VGA-UART

Generated by Doxygen 1.9.6

Chapter 1

CMSIS Cortex-M4

This documentation describes the CMSIS Cortex-M Core Peripheral Access Layer. It consists of:

- Cortex-M Core Register Definitions
- Cortex-M functions
- Cortex-M instructions
- Cortex-M SIMD instructions

The CMSIS Cortex-M4 Core Peripheral Access Layer contains C and assembly functions that ease access to the Cortex-M Core

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

Chapter 3

Data Structure Index

3.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 5

Module Documentation

5.1 CMSIS MISRA-C:2004 Compliance Exceptions

CMSIS violates following MISRA-C2004 Rules:

- Violates MISRA 2004 Required Rule 8.5, object/function definition in header file.
  Function definitions in header files are used to allow 'inlining'.
- Violates MISRA 2004 Required Rule 18.4, declaration of union type or object of union type: '{...}'.
  Unions are used for effective representation of core registers.
- Violates MISRA 2004 Advisory Rule 19.7, Function-like macro defined.
  Function-like macros are used to allow more efficient code.

5.2 CMSIS Core Definitions

Modules

- CMSIS Core Register
- CMSIS Core

Macros

- #define __CM4_CMSIS_VERSION_MAIN (0x02)
- #define __CM4_CMSIS_VERSION_SUB (0x10)
- #define __CM4_CMSIS_VERSION ((__CM4_CMSIS_VERSION_MAIN << 16) | __CM4_CMSIS_VERSION_SUB)
- #define __CORTEX_M (0x04)
- #define **__CORE_CM4_H_DEPENDANT**
- #define __I volatile const
- #define __O volatile
- #define __IO volatile

5.2.1 Detailed Description

This file defines all structures and symbols for CMSIS core:

- CMSIS version number
- Cortex-M core
- Cortex-M core Revision Number

5.2.2 Macro Definition Documentation

5.2.2.1 __CM4_CMSIS_VERSION

```
#define __CM4_CMSIS_VERSION ((__CM4_CMSIS_VERSION_MAIN << 16) | __CM4_CMSIS_VERSION_SUB)
```

CMSIS HAL version number

5.2.2.2 __CM4_CMSIS_VERSION_MAIN

```
#define __CM4_CMSIS_VERSION_MAIN (0x02)
```

[31:16] CMSIS HAL main version

5.2.2.3 __CM4_CMSIS_VERSION_SUB

```
#define __CM4_CMSIS_VERSION_SUB (0x10)
```

[15:0] CMSIS HAL sub version

5.2.2.4 __CORTEX_M

```
#define __CORTEX_M (0x04)
```

Cortex core
__FPU_USED to be checked prior to making use of FPU specific registers and functions standard types definitions
Core Instruction Access
Core Function Access
Compiler specific SIMD Intrinsics

5.2.2.5 __I

```
#define __I volatile const
```

defines 'read only' permissions

5.2.2.6 __IO

```
#define __IO volatile
```

defines 'read / write' permissions

5.2.2.7 __O

```
#define __O volatile
```

defines 'write only' permissions

5.3 CMSIS Core Register

Modules

- CMSIS Core

- #define SCS_BASE (0xE000E000UL)
- #define ITM_BASE (0xE0000000UL)
- #define CoreDebug_BASE (0xE000EDF0UL)
- #define SysTick_BASE (SCS_BASE + 0x0010UL)
- #define NVIC_BASE (SCS_BASE + 0x0100UL)
- #define SCB_BASE (SCS_BASE + 0x0D00UL)
- #define SCnSCB ((SCnSCB_Type *) SCS_BASE)
- #define SCB ((SCB_Type *) SCB_BASE)
- #define SysTick ((SysTick_Type *) SysTick_BASE)
- #define NVIC ((NVIC_Type *) NVIC_BASE)
- #define ITM ((ITM_Type *) ITM_BASE)
- #define CoreDebug ((CoreDebug_Type *) CoreDebug_BASE)
- #define ITM_RXBUFFER_EMPTY 0x5AA55AA5
- volatile int32_t ITM_RxBuffer

5.3.1 Detailed Description

Core Register contain:

- Core Register
- Core NVIC Register
- Core SCB Register
- Core SysTick Register
- Core Debug Register
- Core MPU Register
- Core FPU Register

5.3.2 Macro Definition Documentation

5.3.2.1 CoreDebug

```
#define CoreDebug ((CoreDebug_Type *) CoreDebug_BASE)
```

Core Debug configuration struct

5.3.2.2 CoreDebug_BASE

```
#define CoreDebug_BASE (0xE000EDF0UL)
```

Core Debug Base Address

5.3.2.3 ITM

```
#define ITM ((ITM_Type *) ITM_BASE )
```

ITM configuration struct

5.3.2.4 ITM_BASE

```
#define ITM_BASE (0xE0000000UL)
```

ITM Base Address

5.3.2.5 ITM_RXBUFFER_EMPTY

```
#define ITM_RXBUFFER_EMPTY 0x5AA55AA5
```

value identifying ITM_RxBuffer is ready for next character

5.3.2.6 NVIC

```
#define NVIC ((NVIC_Type *) NVIC_BASE )
```

NVIC configuration struct

5.3.2.7 NVIC_BASE

```
#define NVIC_BASE (SCS_BASE + 0x0100UL)
```

NVIC Base Address

5.3.2.8 SCB

```
#define SCB ((SCB_Type *) SCB_BASE )
```

SCB configuration struct

5.3.2.9 SCB_BASE

```
#define SCB_BASE (SCS_BASE + 0x0D00UL)
```

System Control Block Base Address

5.3.2.10 SCnSCB

```
#define SCnSCB ((SCnSCB_Type *) SCS_BASE )
```

System control Register not in SCB

5.3.2.11 SCS_BASE

```
#define SCS_BASE (0xE000E000UL)
```

System Control Space Base Address

5.3.2.12 SysTick

```
#define SysTick ((SysTick_Type *) SysTick_BASE )
```

SysTick configuration struct

5.3.2.13 SysTick_BASE

```
#define SysTick_BASE (SCS_BASE + 0x0010UL)
```

SysTick Base Address

5.3.3 Variable Documentation

5.3.3.1 ITM_RxBuffer

```
volatile int32_t ITM_RxBuffer  [extern]
```

external variable to receive characters

5.4 CMSIS Core

Modules

- CMSIS NVIC

Data Structures

- union APSR_Type

  Union type to access the Application Program Status Register (APSR).
- union IPSR_Type

  Union type to access the Interrupt Program Status Register (IPSR).
- union xPSR_Type

  Union type to access the Special-Purpose Program Status Registers (xPSR).
- union CONTROL_Type

  Union type to access the Control Registers (CONTROL).

5.4.1 Detailed Description

Type definitions for the Cortex-M Core Registers

5.5 CMSIS NVIC

Modules

- CMSIS SCB

Data Structures

- struct NVIC_Type

  Structure type to access the Nested Vectored Interrupt Controller (NVIC).

Macros

- #define NVIC_STIR_INTID_Pos 0
- #define NVIC_STIR_INTID_Msk (0x1FFUL << NVIC_STIR_INTID_Pos)

5.5.1 Detailed Description

Type definitions for the Cortex-M NVIC Registers

5.5.2 Macro Definition Documentation

5.5.2.1 NVIC_STIR_INTID_Msk

```
#define NVIC_STIR_INTID_Msk (0x1FFUL << NVIC_STIR_INTID_Pos)
```

STIR: INTLINESNUM Mask

5.5.2.2 NVIC_STIR_INTID_Pos

```
#define NVIC_STIR_INTID_Pos 0
```

STIR: INTLINESNUM Position

5.6 CMSIS SCB

Modules

- CMSIS System Control and ID Register not in the SCB

Data Structures

- struct SCB_Type

  Structure type to access the System Control Block (SCB).

Macros

- #define SCB_CPUID_IMPLEMENTER_Pos 24
- #define SCB_CPUID_IMPLEMENTER_Msk (0xFFUL << SCB_CPUID_IMPLEMENTER_Pos)
- #define SCB_CPUID_VARIANT_Pos 20
- #define SCB_CPUID_VARIANT_Msk (0xFUL << SCB_CPUID_VARIANT_Pos)
- #define SCB_CPUID_ARCHITECTURE_Pos 16
- #define SCB_CPUID_ARCHITECTURE_Msk (0xFUL << SCB_CPUID_ARCHITECTURE_Pos)
- #define SCB_CPUID_PARTNO_Pos 4
- #define SCB_CPUID_PARTNO_Msk (0xFFFUL << SCB_CPUID_PARTNO_Pos)
- #define SCB_CPUID_REVISION_Pos 0
- #define SCB_CPUID_REVISION_Msk (0xFUL << SCB_CPUID_REVISION_Pos)
- #define SCB_ICSR_NMIPENDSET_Pos 31
- #define SCB_ICSR_NMIPENDSET_Msk (1UL << SCB_ICSR_NMIPENDSET_Pos)
- #define SCB_ICSR_PENDSVSET_Pos 28
- #define SCB_ICSR_PENDSVSET_Msk (1UL << SCB_ICSR_PENDSVSET_Pos)
- #define SCB_ICSR_PENDSVCLR_Pos 27
- #define SCB_ICSR_PENDSVCLR_Msk (1UL << SCB_ICSR_PENDSVCLR_Pos)
- #define SCB_ICSR_PENDSTSET_Pos 26
- #define SCB_ICSR_PENDSTSET_Msk (1UL << SCB_ICSR_PENDSTSET_Pos)
- #define SCB_ICSR_PENDSTCLR_Pos 25
- #define SCB_ICSR_PENDSTCLR_Msk (1UL << SCB_ICSR_PENDSTCLR_Pos)
- #define SCB_ICSR_ISRPREEMPT_Pos 23
- #define SCB_ICSR_ISRPREEMPT_Msk (1UL << SCB_ICSR_ISRPREEMPT_Pos)
- #define SCB_ICSR_ISRPENDING_Pos 22
- #define SCB_ICSR_ISRPENDING_Msk (1UL << SCB_ICSR_ISRPENDING_Pos)
- #define SCB_ICSR_VECTPENDING_Pos 12
- #define SCB_ICSR_VECTPENDING_Msk (0x1FFUL << SCB_ICSR_VECTPENDING_Pos)
- #define SCB_ICSR_RETTOBASE_Pos 11
- #define SCB_ICSR_RETTOBASE_Msk (1UL << SCB_ICSR_RETTOBASE_Pos)
- #define SCB_ICSR_VECTACTIVE_Pos 0
- #define SCB_ICSR_VECTACTIVE_Msk (0x1FFUL << SCB_ICSR_VECTACTIVE_Pos)
- #define SCB_VTOR_TBLOFF_Pos 7
- #define SCB_VTOR_TBLOFF_Msk (0x1FFFFFFUL << SCB_VTOR_TBLOFF_Pos)
- #define SCB_AIRCR_VECTKEY_Pos 16
- #define SCB_AIRCR_VECTKEY_Msk (0xFFFFUL << SCB_AIRCR_VECTKEY_Pos)
- #define SCB_AIRCR_VECTKEYSTAT_Pos 16
- #define SCB_AIRCR_VECTKEYSTAT_Msk (0xFFFFUL << SCB_AIRCR_VECTKEYSTAT_Pos)
- #define SCB_AIRCR_ENDIANESS_Pos 15
- #define SCB_AIRCR_ENDIANESS_Msk (1UL << SCB_AIRCR_ENDIANESS_Pos)
- #define SCB_AIRCR_PRIGROUP_Pos 8
- #define SCB_AIRCR_PRIGROUP_Msk (7UL << SCB_AIRCR_PRIGROUP_Pos)
- #define SCB_AIRCR_SYSRESETREQ_Pos 2
- #define SCB_AIRCR_SYSRESETREQ_Msk (1UL << SCB_AIRCR_SYSRESETREQ_Pos)
- #define SCB_AIRCR_VECTCLRACTIVE_Pos 1
- #define SCB_AIRCR_VECTCLRACTIVE_Msk (1UL << SCB_AIRCR_VECTCLRACTIVE_Pos)
- #define SCB_AIRCR_VECTRESET_Pos 0
- #define SCB_AIRCR_VECTRESET_Msk (1UL << SCB_AIRCR_VECTRESET_Pos)
- #define SCB_SCR_SEVONPEND_Pos 4

- #define SCB_SCR_SEVONPEND_Msk (1UL << SCB_SCR_SEVONPEND_Pos)
- #define SCB_SCR_SLEEPDEEP_Pos 2
- #define SCB_SCR_SLEEPDEEP_Msk (1UL << SCB_SCR_SLEEPDEEP_Pos)
- #define SCB_SCR_SLEEPONEXIT_Pos 1
- #define SCB_SCR_SLEEPONEXIT_Msk (1UL << SCB_SCR_SLEEPONEXIT_Pos)
- #define SCB_CCR_STKALIGN_Pos 9
- #define SCB_CCR_STKALIGN_Msk (1UL << SCB_CCR_STKALIGN_Pos)
- #define SCB_CCR_BFHFNMIGN_Pos 8
- #define SCB_CCR_BFHFNMIGN_Msk (1UL << SCB_CCR_BFHFNMIGN_Pos)
- #define SCB_CCR_DIV_0_TRP_Pos 4
- #define SCB_CCR_DIV_0_TRP_Msk (1UL << SCB_CCR_DIV_0_TRP_Pos)
- #define SCB_CCR_UNALIGN_TRP_Pos 3
- #define SCB_CCR_UNALIGN_TRP_Msk (1UL << SCB_CCR_UNALIGN_TRP_Pos)
- #define SCB_CCR_USERSETMPEND_Pos 1
- #define SCB_CCR_USERSETMPEND_Msk (1UL << SCB_CCR_USERSETMPEND_Pos)
- #define SCB_CCR_NONBASETHRDENA_Pos 0
- #define SCB_CCR_NONBASETHRDENA_Msk (1UL << SCB_CCR_NONBASETHRDENA_Pos)
- #define SCB_SHCSR_USGFAULTENA_Pos 18
- #define SCB_SHCSR_USGFAULTENA_Msk (1UL << SCB_SHCSR_USGFAULTENA_Pos)
- #define SCB_SHCSR_BUSFAULTENA_Pos 17
- #define SCB_SHCSR_BUSFAULTENA_Msk (1UL << SCB_SHCSR_BUSFAULTENA_Pos)
- #define SCB_SHCSR_MEMFAULTENA_Pos 16
- #define SCB_SHCSR_MEMFAULTENA_Msk (1UL << SCB_SHCSR_MEMFAULTENA_Pos)
- #define SCB_SHCSR_SVCALLPENDED_Pos 15
- #define SCB_SHCSR_SVCALLPENDED_Msk (1UL << SCB_SHCSR_SVCALLPENDED_Pos)
- #define SCB_SHCSR_BUSFAULTPENDED_Pos 14
- #define SCB_SHCSR_BUSFAULTPENDED_Msk (1UL << SCB_SHCSR_BUSFAULTPENDED_Pos)
- #define SCB_SHCSR_MEMFAULTPENDED_Pos 13
- #define SCB_SHCSR_MEMFAULTPENDED_Msk (1UL << SCB_SHCSR_MEMFAULTPENDED_Pos)
- #define SCB_SHCSR_USGFAULTPENDED_Pos 12
- #define SCB_SHCSR_USGFAULTPENDED_Msk (1UL << SCB_SHCSR_USGFAULTPENDED_Pos)
- #define SCB_SHCSR_SYSTICKACT_Pos 11
- #define SCB_SHCSR_SYSTICKACT_Msk (1UL << SCB_SHCSR_SYSTICKACT_Pos)
- #define SCB_SHCSR_PENDSVACT_Pos 10
- #define SCB_SHCSR_PENDSVACT_Msk (1UL << SCB_SHCSR_PENDSVACT_Pos)
- #define SCB_SHCSR_MONITORACT_Pos 8
- #define SCB_SHCSR_MONITORACT_Msk (1UL << SCB_SHCSR_MONITORACT_Pos)
- #define SCB_SHCSR_SVCALLACT_Pos 7
- #define SCB_SHCSR_SVCALLACT_Msk (1UL << SCB_SHCSR_SVCALLACT_Pos)
- #define SCB_SHCSR_USGFAULTACT_Pos 3
- #define SCB_SHCSR_USGFAULTACT_Msk (1UL << SCB_SHCSR_USGFAULTACT_Pos)
- #define SCB_SHCSR_BUSFAULTACT_Pos 1
- #define SCB_SHCSR_BUSFAULTACT_Msk (1UL << SCB_SHCSR_BUSFAULTACT_Pos)
- #define SCB_SHCSR_MEMFAULTACT_Pos 0
- #define SCB_SHCSR_MEMFAULTACT_Msk (1UL << SCB_SHCSR_MEMFAULTACT_Pos)
- #define SCB_CFSR_USGFAULTSR_Pos 16
- #define SCB_CFSR_USGFAULTSR_Msk (0xFFFFUL << SCB_CFSR_USGFAULTSR_Pos)
- #define SCB_CFSR_BUSFAULTSR_Pos 8
- #define SCB_CFSR_BUSFAULTSR_Msk (0xFFUL << SCB_CFSR_BUSFAULTSR_Pos)
- #define SCB_CFSR_MEMFAULTSR_Pos 0
- #define SCB_CFSR_MEMFAULTSR_Msk (0xFFUL << SCB_CFSR_MEMFAULTSR_Pos)
- #define SCB_HFSR_DEBUGEVT_Pos 31
- #define SCB_HFSR_DEBUGEVT_Msk (1UL << SCB_HFSR_DEBUGEVT_Pos)
- #define SCB_HFSR_FORCED_Pos 30
- #define SCB_HFSR_FORCED_Msk (1UL << SCB_HFSR_FORCED_Pos)

- #define SCB_HFSR_VECTTBL_Pos 1
- #define SCB_HFSR_VECTTBL_Msk (1UL << SCB_HFSR_VECTTBL_Pos)
- #define SCB_DFSR_EXTERNAL_Pos 4
- #define SCB_DFSR_EXTERNAL_Msk (1UL << SCB_DFSR_EXTERNAL_Pos)
- #define SCB_DFSR_VCATCH_Pos 3
- #define SCB_DFSR_VCATCH_Msk (1UL << SCB_DFSR_VCATCH_Pos)
- #define SCB_DFSR_DWTTRAP_Pos 2
- #define SCB_DFSR_DWTTRAP_Msk (1UL << SCB_DFSR_DWTTRAP_Pos)
- #define SCB_DFSR_BKPT_Pos 1
- #define SCB_DFSR_BKPT_Msk (1UL << SCB_DFSR_BKPT_Pos)
- #define SCB_DFSR_HALTED_Pos 0
- #define SCB_DFSR_HALTED_Msk (1UL << SCB_DFSR_HALTED_Pos)

5.6.1 Detailed Description

Type definitions for the Cortex-M System Control Block Registers

5.6.2 Macro Definition Documentation

5.6.2.1 SCB_AIRCR_ENDIANESS_Msk

```
#define SCB_AIRCR_ENDIANESS_Msk (1UL << SCB_AIRCR_ENDIANESS_Pos)
```

SCB AIRCR: ENDIANESS Mask

5.6.2.2 SCB_AIRCR_ENDIANESS_Pos

```
#define SCB_AIRCR_ENDIANESS_Pos 15
```

SCB AIRCR: ENDIANESS Position

5.6.2.3 SCB_AIRCR_PRIGROUP_Msk

```
#define SCB_AIRCR_PRIGROUP_Msk (7UL << SCB_AIRCR_PRIGROUP_Pos)
```

SCB AIRCR: PRIGROUP Mask

5.6.2.4 SCB_AIRCR_PRIGROUP_Pos

```
#define SCB_AIRCR_PRIGROUP_Pos 8
```

SCB AIRCR: PRIGROUP Position

5.6.2.5 SCB_AIRCR_SYSRESETREQ_Msk

```
#define SCB_AIRCR_SYSRESETREQ_Msk (1UL << SCB_AIRCR_SYSRESETREQ_Pos)
```

SCB AIRCR: SYSRESETREQ Mask

5.6.2.6 SCB_AIRCR_SYSRESETREQ_Pos

```
#define SCB_AIRCR_SYSRESETREQ_Pos 2
```

SCB AIRCR: SYSRESETREQ Position

5.6.2.7 SCB_AIRCR_VECTCLRACTIVE_Msk

```
#define SCB_AIRCR_VECTCLRACTIVE_Msk (1UL << SCB_AIRCR_VECTCLRACTIVE_Pos)
```

SCB AIRCR: VECTCLRACTIVE Mask

5.6.2.8 SCB_AIRCR_VECTCLRACTIVE_Pos

```
#define SCB_AIRCR_VECTCLRACTIVE_Pos 1
```

SCB AIRCR: VECTCLRACTIVE Position

5.6.2.9 SCB_AIRCR_VECTKEY_Msk

```
#define SCB_AIRCR_VECTKEY_Msk (0xFFFFUL << SCB_AIRCR_VECTKEY_Pos)
```

SCB AIRCR: VECTKEY Mask

5.6.2.10 SCB_AIRCR_VECTKEY_Pos

```
#define SCB_AIRCR_VECTKEY_Pos 16
```

SCB AIRCR: VECTKEY Position

5.6.2.11 SCB_AIRCR_VECTKEYSTAT_Msk

```
#define SCB_AIRCR_VECTKEYSTAT_Msk (0xFFFFUL << SCB_AIRCR_VECTKEYSTAT_Pos)
```

SCB AIRCR: VECTKEYSTAT Mask

5.6.2.12 SCB_AIRCR_VECTKEYSTAT_Pos

```
#define SCB_AIRCR_VECTKEYSTAT_Pos 16
```

SCB AIRCR: VECTKEYSTAT Position

5.6.2.13 SCB_AIRCR_VECTRESET_Msk

```
#define SCB_AIRCR_VECTRESET_Msk (1UL << SCB_AIRCR_VECTRESET_Pos)
```

SCB AIRCR: VECTRESET Mask

5.6.2.14 SCB_AIRCR_VECTRESET_Pos

```
#define SCB_AIRCR_VECTRESET_Pos 0
```

SCB AIRCR: VECTRESET Position

5.6.2.15 SCB_CCR_BFHFNMIGN_Msk

```
#define SCB_CCR_BFHFNMIGN_Msk (1UL << SCB_CCR_BFHFNMIGN_Pos)
```

SCB CCR: BFHFNMIGN Mask

5.6.2.16 SCB_CCR_BFHFNMIGN_Pos

```
#define SCB_CCR_BFHFNMIGN_Pos 8
```

SCB CCR: BFHFNMIGN Position

5.6.2.17 SCB_CCR_DIV_0_TRP_Msk

```
#define SCB_CCR_DIV_0_TRP_Msk (1UL << SCB_CCR_DIV_0_TRP_Pos)
```

SCB CCR: DIV_0_TRP Mask

5.6.2.18 SCB_CCR_DIV_0_TRP_Pos

```
#define SCB_CCR_DIV_0_TRP_Pos 4
```

SCB CCR: DIV_0_TRP Position

5.6.2.19 SCB_CCR_NONBASETHRDENA_Msk

```
#define SCB_CCR_NONBASETHRDENA_Msk (1UL << SCB_CCR_NONBASETHRDENA_Pos)
```

SCB CCR: NONBASETHRDENA Mask

5.6.2.20 SCB_CCR_NONBASETHRDENA_Pos

```
#define SCB_CCR_NONBASETHRDENA_Pos 0
```

SCB CCR: NONBASETHRDENA Position

5.6.2.21 SCB_CCR_STKALIGN_Msk

```
#define SCB_CCR_STKALIGN_Msk (1UL << SCB_CCR_STKALIGN_Pos)
```

SCB CCR: STKALIGN Mask

5.6.2.22 SCB_CCR_STKALIGN_Pos

```
#define SCB_CCR_STKALIGN_Pos 9
```

SCB CCR: STKALIGN Position

5.6.2.23 SCB_CCR_UNALIGN_TRP_Msk

```
#define SCB_CCR_UNALIGN_TRP_Msk (1UL << SCB_CCR_UNALIGN_TRP_Pos)
```

SCB CCR: UNALIGN_TRP Mask

5.6.2.24 SCB_CCR_UNALIGN_TRP_Pos

```
#define SCB_CCR_UNALIGN_TRP_Pos 3
```

SCB CCR: UNALIGN_TRP Position

5.6.2.25 SCB_CCR_USERSETMPEND_Msk

```
#define SCB_CCR_USERSETMPEND_Msk (1UL << SCB_CCR_USERSETMPEND_Pos)
```

SCB CCR: USERSETMPEND Mask

5.6.2.26 SCB_CCR_USERSETMPEND_Pos

```
#define SCB_CCR_USERSETMPEND_Pos 1
```

SCB CCR: USERSETMPEND Position

5.6.2.27 SCB_CFSR_BUSFAULTSR_Msk

```
#define SCB_CFSR_BUSFAULTSR_Msk (0xFFUL << SCB_CFSR_BUSFAULTSR_Pos)
```

SCB CFSR: Bus Fault Status Register Mask

5.6.2.28 SCB_CFSR_BUSFAULTSR_Pos

```
#define SCB_CFSR_BUSFAULTSR_Pos 8
```

SCB CFSR: Bus Fault Status Register Position

5.6.2.29 SCB_CFSR_MEMFAULTSR_Msk

```
#define SCB_CFSR_MEMFAULTSR_Msk (0xFFUL << SCB_CFSR_MEMFAULTSR_Pos)
```

SCB CFSR: Memory Manage Fault Status Register Mask

5.6.2.30 SCB_CFSR_MEMFAULTSR_Pos

```
#define SCB_CFSR_MEMFAULTSR_Pos 0
```

SCB CFSR: Memory Manage Fault Status Register Position

5.6.2.31 SCB_CFSR_USGFAULTSR_Msk

```
#define SCB_CFSR_USGFAULTSR_Msk (0xFFFFUL << SCB_CFSR_USGFAULTSR_Pos)
```

SCB CFSR: Usage Fault Status Register Mask

5.6.2.32 SCB_CFSR_USGFAULTSR_Pos

```
#define SCB_CFSR_USGFAULTSR_Pos 16
```

SCB CFSR: Usage Fault Status Register Position

5.6.2.33 SCB_CPUID_ARCHITECTURE_Msk

```
#define SCB_CPUID_ARCHITECTURE_Msk (0xFUL << SCB_CPUID_ARCHITECTURE_Pos)
```

SCB CPUID: ARCHITECTURE Mask

5.6.2.34 SCB_CPUID_ARCHITECTURE_Pos

```
#define SCB_CPUID_ARCHITECTURE_Pos 16
```

SCB CPUID: ARCHITECTURE Position

5.6.2.35 SCB_CPUID_IMPLEMENTER_Msk

```
#define SCB_CPUID_IMPLEMENTER_Msk (0xFFUL << SCB_CPUID_IMPLEMENTER_Pos)
```

SCB CPUID: IMPLEMENTER Mask

5.6.2.36 SCB_CPUID_IMPLEMENTER_Pos

```
#define SCB_CPUID_IMPLEMENTER_Pos 24
```

SCB CPUID: IMPLEMENTER Position

5.6.2.37 SCB_CPUID_PARTNO_Msk

```
#define SCB_CPUID_PARTNO_Msk (0xFFFUL << SCB_CPUID_PARTNO_Pos)
```

SCB CPUID: PARTNO Mask

5.6.2.38 SCB_CPUID_PARTNO_Pos

```
#define SCB_CPUID_PARTNO_Pos 4
```

SCB CPUID: PARTNO Position

5.6.2.39 SCB_CPUID_REVISION_Msk

```
#define SCB_CPUID_REVISION_Msk (0xFUL << SCB_CPUID_REVISION_Pos)
```

SCB CPUID: REVISION Mask

5.6.2.40 SCB_CPUID_REVISION_Pos

```
#define SCB_CPUID_REVISION_Pos 0
```

SCB CPUID: REVISION Position

5.6.2.41 SCB_CPUID_VARIANT_Msk

```
#define SCB_CPUID_VARIANT_Msk (0xFUL << SCB_CPUID_VARIANT_Pos)
```

SCB CPUID: VARIANT Mask

5.6.2.42 SCB_CPUID_VARIANT_Pos

```
#define SCB_CPUID_VARIANT_Pos 20
```

SCB CPUID: VARIANT Position

5.6.2.43 SCB_DFSR_BKPT_Msk

```
#define SCB_DFSR_BKPT_Msk (1UL << SCB_DFSR_BKPT_Pos)
```

SCB DFSR: BKPT Mask

5.6.2.44 SCB_DFSR_BKPT_Pos

```
#define SCB_DFSR_BKPT_Pos 1
```

SCB DFSR: BKPT Position

5.6.2.45 SCB_DFSR_DWTTRAP_Msk

```
#define SCB_DFSR_DWTTRAP_Msk (1UL << SCB_DFSR_DWTTRAP_Pos)
```

SCB DFSR: DWTTRAP Mask

5.6.2.46 SCB_DFSR_DWTTRAP_Pos

```
#define SCB_DFSR_DWTTRAP_Pos 2
```

SCB DFSR: DWTTRAP Position

5.6.2.47 SCB_DFSR_EXTERNAL_Msk

```
#define SCB_DFSR_EXTERNAL_Msk (1UL << SCB_DFSR_EXTERNAL_Pos)
```

SCB DFSR: EXTERNAL Mask

5.6.2.48 SCB_DFSR_EXTERNAL_Pos

```
#define SCB_DFSR_EXTERNAL_Pos 4
```

SCB DFSR: EXTERNAL Position

5.6.2.49 SCB_DFSR_HALTED_Msk

```
#define SCB_DFSR_HALTED_Msk (1UL << SCB_DFSR_HALTED_Pos)
```

SCB DFSR: HALTED Mask

5.6.2.50 SCB_DFSR_HALTED_Pos

```
#define SCB_DFSR_HALTED_Pos 0
```

SCB DFSR: HALTED Position

5.6.2.51 SCB_DFSR_VCATCH_Msk

```
#define SCB_DFSR_VCATCH_Msk (1UL << SCB_DFSR_VCATCH_Pos)
```

SCB DFSR: VCATCH Mask

5.6.2.52 SCB_DFSR_VCATCH_Pos

```
#define SCB_DFSR_VCATCH_Pos 3
```

SCB DFSR: VCATCH Position

5.6.2.53 SCB_HFSR_DEBUGEVT_Msk

```
#define SCB_HFSR_DEBUGEVT_Msk (1UL << SCB_HFSR_DEBUGEVT_Pos)
```

SCB HFSR: DEBUGEVT Mask

5.6.2.54 SCB_HFSR_DEBUGEVT_Pos

```
#define SCB_HFSR_DEBUGEVT_Pos 31
```

SCB HFSR: DEBUGEVT Position

5.6.2.55 SCB_HFSR_FORCED_Msk

```
#define SCB_HFSR_FORCED_Msk (1UL << SCB_HFSR_FORCED_Pos)
```

SCB HFSR: FORCED Mask

5.6.2.56 SCB_HFSR_FORCED_Pos

```
#define SCB_HFSR_FORCED_Pos 30
```

SCB HFSR: FORCED Position

5.6.2.57 SCB_HFSR_VECTTBL_Msk

```
#define SCB_HFSR_VECTTBL_Msk (1UL << SCB_HFSR_VECTTBL_Pos)
```

SCB HFSR: VECTTBL Mask

5.6.2.58 SCB_HFSR_VECTTBL_Pos

```
#define SCB_HFSR_VECTTBL_Pos 1
```

SCB HFSR: VECTTBL Position

5.6.2.59 SCB_ICSR_ISRPENDING_Msk

```
#define SCB_ICSR_ISRPENDING_Msk (1UL << SCB_ICSR_ISRPENDING_Pos)
```

SCB ICSR: ISRPENDING Mask

5.6.2.60 SCB_ICSR_ISRPENDING_Pos

```
#define SCB_ICSR_ISRPENDING_Pos 22
```

SCB ICSR: ISRPENDING Position

5.6.2.61 SCB_ICSR_ISRPREEMPT_Msk

```
#define SCB_ICSR_ISRPREEMPT_Msk (1UL << SCB_ICSR_ISRPREEMPT_Pos)
```

SCB ICSR: ISRPREEMPT Mask

5.6.2.62 SCB_ICSR_ISRPREEMPT_Pos

```
#define SCB_ICSR_ISRPREEMPT_Pos 23
```

SCB ICSR: ISRPREEMPT Position

5.6.2.63 SCB_ICSR_NMIPENDSET_Msk

```
#define SCB_ICSR_NMIPENDSET_Msk (1UL << SCB_ICSR_NMIPENDSET_Pos)
```

SCB ICSR: NMIPENDSET Mask

5.6.2.64 SCB_ICSR_NMIPENDSET_Pos

```
#define SCB_ICSR_NMIPENDSET_Pos 31
```

SCB ICSR: NMIPENDSET Position

5.6.2.65 SCB_ICSR_PENDSTCLR_Msk

```
#define SCB_ICSR_PENDSTCLR_Msk (1UL << SCB_ICSR_PENDSTCLR_Pos)
```

SCB ICSR: PENDSTCLR Mask

5.6.2.66 SCB_ICSR_PENDSTCLR_Pos

```
#define SCB_ICSR_PENDSTCLR_Pos 25
```

SCB ICSR: PENDSTCLR Position

5.6.2.67 SCB_ICSR_PENDSTSET_Msk

```
#define SCB_ICSR_PENDSTSET_Msk (1UL << SCB_ICSR_PENDSTSET_Pos)
```

SCB ICSR: PENDSTSET Mask

5.6.2.68 SCB_ICSR_PENDSTSET_Pos

```
#define SCB_ICSR_PENDSTSET_Pos 26
```

SCB ICSR: PENDSTSET Position

5.6.2.69 SCB_ICSR_PENDSVCLR_Msk

```
#define SCB_ICSR_PENDSVCLR_Msk (1UL << SCB_ICSR_PENDSVCLR_Pos)
```

SCB ICSR: PENDSVCLR Mask

5.6.2.70 SCB_ICSR_PENDSVCLR_Pos

```
#define SCB_ICSR_PENDSVCLR_Pos 27
```

SCB ICSR: PENDSVCLR Position

5.6.2.71 SCB_ICSR_PENDSVSET_Msk

```
#define SCB_ICSR_PENDSVSET_Msk (1UL << SCB_ICSR_PENDSVSET_Pos)
```

SCB ICSR: PENDSVSET Mask

5.6.2.72 SCB_ICSR_PENDSVSET_Pos

```
#define SCB_ICSR_PENDSVSET_Pos 28
```

SCB ICSR: PENDSVSET Position

5.6.2.73 SCB_ICSR_RETTOBASE_Msk

```
#define SCB_ICSR_RETTOBASE_Msk (1UL << SCB_ICSR_RETTOBASE_Pos)
```

SCB ICSR: RETTOBASE Mask

5.6.2.74 SCB_ICSR_RETTOBASE_Pos

```
#define SCB_ICSR_RETTOBASE_Pos 11
```

SCB ICSR: RETTOBASE Position

5.6.2.75 SCB_ICSR_VECTACTIVE_Msk

```
#define SCB_ICSR_VECTACTIVE_Msk (0x1FFUL << SCB_ICSR_VECTACTIVE_Pos)
```

SCB ICSR: VECTACTIVE Mask

5.6.2.76 SCB_ICSR_VECTACTIVE_Pos

```
#define SCB_ICSR_VECTACTIVE_Pos 0
```

SCB ICSR: VECTACTIVE Position

5.6.2.77 SCB_ICSR_VECTPENDING_Msk

```
#define SCB_ICSR_VECTPENDING_Msk (0x1FFUL << SCB_ICSR_VECTPENDING_Pos)
```

SCB ICSR: VECTPENDING Mask

5.6.2.78 SCB_ICSR_VECTPENDING_Pos

```
#define SCB_ICSR_VECTPENDING_Pos 12
```

SCB ICSR: VECTPENDING Position

5.6.2.79 SCB_SCR_SEVONPEND_Msk

```
#define SCB_SCR_SEVONPEND_Msk (1UL << SCB_SCR_SEVONPEND_Pos)
```

SCB SCR: SEVONPEND Mask

5.6.2.80 SCB_SCR_SEVONPEND_Pos

```
#define SCB_SCR_SEVONPEND_Pos 4
```

SCB SCR: SEVONPEND Position

5.6.2.81 SCB_SCR_SLEEPDEEP_Msk

```
#define SCB_SCR_SLEEPDEEP_Msk (1UL << SCB_SCR_SLEEPDEEP_Pos)
```

SCB SCR: SLEEPDEEP Mask

5.6.2.82 SCB_SCR_SLEEPDEEP_Pos

```
#define SCB_SCR_SLEEPDEEP_Pos 2
```

SCB SCR: SLEEPDEEP Position

5.6.2.83 SCB_SCR_SLEEPONEXIT_Msk

```
#define SCB_SCR_SLEEPONEXIT_Msk (1UL << SCB_SCR_SLEEPONEXIT_Pos)
```

SCB SCR: SLEEPONEXIT Mask

5.6.2.84 SCB_SCR_SLEEPONEXIT_Pos

```
#define SCB_SCR_SLEEPONEXIT_Pos 1
```

SCB SCR: SLEEPONEXIT Position

5.6.2.85 SCB_SHCSR_BUSFAULTACT_Msk

```
#define SCB_SHCSR_BUSFAULTACT_Msk (1UL << SCB_SHCSR_BUSFAULTACT_Pos)
```

SCB SHCSR: BUSFAULTACT Mask

5.6.2.86 SCB_SHCSR_BUSFAULTACT_Pos

```
#define SCB_SHCSR_BUSFAULTACT_Pos 1
```

SCB SHCSR: BUSFAULTACT Position

5.6.2.87 SCB_SHCSR_BUSFAULTENA_Msk

```
#define SCB_SHCSR_BUSFAULTENA_Msk (1UL << SCB_SHCSR_BUSFAULTENA_Pos)
```

SCB SHCSR: BUSFAULTENA Mask

5.6.2.88 SCB_SHCSR_BUSFAULTENA_Pos

```
#define SCB_SHCSR_BUSFAULTENA_Pos 17
```

SCB SHCSR: BUSFAULTENA Position

5.6.2.89 SCB_SHCSR_BUSFAULTPENDED_Msk

```
#define SCB_SHCSR_BUSFAULTPENDED_Msk (1UL << SCB_SHCSR_BUSFAULTPENDED_Pos)
```

SCB SHCSR: BUSFAULTPENDED Mask

5.6.2.90 SCB_SHCSR_BUSFAULTPENDED_Pos

```
#define SCB_SHCSR_BUSFAULTPENDED_Pos 14
```

SCB SHCSR: BUSFAULTPENDED Position

5.6.2.91 SCB_SHCSR_MEMFAULTACT_Msk

```
#define SCB_SHCSR_MEMFAULTACT_Msk (1UL << SCB_SHCSR_MEMFAULTACT_Pos)
```

SCB SHCSR: MEMFAULTACT Mask

5.6.2.92 SCB_SHCSR_MEMFAULTACT_Pos

```
#define SCB_SHCSR_MEMFAULTACT_Pos 0
```

SCB SHCSR: MEMFAULTACT Position

5.6.2.93 SCB_SHCSR_MEMFAULTENA_Msk

```
#define SCB_SHCSR_MEMFAULTENA_Msk (1UL << SCB_SHCSR_MEMFAULTENA_Pos)
```

SCB SHCSR: MEMFAULTENA Mask

5.6.2.94 SCB_SHCSR_MEMFAULTENA_Pos

```
#define SCB_SHCSR_MEMFAULTENA_Pos 16
```

SCB SHCSR: MEMFAULTENA Position

5.6.2.95 SCB_SHCSR_MEMFAULTPENDED_Msk

```
#define SCB_SHCSR_MEMFAULTPENDED_Msk (1UL << SCB_SHCSR_MEMFAULTPENDED_Pos)
```

SCB SHCSR: MEMFAULTPENDED Mask

5.6.2.96 SCB_SHCSR_MEMFAULTPENDED_Pos

```
#define SCB_SHCSR_MEMFAULTPENDED_Pos 13
```

SCB SHCSR: MEMFAULTPENDED Position

5.6.2.97 SCB_SHCSR_MONITORACT_Msk

```
#define SCB_SHCSR_MONITORACT_Msk (1UL << SCB_SHCSR_MONITORACT_Pos)
```

SCB SHCSR: MONITORACT Mask

5.6.2.98 SCB_SHCSR_MONITORACT_Pos

```
#define SCB_SHCSR_MONITORACT_Pos 8
```

SCB SHCSR: MONITORACT Position

5.6.2.99 SCB_SHCSR_PENDSVACT_Msk

```
#define SCB_SHCSR_PENDSVACT_Msk (1UL << SCB_SHCSR_PENDSVACT_Pos)
```

SCB SHCSR: PENDSVACT Mask

5.6.2.100 SCB_SHCSR_PENDSVACT_Pos

```
#define SCB_SHCSR_PENDSVACT_Pos 10
```

SCB SHCSR: PENDSVACT Position

5.6.2.101 SCB_SHCSR_SVCALLACT_Msk

```
#define SCB_SHCSR_SVCALLACT_Msk (1UL << SCB_SHCSR_SVCALLACT_Pos)
```

SCB SHCSR: SVCALLACT Mask

5.6.2.102 SCB_SHCSR_SVCALLACT_Pos

```
#define SCB_SHCSR_SVCALLACT_Pos 7
```

SCB SHCSR: SVCALLACT Position

5.6.2.103 SCB_SHCSR_SVCALLPENDED_Msk

```
#define SCB_SHCSR_SVCALLPENDED_Msk (1UL << SCB_SHCSR_SVCALLPENDED_Pos)
```

SCB SHCSR: SVCALLPENDED Mask

5.6.2.104 SCB_SHCSR_SVCALLPENDED_Pos

```
#define SCB_SHCSR_SVCALLPENDED_Pos 15
```

SCB SHCSR: SVCALLPENDED Position

5.6.2.105 SCB_SHCSR_SYSTICKACT_Msk

```
#define SCB_SHCSR_SYSTICKACT_Msk (1UL << SCB_SHCSR_SYSTICKACT_Pos)
```

SCB SHCSR: SYSTICKACT Mask

5.6.2.106 SCB_SHCSR_SYSTICKACT_Pos

```
#define SCB_SHCSR_SYSTICKACT_Pos 11
```

SCB SHCSR: SYSTICKACT Position

5.6.2.107 SCB_SHCSR_USGFAULTACT_Msk

```
#define SCB_SHCSR_USGFAULTACT_Msk (1UL << SCB_SHCSR_USGFAULTACT_Pos)
```

SCB SHCSR: USGFAULTACT Mask

5.6.2.108 SCB_SHCSR_USGFAULTACT_Pos

```
#define SCB_SHCSR_USGFAULTACT_Pos 3
```

SCB SHCSR: USGFAULTACT Position

5.6.2.109 SCB_SHCSR_USGFAULTENA_Msk

```
#define SCB_SHCSR_USGFAULTENA_Msk (1UL << SCB_SHCSR_USGFAULTENA_Pos)
```

SCB SHCSR: USGFAULTENA Mask

5.6.2.110 SCB_SHCSR_USGFAULTENA_Pos

```
#define SCB_SHCSR_USGFAULTENA_Pos 18
```

SCB SHCSR: USGFAULTENA Position

5.6.2.111 SCB_SHCSR_USGFAULTPENDED_Msk

```
#define SCB_SHCSR_USGFAULTPENDED_Msk (1UL << SCB_SHCSR_USGFAULTPENDED_Pos)
```

SCB SHCSR: USGFAULTPENDED Mask

5.6.2.112 SCB_SHCSR_USGFAULTPENDED_Pos

```
#define SCB_SHCSR_USGFAULTPENDED_Pos 12
```

SCB SHCSR: USGFAULTPENDED Position

5.6.2.113 SCB_VTOR_TBLOFF_Msk

```
#define SCB_VTOR_TBLOFF_Msk (0x1FFFFFFUL << SCB_VTOR_TBLOFF_Pos)
```

SCB VTOR: TBLOFF Mask

5.6.2.114 SCB_VTOR_TBLOFF_Pos

```
#define SCB_VTOR_TBLOFF_Pos 7
```

SCB VTOR: TBLOFF Position

5.7 CMSIS System Control and ID Register not in the SCB

Modules

- CMSIS SysTick

Data Structures

- struct SCnSCB_Type

  Structure type to access the System Control and ID Register not in the SCB.

Macros

- #define SCnSCB_ICTR_INTLINESNUM_Pos 0
- #define SCnSCB_ICTR_INTLINESNUM_Msk (0xFUL << SCnSCB_ICTR_INTLINESNUM_Pos)
- #define SCnSCB_ACTLR_DISOOFP_Pos 9
- #define SCnSCB_ACTLR_DISOOFP_Msk (1UL << SCnSCB_ACTLR_DISOOFP_Pos)
- #define SCnSCB_ACTLR_DISFPCA_Pos 8
- #define SCnSCB_ACTLR_DISFPCA_Msk (1UL << SCnSCB_ACTLR_DISFPCA_Pos)
- #define SCnSCB_ACTLR_DISFOLD_Pos 2
- #define SCnSCB_ACTLR_DISFOLD_Msk (1UL << SCnSCB_ACTLR_DISFOLD_Pos)
- #define SCnSCB_ACTLR_DISDEFWBUF_Pos 1
- #define SCnSCB_ACTLR_DISDEFWBUF_Msk (1UL << SCnSCB_ACTLR_DISDEFWBUF_Pos)
- #define SCnSCB_ACTLR_DISMCYCINT_Pos 0
- #define SCnSCB_ACTLR_DISMCYCINT_Msk (1UL << SCnSCB_ACTLR_DISMCYCINT_Pos)

5.7.1 Detailed Description

Type definitions for the Cortex-M System Control and ID Register not in the SCB

5.7.2 Macro Definition Documentation

5.7.2.1 SCnSCB_ACTLR_DISDEFWBUF_Msk

```
#define SCnSCB_ACTLR_DISDEFWBUF_Msk (1UL << SCnSCB_ACTLR_DISDEFWBUF_Pos)
```

ACTLR: DISDEFWBUF Mask

5.7.2.2 SCnSCB_ACTLR_DISDEFWBUF_Pos

```
#define SCnSCB_ACTLR_DISDEFWBUF_Pos 1
```

ACTLR: DISDEFWBUF Position

5.7.2.3 SCnSCB_ACTLR_DISFOLD_Msk

```
#define SCnSCB_ACTLR_DISFOLD_Msk (1UL << SCnSCB_ACTLR_DISFOLD_Pos)
```

ACTLR: DISFOLD Mask

5.7.2.4 SCnSCB_ACTLR_DISFOLD_Pos

```
#define SCnSCB_ACTLR_DISFOLD_Pos 2
```

ACTLR: DISFOLD Position

5.7.2.5 SCnSCB_ACTLR_DISFPCA_Msk

```
#define SCnSCB_ACTLR_DISFPCA_Msk (1UL << SCnSCB_ACTLR_DISFPCA_Pos)
```

ACTLR: DISFPCA Mask

5.7.2.6 SCnSCB_ACTLR_DISFPCA_Pos

```
#define SCnSCB_ACTLR_DISFPCA_Pos 8
```

ACTLR: DISFPCA Position

5.7.2.7 SCnSCB_ACTLR_DISMCYCINT_Msk

```
#define SCnSCB_ACTLR_DISMCYCINT_Msk (1UL << SCnSCB_ACTLR_DISMCYCINT_Pos)
```

ACTLR: DISMCYCINT Mask

5.7.2.8 SCnSCB_ACTLR_DISMCYCINT_Pos

```
#define SCnSCB_ACTLR_DISMCYCINT_Pos 0
```

ACTLR: DISMCYCINT Position

5.7.2.9 SCnSCB_ACTLR_DISOOFP_Msk

```
#define SCnSCB_ACTLR_DISOOFP_Msk (1UL << SCnSCB_ACTLR_DISOOFP_Pos)
```

ACTLR: DISOOFP Mask

5.7.2.10 SCnSCB_ACTLR_DISOOFP_Pos

```
#define SCnSCB_ACTLR_DISOOFP_Pos 9
```

ACTLR: DISOOFP Position

5.7.2.11 SCnSCB_ICTR_INTLINESNUM_Msk

```
#define SCnSCB_ICTR_INTLINESNUM_Msk (0xFUL << SCnSCB_ICTR_INTLINESNUM_Pos)
```

ICTR: INTLINESNUM Mask

5.7.2.12 SCnSCB_ICTR_INTLINESNUM_Pos

```
#define SCnSCB_ICTR_INTLINESNUM_Pos 0
```

ICTR: INTLINESNUM Position

5.8 CMSIS SysTick

Modules

- CMSIS ITM

Data Structures

- struct SysTick_Type

  Structure type to access the System Timer (SysTick).

Macros

- #define SysTick_CTRL_COUNTFLAG_Pos 16
- #define SysTick_CTRL_COUNTFLAG_Msk (1UL << SysTick_CTRL_COUNTFLAG_Pos)
- #define SysTick_CTRL_CLKSOURCE_Pos 2
- #define SysTick_CTRL_CLKSOURCE_Msk (1UL << SysTick_CTRL_CLKSOURCE_Pos)
- #define SysTick_CTRL_TICKINT_Pos 1
- #define SysTick_CTRL_TICKINT_Msk (1UL << SysTick_CTRL_TICKINT_Pos)
- #define SysTick_CTRL_ENABLE_Pos 0
- #define SysTick_CTRL_ENABLE_Msk (1UL << SysTick_CTRL_ENABLE_Pos)
- #define SysTick_LOAD_RELOAD_Pos 0
- #define SysTick_LOAD_RELOAD_Msk (0xFFFFFFUL << SysTick_LOAD_RELOAD_Pos)
- #define SysTick_VAL_CURRENT_Pos 0
- #define SysTick_VAL_CURRENT_Msk (0xFFFFFFUL << SysTick_VAL_CURRENT_Pos)
- #define SysTick_CALIB_NOREF_Pos 31
- #define SysTick_CALIB_NOREF_Msk (1UL << SysTick_CALIB_NOREF_Pos)
- #define SysTick_CALIB_SKEW_Pos 30
- #define SysTick_CALIB_SKEW_Msk (1UL << SysTick_CALIB_SKEW_Pos)
- #define SysTick_CALIB_TENMS_Pos 0
- #define SysTick_CALIB_TENMS_Msk (0xFFFFFFUL << SysTick_VAL_CURRENT_Pos)

5.8.1 Detailed Description

Type definitions for the Cortex-M System Timer Registers

5.8.2 Macro Definition Documentation

5.8.2.1 SysTick_CALIB_NOREF_Msk

```
#define SysTick_CALIB_NOREF_Msk (1UL << SysTick_CALIB_NOREF_Pos)
```

SysTick CALIB: NOREF Mask

5.8.2.2 SysTick_CALIB_NOREF_Pos

```
#define SysTick_CALIB_NOREF_Pos 31
```

SysTick CALIB: NOREF Position

5.8.2.3 SysTick_CALIB_SKEW_Msk

```
#define SysTick_CALIB_SKEW_Msk (1UL << SysTick_CALIB_SKEW_Pos)
```

SysTick CALIB: SKEW Mask

5.8.2.4 SysTick_CALIB_SKEW_Pos

```
#define SysTick_CALIB_SKEW_Pos 30
```

SysTick CALIB: SKEW Position

5.8.2.5 SysTick_CALIB_TENMS_Msk

```
#define SysTick_CALIB_TENMS_Msk (0xFFFFFFUL << SysTick_VAL_CURRENT_Pos)
```

SysTick CALIB: TENMS Mask

5.8.2.6 SysTick_CALIB_TENMS_Pos

```
#define SysTick_CALIB_TENMS_Pos 0
```

SysTick CALIB: TENMS Position

5.8.2.7 SysTick_CTRL_CLKSOURCE_Msk

```
#define SysTick_CTRL_CLKSOURCE_Msk (1UL << SysTick_CTRL_CLKSOURCE_Pos)
```

SysTick CTRL: CLKSOURCE Mask

5.8.2.8 SysTick_CTRL_CLKSOURCE_Pos

```
#define SysTick_CTRL_CLKSOURCE_Pos 2
```

SysTick CTRL: CLKSOURCE Position

5.8.2.9 SysTick_CTRL_COUNTFLAG_Msk

```
#define SysTick_CTRL_COUNTFLAG_Msk (1UL << SysTick_CTRL_COUNTFLAG_Pos)
```

SysTick CTRL: COUNTFLAG Mask

5.8.2.10 SysTick_CTRL_COUNTFLAG_Pos

```
#define SysTick_CTRL_COUNTFLAG_Pos 16
```

SysTick CTRL: COUNTFLAG Position

5.8.2.11 SysTick_CTRL_ENABLE_Msk

```
#define SysTick_CTRL_ENABLE_Msk (1UL << SysTick_CTRL_ENABLE_Pos)
```

SysTick CTRL: ENABLE Mask

5.8.2.12 SysTick_CTRL_ENABLE_Pos

```
#define SysTick_CTRL_ENABLE_Pos 0
```

SysTick CTRL: ENABLE Position

5.8.2.13 SysTick_CTRL_TICKINT_Msk

```
#define SysTick_CTRL_TICKINT_Msk (1UL << SysTick_CTRL_TICKINT_Pos)
```

SysTick CTRL: TICKINT Mask

5.8.2.14 SysTick_CTRL_TICKINT_Pos

```
#define SysTick_CTRL_TICKINT_Pos 1
```

SysTick CTRL: TICKINT Position

5.8.2.15 SysTick_LOAD_RELOAD_Msk

```
#define SysTick_LOAD_RELOAD_Msk (0xFFFFFFUL << SysTick_LOAD_RELOAD_Pos)
```

SysTick LOAD: RELOAD Mask

5.8.2.16 SysTick_LOAD_RELOAD_Pos

```
#define SysTick_LOAD_RELOAD_Pos 0
```

SysTick LOAD: RELOAD Position

5.8.2.17 SysTick_VAL_CURRENT_Msk

```
#define SysTick_VAL_CURRENT_Msk (0xFFFFFFUL << SysTick_VAL_CURRENT_Pos)
```

SysTick VAL: CURRENT Mask

5.8.2.18 SysTick_VAL_CURRENT_Pos

```
#define SysTick_VAL_CURRENT_Pos 0
```

SysTick VAL: CURRENT Position

5.9 CMSIS ITM

Modules

- CMSIS Core Debug

Data Structures

- struct ITM_Type

  Structure type to access the Instrumentation Trace Macrocell Register (ITM).

Macros

- #define ITM_TPR_PRIVMASK_Pos 0
- #define ITM_TPR_PRIVMASK_Msk (0xFUL << ITM_TPR_PRIVMASK_Pos)
- #define ITM_TCR_BUSY_Pos 23
- #define ITM_TCR_BUSY_Msk (1UL << ITM_TCR_BUSY_Pos)
- #define ITM_TCR_TraceBusID_Pos 16
- #define ITM_TCR_TraceBusID_Msk (0x7FUL << ITM_TCR_TraceBusID_Pos)
- #define ITM_TCR_GTSFREQ_Pos 10
- #define ITM_TCR_GTSFREQ_Msk (3UL << ITM_TCR_GTSFREQ_Pos)
- #define ITM_TCR_TSPrescale_Pos 8
- #define ITM_TCR_TSPrescale_Msk (3UL << ITM_TCR_TSPrescale_Pos)
- #define ITM_TCR_SWOENA_Pos 4
- #define ITM_TCR_SWOENA_Msk (1UL << ITM_TCR_SWOENA_Pos)
- #define ITM_TCR_TXENA_Pos 3
- #define ITM_TCR_TXENA_Msk (1UL << ITM_TCR_TXENA_Pos)
- #define ITM_TCR_SYNCENA_Pos 2
- #define ITM_TCR_SYNCENA_Msk (1UL << ITM_TCR_SYNCENA_Pos)
- #define ITM_TCR_TSENA_Pos 1
- #define ITM_TCR_TSENA_Msk (1UL << ITM_TCR_TSENA_Pos)
- #define ITM_TCR_ITMENA_Pos 0
- #define ITM_TCR_ITMENA_Msk (1UL << ITM_TCR_ITMENA_Pos)

5.9.1 Detailed Description

Type definitions for the Cortex-M Instrumentation Trace Macrocell (ITM)

5.9.2 Macro Definition Documentation

5.9.2.1 ITM_TCR_BUSY_Msk

```
#define ITM_TCR_BUSY_Msk (1UL << ITM_TCR_BUSY_Pos)
```

ITM TCR: BUSY Mask

5.9.2.2 ITM_TCR_BUSY_Pos

```
#define ITM_TCR_BUSY_Pos 23
```

ITM TCR: BUSY Position

5.9.2.3 ITM_TCR_GTSFREQ_Msk

```
#define ITM_TCR_GTSFREQ_Msk (3UL << ITM_TCR_GTSFREQ_Pos)
```

ITM TCR: Global timestamp frequency Mask

5.9.2.4 ITM_TCR_GTSFREQ_Pos

```
#define ITM_TCR_GTSFREQ_Pos 10
```

ITM TCR: Global timestamp frequency Position

5.9.2.5 ITM_TCR_ITMENA_Msk

```
#define ITM_TCR_ITMENA_Msk (1UL << ITM_TCR_ITMENA_Pos)
```

ITM TCR: ITM Enable bit Mask

5.9.2.6 ITM_TCR_ITMENA_Pos

```
#define ITM_TCR_ITMENA_Pos 0
```

ITM TCR: ITM Enable bit Position

5.9.2.7 ITM_TCR_SWOENA_Msk

```
#define ITM_TCR_SWOENA_Msk (1UL << ITM_TCR_SWOENA_Pos)
```

ITM TCR: SWOENA Mask

5.9.2.8 ITM_TCR_SWOENA_Pos

```
#define ITM_TCR_SWOENA_Pos 4
```

ITM TCR: SWOENA Position

5.9.2.9 ITM_TCR_SYNCENA_Msk

```
#define ITM_TCR_SYNCENA_Msk (1UL << ITM_TCR_SYNCENA_Pos)
```

ITM TCR: SYNCENA Mask

5.9.2.10 ITM_TCR_SYNCENA_Pos

```
#define ITM_TCR_SYNCENA_Pos 2
```

ITM TCR: SYNCENA Position

5.9.2.11 ITM_TCR_TraceBusID_Msk

```
#define ITM_TCR_TraceBusID_Msk (0x7FUL << ITM_TCR_TraceBusID_Pos)
```

ITM TCR: ATBID Mask

5.9.2.12 ITM_TCR_TraceBusID_Pos

```
#define ITM_TCR_TraceBusID_Pos 16
```

ITM TCR: ATBID Position

5.9.2.13 ITM_TCR_TSENA_Msk

```
#define ITM_TCR_TSENA_Msk (1UL << ITM_TCR_TSENA_Pos)
```

ITM TCR: TSENA Mask

5.9.2.14 ITM_TCR_TSENA_Pos

```
#define ITM_TCR_TSENA_Pos 1
```

ITM TCR: TSENA Position

5.9.2.15 ITM_TCR_TSPrescale_Msk

```
#define ITM_TCR_TSPrescale_Msk (3UL << ITM_TCR_TSPrescale_Pos)
```

ITM TCR: TSPrescale Mask

5.9.2.16 ITM_TCR_TSPrescale_Pos

```
#define ITM_TCR_TSPrescale_Pos 8
```

ITM TCR: TSPrescale Position

5.9.2.17 ITM_TCR_TXENA_Msk

```
#define ITM_TCR_TXENA_Msk (1UL << ITM_TCR_TXENA_Pos)
```

ITM TCR: TXENA Mask

5.9.2.18 ITM_TCR_TXENA_Pos

```
#define ITM_TCR_TXENA_Pos 3
```

ITM TCR: TXENA Position

5.9.2.19 ITM_TPR_PRIVMASK_Msk

```
#define ITM_TPR_PRIVMASK_Msk (0xFUL << ITM_TPR_PRIVMASK_Pos)
```

ITM TPR: PRIVMASK Mask

5.9.2.20 ITM_TPR_PRIVMASK_Pos

```
#define ITM_TPR_PRIVMASK_Pos 0
```

ITM TPR: PRIVMASK Position

5.10 CMSIS Core Debug

Modules

- CMSIS Core Function Interface
- CMSIS Core NVIC Functions

Data Structures

- struct CoreDebug_Type

  Structure type to access the Core Debug Register (CoreDebug).

Macros

- #define CoreDebug_DHCSR_DBGKEY_Pos 16
- #define CoreDebug_DHCSR_DBGKEY_Msk (0xFFFFUL << CoreDebug_DHCSR_DBGKEY_Pos)
- #define CoreDebug_DHCSR_S_RESET_ST_Pos 25
- #define CoreDebug_DHCSR_S_RESET_ST_Msk (1UL << CoreDebug_DHCSR_S_RESET_ST_Pos)
- #define CoreDebug_DHCSR_S_RETIRE_ST_Pos 24
- #define CoreDebug_DHCSR_S_RETIRE_ST_Msk (1UL << CoreDebug_DHCSR_S_RETIRE_ST_Pos)
- #define CoreDebug_DHCSR_S_LOCKUP_Pos 19
- #define CoreDebug_DHCSR_S_LOCKUP_Msk (1UL << CoreDebug_DHCSR_S_LOCKUP_Pos)
- #define CoreDebug_DHCSR_S_SLEEP_Pos 18
- #define CoreDebug_DHCSR_S_SLEEP_Msk (1UL << CoreDebug_DHCSR_S_SLEEP_Pos)
- #define CoreDebug_DHCSR_S_HALT_Pos 17
- #define CoreDebug_DHCSR_S_HALT_Msk (1UL << CoreDebug_DHCSR_S_HALT_Pos)
- #define CoreDebug_DHCSR_S_REGRDY_Pos 16
- #define CoreDebug_DHCSR_S_REGRDY_Msk (1UL << CoreDebug_DHCSR_S_REGRDY_Pos)
- #define CoreDebug_DHCSR_C_SNAPSTALL_Pos 5
- #define CoreDebug_DHCSR_C_SNAPSTALL_Msk (1UL << CoreDebug_DHCSR_C_SNAPSTALL_Pos)
- #define CoreDebug_DHCSR_C_MASKINTS_Pos 3
- #define CoreDebug_DHCSR_C_MASKINTS_Msk (1UL << CoreDebug_DHCSR_C_MASKINTS_Pos)
- #define CoreDebug_DHCSR_C_STEP_Pos 2
- #define CoreDebug_DHCSR_C_STEP_Msk (1UL << CoreDebug_DHCSR_C_STEP_Pos)
- #define CoreDebug_DHCSR_C_HALT_Pos 1
- #define CoreDebug_DHCSR_C_HALT_Msk (1UL << CoreDebug_DHCSR_C_HALT_Pos)
- #define CoreDebug_DHCSR_C_DEBUGEN_Pos 0
- #define CoreDebug_DHCSR_C_DEBUGEN_Msk (1UL << CoreDebug_DHCSR_C_DEBUGEN_Pos)
- #define CoreDebug_DCRSR_REGWnR_Pos 16
- #define CoreDebug_DCRSR_REGWnR_Msk (1UL << CoreDebug_DCRSR_REGWnR_Pos)
- #define CoreDebug_DCRSR_REGSEL_Pos 0
- #define CoreDebug_DCRSR_REGSEL_Msk (0x1FUL << CoreDebug_DCRSR_REGSEL_Pos)
- #define CoreDebug_DEMCR_TRCENA_Pos 24
- #define CoreDebug_DEMCR_TRCENA_Msk (1UL << CoreDebug_DEMCR_TRCENA_Pos)
- #define CoreDebug_DEMCR_MON_REQ_Pos 19
- #define CoreDebug_DEMCR_MON_REQ_Msk (1UL << CoreDebug_DEMCR_MON_REQ_Pos)
- #define CoreDebug_DEMCR_MON_STEP_Pos 18
- #define CoreDebug_DEMCR_MON_STEP_Msk (1UL << CoreDebug_DEMCR_MON_STEP_Pos)
- #define CoreDebug_DEMCR_MON_PEND_Pos 17
- #define CoreDebug_DEMCR_MON_PEND_Msk (1UL << CoreDebug_DEMCR_MON_PEND_Pos)
- #define CoreDebug_DEMCR_MON_EN_Pos 16
- #define CoreDebug_DEMCR_MON_EN_Msk (1UL << CoreDebug_DEMCR_MON_EN_Pos)
- #define CoreDebug_DEMCR_VC_HARDERR_Pos 10
- #define CoreDebug_DEMCR_VC_HARDERR_Msk (1UL << CoreDebug_DEMCR_VC_HARDERR_Pos)
- #define CoreDebug_DEMCR_VC_INTERR_Pos 9
- #define CoreDebug_DEMCR_VC_INTERR_Msk (1UL << CoreDebug_DEMCR_VC_INTERR_Pos)
- #define CoreDebug_DEMCR_VC_BUSERR_Pos 8
- #define CoreDebug_DEMCR_VC_BUSERR_Msk (1UL << CoreDebug_DEMCR_VC_BUSERR_Pos)
- #define CoreDebug_DEMCR_VC_STATERR_Pos 7
- #define CoreDebug_DEMCR_VC_STATERR_Msk (1UL << CoreDebug_DEMCR_VC_STATERR_Pos)
- #define CoreDebug_DEMCR_VC_CHKERR_Pos 6
- #define CoreDebug_DEMCR_VC_CHKERR_Msk (1UL << CoreDebug_DEMCR_VC_CHKERR_Pos)
- #define CoreDebug_DEMCR_VC_NOCPERR_Pos 5
- #define CoreDebug_DEMCR_VC_NOCPERR_Msk (1UL << CoreDebug_DEMCR_VC_NOCPERR_Pos)
- #define CoreDebug_DEMCR_VC_MMERR_Pos 4
- #define CoreDebug_DEMCR_VC_MMERR_Msk (1UL << CoreDebug_DEMCR_VC_MMERR_Pos)
- #define CoreDebug_DEMCR_VC_CORERESET_Pos 0

- #define CoreDebug_DEMCR_VC_CORERESET_Msk (1UL << CoreDebug_DEMCR_VC_CORERESET_Pos)

- #define ITM_BASE (0xE0000000UL)
- #define CoreDebug_BASE (0xE000EDF0UL)
- #define SysTick_BASE (SCS_BASE + 0x0010UL)
- #define NVIC_BASE (SCS_BASE + 0x0100UL)
- #define SCB_BASE (SCS_BASE + 0x0D00UL)
- #define SCnSCB ((SCnSCB_Type *) SCS_BASE)
- #define SCB ((SCB_Type *) SCB_BASE)
- #define SysTick ((SysTick_Type *) SysTick_BASE)
- #define NVIC ((NVIC_Type *) NVIC_BASE)
- #define ITM ((ITM_Type *) ITM_BASE)
- #define CoreDebug ((CoreDebug_Type *) CoreDebug_BASE)

5.10.1 Detailed Description

Type definitions for the Cortex-M Core Debug Registers

5.10.2 Macro Definition Documentation

5.10.2.1 CoreDebug

```
#define CoreDebug ((CoreDebug_Type *) CoreDebug_BASE)
```

Core Debug configuration struct

5.10.2.2 CoreDebug_BASE

```
#define CoreDebug_BASE (0xE000EDF0UL)
```

Core Debug Base Address

5.10.2.3 CoreDebug_DCRSR_REGSEL_Msk

```
#define CoreDebug_DCRSR_REGSEL_Msk (0x1FUL << CoreDebug_DCRSR_REGSEL_Pos)
```

CoreDebug DCRSR: REGSEL Mask

5.10.2.4 CoreDebug_DCRSR_REGSEL_Pos

```
#define CoreDebug_DCRSR_REGSEL_Pos 0
```

CoreDebug DCRSR: REGSEL Position

5.10.2.5 CoreDebug_DCRSR_REGWnR_Msk

```
#define CoreDebug_DCRSR_REGWnR_Msk (1UL << CoreDebug_DCRSR_REGWnR_Pos)
```

CoreDebug DCRSR: REGWnR Mask

5.10.2.6 CoreDebug_DCRSR_REGWnR_Pos

```
#define CoreDebug_DCRSR_REGWnR_Pos 16
```

CoreDebug DCRSR: REGWnR Position

5.10.2.7 CoreDebug_DEMCR_MON_EN_Msk

```
#define CoreDebug_DEMCR_MON_EN_Msk (1UL << CoreDebug_DEMCR_MON_EN_Pos)
```

CoreDebug DEMCR: MON_EN Mask

5.10.2.8 CoreDebug_DEMCR_MON_EN_Pos

```
#define CoreDebug_DEMCR_MON_EN_Pos 16
```

CoreDebug DEMCR: MON_EN Position

5.10.2.9 CoreDebug_DEMCR_MON_PEND_Msk

```
#define CoreDebug_DEMCR_MON_PEND_Msk (1UL << CoreDebug_DEMCR_MON_PEND_Pos)
```

CoreDebug DEMCR: MON_PEND Mask

5.10.2.10 CoreDebug_DEMCR_MON_PEND_Pos

```
#define CoreDebug_DEMCR_MON_PEND_Pos 17
```

CoreDebug DEMCR: MON_PEND Position

5.10.2.11 CoreDebug_DEMCR_MON_REQ_Msk

```
#define CoreDebug_DEMCR_MON_REQ_Msk (1UL << CoreDebug_DEMCR_MON_REQ_Pos)
```

CoreDebug DEMCR: MON_REQ Mask

5.10.2.12 CoreDebug_DEMCR_MON_REQ_Pos

```
#define CoreDebug_DEMCR_MON_REQ_Pos 19
```

CoreDebug DEMCR: MON_REQ Position

5.10.2.13 CoreDebug_DEMCR_MON_STEP_Msk

```
#define CoreDebug_DEMCR_MON_STEP_Msk (1UL << CoreDebug_DEMCR_MON_STEP_Pos)
```

CoreDebug DEMCR: MON_STEP Mask

5.10.2.14 CoreDebug_DEMCR_MON_STEP_Pos

```
#define CoreDebug_DEMCR_MON_STEP_Pos 18
```

CoreDebug DEMCR: MON_STEP Position

5.10.2.15 CoreDebug_DEMCR_TRCENA_Msk

```
#define CoreDebug_DEMCR_TRCENA_Msk (1UL << CoreDebug_DEMCR_TRCENA_Pos)
```

CoreDebug DEMCR: TRCENA Mask

5.10.2.16 CoreDebug_DEMCR_TRCENA_Pos

```
#define CoreDebug_DEMCR_TRCENA_Pos 24
```

CoreDebug DEMCR: TRCENA Position

5.10.2.17 CoreDebug_DEMCR_VC_BUSERR_Msk

```
#define CoreDebug_DEMCR_VC_BUSERR_Msk (1UL << CoreDebug_DEMCR_VC_BUSERR_Pos)
```

CoreDebug DEMCR: VC_BUSERR Mask

5.10.2.18 CoreDebug_DEMCR_VC_BUSERR_Pos

```
#define CoreDebug_DEMCR_VC_BUSERR_Pos 8
```

CoreDebug DEMCR: VC_BUSERR Position

5.10.2.19 CoreDebug_DEMCR_VC_CHKERR_Msk

```
#define CoreDebug_DEMCR_VC_CHKERR_Msk (1UL << CoreDebug_DEMCR_VC_CHKERR_Pos)
```

CoreDebug DEMCR: VC_CHKERR Mask

5.10.2.20 CoreDebug_DEMCR_VC_CHKERR_Pos

```
#define CoreDebug_DEMCR_VC_CHKERR_Pos 6
```

CoreDebug DEMCR: VC_CHKERR Position

5.10.2.21 CoreDebug_DEMCR_VC_CORERESET_Msk

```
#define CoreDebug_DEMCR_VC_CORERESET_Msk (1UL << CoreDebug_DEMCR_VC_CORERESET_Pos)
```

CoreDebug DEMCR: VC_CORERESET Mask

5.10.2.22 CoreDebug_DEMCR_VC_CORERESET_Pos

```
#define CoreDebug_DEMCR_VC_CORERESET_Pos 0
```

CoreDebug DEMCR: VC_CORERESET Position

5.10.2.23 CoreDebug_DEMCR_VC_HARDERR_Msk

```
#define CoreDebug_DEMCR_VC_HARDERR_Msk (1UL << CoreDebug_DEMCR_VC_HARDERR_Pos)
```

CoreDebug DEMCR: VC_HARDERR Mask

5.10.2.24 CoreDebug_DEMCR_VC_HARDERR_Pos

```
#define CoreDebug_DEMCR_VC_HARDERR_Pos 10
```

CoreDebug DEMCR: VC_HARDERR Position

5.10.2.25 CoreDebug_DEMCR_VC_INTERR_Msk

```
#define CoreDebug_DEMCR_VC_INTERR_Msk (1UL << CoreDebug_DEMCR_VC_INTERR_Pos)
```

CoreDebug DEMCR: VC_INTERR Mask

5.10.2.26 CoreDebug_DEMCR_VC_INTERR_Pos

```
#define CoreDebug_DEMCR_VC_INTERR_Pos 9
```

CoreDebug DEMCR: VC_INTERR Position

5.10.2.27 CoreDebug_DEMCR_VC_MMERR_Msk

```
#define CoreDebug_DEMCR_VC_MMERR_Msk (1UL << CoreDebug_DEMCR_VC_MMERR_Pos)
```

CoreDebug DEMCR: VC_MMERR Mask

5.10.2.28 CoreDebug_DEMCR_VC_MMERR_Pos

```
#define CoreDebug_DEMCR_VC_MMERR_Pos 4
```

CoreDebug DEMCR: VC_MMERR Position

5.10.2.29 CoreDebug_DEMCR_VC_NOCPERR_Msk

```
#define CoreDebug_DEMCR_VC_NOCPERR_Msk (1UL << CoreDebug_DEMCR_VC_NOCPERR_Pos)
```

CoreDebug DEMCR: VC_NOCPERR Mask

5.10.2.30 CoreDebug_DEMCR_VC_NOCPERR_Pos

```
#define CoreDebug_DEMCR_VC_NOCPERR_Pos 5
```

CoreDebug DEMCR: VC_NOCPERR Position

5.10.2.31 CoreDebug_DEMCR_VC_STATERR_Msk

```
#define CoreDebug_DEMCR_VC_STATERR_Msk (1UL << CoreDebug_DEMCR_VC_STATERR_Pos)
```

CoreDebug DEMCR: VC_STATERR Mask

5.10.2.32 CoreDebug_DEMCR_VC_STATERR_Pos

```
#define CoreDebug_DEMCR_VC_STATERR_Pos 7
```

CoreDebug DEMCR: VC_STATERR Position

5.10.2.33 CoreDebug_DHCSR_C_DEBUGEN_Msk

```
#define CoreDebug_DHCSR_C_DEBUGEN_Msk (1UL << CoreDebug_DHCSR_C_DEBUGEN_Pos)
```

CoreDebug DHCSR: C_DEBUGEN Mask

5.10.2.34 CoreDebug_DHCSR_C_DEBUGEN_Pos

```
#define CoreDebug_DHCSR_C_DEBUGEN_Pos 0
```

CoreDebug DHCSR: C_DEBUGEN Position

5.10.2.35 CoreDebug_DHCSR_C_HALT_Msk

```
#define CoreDebug_DHCSR_C_HALT_Msk (1UL << CoreDebug_DHCSR_C_HALT_Pos)
```

CoreDebug DHCSR: C_HALT Mask

5.10.2.36 CoreDebug_DHCSR_C_HALT_Pos

```
#define CoreDebug_DHCSR_C_HALT_Pos 1
```

CoreDebug DHCSR: C_HALT Position

5.10.2.37 CoreDebug_DHCSR_C_MASKINTS_Msk

```
#define CoreDebug_DHCSR_C_MASKINTS_Msk (1UL << CoreDebug_DHCSR_C_MASKINTS_Pos)
```

CoreDebug DHCSR: C_MASKINTS Mask

5.10.2.38 CoreDebug_DHCSR_C_MASKINTS_Pos

```
#define CoreDebug_DHCSR_C_MASKINTS_Pos 3
```

CoreDebug DHCSR: C_MASKINTS Position

5.10.2.39 CoreDebug_DHCSR_C_SNAPSTALL_Msk

```
#define CoreDebug_DHCSR_C_SNAPSTALL_Msk (1UL << CoreDebug_DHCSR_C_SNAPSTALL_Pos)
```

CoreDebug DHCSR: C_SNAPSTALL Mask

5.10.2.40 CoreDebug_DHCSR_C_SNAPSTALL_Pos

```
#define CoreDebug_DHCSR_C_SNAPSTALL_Pos 5
```

CoreDebug DHCSR: C_SNAPSTALL Position

5.10.2.41 CoreDebug_DHCSR_C_STEP_Msk

```
#define CoreDebug_DHCSR_C_STEP_Msk (1UL << CoreDebug_DHCSR_C_STEP_Pos)
```

CoreDebug DHCSR: C_STEP Mask

5.10.2.42 CoreDebug_DHCSR_C_STEP_Pos

```
#define CoreDebug_DHCSR_C_STEP_Pos 2
```

CoreDebug DHCSR: C_STEP Position

5.10.2.43 CoreDebug_DHCSR_DBGKEY_Msk

```
#define CoreDebug_DHCSR_DBGKEY_Msk (0xFFFFUL << CoreDebug_DHCSR_DBGKEY_Pos)
```

CoreDebug DHCSR: DBGKEY Mask

5.10.2.44 CoreDebug_DHCSR_DBGKEY_Pos

```
#define CoreDebug_DHCSR_DBGKEY_Pos 16
```

CoreDebug DHCSR: DBGKEY Position

5.10.2.45 CoreDebug_DHCSR_S_HALT_Msk

```
#define CoreDebug_DHCSR_S_HALT_Msk (1UL << CoreDebug_DHCSR_S_HALT_Pos)
```

CoreDebug DHCSR: S_HALT Mask

5.10.2.46 CoreDebug_DHCSR_S_HALT_Pos

```
#define CoreDebug_DHCSR_S_HALT_Pos 17
```

CoreDebug DHCSR: S_HALT Position

5.10.2.47 CoreDebug_DHCSR_S_LOCKUP_Msk

```
#define CoreDebug_DHCSR_S_LOCKUP_Msk (1UL << CoreDebug_DHCSR_S_LOCKUP_Pos)
```

CoreDebug DHCSR: S_LOCKUP Mask

5.10.2.48 CoreDebug_DHCSR_S_LOCKUP_Pos

```
#define CoreDebug_DHCSR_S_LOCKUP_Pos 19
```

CoreDebug DHCSR: S_LOCKUP Position

5.10.2.49 CoreDebug_DHCSR_S_REGRDY_Msk

```
#define CoreDebug_DHCSR_S_REGRDY_Msk (1UL << CoreDebug_DHCSR_S_REGRDY_Pos)
```

CoreDebug DHCSR: S_REGRDY Mask

5.10.2.50 CoreDebug_DHCSR_S_REGRDY_Pos

```
#define CoreDebug_DHCSR_S_REGRDY_Pos 16
```

CoreDebug DHCSR: S_REGRDY Position

5.10.2.51 CoreDebug_DHCSR_S_RESET_ST_Msk

```
#define CoreDebug_DHCSR_S_RESET_ST_Msk (1UL << CoreDebug_DHCSR_S_RESET_ST_Pos)
```

CoreDebug DHCSR: S_RESET_ST Mask

5.10.2.52 CoreDebug_DHCSR_S_RESET_ST_Pos

```
#define CoreDebug_DHCSR_S_RESET_ST_Pos 25
```

CoreDebug DHCSR: S_RESET_ST Position

5.10.2.53 CoreDebug_DHCSR_S_RETIRE_ST_Msk

```
#define CoreDebug_DHCSR_S_RETIRE_ST_Msk (1UL << CoreDebug_DHCSR_S_RETIRE_ST_Pos)
```

CoreDebug DHCSR: S_RETIRE_ST Mask

5.10.2.54 CoreDebug_DHCSR_S_RETIRE_ST_Pos

```
#define CoreDebug_DHCSR_S_RETIRE_ST_Pos 24
```

CoreDebug DHCSR: S_RETIRE_ST Position

5.10.2.55 CoreDebug_DHCSR_S_SLEEP_Msk

```
#define CoreDebug_DHCSR_S_SLEEP_Msk (1UL << CoreDebug_DHCSR_S_SLEEP_Pos)
```

CoreDebug DHCSR: S_SLEEP Mask

5.10.2.56 CoreDebug_DHCSR_S_SLEEP_Pos

```
#define CoreDebug_DHCSR_S_SLEEP_Pos 18
```

CoreDebug DHCSR: S_SLEEP Position

5.10.2.57 ITM

```
#define ITM ((ITM_Type *) ITM_BASE )
```

ITM configuration struct

5.10.2.58 ITM_BASE

```
#define ITM_BASE (0xE0000000UL)
```

ITM Base Address

5.10.2.59 NVIC

```
#define NVIC ((NVIC_Type *) NVIC_BASE )
```

NVIC configuration struct

5.10.2.60 NVIC_BASE

```
#define NVIC_BASE (SCS_BASE + 0x0100UL)
```

NVIC Base Address

5.10.2.61 SCB

```
#define SCB ((SCB_Type *) SCB_BASE )
```

SCB configuration struct

5.10.2.62 SCB_BASE

```
#define SCB_BASE (SCS_BASE + 0x0D00UL)
```

System Control Block Base Address

5.10.2.63 SCnSCB

```
#define SCnSCB ((SCnSCB_Type *) SCS_BASE )
```

System control Register not in SCB

5.10.2.64 SysTick

```
#define SysTick ((SysTick_Type *) SysTick_BASE )
```

SysTick configuration struct

5.10.2.65 SysTick_BASE

```
#define SysTick_BASE (SCS_BASE + 0x0010UL)
```

SysTick Base Address

5.11 CMSIS Core Function Interface

Modules

- CMSIS Core NVIC Functions
- CMSIS Core Register Access Functions

5.11.1 Detailed Description

Core Function Interface contains:

- Core NVIC Functions
- Core SysTick Functions
- Core Debug Functions
- Core Register Access Functions

5.12 CMSIS Core NVIC Functions

Modules

- CMSIS Core SysTick Functions

5.12.1 Detailed Description

5.13 CMSIS Core SysTick Functions

Modules

- CMSIS Core Debug Functions

5.13.1 Detailed Description

5.14 CMSIS Core Debug Functions

Variables

- uint32_t _reserved0:16
- uint32_t GE:4
- uint32_t _reserved1:7
- uint32_t Q:1
- uint32_t V:1
- uint32_t C:1
- uint32_t Z:1
- uint32_t N:1
- struct {
  uint32_t _reserved0:16
  uint32_t GE:4
  uint32_t _reserved1:7
  uint32_t Q:1
  uint32_t V:1
  uint32_t C:1
  uint32_t Z:1
  uint32_t N:1
  } b

- uint32_t w
- uint32_t ISR:9
- uint32_t _reserved0:23
- struct {
  uint32_t ISR:9
  uint32_t _reserved0:23
  } b

- uint32_t w
- uint32_t ISR:9
- uint32_t _reserved0:7
- uint32_t GE:4
- uint32_t _reserved1:4
- uint32_t T:1
- uint32_t IT:2
- uint32_t Q:1
- uint32_t V:1
- uint32_t C:1
- uint32_t Z:1
- uint32_t N:1
- struct {
  uint32_t ISR:9
  uint32_t _reserved0:7
  uint32_t GE:4
  uint32_t _reserved1:4
  uint32_t T:1
  uint32_t IT:2
  uint32_t Q:1
  uint32_t V:1
  uint32_t C:1
  uint32_t Z:1
  uint32_t N:1
  } b

- uint32_t w
- uint32_t nPRIV:1
- uint32_t SPSEL:1
- uint32_t FPCA:1
- uint32_t _reserved0:29
- struct {
  uint32_t nPRIV:1
  uint32_t SPSEL:1
  uint32_t FPCA:1
  uint32_t _reserved0:29
  } b

- uint32_t w
- __IO uint32_t ISER [8]
- uint32_t **RESERVED0** [24]
- __IO uint32_t ICER [8]
- uint32_t **RSERVED1** [24]
- __IO uint32_t ISPR [8]
- uint32_t **RESERVED2** [24]
- __IO uint32_t ICPR [8]
- uint32_t **RESERVED3** [24]
- __IO uint32_t IABR [8]

- uint32_t **RESERVED4** [56]
- __IO uint8_t IP [240]
- uint32_t **RESERVED5** [644]
- __O uint32_t STIR
- __I uint32_t CPUID
- __IO uint32_t ICSR
- __IO uint32_t VTOR
- __IO uint32_t AIRCR
- __IO uint32_t SCR
- __IO uint32_t CCR
- __IO uint8_t SHP [12]
- __IO uint32_t SHCSR
- __IO uint32_t CFSR
- __IO uint32_t HFSR
- __IO uint32_t DFSR
- __IO uint32_t MMFAR
- __IO uint32_t BFAR
- __IO uint32_t AFSR
- __I uint32_t PFR [2]
- __I uint32_t DFR
- __I uint32_t ADR
- __I uint32_t MMFR [4]
- __I uint32_t ISAR [5]
- uint32_t **RESERVED0** [5]
- __IO uint32_t CPACR
- uint32_t **RESERVED0** [1]
- __I uint32_t ICTR
- __IO uint32_t ACTLR
- __IO uint32_t CTRL
- __IO uint32_t LOAD
- __IO uint32_t VAL
- __I uint32_t CALIB
- __O uint8_t u8
- __O uint16_t u16
- __O uint32_t u32
- union {
  __O uint8_t u8
  __O uint16_t u16
  __O uint32_t u32
  } PORT [32]

- uint32_t **RESERVED0** [864]
- __IO uint32_t TER
- uint32_t **RESERVED1** [15]
- __IO uint32_t TPR
- uint32_t **RESERVED2** [15]
- __IO uint32_t TCR
- __IO uint32_t DHCSR
- __O uint32_t DCRSR
- __IO uint32_t DCRDR
- __IO uint32_t DEMCR

- volatile int32_t ITM_RxBuffer
- #define ITM_RXBUFFER_EMPTY 0x5AA55AA5

5.14.1 Detailed Description

5.14.2 Macro Definition Documentation

5.14.2.1 ITM_RXBUFFER_EMPTY

```
#define ITM_RXBUFFER_EMPTY 0x5AA55AA5
```

value identifying ITM_RxBuffer is ready for next character

5.14.3 Variable Documentation

5.14.3.1 _reserved0 [1/8]

```
uint32_t _reserved0
```

bit: 0..15 Reserved

5.14.3.2 _reserved0 [2/8]

```
uint32_t _reserved0
```

bit: 0..15 Reserved

5.14.3.3 _reserved0 [3/8]

```
uint32_t _reserved0
```

bit: 9..31 Reserved

5.14.3.4 _reserved0 [4/8]

```
uint32_t _reserved0
```

bit: 9..31 Reserved

5.14.3.5 _reserved0 [5/8]

```
uint32_t _reserved0
```

bit: 9..15 Reserved

5.14.3.6 _reserved0 [6/8]

```
uint32_t _reserved0
```

bit: 9..15 Reserved

5.14.3.7 _reserved0 [7/8]

```
uint32_t _reserved0
```

bit: 3..31 Reserved

5.14.3.8 _reserved0 [8/8]

```
uint32_t _reserved0
```

bit: 3..31 Reserved

5.14.3.9 _reserved1 [1/4]

```
uint32_t _reserved1
```

bit: 20..26 Reserved

5.14.3.10 _reserved1 [2/4]

```
uint32_t _reserved1
```

bit: 20..26 Reserved

5.14.3.11 _reserved1 [3/4]

```
uint32_t _reserved1
```

bit: 20..23 Reserved

5.14.3.12 _reserved1 [4/4]

```
uint32_t _reserved1
```

bit: 20..23 Reserved

5.14.3.13 ACTLR

```
__IO uint32_t ACTLR
```

Offset: 0x008 (R/W) Auxiliary Control Register

5.14.3.14 ADR

```
__I uint32_t ADR
```

Offset: 0x04C (R/) Auxiliary Feature Register

5.14.3.15 AFSR

```
__IO uint32_t AFSR
```

Offset: 0x03C (R/W) Auxiliary Fault Status Register

5.14.3.16 AIRCR

```
__IO uint32_t AIRCR
```

Offset: 0x00C (R/W) Application Interrupt and Reset Control Register

5.14.3.17 [1/4]

```
struct { ... } b
```

Structure used for bit access

5.14.3.18 [2/4]

```
struct { ... } b
```

Structure used for bit access

5.14.3.19 [3/4]

```
struct { ...  } b
```

Structure used for bit access

5.14.3.20 [4/4]

```
struct { ...  } b
```

Structure used for bit access

5.14.3.21 BFAR

```
__IO uint32_t BFAR
```

Offset: 0x038 (R/W) BusFault Address Register

5.14.3.22 C [1/4]

```
uint32_t C
```

bit: 29 Carry condition code flag

5.14.3.23 C [2/4]

```
uint32_t C
```

bit: 29 Carry condition code flag

5.14.3.24 C [3/4]

```
uint32_t C
```

bit: 29 Carry condition code flag

5.14.3.25 C [4/4]

```
uint32_t C
```

bit: 29 Carry condition code flag

5.14.3.26 CALIB

```
__I uint32_t CALIB
```

Offset: 0x00C (R/) SysTick Calibration Register

5.14.3.27 CCR

```
__IO uint32_t CCR
```

Offset: 0x014 (R/W) Configuration Control Register

5.14.3.28 CFSR

```
__IO uint32_t CFSR
```

Offset: 0x028 (R/W) Configurable Fault Status Register

5.14.3.29 CPACR

```
__IO uint32_t CPACR
```

Offset: 0x088 (R/W) Coprocessor Access Control Register

5.14.3.30 CPUID

```
__I uint32_t CPUID
```

Offset: 0x000 (R/) CPUID Base Register

5.14.3.31 CTRL

```
__IO uint32_t CTRL
```

Offset: 0x000 (R/W) SysTick Control and Status Register

5.14.3.32 DCRDR

```
__IO uint32_t DCRDR
```

Offset: 0x008 (R/W) Debug Core Register Data Register

5.14.3.33 DCRSR

```
__O uint32_t DCRSR
```

Offset: 0x004 (/W) Debug Core Register Selector Register

5.14.3.34 DEMCR

```
__IO uint32_t DEMCR
```

Offset: 0x00C (R/W) Debug Exception and Monitor Control Register

5.14.3.35 DFR

```
__I uint32_t DFR
```

Offset: 0x048 (R/) Debug Feature Register

5.14.3.36 DFSR

```
__IO uint32_t DFSR
```

Offset: 0x030 (R/W) Debug Fault Status Register

5.14.3.37 DHCSR

```
__IO uint32_t DHCSR
```

Offset: 0x000 (R/W) Debug Halting Control and Status Register

5.14.3.38 FPCA [1/2]

```
uint32_t FPCA
```

bit: 2 FP extension active flag

5.14.3.39 FPCA [2/2]

```
uint32_t FPCA
```

bit: 2 FP extension active flag

5.14.3.40 GE [1/4]

```
uint32_t GE
```

bit: 16..19 Greater than or Equal flags

5.14.3.41 GE [2/4]

```
uint32_t GE
```

bit: 16..19 Greater than or Equal flags

5.14.3.42 GE [3/4]

```
uint32_t GE
```

bit: 16..19 Greater than or Equal flags

5.14.3.43 GE [4/4]

```
uint32_t GE
```

bit: 16..19 Greater than or Equal flags

5.14.3.44 HFSR

```
__IO uint32_t HFSR
```

Offset: 0x02C (R/W) HardFault Status Register

5.14.3.45 IABR

```
__IO uint32_t IABR[8]
```

Offset: 0x200 (R/W) Interrupt Active bit Register

5.14.3.46 ICER

```
__IO uint32_t ICER[8]
```

Offset: 0x080 (R/W) Interrupt Clear Enable Register

5.14.3.47 ICPR

```
__IO uint32_t ICPR[8]
```

Offset: 0x180 (R/W) Interrupt Clear Pending Register

5.14.3.48 ICSR

```
__IO uint32_t ICSR
```

Offset: 0x004 (R/W) Interrupt Control and State Register

5.14.3.49 ICTR

```
__I uint32_t ICTR
```

Offset: 0x004 (R/) Interrupt Controller Type Register

5.14.3.50 IP

```
__IO uint8_t IP[240]
```

Offset: 0x300 (R/W) Interrupt Priority Register (8Bit wide)

5.14.3.51 ISAR

```
__I uint32_t ISAR[5]
```

Offset: 0x060 (R/) Instruction Set Attributes Register

5.14.3.52 ISER

```
__IO uint32_t ISER[8]
```

Offset: 0x000 (R/W) Interrupt Set Enable Register

5.14.3.53 ISPR

```
__IO uint32_t ISPR[8]
```

Offset: 0x100 (R/W) Interrupt Set Pending Register

5.14.3.54 ISR [1/4]

```
uint32_t ISR
```

bit: 0.. 8 Exception number

5.14.3.55 ISR [2/4]

```
uint32_t ISR
```

bit: 0.. 8 Exception number

5.14.3.56 ISR [3/4]

```
uint32_t ISR
```

bit: 0.. 8 Exception number

5.14.3.57 ISR [4/4]

```
uint32_t ISR
```

bit: 0.. 8 Exception number

5.14.3.58 IT [1/2]

```
uint32_t IT
```

bit: 25..26 saved IT state (read 0)

5.14.3.59 IT [2/2]

```
uint32_t IT
```

bit: 25..26 saved IT state (read 0)

5.14.3.60 ITM_RxBuffer

```
volatile int32_t ITM_RxBuffer  [extern]
```

external variable to receive characters

5.14.3.61 LOAD

```
__IO uint32_t LOAD
```

Offset: 0x004 (R/W) SysTick Reload Value Register

5.14.3.62 MMFAR

```
__IO uint32_t MMFAR
```

Offset: 0x034 (R/W) MemManage Fault Address Register

5.14.3.63 MMFR

```
__I uint32_t MMFR[4]
```

Offset: 0x050 (R/) Memory Model Feature Register

5.14.3.64 N [1/4]

```
uint32_t N
```

bit: 31 Negative condition code flag

5.14.3.65 N [2/4]

```
uint32_t N
```

bit: 31 Negative condition code flag

5.14.3.66 N [3/4]

```
uint32_t N
```

bit: 31 Negative condition code flag

5.14.3.67 N [4/4]

```
uint32_t N
```

bit: 31 Negative condition code flag

5.14.3.68 nPRIV [1/2]

```
uint32_t nPRIV
```

bit: 0 Execution privilege in Thread mode

5.14.3.69 nPRIV [2/2]

```
uint32_t nPRIV
```

bit: 0 Execution privilege in Thread mode

5.14.3.70 PFR

```
__I uint32_t PFR[2]
```

Offset: 0x040 (R/) Processor Feature Register

5.14.3.71

```
__O union { ...  } PORT[32]
```

Offset: 0x000 (/W) ITM Stimulus Port Registers

5.14.3.72 Q [1/4]

```
uint32_t Q
```

bit: 27 Saturation condition flag

5.14.3.73 Q [2/4]

```
uint32_t Q
```

bit: 27 Saturation condition flag

5.14.3.74 Q [3/4]

```
uint32_t Q
```

bit: 27 Saturation condition flag

5.14.3.75 Q [4/4]

```
uint32_t Q
```

bit: 27 Saturation condition flag

5.14.3.76 SCR

```
__IO uint32_t SCR
```

Offset: 0x010 (R/W) System Control Register

5.14.3.77 SHCSR

```
__IO uint32_t SHCSR
```

Offset: 0x024 (R/W) System Handler Control and State Register

5.14.3.78 SHP

```
__IO uint8_t SHP[12]
```

Offset: 0x018 (R/W) System Handlers Priority Registers (4-7, 8-11, 12-15)

5.14.3.79 SPSEL [1/2]

```
uint32_t SPSEL
```

bit: 1 Stack to be used

5.14.3.80 SPSEL [2/2]

```
uint32_t SPSEL
```

bit: 1 Stack to be used

5.14.3.81 STIR

```
__O uint32_t STIR
```

Offset: 0xE00 (/W) Software Trigger Interrupt Register

5.14.3.82 T [1/2]

```
uint32_t T
```

bit: 24 Thumb bit (read 0)

5.14.3.83 T [2/2]

```
uint32_t T
```

bit: 24 Thumb bit (read 0)

5.14.3.84 TCR

```
__IO uint32_t TCR
```

Offset: 0xE80 (R/W) ITM Trace Control Register

5.14.3.85 TER

```
__IO uint32_t TER
```

Offset: 0xE00 (R/W) ITM Trace Enable Register

5.14.3.86 TPR

```
__IO uint32_t TPR
```

Offset: 0xE40 (R/W) ITM Trace Privilege Register

5.14.3.87 u16 [1/2]

```
__O uint16_t u16
```

Offset: 0x000 (/W) ITM Stimulus Port 16-bit

5.14.3.88 u16 [2/2]

```
__O uint16_t u16
```

Offset: 0x000 (/W) ITM Stimulus Port 16-bit

5.14.3.89 u32 [1/2]

```
__O uint32_t u32
```

Offset: 0x000 (/W) ITM Stimulus Port 32-bit

5.14.3.90 u32 [2/2]

```
__O uint32_t u32
```

Offset: 0x000 (/W) ITM Stimulus Port 32-bit

5.14.3.91 u8 [1/2]

```
__O uint8_t u8
```

Offset: 0x000 (/W) ITM Stimulus Port 8-bit

5.14.3.92 u8 [2/2]

```
__O uint8_t u8
```

Offset: 0x000 (/W) ITM Stimulus Port 8-bit

5.14.3.93 V [1/4]

```
uint32_t V
```

bit: 28 Overflow condition code flag

5.14.3.94 V [2/4]

```
uint32_t V
```

bit: 28 Overflow condition code flag

5.14.3.95 V [3/4]

```
uint32_t V
```

bit: 28 Overflow condition code flag

5.14.3.96 V [4/4]

```
uint32_t V
```

bit: 28 Overflow condition code flag

5.14.3.97 VAL

```
__IO uint32_t VAL
```

Offset: 0x008 (R/W) SysTick Current Value Register

5.14.3.98 VTOR

```
__IO uint32_t VTOR
```

Offset: 0x008 (R/W) Vector Table Offset Register

5.14.3.99 w [1/4]

```
uint32_t w
```

Type used for word access

5.14.3.100 w [2/4]

```
uint32_t w
```

Type used for word access

5.14.3.101 w [3/4]

```
uint32_t w
```

Type used for word access

5.14.3.102 w [4/4]

```
uint32_t w
```

Type used for word access

5.14.3.103 Z [1/4]

```
uint32_t Z
```

bit: 30 Zero condition code flag

5.14.3.104 Z [2/4]

```
uint32_t Z
```

bit: 30 Zero condition code flag

5.14.3.105 Z [3/4]

```
uint32_t Z
```

bit: 30 Zero condition code flag

5.14.3.106 Z [4/4]

```
uint32_t Z
```

bit: 30 Zero condition code flag

5.15 CMSIS SIMD Intrinsics

Access to dedicated SIMD instructions

5.16 CMSIS Core Register Access Functions

<>

5.17 CMSIS Core Instruction Interface

Access to dedicated instructions

5.18 MISC_Exported_Constants

Modules

- MISC_Vector_Table_Base
- MISC_System_Low_Power
- MISC_Preemption_Priority_Group
- MISC_SysTick_clock_source

5.18.1 Detailed Description

5.19 MISC_Vector_Table_Base

Macros

- #define **NVIC_VectTab_RAM** ((uint32_t)0x20000000)
- #define **NVIC_VectTab_FLASH** ((uint32_t)0x08000000)
- #define IS_NVIC_VECTTAB(VECTTAB)

5.19.1 Detailed Description

5.19.2 Macro Definition Documentation

5.19.2.1 IS_NVIC_VECTTAB

```
#define IS_NVIC_VECTTAB(
              VECTTAB )
```

Value:

```
(((VECTTAB) == NVIC_VectTab_RAM) || \
((VECTTAB) == NVIC_VectTab_FLASH))
```

5.20 MISC_System_Low_Power

Macros

- #define **NVIC_LP_SEVONPEND** ((uint8_t)0x10)
- #define **NVIC_LP_SLEEPDEEP** ((uint8_t)0x04)
- #define **NVIC_LP_SLEEPONEXIT** ((uint8_t)0x02)
- #define IS_NVIC_LP(LP)

5.20.1 Detailed Description

5.20.2 Macro Definition Documentation

5.20.2.1 IS_NVIC_LP

```
#define IS_NVIC_LP(
              LP )
```

Value:

```
(((LP) == NVIC_LP_SEVONPEND) || \
((LP) == NVIC_LP_SLEEPDEEP) || \
((LP) == NVIC_LP_SLEEPONEXIT))
```

5.21 MISC_Preemption_Priority_Group

Macros

- #define NVIC_PriorityGroup_0 ((uint32_t)0x700)
- #define NVIC_PriorityGroup_1 ((uint32_t)0x600)
- #define NVIC_PriorityGroup_2 ((uint32_t)0x500)
- #define NVIC_PriorityGroup_3 ((uint32_t)0x400)
- #define NVIC_PriorityGroup_4 ((uint32_t)0x300)
- #define IS_NVIC_PRIORITY_GROUP(GROUP)
- #define **IS_NVIC_PREEMPTION_PRIORITY**(PRIORITY) ((PRIORITY) < 0x10)
- #define **IS_NVIC_SUB_PRIORITY**(PRIORITY) ((PRIORITY) < 0x10)
- #define **IS_NVIC_OFFSET**(OFFSET) ((OFFSET) < 0x000FFFFF)

5.21.1 Detailed Description

5.21.2 Macro Definition Documentation

5.21.2.1 IS_NVIC_PRIORITY_GROUP

```
#define IS_NVIC_PRIORITY_GROUP(
              GROUP )
```

Value:

```
(((GROUP) == NVIC_PriorityGroup_0) || \
((GROUP) == NVIC_PriorityGroup_1) || \
((GROUP) == NVIC_PriorityGroup_2) || \
((GROUP) == NVIC_PriorityGroup_3) || \
((GROUP) == NVIC_PriorityGroup_4))
```

5.21.2.2 NVIC_PriorityGroup_0

```
#define NVIC_PriorityGroup_0 ((uint32_t)0x700)
```

0 bits for pre-emption priority 4 bits for subpriority

5.21.2.3 NVIC_PriorityGroup_1

```
#define NVIC_PriorityGroup_1 ((uint32_t)0x600)
```

1 bits for pre-emption priority 3 bits for subpriority

5.21.2.4 NVIC_PriorityGroup_2

```
#define NVIC_PriorityGroup_2 ((uint32_t)0x500)
```

2 bits for pre-emption priority 2 bits for subpriority

5.21.2.5 NVIC_PriorityGroup_3

```
#define NVIC_PriorityGroup_3 ((uint32_t)0x400)
```

3 bits for pre-emption priority 1 bits for subpriority

5.21.2.6 NVIC_PriorityGroup_4

```
#define NVIC_PriorityGroup_4 ((uint32_t)0x300)
```

4 bits for pre-emption priority 0 bits for subpriority

5.22 MISC_SysTick_clock_source

Macros

- #define **SysTick_CLKSource_HCLK_Div8** ((uint32_t)0xFFFFFFFB)
- #define **SysTick_CLKSource_HCLK** ((uint32_t)0x00000004)
- #define IS_SYSTICK_CLK_SOURCE(SOURCE)

5.22.1 Detailed Description

5.22.2 Macro Definition Documentation

5.22.2.1 IS_SYSTICK_CLK_SOURCE

```
#define IS_SYSTICK_CLK_SOURCE(
              SOURCE )
```

Value:

```
(((SOURCE) == SysTick_CLKSource_HCLK) || \
((SOURCE) == SysTick_CLKSource_HCLK_Div8))
```

5.23 DMA_Exported_Constants

Modules

- DMA_channel
- DMA_data_transfer_direction
- DMA_data_buffer_size
- DMA_peripheral_incremented_mode
- DMA_memory_incremented_mode
- DMA_peripheral_data_size
- DMA_memory_data_size
- DMA_circular_normal_mode
- DMA_priority_level
- DMA_fifo_direct_mode
- DMA_fifo_threshold_level
- DMA_memory_burst
- DMA_peripheral_burst
- DMA_fifo_status_level
- DMA_flags_definition
- DMA_interrupt_enable_definitions
- DMA_interrupts_definitions
- DMA_peripheral_increment_offset
- DMA_flow_controller_definitions
- DMA_memory_targets_definitions

Macros

- #define IS_DMA_ALL_PERIPH(PERIPH)
- #define IS_DMA_ALL_CONTROLLER(CONTROLLER)

5.23.1 Detailed Description

5.23.2 Macro Definition Documentation

5.23.2.1 IS_DMA_ALL_CONTROLLER

```
#define IS_DMA_ALL_CONTROLLER(
              CONTROLLER )
```

Value:

```
(((CONTROLLER) == DMA1) || \
((CONTROLLER) == DMA2))
```

5.23.2.2 IS_DMA_ALL_PERIPH

```
#define IS_DMA_ALL_PERIPH(
              PERIPH )
```

Value:

```
(((PERIPH) == DMA1_Stream0) || \
((PERIPH) == DMA1_Stream1) || \
((PERIPH) == DMA1_Stream2) || \
((PERIPH) == DMA1_Stream3) || \
((PERIPH) == DMA1_Stream4) || \
((PERIPH) == DMA1_Stream5) || \
((PERIPH) == DMA1_Stream6) || \
((PERIPH) == DMA1_Stream7) || \
((PERIPH) == DMA2_Stream0) || \
((PERIPH) == DMA2_Stream1) || \
((PERIPH) == DMA2_Stream2) || \
((PERIPH) == DMA2_Stream3) || \
((PERIPH) == DMA2_Stream4) || \
((PERIPH) == DMA2_Stream5) || \
((PERIPH) == DMA2_Stream6) || \
((PERIPH) == DMA2_Stream7))
```

5.24 DMA_channel

Macros

- #define **DMA_Channel_0** ((uint32_t)0x00000000)
- #define **DMA_Channel_1** ((uint32_t)0x02000000)
- #define **DMA_Channel_2** ((uint32_t)0x04000000)
- #define **DMA_Channel_3** ((uint32_t)0x06000000)
- #define **DMA_Channel_4** ((uint32_t)0x08000000)
- #define **DMA_Channel_5** ((uint32_t)0x0A000000)
- #define **DMA_Channel_6** ((uint32_t)0x0C000000)
- #define **DMA_Channel_7** ((uint32_t)0x0E000000)
- #define IS_DMA_CHANNEL(CHANNEL)

5.24.1 Detailed Description

5.24.2 Macro Definition Documentation

5.24.2.1 IS_DMA_CHANNEL

```
#define IS_DMA_CHANNEL(
              CHANNEL )
```

Value:

```
(((CHANNEL) == DMA_Channel_0) || \
((CHANNEL) == DMA_Channel_1) || \
((CHANNEL) == DMA_Channel_2) || \
((CHANNEL) == DMA_Channel_3) || \
((CHANNEL) == DMA_Channel_4) || \
((CHANNEL) == DMA_Channel_5) || \
((CHANNEL) == DMA_Channel_6) || \
((CHANNEL) == DMA_Channel_7))
```

5.25 DMA_data_transfer_direction

Macros

- #define **DMA_DIR_PeripheralToMemory** ((uint32_t)0x00000000)
- #define **DMA_DIR_MemoryToPeripheral** ((uint32_t)0x00000040)
- #define **DMA_DIR_MemoryToMemory** ((uint32_t)0x00000080)
- #define IS_DMA_DIRECTION(DIRECTION)

5.25.1 Detailed Description

5.25.2 Macro Definition Documentation

5.25.2.1 IS_DMA_DIRECTION

```
#define IS_DMA_DIRECTION(
              DIRECTION )
```

Value:

```
(((DIRECTION) == DMA_DIR_PeripheralToMemory ) || \
((DIRECTION) == DMA_DIR_MemoryToPeripheral)  || \
((DIRECTION) == DMA_DIR_MemoryToMemory))
```

5.26 DMA_data_buffer_size

Macros

- #define **IS_DMA_BUFFER_SIZE**(SIZE) (((SIZE) >= 0x1) && ((SIZE) < 0x10000))

5.26.1 Detailed Description

5.27 DMA_peripheral_incremented_mode

Macros

- #define **DMA_PeripheralInc_Enable** ((uint32_t)0x00000200)
- #define **DMA_PeripheralInc_Disable** ((uint32_t)0x00000000)
- #define IS_DMA_PERIPHERAL_INC_STATE(STATE)

5.27.1 Detailed Description

5.27.2 Macro Definition Documentation

5.27.2.1 IS_DMA_PERIPHERAL_INC_STATE

```
#define IS_DMA_PERIPHERAL_INC_STATE(
              STATE )
```

Value:

```
(((STATE) == DMA_PeripheralInc_Enable) || \
((STATE) == DMA_PeripheralInc_Disable))
```

5.28 DMA_memory_incremented_mode

Macros

- #define **DMA_MemoryInc_Enable** ((uint32_t)0x00000400)
- #define **DMA_MemoryInc_Disable** ((uint32_t)0x00000000)
- #define IS_DMA_MEMORY_INC_STATE(STATE)

5.28.1 Detailed Description

5.28.2 Macro Definition Documentation

5.28.2.1 IS_DMA_MEMORY_INC_STATE

```
#define IS_DMA_MEMORY_INC_STATE(
              STATE )
```

Value:

```
(((STATE) == DMA_MemoryInc_Enable) || \
((STATE) == DMA_MemoryInc_Disable))
```

5.29 DMA_peripheral_data_size

Macros

- #define **DMA_PeripheralDataSize_Byte** ((uint32_t)0x00000000)
- #define **DMA_PeripheralDataSize_HalfWord** ((uint32_t)0x00000800)
- #define **DMA_PeripheralDataSize_Word** ((uint32_t)0x00001000)
- #define IS_DMA_PERIPHERAL_DATA_SIZE(SIZE)

5.29.1 Detailed Description

5.29.2 Macro Definition Documentation

5.29.2.1 IS_DMA_PERIPHERAL_DATA_SIZE

```
#define IS_DMA_PERIPHERAL_DATA_SIZE(
              SIZE )
```

Value:

```
(((SIZE) == DMA_PeripheralDataSize_Byte)  || \
((SIZE) == DMA_PeripheralDataSize_HalfWord) || \
((SIZE) == DMA_PeripheralDataSize_Word))
```

5.30 DMA_memory_data_size

Macros

- #define **DMA_MemoryDataSize_Byte** ((uint32_t)0x00000000)
- #define **DMA_MemoryDataSize_HalfWord** ((uint32_t)0x00002000)
- #define **DMA_MemoryDataSize_Word** ((uint32_t)0x00004000)
- #define IS_DMA_MEMORY_DATA_SIZE(SIZE)

5.30.1 Detailed Description

5.30.2 Macro Definition Documentation

5.30.2.1 IS_DMA_MEMORY_DATA_SIZE

```
#define IS_DMA_MEMORY_DATA_SIZE(
              SIZE )
```

Value:

```
(((SIZE) == DMA_MemoryDataSize_Byte)  || \
((SIZE) == DMA_MemoryDataSize_HalfWord) || \
((SIZE) == DMA_MemoryDataSize_Word ))
```

5.31 DMA_circular_normal_mode

Macros

- #define **DMA_Mode_Normal** ((uint32_t)0x00000000)
- #define **DMA_Mode_Circular** ((uint32_t)0x00000100)
- #define IS_DMA_MODE(MODE)

5.31.1 Detailed Description

5.31.2 Macro Definition Documentation

5.31.2.1 IS_DMA_MODE

```
#define IS_DMA_MODE(
              MODE )
```

Value:

```
(((MODE) == DMA_Mode_Normal ) || \
((MODE) == DMA_Mode_Circular))
```

5.32 DMA_priority_level

Macros

- #define **DMA_Priority_Low** ((uint32_t)0x00000000)
- #define **DMA_Priority_Medium** ((uint32_t)0x00010000)
- #define **DMA_Priority_High** ((uint32_t)0x00020000)
- #define **DMA_Priority_VeryHigh** ((uint32_t)0x00030000)
- #define IS_DMA_PRIORITY(PRIORITY)

5.32.1 Detailed Description

5.32.2 Macro Definition Documentation

5.32.2.1 IS_DMA_PRIORITY

```
#define IS_DMA_PRIORITY(
              PRIORITY )
```

Value:

```
(((PRIORITY) == DMA_Priority_Low )    || \
((PRIORITY) == DMA_Priority_Medium) || \
((PRIORITY) == DMA_Priority_High)    || \
((PRIORITY) == DMA_Priority_VeryHigh))
```

5.33 DMA_fifo_direct_mode

Macros

- #define **DMA_FIFOMode_Disable** ((uint32_t)0x00000000)
- #define **DMA_FIFOMode_Enable** ((uint32_t)0x00000004)
- #define IS_DMA_FIFO_MODE_STATE(STATE)

5.33.1 Detailed Description

5.33.2 Macro Definition Documentation

5.33.2.1 IS_DMA_FIFO_MODE_STATE

```
#define IS_DMA_FIFO_MODE_STATE(
              STATE )
```

Value:

```
(((STATE) == DMA_FIFOMode_Disable ) || \
((STATE) == DMA_FIFOMode_Enable))
```

5.34 DMA_fifo_threshold_level

Macros

- #define **DMA_FIFOThreshold_1QuarterFull** ((uint32_t)0x00000000)
- #define **DMA_FIFOThreshold_HalfFull** ((uint32_t)0x00000001)
- #define **DMA_FIFOThreshold_3QuartersFull** ((uint32_t)0x00000002)
- #define **DMA_FIFOThreshold_Full** ((uint32_t)0x00000003)
- #define IS_DMA_FIFO_THRESHOLD(THRESHOLD)

5.34.1 Detailed Description

5.34.2 Macro Definition Documentation

5.34.2.1 IS_DMA_FIFO_THRESHOLD

```
#define IS_DMA_FIFO_THRESHOLD(
              THRESHOLD )
```

Value:

```
(((THRESHOLD) == DMA_FIFOThreshold_1QuarterFull ) || \
((THRESHOLD) == DMA_FIFOThreshold_HalfFull)      || \
((THRESHOLD) == DMA_FIFOThreshold_3QuartersFull) || \
((THRESHOLD) == DMA_FIFOThreshold_Full))
```

5.35 DMA_memory_burst

Macros

- #define **DMA_MemoryBurst_Single** ((uint32_t)0x00000000)
- #define **DMA_MemoryBurst_INC4** ((uint32_t)0x00800000)
- #define **DMA_MemoryBurst_INC8** ((uint32_t)0x01000000)
- #define **DMA_MemoryBurst_INC16** ((uint32_t)0x01800000)
- #define IS_DMA_MEMORY_BURST(BURST)

5.35.1 Detailed Description

5.35.2 Macro Definition Documentation

5.35.2.1 IS_DMA_MEMORY_BURST

```
#define IS_DMA_MEMORY_BURST(
              BURST )
```

Value:

```
(((BURST) == DMA_MemoryBurst_Single) || \
((BURST) == DMA_MemoryBurst_INC4)  || \
((BURST) == DMA_MemoryBurst_INC8)  || \
((BURST) == DMA_MemoryBurst_INC16))
```

5.36 DMA_peripheral_burst

Macros

- #define **DMA_PeripheralBurst_Single** ((uint32_t)0x00000000)
- #define **DMA_PeripheralBurst_INC4** ((uint32_t)0x00200000)
- #define **DMA_PeripheralBurst_INC8** ((uint32_t)0x00400000)
- #define **DMA_PeripheralBurst_INC16** ((uint32_t)0x00600000)
- #define IS_DMA_PERIPHERAL_BURST(BURST)

5.36.1 Detailed Description

5.36.2 Macro Definition Documentation

5.36.2.1 IS_DMA_PERIPHERAL_BURST

```
#define IS_DMA_PERIPHERAL_BURST(
              BURST )
```

Value:

```
(((BURST) == DMA_PeripheralBurst_Single) || \
((BURST) == DMA_PeripheralBurst_INC4)  || \
((BURST) == DMA_PeripheralBurst_INC8)  || \
((BURST) == DMA_PeripheralBurst_INC16))
```

5.37 DMA_fifo_status_level

Macros

- #define **DMA_FIFOStatus_Less1QuarterFull** ((uint32_t)0x00000000 << 3)
- #define **DMA_FIFOStatus_1QuarterFull** ((uint32_t)0x00000001 << 3)
- #define **DMA_FIFOStatus_HalfFull** ((uint32_t)0x00000002 << 3)
- #define **DMA_FIFOStatus_3QuartersFull** ((uint32_t)0x00000003 << 3)
- #define **DMA_FIFOStatus_Empty** ((uint32_t)0x00000004 << 3)
- #define **DMA_FIFOStatus_Full** ((uint32_t)0x00000005 << 3)
- #define IS_DMA_FIFO_STATUS(STATUS)

5.37.1 Detailed Description

5.37.2 Macro Definition Documentation

5.37.2.1 IS_DMA_FIFO_STATUS

```
#define IS_DMA_FIFO_STATUS(
              STATUS )
```

Value:

```
(((STATUS) == DMA_FIFOStatus_Less1QuarterFull ) || \
((STATUS) == DMA_FIFOStatus_HalfFull)          || \
((STATUS) == DMA_FIFOStatus_1QuarterFull)      || \
((STATUS) == DMA_FIFOStatus_3QuartersFull)     || \
((STATUS) == DMA_FIFOStatus_Full)              || \
((STATUS) == DMA_FIFOStatus_Empty))
```

5.38 DMA_flags_definition

Macros

- #define **DMA_FLAG_FEIF0** ((uint32_t)0x10800001)
- #define **DMA_FLAG_DMEIF0** ((uint32_t)0x10800004)
- #define **DMA_FLAG_TEIF0** ((uint32_t)0x10000008)
- #define **DMA_FLAG_HTIF0** ((uint32_t)0x10000010)
- #define **DMA_FLAG_TCIF0** ((uint32_t)0x10000020)
- #define **DMA_FLAG_FEIF1** ((uint32_t)0x10000040)
- #define **DMA_FLAG_DMEIF1** ((uint32_t)0x10000100)
- #define **DMA_FLAG_TEIF1** ((uint32_t)0x10000200)
- #define **DMA_FLAG_HTIF1** ((uint32_t)0x10000400)
- #define **DMA_FLAG_TCIF1** ((uint32_t)0x10000800)
- #define **DMA_FLAG_FEIF2** ((uint32_t)0x10010000)
- #define **DMA_FLAG_DMEIF2** ((uint32_t)0x10040000)
- #define **DMA_FLAG_TEIF2** ((uint32_t)0x10080000)
- #define **DMA_FLAG_HTIF2** ((uint32_t)0x10100000)
- #define **DMA_FLAG_TCIF2** ((uint32_t)0x10200000)
- #define **DMA_FLAG_FEIF3** ((uint32_t)0x10400000)
- #define **DMA_FLAG_DMEIF3** ((uint32_t)0x11000000)
- #define **DMA_FLAG_TEIF3** ((uint32_t)0x12000000)
- #define **DMA_FLAG_HTIF3** ((uint32_t)0x14000000)
- #define **DMA_FLAG_TCIF3** ((uint32_t)0x18000000)
- #define **DMA_FLAG_FEIF4** ((uint32_t)0x20000001)
- #define **DMA_FLAG_DMEIF4** ((uint32_t)0x20000004)
- #define **DMA_FLAG_TEIF4** ((uint32_t)0x20000008)
- #define **DMA_FLAG_HTIF4** ((uint32_t)0x20000010)
- #define **DMA_FLAG_TCIF4** ((uint32_t)0x20000020)
- #define **DMA_FLAG_FEIF5** ((uint32_t)0x20000040)
- #define **DMA_FLAG_DMEIF5** ((uint32_t)0x20000100)
- #define **DMA_FLAG_TEIF5** ((uint32_t)0x20000200)
- #define **DMA_FLAG_HTIF5** ((uint32_t)0x20000400)
- #define **DMA_FLAG_TCIF5** ((uint32_t)0x20000800)
- #define **DMA_FLAG_FEIF6** ((uint32_t)0x20010000)
- #define **DMA_FLAG_DMEIF6** ((uint32_t)0x20040000)
- #define **DMA_FLAG_TEIF6** ((uint32_t)0x20080000)
- #define **DMA_FLAG_HTIF6** ((uint32_t)0x20100000)
- #define **DMA_FLAG_TCIF6** ((uint32_t)0x20200000)
- #define **DMA_FLAG_FEIF7** ((uint32_t)0x20400000)
- #define **DMA_FLAG_DMEIF7** ((uint32_t)0x21000000)
- #define **DMA_FLAG_TEIF7** ((uint32_t)0x22000000)
- #define **DMA_FLAG_HTIF7** ((uint32_t)0x24000000)
- #define **DMA_FLAG_TCIF7** ((uint32_t)0x28000000)
- #define IS_DMA_CLEAR_FLAG(FLAG)
- #define IS_DMA_GET_FLAG(FLAG)

5.38.1 Detailed Description

5.38.2 Macro Definition Documentation

5.38.2.1 IS_DMA_CLEAR_FLAG

```
#define IS_DMA_CLEAR_FLAG(
              FLAG )
```

Value:

```
((((FLAG) & 0x30000000) != 0x30000000) && (((FLAG) & 0x30000000) != 0) && \
(((FLAG) & 0xC082F082) == 0x00) && ((FLAG) != 0x00))
```

5.38.2.2 IS_DMA_GET_FLAG

```
#define IS_DMA_GET_FLAG(
              FLAG )
```

Value:

```
(((FLAG) == DMA_FLAG_TCIF0)  || ((FLAG) == DMA_FLAG_HTIF0)  || \
((FLAG) == DMA_FLAG_TEIF0)  || ((FLAG) == DMA_FLAG_DMEIF0) || \
((FLAG) == DMA_FLAG_FEIF0)  || ((FLAG) == DMA_FLAG_TCIF1)  || \
((FLAG) == DMA_FLAG_HTIF1)  || ((FLAG) == DMA_FLAG_TEIF1)  || \
((FLAG) == DMA_FLAG_DMEIF1) || ((FLAG) == DMA_FLAG_FEIF1)  || \
((FLAG) == DMA_FLAG_TCIF2)  || ((FLAG) == DMA_FLAG_HTIF2)  || \
((FLAG) == DMA_FLAG_TEIF2)  || ((FLAG) == DMA_FLAG_DMEIF2) || \
((FLAG) == DMA_FLAG_FEIF2)  || ((FLAG) == DMA_FLAG_TCIF3)  || \
((FLAG) == DMA_FLAG_HTIF3)  || ((FLAG) == DMA_FLAG_TEIF3)  || \
((FLAG) == DMA_FLAG_DMEIF3) || ((FLAG) == DMA_FLAG_FEIF3)  || \
((FLAG) == DMA_FLAG_TCIF4)  || ((FLAG) == DMA_FLAG_HTIF4)  || \
((FLAG) == DMA_FLAG_TEIF4)  || ((FLAG) == DMA_FLAG_DMEIF4) || \
((FLAG) == DMA_FLAG_FEIF4)  || ((FLAG) == DMA_FLAG_TCIF5)  || \
((FLAG) == DMA_FLAG_HTIF5)  || ((FLAG) == DMA_FLAG_TEIF5)  || \
((FLAG) == DMA_FLAG_DMEIF5) || ((FLAG) == DMA_FLAG_FEIF5)  || \
((FLAG) == DMA_FLAG_TCIF6)  || ((FLAG) == DMA_FLAG_HTIF6)  || \
((FLAG) == DMA_FLAG_TEIF6)  || ((FLAG) == DMA_FLAG_DMEIF6) || \
((FLAG) == DMA_FLAG_FEIF6)  || ((FLAG) == DMA_FLAG_TCIF7)  || \
((FLAG) == DMA_FLAG_HTIF7)  || ((FLAG) == DMA_FLAG_TEIF7)  || \
((FLAG) == DMA_FLAG_DMEIF7) || ((FLAG) == DMA_FLAG_FEIF7))
```

5.39 DMA_interrupt_enable_definitions

Macros

- #define **DMA_IT_TC** ((uint32_t)0x00000010)
- #define **DMA_IT_HT** ((uint32_t)0x00000008)
- #define **DMA_IT_TE** ((uint32_t)0x00000004)
- #define **DMA_IT_DME** ((uint32_t)0x00000002)
- #define **DMA_IT_FE** ((uint32_t)0x00000080)
- #define **IS_DMA_CONFIG_IT**(IT) ((((IT) & 0xFFFFFF61) == 0x00) && ((IT) != 0x00))

5.39.1 Detailed Description

5.40 DMA_interrupts_definitions

Macros

- #define **DMA_IT_FEIF0** ((uint32_t)0x90000001)
- #define **DMA_IT_DMEIF0** ((uint32_t)0x10001004)

- #define **DMA_IT_TEIF0** ((uint32_t)0x10002008)
- #define **DMA_IT_HTIF0** ((uint32_t)0x10004010)
- #define **DMA_IT_TCIF0** ((uint32_t)0x10008020)
- #define **DMA_IT_FEIF1** ((uint32_t)0x90000040)
- #define **DMA_IT_DMEIF1** ((uint32_t)0x10001100)
- #define **DMA_IT_TEIF1** ((uint32_t)0x10002200)
- #define **DMA_IT_HTIF1** ((uint32_t)0x10004400)
- #define **DMA_IT_TCIF1** ((uint32_t)0x10008800)
- #define **DMA_IT_FEIF2** ((uint32_t)0x90010000)
- #define **DMA_IT_DMEIF2** ((uint32_t)0x10041000)
- #define **DMA_IT_TEIF2** ((uint32_t)0x10082000)
- #define **DMA_IT_HTIF2** ((uint32_t)0x10104000)
- #define **DMA_IT_TCIF2** ((uint32_t)0x10208000)
- #define **DMA_IT_FEIF3** ((uint32_t)0x90400000)
- #define **DMA_IT_DMEIF3** ((uint32_t)0x11001000)
- #define **DMA_IT_TEIF3** ((uint32_t)0x12002000)
- #define **DMA_IT_HTIF3** ((uint32_t)0x14004000)
- #define **DMA_IT_TCIF3** ((uint32_t)0x18008000)
- #define **DMA_IT_FEIF4** ((uint32_t)0xA0000001)
- #define **DMA_IT_DMEIF4** ((uint32_t)0x20001004)
- #define **DMA_IT_TEIF4** ((uint32_t)0x20002008)
- #define **DMA_IT_HTIF4** ((uint32_t)0x20004010)
- #define **DMA_IT_TCIF4** ((uint32_t)0x20008020)
- #define **DMA_IT_FEIF5** ((uint32_t)0xA0000040)
- #define **DMA_IT_DMEIF5** ((uint32_t)0x20001100)
- #define **DMA_IT_TEIF5** ((uint32_t)0x20002200)
- #define **DMA_IT_HTIF5** ((uint32_t)0x20004400)
- #define **DMA_IT_TCIF5** ((uint32_t)0x20008800)
- #define **DMA_IT_FEIF6** ((uint32_t)0xA0010000)
- #define **DMA_IT_DMEIF6** ((uint32_t)0x20041000)
- #define **DMA_IT_TEIF6** ((uint32_t)0x20082000)
- #define **DMA_IT_HTIF6** ((uint32_t)0x20104000)
- #define **DMA_IT_TCIF6** ((uint32_t)0x20208000)
- #define **DMA_IT_FEIF7** ((uint32_t)0xA0400000)
- #define **DMA_IT_DMEIF7** ((uint32_t)0x21001000)
- #define **DMA_IT_TEIF7** ((uint32_t)0x22002000)
- #define **DMA_IT_HTIF7** ((uint32_t)0x24004000)
- #define **DMA_IT_TCIF7** ((uint32_t)0x28008000)
- #define IS_DMA_CLEAR_IT(IT)
- #define IS_DMA_GET_IT(IT)

5.40.1 Detailed Description

5.40.2 Macro Definition Documentation

5.40.2.1 IS_DMA_CLEAR_IT

```
#define IS_DMA_CLEAR_IT(
              IT )
```

Value:

```
((((IT) & 0x30000000) != 0x30000000) && \
(((IT) & 0x30000000) != 0) && ((IT) != 0x00) && \
(((IT) & 0x40820082) == 0x00))
```

5.40.2.2 IS_DMA_GET_IT

```
#define IS_DMA_GET_IT(
              IT )
```

Value:

```
(((IT) == DMA_IT_TCIF0) || ((IT) == DMA_IT_HTIF0)  || \
((IT) == DMA_IT_TEIF0) || ((IT) == DMA_IT_DMEIF0) || \
((IT) == DMA_IT_FEIF0) || ((IT) == DMA_IT_TCIF1)  || \
((IT) == DMA_IT_HTIF1) || ((IT) == DMA_IT_TEIF1)  || \
((IT) == DMA_IT_DMEIF1)|| ((IT) == DMA_IT_FEIF1)  || \
((IT) == DMA_IT_TCIF2) || ((IT) == DMA_IT_HTIF2)  || \
((IT) == DMA_IT_TEIF2) || ((IT) == DMA_IT_DMEIF2) || \
((IT) == DMA_IT_FEIF2) || ((IT) == DMA_IT_TCIF3)  || \
((IT) == DMA_IT_HTIF3) || ((IT) == DMA_IT_TEIF3)  || \
((IT) == DMA_IT_DMEIF3)|| ((IT) == DMA_IT_FEIF3)  || \
((IT) == DMA_IT_TCIF4) || ((IT) == DMA_IT_HTIF4)  || \
((IT) == DMA_IT_TEIF4) || ((IT) == DMA_IT_DMEIF4) || \
((IT) == DMA_IT_FEIF4) || ((IT) == DMA_IT_TCIF5)  || \
((IT) == DMA_IT_HTIF5) || ((IT) == DMA_IT_TEIF5)  || \
((IT) == DMA_IT_DMEIF5)|| ((IT) == DMA_IT_FEIF5)  || \
((IT) == DMA_IT_TCIF6) || ((IT) == DMA_IT_HTIF6)  || \
((IT) == DMA_IT_TEIF6) || ((IT) == DMA_IT_DMEIF6) || \
((IT) == DMA_IT_FEIF6) || ((IT) == DMA_IT_TCIF7)  || \
((IT) == DMA_IT_HTIF7) || ((IT) == DMA_IT_TEIF7)  || \
((IT) == DMA_IT_DMEIF7)|| ((IT) == DMA_IT_FEIF7))
```

5.41 DMA_peripheral_increment_offset

Macros

- #define **DMA_PINCOS_Psize** ((uint32_t)0x00000000)
- #define **DMA_PINCOS_WordAligned** ((uint32_t)0x00008000)
- #define IS_DMA_PINCOS_SIZE(SIZE)

5.41.1 Detailed Description

5.41.2 Macro Definition Documentation

5.41.2.1 IS_DMA_PINCOS_SIZE

```
#define IS_DMA_PINCOS_SIZE(
              SIZE )
```

Value:

```
(((SIZE) == DMA_PINCOS_Psize) || \
((SIZE) == DMA_PINCOS_WordAligned))
```

5.42 DMA_flow_controller_definitions

Macros

- #define **DMA_FlowCtrl_Memory** ((uint32_t)0x00000000)
- #define **DMA_FlowCtrl_Peripheral** ((uint32_t)0x00000020)
- #define IS_DMA_FLOW_CTRL(CTRL)

5.42.1 Detailed Description

5.42.2 Macro Definition Documentation

5.42.2.1 IS_DMA_FLOW_CTRL

```
#define IS_DMA_FLOW_CTRL(
              CTRL )
```

Value:

```
(((CTRL) == DMA_FlowCtrl_Memory) || \
((CTRL) == DMA_FlowCtrl_Peripheral))
```

5.43 DMA_memory_targets_definitions

Macros

- #define **DMA_Memory_0** ((uint32_t)0x00000000)
- #define **DMA_Memory_1** ((uint32_t)0x00080000)
- #define **IS_DMA_CURRENT_MEM**(MEM) (((MEM) == DMA_Memory_0) || ((MEM) == DMA_Memory_1))

5.43.1 Detailed Description

5.44 GPIO_Exported_Constants

Modules

- GPIO_pins_define
- GPIO_Pin_sources
- GPIO_Alternat_function_selection_define
- GPIO_Legacy

5.44.1 Detailed Description

5.45 GPIO_pins_define

Macros

- #define **GPIO_Pin_0** ((uint16_t)0x0001) /* Pin 0 selected */
- #define **GPIO_Pin_1** ((uint16_t)0x0002) /* Pin 1 selected */
- #define **GPIO_Pin_2** ((uint16_t)0x0004) /* Pin 2 selected */
- #define **GPIO_Pin_3** ((uint16_t)0x0008) /* Pin 3 selected */
- #define **GPIO_Pin_4** ((uint16_t)0x0010) /* Pin 4 selected */
- #define **GPIO_Pin_5** ((uint16_t)0x0020) /* Pin 5 selected */
- #define **GPIO_Pin_6** ((uint16_t)0x0040) /* Pin 6 selected */
- #define **GPIO_Pin_7** ((uint16_t)0x0080) /* Pin 7 selected */
- #define **GPIO_Pin_8** ((uint16_t)0x0100) /* Pin 8 selected */
- #define **GPIO_Pin_9** ((uint16_t)0x0200) /* Pin 9 selected */
- #define **GPIO_Pin_10** ((uint16_t)0x0400) /* Pin 10 selected */
- #define **GPIO_Pin_11** ((uint16_t)0x0800) /* Pin 11 selected */
- #define **GPIO_Pin_12** ((uint16_t)0x1000) /* Pin 12 selected */
- #define **GPIO_Pin_13** ((uint16_t)0x2000) /* Pin 13 selected */
- #define **GPIO_Pin_14** ((uint16_t)0x4000) /* Pin 14 selected */
- #define **GPIO_Pin_15** ((uint16_t)0x8000) /* Pin 15 selected */
- #define **GPIO_Pin_All** ((uint16_t)0xFFFF) /* All pins selected */
- #define **IS_GPIO_PIN**(PIN) ((((PIN) & (uint16_t)0x00) == 0x00) && ((PIN) != (uint16_t)0x00))
- #define IS_GET_GPIO_PIN(PIN)

5.45.1 Detailed Description

5.45.2 Macro Definition Documentation

5.45.2.1 IS_GET_GPIO_PIN

```
#define IS_GET_GPIO_PIN(
              PIN )
```

Value:

```
(((PIN) == GPIO_Pin_0) || \
((PIN) == GPIO_Pin_1) || \
((PIN) == GPIO_Pin_2) || \
((PIN) == GPIO_Pin_3) || \
((PIN) == GPIO_Pin_4) || \
((PIN) == GPIO_Pin_5) || \
((PIN) == GPIO_Pin_6) || \
((PIN) == GPIO_Pin_7) || \
((PIN) == GPIO_Pin_8) || \
((PIN) == GPIO_Pin_9) || \
((PIN) == GPIO_Pin_10) || \
((PIN) == GPIO_Pin_11) || \
((PIN) == GPIO_Pin_12) || \
((PIN) == GPIO_Pin_13) || \
((PIN) == GPIO_Pin_14) || \
((PIN) == GPIO_Pin_15))
```

5.46 GPIO_Pin_sources

Macros

- #define **GPIO_PinSource0** ((uint8_t)0x00)
- #define **GPIO_PinSource1** ((uint8_t)0x01)
- #define **GPIO_PinSource2** ((uint8_t)0x02)
- #define **GPIO_PinSource3** ((uint8_t)0x03)
- #define **GPIO_PinSource4** ((uint8_t)0x04)
- #define **GPIO_PinSource5** ((uint8_t)0x05)
- #define **GPIO_PinSource6** ((uint8_t)0x06)
- #define **GPIO_PinSource7** ((uint8_t)0x07)
- #define **GPIO_PinSource8** ((uint8_t)0x08)
- #define **GPIO_PinSource9** ((uint8_t)0x09)
- #define **GPIO_PinSource10** ((uint8_t)0x0A)
- #define **GPIO_PinSource11** ((uint8_t)0x0B)
- #define **GPIO_PinSource12** ((uint8_t)0x0C)
- #define **GPIO_PinSource13** ((uint8_t)0x0D)
- #define **GPIO_PinSource14** ((uint8_t)0x0E)
- #define **GPIO_PinSource15** ((uint8_t)0x0F)
- #define IS_GPIO_PIN_SOURCE(PINSOURCE)

5.46.1 Detailed Description

5.46.2 Macro Definition Documentation

5.46.2.1 IS_GPIO_PIN_SOURCE

```
#define IS_GPIO_PIN_SOURCE(
              PINSOURCE )
```

Value:

```
(((PINSOURCE) == GPIO_PinSource0) || \
((PINSOURCE) == GPIO_PinSource1) || \
((PINSOURCE) == GPIO_PinSource2) || \
((PINSOURCE) == GPIO_PinSource3) || \
((PINSOURCE) == GPIO_PinSource4) || \
((PINSOURCE) == GPIO_PinSource5) || \
((PINSOURCE) == GPIO_PinSource6) || \
((PINSOURCE) == GPIO_PinSource7) || \
((PINSOURCE) == GPIO_PinSource8) || \
((PINSOURCE) == GPIO_PinSource9) || \
((PINSOURCE) == GPIO_PinSource10) || \
((PINSOURCE) == GPIO_PinSource11) || \
((PINSOURCE) == GPIO_PinSource12) || \
((PINSOURCE) == GPIO_PinSource13) || \
((PINSOURCE) == GPIO_PinSource14) || \
((PINSOURCE) == GPIO_PinSource15))
```

5.47 GPIO_Alternat_function_selection_define

Macros

- #define **GPIO_AF_RTC_50Hz** ((uint8_t)0x00) /* RTC_50Hz Alternate Function mapping */

  AF 0 selection

- #define **GPIO_AF_MCO** ((uint8_t)0x00) /* MCO (MCO1 and MCO2) Alternate Function mapping */
- #define **GPIO_AF_TAMPER** ((uint8_t)0x00) /* TAMPER (TAMPER_1 and TAMPER_2) Alternate Function mapping */
- #define **GPIO_AF_SWJ** ((uint8_t)0x00) /* SWJ (SWD and JTAG) Alternate Function mapping */
- #define **GPIO_AF_TRACE** ((uint8_t)0x00) /* TRACE Alternate Function mapping */
- #define **GPIO_AF_TIM1** ((uint8_t)0x01) /* TIM1 Alternate Function mapping */

  AF 1 selection

- #define **GPIO_AF_TIM2** ((uint8_t)0x01) /* TIM2 Alternate Function mapping */
- #define **GPIO_AF_TIM3** ((uint8_t)0x02) /* TIM3 Alternate Function mapping */

  AF 2 selection

- #define **GPIO_AF_TIM4** ((uint8_t)0x02) /* TIM4 Alternate Function mapping */
- #define **GPIO_AF_TIM5** ((uint8_t)0x02) /* TIM5 Alternate Function mapping */
- #define **GPIO_AF_TIM8** ((uint8_t)0x03) /* TIM8 Alternate Function mapping */

  AF 3 selection

- #define **GPIO_AF_TIM9** ((uint8_t)0x03) /* TIM9 Alternate Function mapping */
- #define **GPIO_AF_TIM10** ((uint8_t)0x03) /* TIM10 Alternate Function mapping */
- #define **GPIO_AF_TIM11** ((uint8_t)0x03) /* TIM11 Alternate Function mapping */
- #define **GPIO_AF_I2C1** ((uint8_t)0x04) /* I2C1 Alternate Function mapping */

  AF 4 selection

- #define **GPIO_AF_I2C2** ((uint8_t)0x04) /* I2C2 Alternate Function mapping */
- #define **GPIO_AF_I2C3** ((uint8_t)0x04) /* I2C3 Alternate Function mapping */
- #define **GPIO_AF_SPI1** ((uint8_t)0x05) /* SPI1 Alternate Function mapping */

  AF 5 selection

- #define **GPIO_AF_SPI2** ((uint8_t)0x05) /* SPI2/I2S2 Alternate Function mapping */
- #define **GPIO_AF_SPI3** ((uint8_t)0x06) /* SPI3/I2S3 Alternate Function mapping */

  AF 6 selection

- #define **GPIO_AF_USART1** ((uint8_t)0x07) /* USART1 Alternate Function mapping */

  AF 7 selection

- #define **GPIO_AF_USART2** ((uint8_t)0x07) /* USART2 Alternate Function mapping */
- #define **GPIO_AF_USART3** ((uint8_t)0x07) /* USART3 Alternate Function mapping */
- #define **GPIO_AF_I2S3ext** ((uint8_t)0x07) /* I2S3ext Alternate Function mapping */
- #define **GPIO_AF_UART4** ((uint8_t)0x08) /* UART4 Alternate Function mapping */

  AF 8 selection

- #define **GPIO_AF_UART5** ((uint8_t)0x08) /* UART5 Alternate Function mapping */
- #define **GPIO_AF_USART6** ((uint8_t)0x08) /* USART6 Alternate Function mapping */
- #define **GPIO_AF_CAN1** ((uint8_t)0x09) /* CAN1 Alternate Function mapping */

  AF 9 selection.
- #define **GPIO_AF_CAN2** ((uint8_t)0x09) /* CAN2 Alternate Function mapping */
- #define **GPIO_AF_TIM12** ((uint8_t)0x09) /* TIM12 Alternate Function mapping */

- #define **GPIO_AF_TIM13** ((uint8_t)0x09) /* TIM13 Alternate Function mapping */
- #define **GPIO_AF_TIM14** ((uint8_t)0x09) /* TIM14 Alternate Function mapping */
- #define **GPIO_AF_OTG_FS** ((uint8_t)0xA) /* OTG_FS Alternate Function mapping */

  AF 10 selection

- #define **GPIO_AF_OTG_HS** ((uint8_t)0xA) /* OTG_HS Alternate Function mapping */
- #define **GPIO_AF_ETH** ((uint8_t)0x0B) /* ETHERNET Alternate Function mapping */

  AF 11 selection

- #define **GPIO_AF_FSMC** ((uint8_t)0xC) /* FSMC Alternate Function mapping */

  AF 12 selection

- #define **GPIO_AF_OTG_HS_FS** ((uint8_t)0xC) /* OTG HS configured in FS, Alternate Function mapping */
- #define **GPIO_AF_SDIO** ((uint8_t)0xC) /* SDIO Alternate Function mapping */
- #define **GPIO_AF_DCMI** ((uint8_t)0x0D) /* DCMI Alternate Function mapping */

  AF 13 selection

- #define **GPIO_AF_EVENTOUT** ((uint8_t)0x0F) /* EVENTOUT Alternate Function mapping */

  AF 15 selection

- #define IS_GPIO_AF(AF)

5.47.1 Detailed Description

5.47.2 Macro Definition Documentation

5.47.2.1 IS_GPIO_AF

```
#define IS_GPIO_AF(
              AF )
```

Value:

```
(((AF) == GPIO_AF_RTC_50Hz)  || ((AF) == GPIO_AF_TIM14)  || \
((AF) == GPIO_AF_MCO)        || ((AF) == GPIO_AF_TAMPER) || \
((AF) == GPIO_AF_SWJ)        || ((AF) == GPIO_AF_TRACE)  || \
((AF) == GPIO_AF_TIM1)       || ((AF) == GPIO_AF_TIM2)   || \
((AF) == GPIO_AF_TIM3)       || ((AF) == GPIO_AF_TIM4)   || \
((AF) == GPIO_AF_TIM5)       || ((AF) == GPIO_AF_TIM8)   || \
((AF) == GPIO_AF_I2C1)       || ((AF) == GPIO_AF_I2C2)   || \
((AF) == GPIO_AF_I2C3)       || ((AF) == GPIO_AF_SPI1)   || \
((AF) == GPIO_AF_SPI2)       || ((AF) == GPIO_AF_TIM13)  || \
((AF) == GPIO_AF_SPI3)       || ((AF) == GPIO_AF_TIM14)  || \
((AF) == GPIO_AF_USART1)     || ((AF) == GPIO_AF_USART2) || \
((AF) == GPIO_AF_USART3)     || ((AF) == GPIO_AF_UART4)  || \
((AF) == GPIO_AF_UART5)      || ((AF) == GPIO_AF_USART6) || \
((AF) == GPIO_AF_CAN1)       || ((AF) == GPIO_AF_CAN2)   || \
((AF) == GPIO_AF_OTG_FS)     || ((AF) == GPIO_AF_OTG_HS) || \
((AF) == GPIO_AF_ETH)        || ((AF) == GPIO_AF_FSMC)   || \
((AF) == GPIO_AF_OTG_HS_FS)  || ((AF) == GPIO_AF_SDIO)   || \
((AF) == GPIO_AF_DCMI)       || ((AF) == GPIO_AF_EVENTOUT))
```

5.48 GPIO_Legacy

Macros

- #define **GPIO_Mode_AIN** GPIO_Mode_AN
- #define **GPIO_AF_OTG1_FS** GPIO_AF_OTG_FS
- #define **GPIO_AF_OTG2_HS** GPIO_AF_OTG_HS
- #define **GPIO_AF_OTG2_FS** GPIO_AF_OTG_HS_FS

5.48.1 Detailed Description

5.49 RCC_Exported_Constants

Modules

- RCC_HSE_configuration
- RCC_PLL_Clock_Source
- RCC_System_Clock_Source
- RCC_AHB_Clock_Source
- RCC_APB1_APB2_Clock_Source
- RCC_Interrupt_Source
- RCC_LSE_Configuration
- RCC_RTC_Clock_Source
- RCC_I2S_Clock_Source
- RCC_AHB1_Peripherals
- RCC_AHB2_Peripherals
- RCC_AHB3_Peripherals
- RCC_APB1_Peripherals
- RCC_APB2_Peripherals
- RCC_MCO1_Clock_Source_Prescaler
- RCC_MCO2_Clock_Source_Prescaler
- RCC_Flag

5.49.1 Detailed Description

5.50 RCC_HSE_configuration

Macros

- #define **RCC_HSE_OFF** ((uint8_t)0x00)
- #define **RCC_HSE_ON** ((uint8_t)0x01)
- #define **RCC_HSE_Bypass** ((uint8_t)0x05)
- #define IS_RCC_HSE(HSE)

5.50.1 Detailed Description

5.50.2 Macro Definition Documentation

5.50.2.1 IS_RCC_HSE

```
#define IS_RCC_HSE(
               HSE )
```

Value:

```
(((HSE) == RCC_HSE_OFF) || ((HSE) == RCC_HSE_ON) || \
((HSE) == RCC_HSE_Bypass))
```

5.51 RCC_PLL_Clock_Source

Macros

- #define **RCC_PLLSource_HSI** ((uint32_t)0x00000000)
- #define **RCC_PLLSource_HSE** ((uint32_t)0x00400000)
- #define IS_RCC_PLL_SOURCE(SOURCE)
- #define **IS_RCC_PLLM_VALUE**(VALUE) ((VALUE) <= 63)
- #define **IS_RCC_PLLN_VALUE**(VALUE) ((192 <= (VALUE)) && ((VALUE) <= 432))
- #define **IS_RCC_PLLP_VALUE**(VALUE) (((VALUE) == 2) || ((VALUE) == 4) || ((VALUE) == 6) || ((VALUE) == 8))
- #define **IS_RCC_PLLQ_VALUE**(VALUE) ((4 <= (VALUE)) && ((VALUE) <= 15))
- #define **IS_RCC_PLLI2SN_VALUE**(VALUE) ((192 <= (VALUE)) && ((VALUE) <= 432))
- #define **IS_RCC_PLLI2SR_VALUE**(VALUE) ((2 <= (VALUE)) && ((VALUE) <= 7))

5.51.1 Detailed Description

5.51.2 Macro Definition Documentation

5.51.2.1 IS_RCC_PLL_SOURCE

```
#define IS_RCC_PLL_SOURCE(
               SOURCE )
```

Value:

```
(((SOURCE) == RCC_PLLSource_HSI) || \
((SOURCE) == RCC_PLLSource_HSE))
```

5.52 RCC_System_Clock_Source

Macros

- #define **RCC_SYSCLKSource_HSI** ((uint32_t)0x00000000)
- #define **RCC_SYSCLKSource_HSE** ((uint32_t)0x00000001)
- #define **RCC_SYSCLKSource_PLLCLK** ((uint32_t)0x00000002)
- #define IS_RCC_SYSCLK_SOURCE(SOURCE)

5.52.1 Detailed Description

5.52.2 Macro Definition Documentation

5.52.2.1 IS_RCC_SYSCLK_SOURCE

```
#define IS_RCC_SYSCLK_SOURCE(
              SOURCE )
```

Value:

```
(((SOURCE) == RCC_SYSCLKSource_HSI) || \
((SOURCE) == RCC_SYSCLKSource_HSE) || \
((SOURCE) == RCC_SYSCLKSource_PLLCLK))
```

5.53 RCC_AHB_Clock_Source

Macros

- #define **RCC_SYSCLK_Div1** ((uint32_t)0x00000000)
- #define **RCC_SYSCLK_Div2** ((uint32_t)0x00000080)
- #define **RCC_SYSCLK_Div4** ((uint32_t)0x00000090)
- #define **RCC_SYSCLK_Div8** ((uint32_t)0x000000A0)
- #define **RCC_SYSCLK_Div16** ((uint32_t)0x000000B0)
- #define **RCC_SYSCLK_Div64** ((uint32_t)0x000000C0)
- #define **RCC_SYSCLK_Div128** ((uint32_t)0x000000D0)
- #define **RCC_SYSCLK_Div256** ((uint32_t)0x000000E0)
- #define **RCC_SYSCLK_Div512** ((uint32_t)0x000000F0)
- #define IS_RCC_HCLK(HCLK)

5.53.1 Detailed Description

5.53.2 Macro Definition Documentation

5.53.2.1 IS_RCC_HCLK

```
#define IS_RCC_HCLK(
              HCLK )
```

Value:

```
(((HCLK) == RCC_SYSCLK_Div1) || ((HCLK) == RCC_SYSCLK_Div2) || \
((HCLK) == RCC_SYSCLK_Div4) || ((HCLK) == RCC_SYSCLK_Div8) || \
((HCLK) == RCC_SYSCLK_Div16) || ((HCLK) == RCC_SYSCLK_Div64) || \
((HCLK) == RCC_SYSCLK_Div128) || ((HCLK) == RCC_SYSCLK_Div256) || \
((HCLK) == RCC_SYSCLK_Div512))
```

5.54 RCC_APB1_APB2_Clock_Source

Macros

- #define **RCC_HCLK_Div1** ((uint32_t)0x00000000)
- #define **RCC_HCLK_Div2** ((uint32_t)0x00001000)
- #define **RCC_HCLK_Div4** ((uint32_t)0x00001400)
- #define **RCC_HCLK_Div8** ((uint32_t)0x00001800)
- #define **RCC_HCLK_Div16** ((uint32_t)0x00001C00)
- #define IS_RCC_PCLK(PCLK)

5.54.1 Detailed Description

5.54.2 Macro Definition Documentation

5.54.2.1 IS_RCC_PCLK

```
#define IS_RCC_PCLK(
              PCLK )
```

Value:

```
(((PCLK) == RCC_HCLK_Div1) || ((PCLK) == RCC_HCLK_Div2) || \
((PCLK) == RCC_HCLK_Div4) || ((PCLK) == RCC_HCLK_Div8) || \
((PCLK) == RCC_HCLK_Div16))
```

5.55 RCC_Interrupt_Source

Macros

- #define **RCC_IT_LSIRDY** ((uint8_t)0x01)
- #define **RCC_IT_LSERDY** ((uint8_t)0x02)
- #define **RCC_IT_HSIRDY** ((uint8_t)0x04)
- #define **RCC_IT_HSERDY** ((uint8_t)0x08)
- #define **RCC_IT_PLLRDY** ((uint8_t)0x10)
- #define **RCC_IT_PLLI2SRDY** ((uint8_t)0x20)
- #define **RCC_IT_CSS** ((uint8_t)0x80)
- #define **IS_RCC_IT**(IT) ((((IT) & (uint8_t)0xC0) == 0x00) && ((IT) != 0x00))
- #define IS_RCC_GET_IT(IT)
- #define **IS_RCC_CLEAR_IT**(IT) ((((IT) & (uint8_t)0x40) == 0x00) && ((IT) != 0x00))

5.55.1 Detailed Description

5.55.2 Macro Definition Documentation

5.55.2.1 IS_RCC_GET_IT

```
#define IS_RCC_GET_IT(
              IT )
```

Value:

```
(((IT) == RCC_IT_LSIRDY) || ((IT) == RCC_IT_LSERDY) || \
((IT) == RCC_IT_HSIRDY) || ((IT) == RCC_IT_HSERDY) || \
((IT) == RCC_IT_PLLRDY) || ((IT) == RCC_IT_CSS) || \
((IT) == RCC_IT_PLLI2SRDY))
```

5.56 RCC_LSE_Configuration

Macros

- #define **RCC_LSE_OFF** ((uint8_t)0x00)
- #define **RCC_LSE_ON** ((uint8_t)0x01)
- #define **RCC_LSE_Bypass** ((uint8_t)0x04)
- #define IS_RCC_LSE(LSE)

5.56.1 Detailed Description

5.56.2 Macro Definition Documentation

5.56.2.1 IS_RCC_LSE

```
#define IS_RCC_LSE(
              LSE )
```

Value:

```
(((LSE) == RCC_LSE_OFF) || ((LSE) == RCC_LSE_ON) || \
((LSE) == RCC_LSE_Bypass))
```

5.57 RCC_RTC_Clock_Source

Macros

- #define **RCC_RTCCLKSource_LSE** ((uint32_t)0x00000100)
- #define **RCC_RTCCLKSource_LSI** ((uint32_t)0x00000200)
- #define **RCC_RTCCLKSource_HSE_Div2** ((uint32_t)0x00020300)
- #define **RCC_RTCCLKSource_HSE_Div3** ((uint32_t)0x00030300)
- #define **RCC_RTCCLKSource_HSE_Div4** ((uint32_t)0x00040300)
- #define **RCC_RTCCLKSource_HSE_Div5** ((uint32_t)0x00050300)
- #define **RCC_RTCCLKSource_HSE_Div6** ((uint32_t)0x00060300)
- #define **RCC_RTCCLKSource_HSE_Div7** ((uint32_t)0x00070300)
- #define **RCC_RTCCLKSource_HSE_Div8** ((uint32_t)0x00080300)
- #define **RCC_RTCCLKSource_HSE_Div9** ((uint32_t)0x00090300)

- #define **RCC_RTCCLKSource_HSE_Div10** ((uint32_t)0x000A0300)
- #define **RCC_RTCCLKSource_HSE_Div11** ((uint32_t)0x000B0300)
- #define **RCC_RTCCLKSource_HSE_Div12** ((uint32_t)0x000C0300)
- #define **RCC_RTCCLKSource_HSE_Div13** ((uint32_t)0x000D0300)
- #define **RCC_RTCCLKSource_HSE_Div14** ((uint32_t)0x000E0300)
- #define **RCC_RTCCLKSource_HSE_Div15** ((uint32_t)0x000F0300)
- #define **RCC_RTCCLKSource_HSE_Div16** ((uint32_t)0x00100300)
- #define **RCC_RTCCLKSource_HSE_Div17** ((uint32_t)0x00110300)
- #define **RCC_RTCCLKSource_HSE_Div18** ((uint32_t)0x00120300)
- #define **RCC_RTCCLKSource_HSE_Div19** ((uint32_t)0x00130300)
- #define **RCC_RTCCLKSource_HSE_Div20** ((uint32_t)0x00140300)
- #define **RCC_RTCCLKSource_HSE_Div21** ((uint32_t)0x00150300)
- #define **RCC_RTCCLKSource_HSE_Div22** ((uint32_t)0x00160300)
- #define **RCC_RTCCLKSource_HSE_Div23** ((uint32_t)0x00170300)
- #define **RCC_RTCCLKSource_HSE_Div24** ((uint32_t)0x00180300)
- #define **RCC_RTCCLKSource_HSE_Div25** ((uint32_t)0x00190300)
- #define **RCC_RTCCLKSource_HSE_Div26** ((uint32_t)0x001A0300)
- #define **RCC_RTCCLKSource_HSE_Div27** ((uint32_t)0x001B0300)
- #define **RCC_RTCCLKSource_HSE_Div28** ((uint32_t)0x001C0300)
- #define **RCC_RTCCLKSource_HSE_Div29** ((uint32_t)0x001D0300)
- #define **RCC_RTCCLKSource_HSE_Div30** ((uint32_t)0x001E0300)
- #define **RCC_RTCCLKSource_HSE_Div31** ((uint32_t)0x001F0300)
- #define **IS_RCC_RTCCLK_SOURCE**(SOURCE)

5.57.1 Detailed Description

5.58 RCC_I2S_Clock_Source

Macros

- #define **RCC_I2S2CLKSource_PLLI2S** ((uint8_t)0x00)
- #define **RCC_I2S2CLKSource_Ext** ((uint8_t)0x01)
- #define **IS_RCC_I2SCLK_SOURCE**(SOURCE) (((SOURCE) == RCC_I2S2CLKSource_PLLI2S) || ((SOURCE) == RCC_I2S2CLKSource_Ext))

5.58.1 Detailed Description

5.59 RCC_AHB1_Peripherals

Macros

- #define **RCC_AHB1Periph_GPIOA** ((uint32_t)0x00000001)
- #define **RCC_AHB1Periph_GPIOB** ((uint32_t)0x00000002)
- #define **RCC_AHB1Periph_GPIOC** ((uint32_t)0x00000004)
- #define **RCC_AHB1Periph_GPIOD** ((uint32_t)0x00000008)
- #define **RCC_AHB1Periph_GPIOE** ((uint32_t)0x00000010)
- #define **RCC_AHB1Periph_GPIOF** ((uint32_t)0x00000020)
- #define **RCC_AHB1Periph_GPIOG** ((uint32_t)0x00000040)
- #define **RCC_AHB1Periph_GPIOH** ((uint32_t)0x00000080)

- #define **RCC_AHB1Periph_GPIOI** ((uint32_t)0x00000100)
- #define **RCC_AHB1Periph_CRC** ((uint32_t)0x00001000)
- #define **RCC_AHB1Periph_FLITF** ((uint32_t)0x00008000)
- #define **RCC_AHB1Periph_SRAM1** ((uint32_t)0x00010000)
- #define **RCC_AHB1Periph_SRAM2** ((uint32_t)0x00020000)
- #define **RCC_AHB1Periph_BKPSRAM** ((uint32_t)0x00040000)
- #define **RCC_AHB1Periph_CCMDATARAMEN** ((uint32_t)0x00100000)
- #define **RCC_AHB1Periph_DMA1** ((uint32_t)0x00200000)
- #define **RCC_AHB1Periph_DMA2** ((uint32_t)0x00400000)
- #define **RCC_AHB1Periph_ETH_MAC** ((uint32_t)0x02000000)
- #define **RCC_AHB1Periph_ETH_MAC_Tx** ((uint32_t)0x04000000)
- #define **RCC_AHB1Periph_ETH_MAC_Rx** ((uint32_t)0x08000000)
- #define **RCC_AHB1Periph_ETH_MAC_PTP** ((uint32_t)0x10000000)
- #define **RCC_AHB1Periph_OTG_HS** ((uint32_t)0x20000000)
- #define **RCC_AHB1Periph_OTG_HS_ULPI** ((uint32_t)0x40000000)
- #define **IS_RCC_AHB1_CLOCK_PERIPH**(PERIPH) ((((PERIPH) & 0x818BEE00) == 0x00) && ((PERIPH) != 0x00))
- #define **IS_RCC_AHB1_RESET_PERIPH**(PERIPH) ((((PERIPH) & 0xDD9FEE00) == 0x00) && ((PERIPH) != 0x00))
- #define **IS_RCC_AHB1_LPMODE_PERIPH**(PERIPH) ((((PERIPH) & 0x81986E00) == 0x00) && ((PERIPH) != 0x00))

5.59.1 Detailed Description

5.60 RCC_AHB2_Peripherals

Macros

- #define **RCC_AHB2Periph_DCMI** ((uint32_t)0x00000001)
- #define **RCC_AHB2Periph_CRYP** ((uint32_t)0x00000010)
- #define **RCC_AHB2Periph_HASH** ((uint32_t)0x00000020)
- #define **RCC_AHB2Periph_RNG** ((uint32_t)0x00000040)
- #define **RCC_AHB2Periph_OTG_FS** ((uint32_t)0x00000080)
- #define **IS_RCC_AHB2_PERIPH**(PERIPH) ((((PERIPH) & 0xFFFFFF0E) == 0x00) && ((PERIPH) != 0x00))

5.60.1 Detailed Description

5.61 RCC_AHB3_Peripherals

Macros

- #define **RCC_AHB3Periph_FSMC** ((uint32_t)0x00000001)
- #define **IS_RCC_AHB3_PERIPH**(PERIPH) ((((PERIPH) & 0xFFFFFFFE) == 0x00) && ((PERIPH) != 0x00))

5.61.1 Detailed Description

5.62 RCC_APB1_Peripherals

Macros

- #define **RCC_APB1Periph_TIM2** ((uint32_t)0x00000001)
- #define **RCC_APB1Periph_TIM3** ((uint32_t)0x00000002)
- #define **RCC_APB1Periph_TIM4** ((uint32_t)0x00000004)
- #define **RCC_APB1Periph_TIM5** ((uint32_t)0x00000008)
- #define **RCC_APB1Periph_TIM6** ((uint32_t)0x00000010)
- #define **RCC_APB1Periph_TIM7** ((uint32_t)0x00000020)
- #define **RCC_APB1Periph_TIM12** ((uint32_t)0x00000040)
- #define **RCC_APB1Periph_TIM13** ((uint32_t)0x00000080)
- #define **RCC_APB1Periph_TIM14** ((uint32_t)0x00000100)
- #define **RCC_APB1Periph_WWDG** ((uint32_t)0x00000800)
- #define **RCC_APB1Periph_SPI2** ((uint32_t)0x00004000)
- #define **RCC_APB1Periph_SPI3** ((uint32_t)0x00008000)
- #define **RCC_APB1Periph_USART2** ((uint32_t)0x00020000)
- #define **RCC_APB1Periph_USART3** ((uint32_t)0x00040000)
- #define **RCC_APB1Periph_UART4** ((uint32_t)0x00080000)
- #define **RCC_APB1Periph_UART5** ((uint32_t)0x00100000)
- #define **RCC_APB1Periph_I2C1** ((uint32_t)0x00200000)
- #define **RCC_APB1Periph_I2C2** ((uint32_t)0x00400000)
- #define **RCC_APB1Periph_I2C3** ((uint32_t)0x00800000)
- #define **RCC_APB1Periph_CAN1** ((uint32_t)0x02000000)
- #define **RCC_APB1Periph_CAN2** ((uint32_t)0x04000000)
- #define **RCC_APB1Periph_PWR** ((uint32_t)0x10000000)
- #define **RCC_APB1Periph_DAC** ((uint32_t)0x20000000)
- #define **IS_RCC_APB1_PERIPH**(PERIPH) ((((PERIPH) & 0xC9013600) == 0x00) && ((PERIPH) != 0x00))

5.62.1 Detailed Description

5.63 RCC_APB2_Peripherals

Macros

- #define **RCC_APB2Periph_TIM1** ((uint32_t)0x00000001)
- #define **RCC_APB2Periph_TIM8** ((uint32_t)0x00000002)
- #define **RCC_APB2Periph_USART1** ((uint32_t)0x00000010)
- #define **RCC_APB2Periph_USART6** ((uint32_t)0x00000020)
- #define **RCC_APB2Periph_ADC** ((uint32_t)0x00000100)
- #define **RCC_APB2Periph_ADC1** ((uint32_t)0x00000100)
- #define **RCC_APB2Periph_ADC2** ((uint32_t)0x00000200)
- #define **RCC_APB2Periph_ADC3** ((uint32_t)0x00000400)
- #define **RCC_APB2Periph_SDIO** ((uint32_t)0x00000800)
- #define **RCC_APB2Periph_SPI1** ((uint32_t)0x00001000)
- #define **RCC_APB2Periph_SYSCFG** ((uint32_t)0x00004000)
- #define **RCC_APB2Periph_TIM9** ((uint32_t)0x00010000)
- #define **RCC_APB2Periph_TIM10** ((uint32_t)0x00020000)
- #define **RCC_APB2Periph_TIM11** ((uint32_t)0x00040000)
- #define **IS_RCC_APB2_PERIPH**(PERIPH) ((((PERIPH) & 0xFFF8A0CC) == 0x00) && ((PERIPH) != 0x00))
- #define **IS_RCC_APB2_RESET_PERIPH**(PERIPH) ((((PERIPH) & 0xFFF8A6CC) == 0x00) && ((PERIPH) != 0x00))

5.63.1 Detailed Description

5.64 RCC_MCO1_Clock_Source_Prescaler

Macros

- #define **RCC_MCO1Source_HSI** ((uint32_t)0x00000000)
- #define **RCC_MCO1Source_LSE** ((uint32_t)0x00200000)
- #define **RCC_MCO1Source_HSE** ((uint32_t)0x00400000)
- #define **RCC_MCO1Source_PLLCLK** ((uint32_t)0x00600000)
- #define **RCC_MCO1Div_1** ((uint32_t)0x00000000)
- #define **RCC_MCO1Div_2** ((uint32_t)0x04000000)
- #define **RCC_MCO1Div_3** ((uint32_t)0x05000000)
- #define **RCC_MCO1Div_4** ((uint32_t)0x06000000)
- #define **RCC_MCO1Div_5** ((uint32_t)0x07000000)
- #define IS_RCC_MCO1SOURCE(SOURCE)
- #define IS_RCC_MCO1DIV(DIV)

5.64.1 Detailed Description

5.64.2 Macro Definition Documentation

5.64.2.1 IS_RCC_MCO1DIV

```
#define IS_RCC_MCO1DIV(
              DIV )
```

Value:

```
((DIV) == RCC_MCO1Div_1) || ((DIV) == RCC_MCO1Div_2) || \
((DIV) == RCC_MCO1Div_3) || ((DIV) == RCC_MCO1Div_4) || \
((DIV) == RCC_MCO1Div_5))
```

5.64.2.2 IS_RCC_MCO1SOURCE

```
#define IS_RCC_MCO1SOURCE(
              SOURCE )
```

Value:

```
(((SOURCE) == RCC_MCO1Source_HSI) || ((SOURCE) == RCC_MCO1Source_LSE) ||
    \
((SOURCE) == RCC_MCO1Source_HSE) || ((SOURCE) == RCC_MCO1Source_PLLCLK))
```

5.65 RCC_MCO2_Clock_Source_Prescaler

Macros

- #define **RCC_MCO2Source_SYSCLK** ((uint32_t)0x00000000)
- #define **RCC_MCO2Source_PLLI2SCLK** ((uint32_t)0x40000000)
- #define **RCC_MCO2Source_HSE** ((uint32_t)0x80000000)
- #define **RCC_MCO2Source_PLLCLK** ((uint32_t)0xC0000000)
- #define **RCC_MCO2Div_1** ((uint32_t)0x00000000)
- #define **RCC_MCO2Div_2** ((uint32_t)0x20000000)
- #define **RCC_MCO2Div_3** ((uint32_t)0x28000000)
- #define **RCC_MCO2Div_4** ((uint32_t)0x30000000)
- #define **RCC_MCO2Div_5** ((uint32_t)0x38000000)
- #define IS_RCC_MCO2SOURCE(SOURCE)
- #define IS_RCC_MCO2DIV(DIV)

5.65.1 Detailed Description

5.65.2 Macro Definition Documentation

5.65.2.1 IS_RCC_MCO2DIV

```
#define IS_RCC_MCO2DIV(
              DIV )
```

Value:

```
                          (((DIV) == RCC_MCO2Div_1) || ((DIV) == RCC_MCO2Div_2) || \
                          ((DIV) == RCC_MCO2Div_3) || ((DIV) == RCC_MCO2Div_4) || \
                          ((DIV) == RCC_MCO2Div_5))
```

5.65.2.2 IS_RCC_MCO2SOURCE

```
#define IS_RCC_MCO2SOURCE(
              SOURCE )
```

Value:

```
                                 (((SOURCE) == RCC_MCO2Source_SYSCLK) || ((SOURCE) ==
    RCC_MCO2Source_PLLI2SCLK)|| \
                                 ((SOURCE) == RCC_MCO2Source_HSE) || ((SOURCE) == RCC_MCO2Source_PLLCLK))
```

5.66 RCC_Flag

Macros

- #define **RCC_FLAG_HSIRDY** ((uint8_t)0x21)
- #define **RCC_FLAG_HSERDY** ((uint8_t)0x31)
- #define **RCC_FLAG_PLLRDY** ((uint8_t)0x39)
- #define **RCC_FLAG_PLLI2SRDY** ((uint8_t)0x3B)
- #define **RCC_FLAG_LSERDY** ((uint8_t)0x41)
- #define **RCC_FLAG_LSIRDY** ((uint8_t)0x61)
- #define **RCC_FLAG_BORRST** ((uint8_t)0x79)
- #define **RCC_FLAG_PINRST** ((uint8_t)0x7A)
- #define **RCC_FLAG_PORRST** ((uint8_t)0x7B)
- #define **RCC_FLAG_SFTRST** ((uint8_t)0x7C)
- #define **RCC_FLAG_IWDGRST** ((uint8_t)0x7D)
- #define **RCC_FLAG_WWDGRST** ((uint8_t)0x7E)
- #define **RCC_FLAG_LPWRRST** ((uint8_t)0x7F)
- #define IS_RCC_FLAG(FLAG)
- #define **IS_RCC_CALIBRATION_VALUE**(VALUE) ((VALUE) <= 0x1F)

5.66.1 Detailed Description

5.66.2 Macro Definition Documentation

5.66.2.1 IS_RCC_FLAG

```
#define IS_RCC_FLAG(
              FLAG )
```

Value:

```
(((FLAG) == RCC_FLAG_HSIRDY) || ((FLAG) == RCC_FLAG_HSERDY) || \
((FLAG) == RCC_FLAG_PLLRDY) || ((FLAG) == RCC_FLAG_LSERDY) || \
((FLAG) == RCC_FLAG_LSIRDY) || ((FLAG) == RCC_FLAG_BORRST) || \
((FLAG) == RCC_FLAG_PINRST) || ((FLAG) == RCC_FLAG_PORRST) || \
((FLAG) == RCC_FLAG_SFTRST) || ((FLAG) == RCC_FLAG_IWDGRST)|| \
((FLAG) == RCC_FLAG_WWDGRST)|| ((FLAG) == RCC_FLAG_LPWRRST)|| \
((FLAG) == RCC_FLAG_PLLI2SRDY))
```

5.67 TIM_Exported_constants

Modules

- TIM_Output_Compare_and_PWM_modes
- TIM_One_Pulse_Mode
- TIM_Channel
- TIM_Clock_Division_CKD
- TIM_Counter_Mode
- TIM_Output_Compare_Polarity
- TIM_Output_Compare_N_Polarity

- TIM_Output_Compare_State
- TIM_Output_Compare_N_State
- TIM_Capture_Compare_State
- TIM_Capture_Compare_N_State
- TIM_Break_Input_enable_disable
- TIM_Break_Polarity
- TIM_AOE_Bit_Set_Reset
- TIM_Lock_level
- TIM_OSSI_Off_State_Selection_for_Idle_mode_state
- TIM_OSSR_Off_State_Selection_for_Run_mode_state
- TIM_Output_Compare_Idle_State
- TIM_Output_Compare_N_Idle_State
- TIM_Input_Capture_Polarity
- TIM_Input_Capture_Selection
- TIM_Input_Capture_Prescaler
- TIM_interrupt_sources
- TIM_DMA_Base_address
- TIM_DMA_Burst_Length
- TIM_DMA_sources
- TIM_External_Trigger_Prescaler
- TIM_Internal_Trigger_Selection
- TIM_TIx_External_Clock_Source
- TIM_External_Trigger_Polarity
- TIM_Prescaler_Reload_Mode
- TIM_Forced_Action
- TIM_Encoder_Mode
- TIM_Event_Source
- TIM_Update_Source
- TIM_Output_Compare_Preload_State
- TIM_Output_Compare_Fast_State
- TIM_Output_Compare_Clear_State
- TIM_Trigger_Output_Source
- TIM_Slave_Mode
- TIM_Master_Slave_Mode
- TIM_Remap
- TIM_Flags
- TIM_Input_Capture_Filer_Value
- TIM_External_Trigger_Filter
- TIM_Legacy

Macros

- #define IS_TIM_ALL_PERIPH(PERIPH)
- #define IS_TIM_LIST1_PERIPH(PERIPH)
- #define IS_TIM_LIST2_PERIPH(PERIPH)
- #define IS_TIM_LIST3_PERIPH(PERIPH)
- #define IS_TIM_LIST4_PERIPH(PERIPH)
- #define IS_TIM_LIST5_PERIPH(PERIPH)
- #define IS_TIM_LIST6_PERIPH(TIMx)

5.67.1 Detailed Description

5.67.2 Macro Definition Documentation

5.67.2.1 IS_TIM_ALL_PERIPH

```
#define IS_TIM_ALL_PERIPH(
              PERIPH )
```

Value:

```
(((PERIPH) == TIM1) || \
((PERIPH) == TIM2) || \
((PERIPH) == TIM3) || \
((PERIPH) == TIM4) || \
((PERIPH) == TIM5) || \
((PERIPH) == TIM6) || \
((PERIPH) == TIM7) || \
((PERIPH) == TIM8) || \
((PERIPH) == TIM9) || \
((PERIPH) == TIM10) || \
((PERIPH) == TIM11) || \
((PERIPH) == TIM12) || \
(((PERIPH) == TIM13) || \
((PERIPH) == TIM14)))
```

5.67.2.2 IS_TIM_LIST1_PERIPH

```
#define IS_TIM_LIST1_PERIPH(
              PERIPH )
```

Value:

```
(((PERIPH) == TIM1) || \
((PERIPH) == TIM2) || \
((PERIPH) == TIM3) || \
((PERIPH) == TIM4) || \
((PERIPH) == TIM5) || \
((PERIPH) == TIM8) || \
((PERIPH) == TIM9) || \
((PERIPH) == TIM10) || \
((PERIPH) == TIM11) || \
((PERIPH) == TIM12) || \
((PERIPH) == TIM13) || \
((PERIPH) == TIM14))
```

5.67.2.3 IS_TIM_LIST2_PERIPH

```
#define IS_TIM_LIST2_PERIPH(
              PERIPH )
```

Value:

```
(((PERIPH) == TIM1) || \
((PERIPH) == TIM2) || \
((PERIPH) == TIM3) || \
((PERIPH) == TIM4) || \
((PERIPH) == TIM5) || \
((PERIPH) == TIM8) || \
((PERIPH) == TIM9) || \
((PERIPH) == TIM12))
```

5.67.2.4 IS_TIM_LIST3_PERIPH

```
#define IS_TIM_LIST3_PERIPH(
              PERIPH )
```

Value:

```
                                  (((PERIPH) == TIM1) || \
                                  ((PERIPH) == TIM2) || \
                                  ((PERIPH) == TIM3) || \
                                  ((PERIPH) == TIM4) || \
                                  ((PERIPH) == TIM5) || \
                                  ((PERIPH) == TIM8))
```

5.67.2.5 IS_TIM_LIST4_PERIPH

```
#define IS_TIM_LIST4_PERIPH(
              PERIPH )
```

Value:

```
                                  (((PERIPH) == TIM1) || \
                                  ((PERIPH) == TIM8))
```

5.67.2.6 IS_TIM_LIST5_PERIPH

```
#define IS_TIM_LIST5_PERIPH(
              PERIPH )
```

Value:

```
                                  (((PERIPH) == TIM1) || \
                                  ((PERIPH) == TIM2) || \
                                  ((PERIPH) == TIM3) || \
                                  ((PERIPH) == TIM4) || \
                                  ((PERIPH) == TIM5) || \
                                  ((PERIPH) == TIM6) || \
                                  ((PERIPH) == TIM7) || \
                                  ((PERIPH) == TIM8))
```

5.67.2.7 IS_TIM_LIST6_PERIPH

```
#define IS_TIM_LIST6_PERIPH(
              TIMx )
```

Value:

```
                                (((TIMx) == TIM2) || \
                                ((TIMx) == TIM5) || \
                                ((TIMx) == TIM11))
```

5.68 TIM_Output_Compare_and_PWM_modes

Macros

- #define **TIM_OCMode_Timing** ((uint16_t)0x0000)
- #define **TIM_OCMode_Active** ((uint16_t)0x0010)
- #define **TIM_OCMode_Inactive** ((uint16_t)0x0020)
- #define **TIM_OCMode_Toggle** ((uint16_t)0x0030)
- #define **TIM_OCMode_PWM1** ((uint16_t)0x0060)
- #define **TIM_OCMode_PWM2** ((uint16_t)0x0070)
- #define IS_TIM_OC_MODE(MODE)
- #define IS_TIM_OCM(MODE)

5.68.1 Detailed Description

5.68.2 Macro Definition Documentation

5.68.2.1 IS_TIM_OC_MODE

```
#define IS_TIM_OC_MODE(
              MODE )
```

Value:

```
(((MODE) == TIM_OCMode_Timing) || \
((MODE) == TIM_OCMode_Active) || \
((MODE) == TIM_OCMode_Inactive) || \
((MODE) == TIM_OCMode_Toggle)|| \
((MODE) == TIM_OCMode_PWM1) || \
((MODE) == TIM_OCMode_PWM2))
```

5.68.2.2 IS_TIM_OCM

```
#define IS_TIM_OCM(
              MODE )
```

Value:

```
(((MODE) == TIM_OCMode_Timing) || \
((MODE) == TIM_OCMode_Active) || \
((MODE) == TIM_OCMode_Inactive) || \
((MODE) == TIM_OCMode_Toggle)|| \
((MODE) == TIM_OCMode_PWM1) || \
((MODE) == TIM_OCMode_PWM2) ||    \
((MODE) == TIM_ForcedAction_Active) || \
((MODE) == TIM_ForcedAction_InActive))
```

5.69 TIM_One_Pulse_Mode

Macros

- #define **TIM_OPMode_Single** ((uint16_t)0x0008)
- #define **TIM_OPMode_Repetitive** ((uint16_t)0x0000)
- #define IS_TIM_OPM_MODE(MODE)

5.69.1 Detailed Description

5.69.2 Macro Definition Documentation

5.69.2.1 IS_TIM_OPM_MODE

```
#define IS_TIM_OPM_MODE(
              MODE )
```

Value:

```
((MODE) == TIM_OPMode_Single) || \
((MODE) == TIM_OPMode_Repetitive))
```

5.70 TIM_Channel

Macros

- #define **TIM_Channel_1** ((uint16_t)0x0000)
- #define **TIM_Channel_2** ((uint16_t)0x0004)
- #define **TIM_Channel_3** ((uint16_t)0x0008)
- #define **TIM_Channel_4** ((uint16_t)0x000C)
- #define IS_TIM_CHANNEL(CHANNEL)
- #define IS_TIM_PWMI_CHANNEL(CHANNEL)
- #define IS_TIM_COMPLEMENTARY_CHANNEL(CHANNEL)

5.70.1 Detailed Description

5.70.2 Macro Definition Documentation

5.70.2.1 IS_TIM_CHANNEL

```
#define IS_TIM_CHANNEL(
              CHANNEL )
```

Value:

```
(((CHANNEL) == TIM_Channel_1) || \
((CHANNEL) == TIM_Channel_2) || \
((CHANNEL) == TIM_Channel_3) || \
((CHANNEL) == TIM_Channel_4))
```

5.70.2.2 IS_TIM_COMPLEMENTARY_CHANNEL

```
#define IS_TIM_COMPLEMENTARY_CHANNEL(
              CHANNEL )
```

Value:

```
(((CHANNEL) == TIM_Channel_1) || \
((CHANNEL) == TIM_Channel_2) || \
((CHANNEL) == TIM_Channel_3))
```

5.70.2.3 IS_TIM_PWMI_CHANNEL

```
#define IS_TIM_PWMI_CHANNEL(
              CHANNEL )
```

Value:

```
(((CHANNEL) == TIM_Channel_1) || \
((CHANNEL) == TIM_Channel_2))
```

5.71 TIM_Clock_Division_CKD

Macros

- #define **TIM_CKD_DIV1** ((uint16_t)0x0000)
- #define **TIM_CKD_DIV2** ((uint16_t)0x0100)
- #define **TIM_CKD_DIV4** ((uint16_t)0x0200)
- #define IS_TIM_CKD_DIV(DIV)

5.71.1 Detailed Description

5.71.2 Macro Definition Documentation

5.71.2.1 IS_TIM_CKD_DIV

```
#define IS_TIM_CKD_DIV(
              DIV )
```

Value:

```
(((DIV) == TIM_CKD_DIV1) || \
((DIV) == TIM_CKD_DIV2) || \
((DIV) == TIM_CKD_DIV4))
```

5.72 TIM_Counter_Mode

Macros

- #define **TIM_CounterMode_Up** ((uint16_t)0x0000)
- #define **TIM_CounterMode_Down** ((uint16_t)0x0010)
- #define **TIM_CounterMode_CenterAligned1** ((uint16_t)0x0020)
- #define **TIM_CounterMode_CenterAligned2** ((uint16_t)0x0040)
- #define **TIM_CounterMode_CenterAligned3** ((uint16_t)0x0060)
- #define IS_TIM_COUNTER_MODE(MODE)

5.72.1 Detailed Description

5.72.2 Macro Definition Documentation

5.72.2.1 IS_TIM_COUNTER_MODE

```
#define IS_TIM_COUNTER_MODE(
              MODE )
```

Value:

```
(((MODE) == TIM_CounterMode_Up) ||  \
((MODE) == TIM_CounterMode_Down) || \
((MODE) == TIM_CounterMode_CenterAligned1) || \
((MODE) == TIM_CounterMode_CenterAligned2) || \
((MODE) == TIM_CounterMode_CenterAligned3))
```

5.73 TIM_Output_Compare_Polarity

Macros

- #define **TIM_OCPolarity_High** ((uint16_t)0x0000)
- #define **TIM_OCPolarity_Low** ((uint16_t)0x0002)
- #define IS_TIM_OC_POLARITY(POLARITY)

5.73.1 Detailed Description

5.73.2 Macro Definition Documentation

5.73.2.1 IS_TIM_OC_POLARITY

```
#define IS_TIM_OC_POLARITY(
              POLARITY )
```

Value:

```
(((POLARITY) == TIM_OCPolarity_High) || \
((POLARITY) == TIM_OCPolarity_Low))
```

5.74 TIM_Output_Compare_N_Polarity

Macros

- #define **TIM_OCNPolarity_High** ((uint16_t)0x0000)
- #define **TIM_OCNPolarity_Low** ((uint16_t)0x0008)
- #define IS_TIM_OCN_POLARITY(POLARITY)

5.74.1 Detailed Description

5.74.2 Macro Definition Documentation

5.74.2.1 IS_TIM_OCN_POLARITY

```
#define IS_TIM_OCN_POLARITY(
              POLARITY )
```

Value:

```
(((POLARITY) == TIM_OCNPolarity_High) || \
((POLARITY) == TIM_OCNPolarity_Low))
```

5.75 TIM_Output_Compare_State

Macros

- #define **TIM_OutputState_Disable** ((uint16_t)0x0000)
- #define **TIM_OutputState_Enable** ((uint16_t)0x0001)
- #define IS_TIM_OUTPUT_STATE(STATE)

5.75.1 Detailed Description

5.75.2 Macro Definition Documentation

5.75.2.1 IS_TIM_OUTPUT_STATE

```
#define IS_TIM_OUTPUT_STATE(
              STATE )
```

Value:

```
(((STATE) == TIM_OutputState_Disable) || \
((STATE) == TIM_OutputState_Enable))
```

5.76 TIM_Output_Compare_N_State

Macros

- #define **TIM_OutputNState_Disable** ((uint16_t)0x0000)
- #define **TIM_OutputNState_Enable** ((uint16_t)0x0004)
- #define IS_TIM_OUTPUTN_STATE(STATE)

5.76.1 Detailed Description

5.76.2 Macro Definition Documentation

5.76.2.1 IS_TIM_OUTPUTN_STATE

```
#define IS_TIM_OUTPUTN_STATE(
              STATE )
```

Value:

```
(((STATE) == TIM_OutputNState_Disable) || \
((STATE) == TIM_OutputNState_Enable))
```

5.77 TIM_Capture_Compare_State

Macros

- #define **TIM_CCx_Enable** ((uint16_t)0x0001)
- #define **TIM_CCx_Disable** ((uint16_t)0x0000)
- #define IS_TIM_CCX(CCX)

5.77.1 Detailed Description

5.77.2 Macro Definition Documentation

5.77.2.1 IS_TIM_CCX

```
#define IS_TIM_CCX(
              CCX )
```

Value:

```
(((CCX) == TIM_CCx_Enable) || \
((CCX) == TIM_CCx_Disable))
```

5.78 TIM_Capture_Compare_N_State

Macros

- #define **TIM_CCxN_Enable** ((uint16_t)0x0004)
- #define **TIM_CCxN_Disable** ((uint16_t)0x0000)
- #define IS_TIM_CCXN(CCXN)

5.78.1 Detailed Description

5.78.2 Macro Definition Documentation

5.78.2.1 IS_TIM_CCXN

```
#define IS_TIM_CCXN(
              CCXN )
```

Value:

```
                        (((CCXN) == TIM_CCxN_Enable) || \
                        ((CCXN) == TIM_CCxN_Disable))
```

5.79 TIM_Break_Input_enable_disable

Macros

- #define **TIM_Break_Enable** ((uint16_t)0x1000)
- #define **TIM_Break_Disable** ((uint16_t)0x0000)
- #define IS_TIM_BREAK_STATE(STATE)

5.79.1 Detailed Description

5.79.2 Macro Definition Documentation

5.79.2.1 IS_TIM_BREAK_STATE

```
#define IS_TIM_BREAK_STATE(
              STATE )
```

Value:

```
                                ((STATE) == TIM_Break_Enable) || \
                                ((STATE) == TIM_Break_Disable))
```

5.80 TIM_Break_Polarity

Macros

- #define **TIM_BreakPolarity_Low** ((uint16_t)0x0000)
- #define **TIM_BreakPolarity_High** ((uint16_t)0x2000)
- #define IS_TIM_BREAK_POLARITY(POLARITY)

5.80.1 Detailed Description

5.80.2 Macro Definition Documentation

5.80.2.1 IS_TIM_BREAK_POLARITY

```
#define IS_TIM_BREAK_POLARITY(
              POLARITY )
```

Value:

```
(((POLARITY) == TIM_BreakPolarity_Low) || \
((POLARITY) == TIM_BreakPolarity_High))
```

5.81 TIM_AOE_Bit_Set_Reset

Macros

- #define **TIM_AutomaticOutput_Enable** ((uint16_t)0x4000)
- #define **TIM_AutomaticOutput_Disable** ((uint16_t)0x0000)
- #define IS_TIM_AUTOMATIC_OUTPUT_STATE(STATE)

5.81.1 Detailed Description

5.81.2 Macro Definition Documentation

5.81.2.1 IS_TIM_AUTOMATIC_OUTPUT_STATE

```
#define IS_TIM_AUTOMATIC_OUTPUT_STATE(
              STATE )
```

Value:

```
(((STATE) == TIM_AutomaticOutput_Enable) || \
((STATE) == TIM_AutomaticOutput_Disable))
```

5.82 TIM_Lock_level

Macros

- #define **TIM_LOCKLevel_OFF** ((uint16_t)0x0000)
- #define **TIM_LOCKLevel_1** ((uint16_t)0x0100)
- #define **TIM_LOCKLevel_2** ((uint16_t)0x0200)
- #define **TIM_LOCKLevel_3** ((uint16_t)0x0300)
- #define IS_TIM_LOCK_LEVEL(LEVEL)

5.82.1 Detailed Description

5.82.2 Macro Definition Documentation

5.82.2.1 IS_TIM_LOCK_LEVEL

```
#define IS_TIM_LOCK_LEVEL(
              LEVEL )
```

Value:

```
(((LEVEL) == TIM_LOCKLevel_OFF) || \
((LEVEL) == TIM_LOCKLevel_1) || \
((LEVEL) == TIM_LOCKLevel_2) || \
((LEVEL) == TIM_LOCKLevel_3))
```

5.83 TIM_OSSI_Off_State_Selection_for_Idle_mode_state

Macros

- #define **TIM_OSSIState_Enable** ((uint16_t)0x0400)
- #define **TIM_OSSIState_Disable** ((uint16_t)0x0000)
- #define IS_TIM_OSSI_STATE(STATE)

5.83.1 Detailed Description

5.83.2 Macro Definition Documentation

5.83.2.1 IS_TIM_OSSI_STATE

```
#define IS_TIM_OSSI_STATE(
              STATE )
```

Value:

```
(((STATE) == TIM_OSSIState_Enable) || \
((STATE) == TIM_OSSIState_Disable))
```

5.84 TIM_OSSR_Off_State_Selection_for_Run_mode_state

Macros

- #define **TIM_OSSRState_Enable** ((uint16_t)0x0800)
- #define **TIM_OSSRState_Disable** ((uint16_t)0x0000)
- #define IS_TIM_OSSR_STATE(STATE)

5.84.1 Detailed Description

5.84.2 Macro Definition Documentation

5.84.2.1 IS_TIM_OSSR_STATE

```
#define IS_TIM_OSSR_STATE(
              STATE )
```

Value:

```
                                ((STATE) == TIM_OSSRState_Enable) || \
                                ((STATE) == TIM_OSSRState_Disable))
```

5.85 TIM_Output_Compare_Idle_State

Macros

- #define **TIM_OCIdleState_Set** ((uint16_t)0x0100)
- #define **TIM_OCIdleState_Reset** ((uint16_t)0x0000)
- #define IS_TIM_OCIDLE_STATE(STATE)

5.85.1 Detailed Description

5.85.2 Macro Definition Documentation

5.85.2.1 IS_TIM_OCIDLE_STATE

```
#define IS_TIM_OCIDLE_STATE(
              STATE )
```

Value:

```
                                ((STATE) == TIM_OCIdleState_Set) || \
                                ((STATE) == TIM_OCIdleState_Reset))
```

5.86 TIM_Output_Compare_N_Idle_State

Macros

- #define **TIM_OCNIdleState_Set** ((uint16_t)0x0200)
- #define **TIM_OCNIdleState_Reset** ((uint16_t)0x0000)
- #define IS_TIM_OCNIDLE_STATE(STATE)

5.86.1 Detailed Description

5.86.2 Macro Definition Documentation

5.86.2.1 IS_TIM_OCNIDLE_STATE

```
#define IS_TIM_OCNIDLE_STATE(
              STATE )
```

Value:

```
(((STATE) == TIM_OCNIdleState_Set) || \
((STATE) == TIM_OCNIdleState_Reset))
```

5.87 TIM_Input_Capture_Polarity

Macros

- #define **TIM_ICPolarity_Rising** ((uint16_t)0x0000)
- #define **TIM_ICPolarity_Falling** ((uint16_t)0x0002)
- #define **TIM_ICPolarity_BothEdge** ((uint16_t)0x000A)
- #define IS_TIM_IC_POLARITY(POLARITY)

5.87.1 Detailed Description

5.87.2 Macro Definition Documentation

5.87.2.1 IS_TIM_IC_POLARITY

```
#define IS_TIM_IC_POLARITY(
              POLARITY )
```

Value:

```
(((POLARITY) == TIM_ICPolarity_Rising) || \
((POLARITY) == TIM_ICPolarity_Falling)|| \
((POLARITY) == TIM_ICPolarity_BothEdge))
```

5.88 TIM_Input_Capture_Selection

Macros

- #define TIM_ICSelection_DirectTI ((uint16_t)0x0001)
- #define TIM_ICSelection_IndirectTI ((uint16_t)0x0002)
- #define TIM_ICSelection_TRC ((uint16_t)0x0003)
- #define IS_TIM_IC_SELECTION(SELECTION)

5.88.1 Detailed Description

5.88.2 Macro Definition Documentation

5.88.2.1 IS_TIM_IC_SELECTION

```
#define IS_TIM_IC_SELECTION(
              SELECTION )
```

Value:

```
(((SELECTION) == TIM_ICSelection_DirectTI) || \
((SELECTION) == TIM_ICSelection_IndirectTI) || \
((SELECTION) == TIM_ICSelection_TRC))
```

5.88.2.2 TIM_ICSelection_DirectTI

```
#define TIM_ICSelection_DirectTI ((uint16_t)0x0001)
```

TIM Input 1, 2, 3 or 4 is selected to be connected to IC1, IC2, IC3 or IC4, respectively

5.88.2.3 TIM_ICSelection_IndirectTI

```
#define TIM_ICSelection_IndirectTI ((uint16_t)0x0002)
```

TIM Input 1, 2, 3 or 4 is selected to be connected to IC2, IC1, IC4 or IC3, respectively.

5.88.2.4 TIM_ICSelection_TRC

```
#define TIM_ICSelection_TRC ((uint16_t)0x0003)
```

TIM Input 1, 2, 3 or 4 is selected to be connected to TRC.

5.89 TIM_Input_Capture_Prescaler

Macros

- #define TIM_ICPSC_DIV1 ((uint16_t)0x0000)
- #define TIM_ICPSC_DIV2 ((uint16_t)0x0004)
- #define TIM_ICPSC_DIV4 ((uint16_t)0x0008)
- #define TIM_ICPSC_DIV8 ((uint16_t)0x000C)
- #define IS_TIM_IC_PRESCALER(PRESCALER)

5.89.1 Detailed Description

5.89.2 Macro Definition Documentation

5.89.2.1 IS_TIM_IC_PRESCALER

```
#define IS_TIM_IC_PRESCALER(
              PRESCALER )
```

Value:

```
(((PRESCALER) == TIM_ICPSC_DIV1) || \
((PRESCALER) == TIM_ICPSC_DIV2) || \
((PRESCALER) == TIM_ICPSC_DIV4) || \
((PRESCALER) == TIM_ICPSC_DIV8))
```

5.89.2.2 TIM_ICPSC_DIV1

```
#define TIM_ICPSC_DIV1 ((uint16_t)0x0000)
```

Capture performed each time an edge is detected on the capture input.

5.89.2.3 TIM_ICPSC_DIV2

```
#define TIM_ICPSC_DIV2 ((uint16_t)0x0004)
```

Capture performed once every 2 events.

5.89.2.4 TIM_ICPSC_DIV4

```
#define TIM_ICPSC_DIV4 ((uint16_t)0x0008)
```

Capture performed once every 4 events.

5.89.2.5 TIM_ICPSC_DIV8

```
#define TIM_ICPSC_DIV8 ((uint16_t)0x000C)
```

Capture performed once every 8 events.

5.90 TIM_interrupt_sources

Macros

- #define **TIM_IT_Update** ((uint16_t)0x0001)
- #define **TIM_IT_CC1** ((uint16_t)0x0002)
- #define **TIM_IT_CC2** ((uint16_t)0x0004)
- #define **TIM_IT_CC3** ((uint16_t)0x0008)
- #define **TIM_IT_CC4** ((uint16_t)0x0010)
- #define **TIM_IT_COM** ((uint16_t)0x0020)
- #define **TIM_IT_Trigger** ((uint16_t)0x0040)
- #define **TIM_IT_Break** ((uint16_t)0x0080)
- #define **IS_TIM_IT**(IT) ((((IT) & (uint16_t)0xFF00) == 0x0000) && ((IT) != 0x0000))
- #define IS_TIM_GET_IT(IT)

5.90.1 Detailed Description

5.90.2 Macro Definition Documentation

5.90.2.1 IS_TIM_GET_IT

```
#define IS_TIM_GET_IT(
              IT )
```

Value:

```
(((IT) == TIM_IT_Update) || \
((IT) == TIM_IT_CC1) || \
((IT) == TIM_IT_CC2) || \
((IT) == TIM_IT_CC3) || \
((IT) == TIM_IT_CC4) || \
((IT) == TIM_IT_COM) || \
((IT) == TIM_IT_Trigger) || \
((IT) == TIM_IT_Break))
```

5.91 TIM_DMA_Base_address

Macros

- #define **TIM_DMABase_CR1** ((uint16_t)0x0000)
- #define **TIM_DMABase_CR2** ((uint16_t)0x0001)
- #define **TIM_DMABase_SMCR** ((uint16_t)0x0002)
- #define **TIM_DMABase_DIER** ((uint16_t)0x0003)
- #define **TIM_DMABase_SR** ((uint16_t)0x0004)
- #define **TIM_DMABase_EGR** ((uint16_t)0x0005)
- #define **TIM_DMABase_CCMR1** ((uint16_t)0x0006)
- #define **TIM_DMABase_CCMR2** ((uint16_t)0x0007)
- #define **TIM_DMABase_CCER** ((uint16_t)0x0008)
- #define **TIM_DMABase_CNT** ((uint16_t)0x0009)
- #define **TIM_DMABase_PSC** ((uint16_t)0x000A)
- #define **TIM_DMABase_ARR** ((uint16_t)0x000B)
- #define **TIM_DMABase_RCR** ((uint16_t)0x000C)
- #define **TIM_DMABase_CCR1** ((uint16_t)0x000D)
- #define **TIM_DMABase_CCR2** ((uint16_t)0x000E)
- #define **TIM_DMABase_CCR3** ((uint16_t)0x000F)
- #define **TIM_DMABase_CCR4** ((uint16_t)0x0010)
- #define **TIM_DMABase_BDTR** ((uint16_t)0x0011)
- #define **TIM_DMABase_DCR** ((uint16_t)0x0012)
- #define **TIM_DMABase_OR** ((uint16_t)0x0013)
- #define IS_TIM_DMA_BASE(BASE)

5.91.1 Detailed Description

5.91.2 Macro Definition Documentation

5.91.2.1 IS_TIM_DMA_BASE

```
#define IS_TIM_DMA_BASE(
              BASE )
```

Value:

```
(((BASE) == TIM_DMABase_CR1) || \
((BASE) == TIM_DMABase_CR2) || \
((BASE) == TIM_DMABase_SMCR) || \
((BASE) == TIM_DMABase_DIER) || \
((BASE) == TIM_DMABase_SR) || \
((BASE) == TIM_DMABase_EGR) || \
((BASE) == TIM_DMABase_CCMR1) || \
((BASE) == TIM_DMABase_CCMR2) || \
((BASE) == TIM_DMABase_CCER) || \
((BASE) == TIM_DMABase_CNT) || \
((BASE) == TIM_DMABase_PSC) || \
((BASE) == TIM_DMABase_ARR) || \
((BASE) == TIM_DMABase_RCR) || \
((BASE) == TIM_DMABase_CCR1) || \
((BASE) == TIM_DMABase_CCR2) || \
((BASE) == TIM_DMABase_CCR3) || \
((BASE) == TIM_DMABase_CCR4) || \
((BASE) == TIM_DMABase_BDTR) || \
((BASE) == TIM_DMABase_DCR) || \
((BASE) == TIM_DMABase_OR))
```

5.92 TIM_DMA_Burst_Length

Macros

- #define **TIM_DMABurstLength_1Transfer** ((uint16_t)0x0000)
- #define **TIM_DMABurstLength_2Transfers** ((uint16_t)0x0100)
- #define **TIM_DMABurstLength_3Transfers** ((uint16_t)0x0200)
- #define **TIM_DMABurstLength_4Transfers** ((uint16_t)0x0300)
- #define **TIM_DMABurstLength_5Transfers** ((uint16_t)0x0400)
- #define **TIM_DMABurstLength_6Transfers** ((uint16_t)0x0500)
- #define **TIM_DMABurstLength_7Transfers** ((uint16_t)0x0600)
- #define **TIM_DMABurstLength_8Transfers** ((uint16_t)0x0700)
- #define **TIM_DMABurstLength_9Transfers** ((uint16_t)0x0800)
- #define **TIM_DMABurstLength_10Transfers** ((uint16_t)0x0900)
- #define **TIM_DMABurstLength_11Transfers** ((uint16_t)0x0A00)
- #define **TIM_DMABurstLength_12Transfers** ((uint16_t)0x0B00)
- #define **TIM_DMABurstLength_13Transfers** ((uint16_t)0x0C00)
- #define **TIM_DMABurstLength_14Transfers** ((uint16_t)0x0D00)
- #define **TIM_DMABurstLength_15Transfers** ((uint16_t)0x0E00)
- #define **TIM_DMABurstLength_16Transfers** ((uint16_t)0x0F00)
- #define **TIM_DMABurstLength_17Transfers** ((uint16_t)0x1000)
- #define **TIM_DMABurstLength_18Transfers** ((uint16_t)0x1100)
- #define IS_TIM_DMA_LENGTH(LENGTH)

5.92.1 Detailed Description

5.92.2 Macro Definition Documentation

5.92.2.1 IS_TIM_DMA_LENGTH

```
#define IS_TIM_DMA_LENGTH(
              LENGTH )
```

Value:

```
(((LENGTH) == TIM_DMABurstLength_1Transfer) || \
((LENGTH) == TIM_DMABurstLength_2Transfers) || \
((LENGTH) == TIM_DMABurstLength_3Transfers) || \
((LENGTH) == TIM_DMABurstLength_4Transfers) || \
((LENGTH) == TIM_DMABurstLength_5Transfers) || \
((LENGTH) == TIM_DMABurstLength_6Transfers) || \
((LENGTH) == TIM_DMABurstLength_7Transfers) || \
((LENGTH) == TIM_DMABurstLength_8Transfers) || \
((LENGTH) == TIM_DMABurstLength_9Transfers) || \
((LENGTH) == TIM_DMABurstLength_10Transfers) || \
((LENGTH) == TIM_DMABurstLength_11Transfers) || \
((LENGTH) == TIM_DMABurstLength_12Transfers) || \
((LENGTH) == TIM_DMABurstLength_13Transfers) || \
((LENGTH) == TIM_DMABurstLength_14Transfers) || \
((LENGTH) == TIM_DMABurstLength_15Transfers) || \
((LENGTH) == TIM_DMABurstLength_16Transfers) || \
((LENGTH) == TIM_DMABurstLength_17Transfers) || \
((LENGTH) == TIM_DMABurstLength_18Transfers))
```

5.93 TIM_DMA_sources

Macros

- #define **TIM_DMA_Update** ((uint16_t)0x0100)
- #define **TIM_DMA_CC1** ((uint16_t)0x0200)
- #define **TIM_DMA_CC2** ((uint16_t)0x0400)
- #define **TIM_DMA_CC3** ((uint16_t)0x0800)
- #define **TIM_DMA_CC4** ((uint16_t)0x1000)
- #define **TIM_DMA_COM** ((uint16_t)0x2000)
- #define **TIM_DMA_Trigger** ((uint16_t)0x4000)
- #define **IS_TIM_DMA_SOURCE**(SOURCE) ((((SOURCE) & (uint16_t)0x80FF) == 0x0000) && ((SOURCE) != 0x0000))

5.93.1 Detailed Description

5.94 TIM_External_Trigger_Prescaler

Macros

- #define **TIM_ExtTRGPSC_OFF** ((uint16_t)0x0000)
- #define **TIM_ExtTRGPSC_DIV2** ((uint16_t)0x1000)
- #define **TIM_ExtTRGPSC_DIV4** ((uint16_t)0x2000)
- #define **TIM_ExtTRGPSC_DIV8** ((uint16_t)0x3000)
- #define IS_TIM_EXT_PRESCALER(PRESCALER)

5.94.1 Detailed Description

5.94.2 Macro Definition Documentation

5.94.2.1 IS_TIM_EXT_PRESCALER

```
#define IS_TIM_EXT_PRESCALER(
              PRESCALER )
```

Value:

```
(((PRESCALER) == TIM_ExtTRGPSC_OFF) || \
((PRESCALER) == TIM_ExtTRGPSC_DIV2) || \
((PRESCALER) == TIM_ExtTRGPSC_DIV4) || \
((PRESCALER) == TIM_ExtTRGPSC_DIV8))
```

5.95 TIM_Internal_Trigger_Selection

Macros

- #define **TIM_TS_ITR0** ((uint16_t)0x0000)
- #define **TIM_TS_ITR1** ((uint16_t)0x0010)
- #define **TIM_TS_ITR2** ((uint16_t)0x0020)
- #define **TIM_TS_ITR3** ((uint16_t)0x0030)
- #define **TIM_TS_TI1F_ED** ((uint16_t)0x0040)
- #define **TIM_TS_TI1FP1** ((uint16_t)0x0050)
- #define **TIM_TS_TI2FP2** ((uint16_t)0x0060)
- #define **TIM_TS_ETRF** ((uint16_t)0x0070)
- #define IS_TIM_TRIGGER_SELECTION(SELECTION)
- #define IS_TIM_INTERNAL_TRIGGER_SELECTION(SELECTION)

5.95.1 Detailed Description

5.95.2 Macro Definition Documentation

5.95.2.1 IS_TIM_INTERNAL_TRIGGER_SELECTION

```
#define IS_TIM_INTERNAL_TRIGGER_SELECTION(
              SELECTION )
```

Value:

```
(((SELECTION) == TIM_TS_ITR0) || \
((SELECTION) == TIM_TS_ITR1) || \
((SELECTION) == TIM_TS_ITR2) || \
((SELECTION) == TIM_TS_ITR3))
```

5.95.2.2 IS_TIM_TRIGGER_SELECTION

```
#define IS_TIM_TRIGGER_SELECTION(
              SELECTION )
```

Value:

```
(((SELECTION) == TIM_TS_ITR0) || \
((SELECTION) == TIM_TS_ITR1) || \
((SELECTION) == TIM_TS_ITR2) || \
((SELECTION) == TIM_TS_ITR3) || \
((SELECTION) == TIM_TS_TI1F_ED) || \
((SELECTION) == TIM_TS_TI1FP1) || \
((SELECTION) == TIM_TS_TI2FP2) || \
((SELECTION) == TIM_TS_ETRF))
```

5.96 TIM_TIx_External_Clock_Source

Macros

- #define **TIM_TIxExternalCLK1Source_TI1** ((uint16_t)0x0050)
- #define **TIM_TIxExternalCLK1Source_TI2** ((uint16_t)0x0060)
- #define **TIM_TIxExternalCLK1Source_TI1ED** ((uint16_t)0x0040)

5.96.1 Detailed Description

5.97 TIM_External_Trigger_Polarity

Macros

- #define **TIM_ExtTRGPolarity_Inverted** ((uint16_t)0x8000)
- #define **TIM_ExtTRGPolarity_NonInverted** ((uint16_t)0x0000)
- #define IS_TIM_EXT_POLARITY(POLARITY)

5.97.1 Detailed Description

5.97.2 Macro Definition Documentation

5.97.2.1 IS_TIM_EXT_POLARITY

```
#define IS_TIM_EXT_POLARITY(
              POLARITY )
```

Value:

```
(((POLARITY) == TIM_ExtTRGPolarity_Inverted) || \
((POLARITY) == TIM_ExtTRGPolarity_NonInverted))
```

5.98 TIM_Prescaler_Reload_Mode

Macros

- #define **TIM_PSCReloadMode_Update** ((uint16_t)0x0000)
- #define **TIM_PSCReloadMode_Immediate** ((uint16_t)0x0001)
- #define IS_TIM_PRESCALER_RELOAD(RELOAD)

5.98.1 Detailed Description

5.98.2 Macro Definition Documentation

5.98.2.1 IS_TIM_PRESCALER_RELOAD

```
#define IS_TIM_PRESCALER_RELOAD(
              RELOAD )
```

Value:

```
(((RELOAD) == TIM_PSCReloadMode_Update) || \
((RELOAD) == TIM_PSCReloadMode_Immediate))
```

5.99 TIM_Forced_Action

Macros

- #define **TIM_ForcedAction_Active** ((uint16_t)0x0050)
- #define **TIM_ForcedAction_InActive** ((uint16_t)0x0040)
- #define IS_TIM_FORCED_ACTION(ACTION)

5.99.1 Detailed Description

5.99.2 Macro Definition Documentation

5.99.2.1 IS_TIM_FORCED_ACTION

```
#define IS_TIM_FORCED_ACTION(
              ACTION )
```

Value:

```
(((ACTION) == TIM_ForcedAction_Active) || \
((ACTION) == TIM_ForcedAction_InActive))
```

5.100 TIM_Encoder_Mode

Macros

- #define **TIM_EncoderMode_TI1** ((uint16_t)0x0001)
- #define **TIM_EncoderMode_TI2** ((uint16_t)0x0002)
- #define **TIM_EncoderMode_TI12** ((uint16_t)0x0003)
- #define IS_TIM_ENCODER_MODE(MODE)

5.100.1 Detailed Description

5.100.2 Macro Definition Documentation

5.100.2.1 IS_TIM_ENCODER_MODE

```
#define IS_TIM_ENCODER_MODE(
              MODE )
```

Value:

```
(((MODE) == TIM_EncoderMode_TI1) || \
((MODE) == TIM_EncoderMode_TI2) || \
((MODE) == TIM_EncoderMode_TI12))
```

5.101 TIM_Event_Source

Macros

- #define **TIM_EventSource_Update** ((uint16_t)0x0001)
- #define **TIM_EventSource_CC1** ((uint16_t)0x0002)
- #define **TIM_EventSource_CC2** ((uint16_t)0x0004)
- #define **TIM_EventSource_CC3** ((uint16_t)0x0008)
- #define **TIM_EventSource_CC4** ((uint16_t)0x0010)
- #define **TIM_EventSource_COM** ((uint16_t)0x0020)
- #define **TIM_EventSource_Trigger** ((uint16_t)0x0040)
- #define **TIM_EventSource_Break** ((uint16_t)0x0080)
- #define **IS_TIM_EVENT_SOURCE**(SOURCE) ((((SOURCE) & (uint16_t)0xFF00) == 0x0000) && ((SOURCE) != 0x0000))

5.101.1 Detailed Description

5.102 TIM_Update_Source

Macros

- #define TIM_UpdateSource_Global ((uint16_t)0x0000)
- #define TIM_UpdateSource_Regular ((uint16_t)0x0001)
- #define IS_TIM_UPDATE_SOURCE(SOURCE)

5.102.1 Detailed Description

5.102.2 Macro Definition Documentation

5.102.2.1 IS_TIM_UPDATE_SOURCE

```
#define IS_TIM_UPDATE_SOURCE(
              SOURCE )
```

Value:

```
(((SOURCE) == TIM_UpdateSource_Global) || \
((SOURCE) == TIM_UpdateSource_Regular))
```

5.102.2.2 TIM_UpdateSource_Global

```
#define TIM_UpdateSource_Global ((uint16_t)0x0000)
```

Source of update is the counter overflow/underflow or the setting of UG bit, or an update generation through the slave mode controller.

5.102.2.3 TIM_UpdateSource_Regular

```
#define TIM_UpdateSource_Regular ((uint16_t)0x0001)
```

Source of update is counter overflow/underflow.

5.103 TIM_Output_Compare_Preload_State

Macros

- #define **TIM_OCPreload_Enable** ((uint16_t)0x0008)
- #define **TIM_OCPreload_Disable** ((uint16_t)0x0000)
- #define IS_TIM_OCPRELOAD_STATE(STATE)

5.103.1 Detailed Description

5.103.2 Macro Definition Documentation

5.103.2.1 IS_TIM_OCPRELOAD_STATE

```
#define IS_TIM_OCPRELOAD_STATE(
              STATE )
```

Value:

```
(((STATE) == TIM_OCPreload_Enable) || \
((STATE) == TIM_OCPreload_Disable))
```

5.104 TIM_Output_Compare_Fast_State

Macros

- #define **TIM_OCFast_Enable** ((uint16_t)0x0004)
- #define **TIM_OCFast_Disable** ((uint16_t)0x0000)
- #define IS_TIM_OCFAST_STATE(STATE)

5.104.1 Detailed Description

5.104.2 Macro Definition Documentation

5.104.2.1 IS_TIM_OCFAST_STATE

```
#define IS_TIM_OCFAST_STATE(
              STATE )
```

Value:

```
(((STATE) == TIM_OCFast_Enable) || \
((STATE) == TIM_OCFast_Disable))
```

5.105 TIM_Output_Compare_Clear_State

Macros

- #define **TIM_OCClear_Enable** ((uint16_t)0x0080)
- #define **TIM_OCClear_Disable** ((uint16_t)0x0000)
- #define IS_TIM_OCCLEAR_STATE(STATE)

5.105.1 Detailed Description

5.105.2 Macro Definition Documentation

5.105.2.1 IS_TIM_OCCLEAR_STATE

```
#define IS_TIM_OCCLEAR_STATE(
              STATE )
```

Value:

```
(((STATE) == TIM_OCClear_Enable) || \
((STATE) == TIM_OCClear_Disable))
```

5.106 TIM_Trigger_Output_Source

Macros

- #define **TIM_TRGOSource_Reset** ((uint16_t)0x0000)
- #define **TIM_TRGOSource_Enable** ((uint16_t)0x0010)
- #define **TIM_TRGOSource_Update** ((uint16_t)0x0020)
- #define **TIM_TRGOSource_OC1** ((uint16_t)0x0030)
- #define **TIM_TRGOSource_OC1Ref** ((uint16_t)0x0040)
- #define **TIM_TRGOSource_OC2Ref** ((uint16_t)0x0050)
- #define **TIM_TRGOSource_OC3Ref** ((uint16_t)0x0060)
- #define **TIM_TRGOSource_OC4Ref** ((uint16_t)0x0070)
- #define IS_TIM_TRGO_SOURCE(SOURCE)

5.106.1 Detailed Description

5.106.2 Macro Definition Documentation

5.106.2.1 IS_TIM_TRGO_SOURCE

```
#define IS_TIM_TRGO_SOURCE(
              SOURCE )
```

Value:

```
(((SOURCE) == TIM_TRGOSource_Reset) || \
((SOURCE) == TIM_TRGOSource_Enable) || \
((SOURCE) == TIM_TRGOSource_Update) || \
((SOURCE) == TIM_TRGOSource_OC1) || \
((SOURCE) == TIM_TRGOSource_OC1Ref) || \
((SOURCE) == TIM_TRGOSource_OC2Ref) || \
((SOURCE) == TIM_TRGOSource_OC3Ref) || \
((SOURCE) == TIM_TRGOSource_OC4Ref))
```

5.107 TIM_Slave_Mode

Macros

- #define **TIM_SlaveMode_Reset** ((uint16_t)0x0004)
- #define **TIM_SlaveMode_Gated** ((uint16_t)0x0005)
- #define **TIM_SlaveMode_Trigger** ((uint16_t)0x0006)
- #define **TIM_SlaveMode_External1** ((uint16_t)0x0007)
- #define IS_TIM_SLAVE_MODE(MODE)

5.107.1 Detailed Description

5.107.2 Macro Definition Documentation

5.107.2.1 IS_TIM_SLAVE_MODE

```
#define IS_TIM_SLAVE_MODE(
              MODE )
```

Value:

```
(((MODE) == TIM_SlaveMode_Reset) || \
((MODE) == TIM_SlaveMode_Gated) || \
((MODE) == TIM_SlaveMode_Trigger) || \
((MODE) == TIM_SlaveMode_External1))
```

5.108 TIM_Master_Slave_Mode

Macros

- #define **TIM_MasterSlaveMode_Enable** ((uint16_t)0x0080)
- #define **TIM_MasterSlaveMode_Disable** ((uint16_t)0x0000)
- #define IS_TIM_MSM_STATE(STATE)

5.108.1 Detailed Description

5.108.2 Macro Definition Documentation

5.108.2.1 IS_TIM_MSM_STATE

```
#define IS_TIM_MSM_STATE(
              STATE )
```

Value:

```
(((STATE) == TIM_MasterSlaveMode_Enable) || \
((STATE) == TIM_MasterSlaveMode_Disable))
```

5.109 TIM_Remap

Macros

- #define **TIM2_TIM8_TRGO** ((uint16_t)0x0000)
- #define **TIM2_ETH_PTP** ((uint16_t)0x0400)
- #define **TIM2_USBFS_SOF** ((uint16_t)0x0800)
- #define **TIM2_USBHS_SOF** ((uint16_t)0x0C00)
- #define **TIM5_GPIO** ((uint16_t)0x0000)
- #define **TIM5_LSI** ((uint16_t)0x0040)
- #define **TIM5_LSE** ((uint16_t)0x0080)
- #define **TIM5_RTC** ((uint16_t)0x00C0)
- #define **TIM11_GPIO** ((uint16_t)0x0000)
- #define **TIM11_HSE** ((uint16_t)0x0002)
- #define IS_TIM_REMAP(TIM_REMAP)

5.109.1 Detailed Description

5.109.2 Macro Definition Documentation

5.109.2.1 IS_TIM_REMAP

```
#define IS_TIM_REMAP(
              TIM_REMAP )
```

Value:

```
(((TIM_REMAP) == TIM2_TIM8_TRGO)||\
((TIM_REMAP) == TIM2_ETH_PTP)||\
((TIM_REMAP) == TIM2_USBFS_SOF)||\
((TIM_REMAP) == TIM2_USBHS_SOF)||\
((TIM_REMAP) == TIM5_GPIO)||\
((TIM_REMAP) == TIM5_LSI)||\
((TIM_REMAP) == TIM5_LSE)||\
((TIM_REMAP) == TIM5_RTC)||\
((TIM_REMAP) == TIM11_GPIO)||\
((TIM_REMAP) == TIM11_HSE))
```

5.110 TIM_Flags

Macros

- #define **TIM_FLAG_Update** ((uint16_t)0x0001)
- #define **TIM_FLAG_CC1** ((uint16_t)0x0002)
- #define **TIM_FLAG_CC2** ((uint16_t)0x0004)
- #define **TIM_FLAG_CC3** ((uint16_t)0x0008)
- #define **TIM_FLAG_CC4** ((uint16_t)0x0010)
- #define **TIM_FLAG_COM** ((uint16_t)0x0020)
- #define **TIM_FLAG_Trigger** ((uint16_t)0x0040)
- #define **TIM_FLAG_Break** ((uint16_t)0x0080)
- #define **TIM_FLAG_CC1OF** ((uint16_t)0x0200)
- #define **TIM_FLAG_CC2OF** ((uint16_t)0x0400)
- #define **TIM_FLAG_CC3OF** ((uint16_t)0x0800)
- #define **TIM_FLAG_CC4OF** ((uint16_t)0x1000)
- #define IS_TIM_GET_FLAG(FLAG)

5.110.1 Detailed Description

5.110.2 Macro Definition Documentation

5.110.2.1 IS_TIM_GET_FLAG

```
#define IS_TIM_GET_FLAG(
              FLAG )
```

Value:

```
(((FLAG) == TIM_FLAG_Update) || \
((FLAG) == TIM_FLAG_CC1) || \
((FLAG) == TIM_FLAG_CC2) || \
((FLAG) == TIM_FLAG_CC3) || \
((FLAG) == TIM_FLAG_CC4) || \
((FLAG) == TIM_FLAG_COM) || \
((FLAG) == TIM_FLAG_Trigger) || \
((FLAG) == TIM_FLAG_Break) || \
((FLAG) == TIM_FLAG_CC1OF) || \
((FLAG) == TIM_FLAG_CC2OF) || \
((FLAG) == TIM_FLAG_CC3OF) || \
((FLAG) == TIM_FLAG_CC4OF))
```

5.111 TIM_Input_Capture_Filer_Value

Macros

- #define **IS_TIM_IC_FILTER**(ICFILTER) ((ICFILTER) <= 0xF)

5.111.1 Detailed Description

5.112 TIM_External_Trigger_Filter

Macros

- #define **IS_TIM_EXT_FILTER**(EXTFILTER) ((EXTFILTER) <= 0xF)

5.112.1 Detailed Description

5.113 TIM_Legacy

Macros

- #define **TIM_DMABurstLength_1Byte** TIM_DMABurstLength_1Transfer
- #define **TIM_DMABurstLength_2Bytes** TIM_DMABurstLength_2Transfers
- #define **TIM_DMABurstLength_3Bytes** TIM_DMABurstLength_3Transfers
- #define **TIM_DMABurstLength_4Bytes** TIM_DMABurstLength_4Transfers
- #define **TIM_DMABurstLength_5Bytes** TIM_DMABurstLength_5Transfers
- #define **TIM_DMABurstLength_6Bytes** TIM_DMABurstLength_6Transfers
- #define **TIM_DMABurstLength_7Bytes** TIM_DMABurstLength_7Transfers
- #define **TIM_DMABurstLength_8Bytes** TIM_DMABurstLength_8Transfers
- #define **TIM_DMABurstLength_9Bytes** TIM_DMABurstLength_9Transfers
- #define **TIM_DMABurstLength_10Bytes** TIM_DMABurstLength_10Transfers
- #define **TIM_DMABurstLength_11Bytes** TIM_DMABurstLength_11Transfers
- #define **TIM_DMABurstLength_12Bytes** TIM_DMABurstLength_12Transfers
- #define **TIM_DMABurstLength_13Bytes** TIM_DMABurstLength_13Transfers
- #define **TIM_DMABurstLength_14Bytes** TIM_DMABurstLength_14Transfers
- #define **TIM_DMABurstLength_15Bytes** TIM_DMABurstLength_15Transfers
- #define **TIM_DMABurstLength_16Bytes** TIM_DMABurstLength_16Transfers
- #define **TIM_DMABurstLength_17Bytes** TIM_DMABurstLength_17Transfers
- #define **TIM_DMABurstLength_18Bytes** TIM_DMABurstLength_18Transfers

5.113.1 Detailed Description

5.114 MISC

MISC driver modules.

Modules

- MISC_Exported_Constants
- MISC_Private_Functions

Data Structures

- struct NVIC_InitTypeDef

  NVIC Init Structure definition

Macros

- #define **AIRCR_VECTKEY_MASK** ((uint32_t)0x05FA0000)

Functions

- void NVIC_PriorityGroupConfig (uint32_t NVIC_PriorityGroup)

  Configures the priority grouping: pre-emption priority and subpriority.
- void NVIC_Init (NVIC_InitTypeDef *NVIC_InitStruct)

  Initializes the NVIC peripheral according to the specified parameters in the NVIC_InitStruct.
- void NVIC_SetVectorTable (uint32_t NVIC_VectTab, uint32_t Offset)

  Sets the vector table location and Offset.
- void NVIC_SystemLPConfig (uint8_t LowPowerMode, FunctionalState NewState)

  Selects the condition for the system to enter low power mode.
- void SysTick_CLKSourceConfig (uint32_t SysTick_CLKSource)

  Configures the SysTick clock source.

5.114.1 Detailed Description

MISC driver modules.

5.114.2 Function Documentation

5.114.2.1 NVIC_Init()

```
void NVIC_Init (
            NVIC_InitTypeDef * NVIC_InitStruct )
```

Initializes the NVIC peripheral according to the specified parameters in the NVIC_InitStruct.

Note

To configure interrupts priority correctly, the NVIC_PriorityGroupConfig() function should be called before.

Parameters

NVIC_InitStruct	pointer to a NVIC_InitTypeDef structure that contains the configuration information for the specified NVIC peripheral.

Return values

None	

5.114.2.2 NVIC_PriorityGroupConfig()

```
void NVIC_PriorityGroupConfig (
             uint32_t NVIC_PriorityGroup )
```

Configures the priority grouping: pre-emption priority and subpriority.

Parameters

NVIC_PriorityGroup	specifies the priority grouping bits length. This parameter can be one of the following values: • NVIC_PriorityGroup_0: 0 bits for pre-emption priority 4 bits for subpriority • NVIC_PriorityGroup_1: 1 bits for pre-emption priority 3 bits for subpriority • NVIC_PriorityGroup_2: 2 bits for pre-emption priority 2 bits for subpriority • NVIC_PriorityGroup_3: 3 bits for pre-emption priority 1 bits for subpriority • NVIC_PriorityGroup_4: 4 bits for pre-emption priority 0 bits for subpriority

Note

When the NVIC_PriorityGroup_0 is selected, IRQ pre-emption is no more possible. The pending IRQ priority will be managed only by the subpriority.

Return values

None	

5.114.2.3 NVIC_SetVectorTable()

```
void NVIC_SetVectorTable (
             uint32_t NVIC_VectTab,
             uint32_t Offset )
```

Sets the vector table location and Offset.

Parameters

NVIC_VectTab	specifies if the vector table is in RAM or FLASH memory. This parameter can be one of the following values: • NVIC_VectTab_RAM: Vector Table in internal SRAM. • NVIC_VectTab_FLASH: Vector Table in internal FLASH.
Offset	Vector Table base offset field. This value must be a multiple of 0x200.

Return values

None	

5.114.2.4 NVIC_SystemLPConfig()

```
void NVIC_SystemLPConfig (
             uint8_t LowPowerMode,
             FunctionalState NewState )
```

Selects the condition for the system to enter low power mode.

Parameters

LowPowerMode	Specifies the new mode for the system to enter low power mode. This parameter can be one of the following values: • NVIC_LP_SEVONPEND: Low Power SEV on Pend. • NVIC_LP_SLEEPDEEP: Low Power DEEPSLEEP request. • NVIC_LP_SLEEPONEXIT: Low Power Sleep on Exit.
NewState	new state of LP condition. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.114.2.5 SysTick_CLKSourceConfig()

```
void SysTick_CLKSourceConfig (
             uint32_t SysTick_CLKSource )
```

Configures the SysTick clock source.

Parameters

SysTick_CLKSource	specifies the SysTick clock source. This parameter can be one of the following values: • SysTick_CLKSource_HCLK_Div8: AHB clock divided by 8 selected as SysTick clock source. • SysTick_CLKSource_HCLK: AHB clock selected as SysTick clock source.

Return values

None	

5.115 MISC_Private_Functions

Functions

- void NVIC_PriorityGroupConfig (uint32_t NVIC_PriorityGroup)

  Configures the priority grouping: pre-emption priority and subpriority.
- void NVIC_Init (NVIC_InitTypeDef *NVIC_InitStruct)

  Initializes the NVIC peripheral according to the specified parameters in the NVIC_InitStruct.
- void NVIC_SetVectorTable (uint32_t NVIC_VectTab, uint32_t Offset)

  Sets the vector table location and Offset.
- void NVIC_SystemLPConfig (uint8_t LowPowerMode, FunctionalState NewState)

  Selects the condition for the system to enter low power mode.
- void SysTick_CLKSourceConfig (uint32_t SysTick_CLKSource)

  Configures the SysTick clock source.

5.115.1 Detailed Description

5.115.2 Function Documentation

5.115.2.1 NVIC_Init()

```
void NVIC_Init (
            NVIC_InitTypeDef * NVIC_InitStruct )
```

Initializes the NVIC peripheral according to the specified parameters in the NVIC_InitStruct.

Note

To configure interrupts priority correctly, the NVIC_PriorityGroupConfig() function should be called before.

Parameters

NVIC_InitStruct	pointer to a NVIC_InitTypeDef structure that contains the configuration information for the specified NVIC peripheral.

Return values

None	

5.115.2.2 NVIC_PriorityGroupConfig()

```
void NVIC_PriorityGroupConfig (
             uint32_t NVIC_PriorityGroup )
```

Configures the priority grouping: pre-emption priority and subpriority.

Parameters

NVIC_PriorityGroup	specifies the priority grouping bits length. This parameter can be one of the following values: • NVIC_PriorityGroup_0: 0 bits for pre-emption priority 4 bits for subpriority • NVIC_PriorityGroup_1: 1 bits for pre-emption priority 3 bits for subpriority • NVIC_PriorityGroup_2: 2 bits for pre-emption priority 2 bits for subpriority • NVIC_PriorityGroup_3: 3 bits for pre-emption priority 1 bits for subpriority • NVIC_PriorityGroup_4: 4 bits for pre-emption priority 0 bits for subpriority

Note

When the NVIC_PriorityGroup_0 is selected, IRQ pre-emption is no more possible. The pending IRQ priority will be managed only by the subpriority.

Return values

None	

5.115.2.3 NVIC_SetVectorTable()

```
void NVIC_SetVectorTable (
             uint32_t NVIC_VectTab,
             uint32_t Offset )
```

Sets the vector table location and Offset.

Parameters

NVIC_VectTab	specifies if the vector table is in RAM or FLASH memory. This parameter can be one of the following values: • NVIC_VectTab_RAM: Vector Table in internal SRAM. • NVIC_VectTab_FLASH: Vector Table in internal FLASH.
Offset	Vector Table base offset field. This value must be a multiple of 0x200.

Return values

None	

5.115.2.4 NVIC_SystemLPConfig()

```
void NVIC_SystemLPConfig (
             uint8_t LowPowerMode,
             FunctionalState NewState )
```

Selects the condition for the system to enter low power mode.

Parameters

LowPowerMode	Specifies the new mode for the system to enter low power mode. This parameter can be one of the following values: • NVIC_LP_SEVONPEND: Low Power SEV on Pend. • NVIC_LP_SLEEPDEEP: Low Power DEEPSLEEP request. • NVIC_LP_SLEEPONEXIT: Low Power Sleep on Exit.
NewState	new state of LP condition. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.115.2.5 SysTick_CLKSourceConfig()

```
void SysTick_CLKSourceConfig (
             uint32_t SysTick_CLKSource )
```

Configures the SysTick clock source.

Parameters

SysTick_CLKSource	specifies the SysTick clock source. This parameter can be one of the following values: • SysTick_CLKSource_HCLK_Div8: AHB clock divided by 8 selected as SysTick clock source. • SysTick_CLKSource_HCLK: AHB clock selected as SysTick clock source.

Return values

None	

5.116 DMA

DMA driver modules.

Modules

- DMA_Exported_Constants
- DMA_Private_Functions

Data Structures

- struct DMA_InitTypeDef

  DMA Init structure definition.

Macros

- #define TRANSFER_IT_ENABLE_MASK
- #define DMA_Stream0_IT_MASK
- #define **DMA_Stream1_IT_MASK** (uint32_t)(DMA_Stream0_IT_MASK << 6)
- #define **DMA_Stream2_IT_MASK** (uint32_t)(DMA_Stream0_IT_MASK << 16)
- #define **DMA_Stream3_IT_MASK** (uint32_t)(DMA_Stream0_IT_MASK << 22)
- #define **DMA_Stream4_IT_MASK** (uint32_t)(DMA_Stream0_IT_MASK | (uint32_t)0x20000000)
- #define **DMA_Stream5_IT_MASK** (uint32_t)(DMA_Stream1_IT_MASK | (uint32_t)0x20000000)
- #define **DMA_Stream6_IT_MASK** (uint32_t)(DMA_Stream2_IT_MASK | (uint32_t)0x20000000)
- #define **DMA_Stream7_IT_MASK** (uint32_t)(DMA_Stream3_IT_MASK | (uint32_t)0x20000000)
- #define **TRANSFER_IT_MASK** (uint32_t)0x0F3C0F3C
- #define **HIGH_ISR_MASK** (uint32_t)0x20000000
- #define **RESERVED_MASK** (uint32_t)0x0F7D0F7D

Functions

- void DMA_DeInit (DMA_Stream_TypeDef *DMAy_Streamx)

  Deinitialize the DMAy Streamx registers to their default reset values.
- void DMA_Init (DMA_Stream_TypeDef *DMAy_Streamx, DMA_InitTypeDef *DMA_InitStruct)

  Initializes the DMAy Streamx according to the specified parameters in the DMA_InitStruct structure.
- void DMA_StructInit (DMA_InitTypeDef *DMA_InitStruct)

  Fills each DMA_InitStruct member with its default value.
- void DMA_Cmd (DMA_Stream_TypeDef *DMAy_Streamx, FunctionalState NewState)

  Enables or disables the specified DMAy Streamx.
- void DMA_PeriphIncOffsetSizeConfig (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_Pincos)

  Configures, when the PINC (Peripheral Increment address mode) bit is set, if the peripheral address should be incremented with the data size (configured with PSIZE bits) or by a fixed offset equal to 4 (32-bit aligned addresses).
- void DMA_FlowControllerConfig (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_FlowCtrl)

  Configures, when the DMAy Streamx is disabled, the flow controller for the next transactions (Peripheral or Memory).
- void DMA_SetCurrDataCounter (DMA_Stream_TypeDef *DMAy_Streamx, uint16_t Counter)

  Writes the number of data units to be transferred on the DMAy Streamx.
- uint16_t DMA_GetCurrDataCounter (DMA_Stream_TypeDef *DMAy_Streamx)

  Returns the number of remaining data units in the current DMAy Streamx transfer.
- void DMA_DoubleBufferModeConfig (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t Memory1BaseAddr, uint32_t DMA_CurrentMemory)

  Configures, when the DMAy Streamx is disabled, the double buffer mode and the current memory target.
- void DMA_DoubleBufferModeCmd (DMA_Stream_TypeDef *DMAy_Streamx, FunctionalState NewState)

  Enables or disables the double buffer mode for the selected DMA stream.
- void DMA_MemoryTargetConfig (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t MemoryBaseAddr, uint32_t DMA_MemoryTarget)

  Configures the Memory address for the next buffer transfer in double buffer mode (for dynamic use). This function can be called when the DMA Stream is enabled and when the transfer is ongoing.
- uint32_t DMA_GetCurrentMemoryTarget (DMA_Stream_TypeDef *DMAy_Streamx)

  Returns the current memory target used by double buffer transfer.
- FunctionalState DMA_GetCmdStatus (DMA_Stream_TypeDef *DMAy_Streamx)

  Returns the status of EN bit for the specified DMAy Streamx.
- uint32_t DMA_GetFIFOStatus (DMA_Stream_TypeDef *DMAy_Streamx)

  Returns the current DMAy Streamx FIFO filled level.
- FlagStatus DMA_GetFlagStatus (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_FLAG)

  Checks whether the specified DMAy Streamx flag is set or not.
- void DMA_ClearFlag (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_FLAG)

  Clears the DMAy Streamx's pending flags.
- void DMA_ITConfig (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_IT, FunctionalState NewState)

  Enables or disables the specified DMAy Streamx interrupts.
- ITStatus DMA_GetITStatus (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_IT)

  Checks whether the specified DMAy Streamx interrupt has occurred or not.
- void DMA_ClearITPendingBit (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_IT)

  Clears the DMAy Streamx's interrupt pending bits.

5.116.1 Detailed Description

DMA driver modules.

5.116.2 Macro Definition Documentation

5.116.2.1 DMA_Stream0_IT_MASK

```
#define DMA_Stream0_IT_MASK
```

Value:

```
(uint32_t)(DMA_LISR_FEIF0 | DMA_LISR_DMEIF0 | \
DMA_LISR_TEIF0 | DMA_LISR_HTIF0 | \
DMA_LISR_TCIF0)
```

5.116.2.2 TRANSFER_IT_ENABLE_MASK

```
#define TRANSFER_IT_ENABLE_MASK
```

Value:

```
(uint32_t)(DMA_SxCR_TCIE | DMA_SxCR_HTIE | \
DMA_SxCR_TEIE | DMA_SxCR_DMEIE)
```

5.116.3 Function Documentation

5.116.3.1 DMA_ClearFlag()

```
void DMA_ClearFlag (
            DMA_Stream_TypeDef * DMAy_Streamx,
            uint32_t DMA_FLAG )
```

Clears the DMAy Streamx's pending flags.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
DMA_FLAG	specifies the flag to clear. This parameter can be any combination of the following values: • DMA_FLAG_TCIFx: Streamx transfer complete flag • DMA_FLAG_HTIFx: Streamx half transfer complete flag • DMA_FLAG_TEIFx: Streamx transfer error flag • DMA_FLAG_DMEIFx: Streamx direct mode error flag • DMA_FLAG_FEIFx: Streamx FIFO error flag Where x can be 0 to 7 to select the DMA Stream.

Return values

None	

5.116.3.2 DMA_ClearITPendingBit()

```
void DMA_ClearITPendingBit (
            DMA_Stream_TypeDef * DMAy_Streamx,
            uint32_t DMA_IT )
```

Clears the DMAy Streamx's interrupt pending bits.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
DMA_IT	specifies the DMA interrupt pending bit to clear. This parameter can be any combination of the following values: • DMA_IT_TCIFx: Streamx transfer complete interrupt • DMA_IT_HTIFx: Streamx half transfer complete interrupt • DMA_IT_TEIFx: Streamx transfer error interrupt • DMA_IT_DMEIFx: Streamx direct mode error interrupt • DMA_IT_FEIFx: Streamx FIFO error interrupt Where x can be 0 to 7 to select the DMA Stream.

Return values

None	

5.116.3.3 DMA_Cmd()

```
void DMA_Cmd (
            DMA_Stream_TypeDef * DMAy_Streamx,
            FunctionalState NewState )
```

Enables or disables the specified DMAy Streamx.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
NewState	new state of the DMAy Streamx. This parameter can be: ENABLE or DISABLE.

Note

This function may be used to perform Pause-Resume operation. When a transfer is ongoing, calling this function to disable the Stream will cause the transfer to be paused. All configuration registers and the number of remaining data will be preserved. When calling again this function to re-enable the Stream, the transfer will be resumed from the point where it was paused.

After configuring the DMA Stream (DMA_Init() function) and enabling the stream, it is recommended to check (or wait until) the DMA Stream is effectively enabled. A Stream may remain disabled if a configuration parameter is wrong. After disabling a DMA Stream, it is also recommended to check (or wait until) the DMA Stream is effectively disabled. If a Stream is disabled while a data transfer is ongoing, the current data will be transferred and the Stream will be effectively disabled only after the transfer of this single data is finished.

Return values

None	

5.116.3.4 DMA_DeInit()

```
void DMA_DeInit (
             DMA_Stream_TypeDef * DMAy_Streamx )
```

Deinitialize the DMAy Streamx registers to their default reset values.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.

Return values

None	

5.116.3.5 DMA_DoubleBufferModeCmd()

```
void DMA_DoubleBufferModeCmd (
             DMA_Stream_TypeDef * DMAy_Streamx,
             FunctionalState NewState )
```

Enables or disables the double buffer mode for the selected DMA stream.

Note

This function can be called only when the DMA Stream is disabled.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
NewState	new state of the DMAy Streamx double buffer mode. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.116.3.6 DMA_DoubleBufferModeConfig()

```
void DMA_DoubleBufferModeConfig (
             DMA_Stream_TypeDef * DMAy_Streamx,
             uint32_t Memory1BaseAddr,
             uint32_t DMA_CurrentMemory )
```

Configures, when the DMAy Streamx is disabled, the double buffer mode and the current memory target.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
Memory1BaseAddr	the base address of the second buffer (Memory 1)
DMA_CurrentMemory	specifies which memory will be first buffer for the transactions when the Stream will be enabled. This parameter can be one of the following values: • DMA_Memory_0: Memory 0 is the current buffer. • DMA_Memory_1: Memory 1 is the current buffer.

Note

Memory0BaseAddr is set by the DMA structure configuration in DMA_Init().

Return values

None	

5.116.3.7 DMA_FlowControllerConfig()

```
void DMA_FlowControllerConfig (
             DMA_Stream_TypeDef * DMAy_Streamx,
             uint32_t DMA_FlowCtrl )
```

Configures, when the DMAy Streamx is disabled, the flow controller for the next transactions (Peripheral or Memory).

Note

Before enabling this feature, check if the used peripheral supports the Flow Controller mode or not.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
DMA_FlowCtrl	specifies the DMA flow controller. This parameter can be one of the following values: • DMA_FlowCtrl_Memory: DMAy_Streamx transactions flow controller is the DMA controller. • DMA_FlowCtrl_Peripheral: DMAy_Streamx transactions flow controller is the peripheral.

Return values

None	

5.116.3.8 DMA_GetCmdStatus()

```
FunctionalState DMA_GetCmdStatus (
            DMA_Stream_TypeDef * DMAy_Streamx )
```

Returns the status of EN bit for the specified DMAy Streamx.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.

Note

After configuring the DMA Stream (DMA_Init() function) and enabling the stream, it is recommended to check (or wait until) the DMA Stream is effectively enabled. A Stream may remain disabled if a configuration parameter is wrong. After disabling a DMA Stream, it is also recommended to check (or wait until) the DMA Stream is effectively disabled. If a Stream is disabled while a data transfer is ongoing, the current data will be transferred and the Stream will be effectively disabled only after the transfer of this single data is finished.

Return values

Current	state of the DMAy Streamx (ENABLE or DISABLE).

5.116.3.9 DMA_GetCurrDataCounter()

```
uint16_t DMA_GetCurrDataCounter (
            DMA_Stream_TypeDef * DMAy_Streamx )
```

Returns the number of remaining data units in the current DMAy Streamx transfer.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.

Return values

The	number of remaining data units in the current DMAy Streamx transfer.

5.116.3.10 DMA_GetCurrentMemoryTarget()

```
uint32_t DMA_GetCurrentMemoryTarget (
            DMA_Stream_TypeDef * DMAy_Streamx )
```

Returns the current memory target used by double buffer transfer.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.

Return values

The	memory target number: 0 for Memory0 or 1 for Memory1.

5.116.3.11 DMA_GetFIFOStatus()

```
uint32_t DMA_GetFIFOStatus (
            DMA_Stream_TypeDef * DMAy_Streamx )
```

Returns the current DMAy Streamx FIFO filled level.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.

Return values

The	FIFO filling state. • DMA_FIFOStatus_Less1QuarterFull: when FIFO is less than 1 quarter-full and not empty. • DMA_FIFOStatus_1QuarterFull: if more than 1 quarter-full. • DMA_FIFOStatus_HalfFull: if more than 1 half-full. • DMA_FIFOStatus_3QuartersFull: if more than 3 quarters-full. • DMA_FIFOStatus_Empty: when FIFO is empty • DMA_FIFOStatus_Full: when FIFO is full

5.116.3.12 DMA_GetFlagStatus()

```
FlagStatus DMA_GetFlagStatus (
              DMA_Stream_TypeDef * DMAy_Streamx,
              uint32_t DMA_FLAG )
```

Checks whether the specified DMAy Streamx flag is set or not.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
DMA_FLAG	specifies the flag to check. This parameter can be one of the following values: • DMA_FLAG_TCIFx: Streamx transfer complete flag • DMA_FLAG_HTIFx: Streamx half transfer complete flag • DMA_FLAG_TEIFx: Streamx transfer error flag • DMA_FLAG_DMEIFx: Streamx direct mode error flag • DMA_FLAG_FEIFx: Streamx FIFO error flag Where x can be 0 to 7 to select the DMA Stream.

Return values

The	new state of DMA_FLAG (SET or RESET).

5.116.3.13 DMA_GetITStatus()

```
ITStatus DMA_GetITStatus (
              DMA_Stream_TypeDef * DMAy_Streamx,
              uint32_t DMA_IT )
```

Checks whether the specified DMAy Streamx interrupt has occurred or not.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
DMA_IT	specifies the DMA interrupt source to check. This parameter can be one of the following values: • DMA_IT_TCIFx: Streamx transfer complete interrupt • DMA_IT_HTIFx: Streamx half transfer complete interrupt • DMA_IT_TEIFx: Streamx transfer error interrupt • DMA_IT_DMEIFx: Streamx direct mode error interrupt • DMA_IT_FEIFx: Streamx FIFO error interrupt Where x can be 0 to 7 to select the DMA Stream.

Return values

The	new state of DMA_IT (SET or RESET).

5.116.3.14 DMA_Init()

```
void DMA_Init (
             DMA_Stream_TypeDef * DMAy_Streamx,
             DMA_InitTypeDef * DMA_InitStruct )
```

Initializes the DMAy Streamx according to the specified parameters in the DMA_InitStruct structure.

Note

Before calling this function, it is recommended to check that the Stream is actually disabled using the function DMA_GetCmdStatus().

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
DMA_InitStruct	pointer to a DMA_InitTypeDef structure that contains the configuration information for the specified DMA Stream.

Return values

None	

5.116.3.15 DMA_ITConfig()

```
void DMA_ITConfig (
              DMA_Stream_TypeDef * DMAy_Streamx,
              uint32_t DMA_IT,
              FunctionalState NewState )
```

Enables or disables the specified DMAy Streamx interrupts.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
DMA_IT	specifies the DMA interrupt sources to be enabled or disabled. This parameter can be any combination of the following values: • DMA_IT_TC: Transfer complete interrupt mask • DMA_IT_HT: Half transfer complete interrupt mask • DMA_IT_TE: Transfer error interrupt mask • DMA_IT_FE: FIFO error interrupt mask
NewState	new state of the specified DMA interrupts. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.116.3.16 DMA_MemoryTargetConfig()

```
void DMA_MemoryTargetConfig (
              DMA_Stream_TypeDef * DMAy_Streamx,
              uint32_t MemoryBaseAddr,
              uint32_t DMA_MemoryTarget )
```

Configures the Memory address for the next buffer transfer in double buffer mode (for dynamic use). This function can be called when the DMA Stream is enabled and when the transfer is ongoing.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
MemoryBaseAddr	The base address of the target memory buffer
DMA_MemoryTarget	Next memory target to be used. This parameter can be one of the following values: • DMA_Memory_0: To use the memory address 0 • DMA_Memory_1: To use the memory address 1

Note

It is not allowed to modify the Base Address of a target Memory when this target is involved in the current transfer. ie. If the DMA Stream is currently transferring to/from Memory 1, then it not possible to modify Base address of Memory 1, but it is possible to modify Base address of Memory 0. To know which Memory is currently used, you can use the function DMA_GetCurrentMemoryTarget().

Return values

None	

5.116.3.17 DMA_PeriphIncOffsetSizeConfig()

```
void DMA_PeriphIncOffsetSizeConfig (
             DMA_Stream_TypeDef * DMAy_Streamx,
             uint32_t DMA_Pincos )
```

Configures, when the PINC (Peripheral Increment address mode) bit is set, if the peripheral address should be incremented with the data size (configured with PSIZE bits) or by a fixed offset equal to 4 (32-bit aligned addresses).

Note

This function has no effect if the Peripheral Increment mode is disabled.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
DMA_Pincos	specifies the Peripheral increment offset size. This parameter can be one of the following values: • DMA_PINCOS_Psize: Peripheral address increment is done accordingly to PSIZE parameter. • DMA_PINCOS_WordAligned: Peripheral address increment offset is fixed to 4 (32-bit aligned addresses).

Return values

None	

5.116.3.18 DMA_SetCurrDataCounter()

```
void DMA_SetCurrDataCounter (
             DMA_Stream_TypeDef * DMAy_Streamx,
             uint16_t Counter )
```

Writes the number of data units to be transferred on the DMAy Streamx.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
Counter	Number of data units to be transferred (from 0 to 65535) Number of data items depends only on the Peripheral data format.

Note

If Peripheral data format is Bytes: number of data units is equal to total number of bytes to be transferred.

If Peripheral data format is Half-Word: number of data units is
equal to total number of bytes to be transferred / 2.

If Peripheral data format is Word: number of data units is equal to total number of bytes to be transferred / 4.

In Memory-to-Memory transfer mode, the memory buffer pointed by DMAy_SxPAR register is considered as Peripheral.

Return values

The	number of remaining data units in the current DMAy Streamx transfer.

5.116.3.19 DMA_StructInit()

```
void DMA_StructInit (
            DMA_InitTypeDef * DMA_InitStruct )
```

Fills each DMA_InitStruct member with its default value.

Parameters

DMA_InitStruct	: pointer to a DMA_InitTypeDef structure which will be initialized.

Return values

None	

5.117 DMA_Private_Functions

Modules

- Initialization and Configuration functions

  Initialization and Configuration functions.
- Data Counter functions

  Data Counter functions.
- Double Buffer mode functions

Double Buffer mode functions.

- Interrupts and flags management functions

  Interrupts and flags management functions.

5.117.1 Detailed Description

5.118 Initialization and Configuration functions

Initialization and Configuration functions.

Functions

- void DMA_DeInit (DMA_Stream_TypeDef *DMAy_Streamx)

  Deinitialize the DMAy Streamx registers to their default reset values.
- void DMA_Init (DMA_Stream_TypeDef *DMAy_Streamx, DMA_InitTypeDef *DMA_InitStruct)

  Initializes the DMAy Streamx according to the specified parameters in the DMA_InitStruct structure.
- void DMA_StructInit (DMA_InitTypeDef *DMA_InitStruct)

  Fills each DMA_InitStruct member with its default value.
- void DMA_Cmd (DMA_Stream_TypeDef *DMAy_Streamx, FunctionalState NewState)

  Enables or disables the specified DMAy Streamx.
- void DMA_PeriphIncOffsetSizeConfig (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_Pincos)

  Configures, when the PINC (Peripheral Increment address mode) bit is set, if the peripheral address should be incremented with the data size (configured with PSIZE bits) or by a fixed offset equal to 4 (32-bit aligned addresses).
- void DMA_FlowControllerConfig (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_FlowCtrl)

  Configures, when the DMAy Streamx is disabled, the flow controller for the next transactions (Peripheral or Memory).

5.118.1 Detailed Description

Initialization and Configuration functions.

```
 ===================================================================================
                 Initialization and Configuration functions
 ===================================================================================

  This subsection provides functions allowing to initialize the DMA Stream source
  and destination addresses, incrementation and data sizes, transfer direction,
  buffer size, circular/normal mode selection, memory-to-memory mode selection
  and Stream priority value.

  The DMA_Init() function follows the DMA configuration procedures as described in
  reference manual (RM0090) except the first point: waiting on EN bit to be reset.
  This condition should be checked by user application using the function DMA_GetCmdStatus()
  before calling the DMA_Init() function.
```

5.118.2 Function Documentation

5.118.2.1 DMA_Cmd()

```
void DMA_Cmd (
             DMA_Stream_TypeDef * DMAy_Streamx,
             FunctionalState NewState )
```

Enables or disables the specified DMAy Streamx.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
NewState	new state of the DMAy Streamx. This parameter can be: ENABLE or DISABLE.

Note

This function may be used to perform Pause-Resume operation. When a transfer is ongoing, calling this function to disable the Stream will cause the transfer to be paused. All configuration registers and the number of remaining data will be preserved. When calling again this function to re-enable the Stream, the transfer will be resumed from the point where it was paused.

After configuring the DMA Stream (DMA_Init() function) and enabling the stream, it is recommended to check (or wait until) the DMA Stream is effectively enabled. A Stream may remain disabled if a configuration parameter is wrong. After disabling a DMA Stream, it is also recommended to check (or wait until) the DMA Stream is effectively disabled. If a Stream is disabled while a data transfer is ongoing, the current data will be transferred and the Stream will be effectively disabled only after the transfer of this single data is finished.

Return values

None	

5.118.2.2 DMA_DeInit()

```
void DMA_DeInit (
             DMA_Stream_TypeDef * DMAy_Streamx )
```

Deinitialize the DMAy Streamx registers to their default reset values.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.

Return values

None	

5.118.2.3 DMA_FlowControllerConfig()

```
void DMA_FlowControllerConfig (
             DMA_Stream_TypeDef * DMAy_Streamx,
             uint32_t DMA_FlowCtrl )
```

Configures, when the DMAy Streamx is disabled, the flow controller for the next transactions (Peripheral or Memory).

Note

Before enabling this feature, check if the used peripheral supports the Flow Controller mode or not.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
DMA_FlowCtrl	specifies the DMA flow controller. This parameter can be one of the following values: • DMA_FlowCtrl_Memory: DMAy_Streamx transactions flow controller is the DMA controller. • DMA_FlowCtrl_Peripheral: DMAy_Streamx transactions flow controller is the peripheral.

Return values

None	

5.118.2.4 DMA_Init()

```
void DMA_Init (
             DMA_Stream_TypeDef * DMAy_Streamx,
             DMA_InitTypeDef * DMA_InitStruct )
```

Initializes the DMAy Streamx according to the specified parameters in the DMA_InitStruct structure.

Note

Before calling this function, it is recommended to check that the Stream is actually disabled using the function DMA_GetCmdStatus().

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
DMA_InitStruct	pointer to a DMA_InitTypeDef structure that contains the configuration information for the specified DMA Stream.

Return values

None	

5.118.2.5 DMA_PeriphIncOffsetSizeConfig()

```
void DMA_PeriphIncOffsetSizeConfig (
            DMA_Stream_TypeDef * DMAy_Streamx,
            uint32_t DMA_Pincos )
```

Configures, when the PINC (Peripheral Increment address mode) bit is set, if the peripheral address should be incremented with the data size (configured with PSIZE bits) or by a fixed offset equal to 4 (32-bit aligned addresses).

Note

This function has no effect if the Peripheral Increment mode is disabled.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
DMA_Pincos	specifies the Peripheral increment offset size. This parameter can be one of the following values: • DMA_PINCOS_Psize: Peripheral address increment is done accordingly to PSIZE parameter. • DMA_PINCOS_WordAligned: Peripheral address increment offset is fixed to 4 (32-bit aligned addresses).

Return values

None	

5.118.2.6 DMA_StructInit()

```
void DMA_StructInit (
            DMA_InitTypeDef * DMA_InitStruct )
```

Fills each DMA_InitStruct member with its default value.

Parameters

DMA_InitStruct	: pointer to a DMA_InitTypeDef structure which will be initialized.

Return values

None	

5.119 Data Counter functions

Data Counter functions.

Functions

- void DMA_SetCurrDataCounter (DMA_Stream_TypeDef *DMAy_Streamx, uint16_t Counter)
  Writes the number of data units to be transferred on the DMAy Streamx.
- uint16_t DMA_GetCurrDataCounter (DMA_Stream_TypeDef *DMAy_Streamx)
  Returns the number of remaining data units in the current DMAy Streamx transfer.

5.119.1 Detailed Description

Data Counter functions.

```
 ===================================================================================
                           Data Counter functions
 ===================================================================================

  This subsection provides function allowing to configure and read the buffer size
  (number of data to be transferred).

  The DMA data counter can be written only when the DMA Stream is disabled
  (ie. after transfer complete event).

  The following function can be used to write the Stream data counter value:
    - void DMA_SetCurrDataCounter(DMA_Stream_TypeDef* DMAy_Streamx, uint16_t Counter);

@note It is advised to use this function rather than DMA_Init() in situations where
      only the Data buffer needs to be reloaded.

@note If the Source and Destination Data Sizes are different, then the value written in
      data counter, expressing the number of transfers, is relative to the number of
      transfers from the Peripheral point of view.
      ie. If Memory data size is Word, Peripheral data size is Half-Words, then the value
      to be configured in the data counter is the number of Half-Words to be transferred
      from/to the peripheral.

  The DMA data counter can be read to indicate the number of remaining transfers for
  the relative DMA Stream. This counter is decremented at the end of each data
  transfer and when the transfer is complete:
   - If Normal mode is selected: the counter is set to 0.
   - If Circular mode is selected: the counter is reloaded with the initial value
     (configured before enabling the DMA Stream)

  The following function can be used to read the Stream data counter value:
     - uint16_t DMA_GetCurrDataCounter(DMA_Stream_TypeDef* DMAy_Streamx);
```

5.119.2 Function Documentation

5.119.2.1 DMA_GetCurrDataCounter()

```
uint16_t DMA_GetCurrDataCounter (
            DMA_Stream_TypeDef * DMAy_Streamx )
```

Returns the number of remaining data units in the current DMAy Streamx transfer.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.

Return values

The	number of remaining data units in the current DMAy Streamx transfer.

5.119.2.2 DMA_SetCurrDataCounter()

```
void DMA_SetCurrDataCounter (
             DMA_Stream_TypeDef * DMAy_Streamx,
             uint16_t Counter )
```

Writes the number of data units to be transferred on the DMAy Streamx.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
Counter	Number of data units to be transferred (from 0 to 65535) Number of data items depends only on the Peripheral data format.

Note

If Peripheral data format is Bytes: number of data units is equal to total number of bytes to be transferred.

If Peripheral data format is Half-Word: number of data units is
equal to total number of bytes to be transferred / 2.

If Peripheral data format is Word: number of data units is equal to total number of bytes to be transferred / 4.

In Memory-to-Memory transfer mode, the memory buffer pointed by DMAy_SxPAR register is considered as Peripheral.

Return values

The	number of remaining data units in the current DMAy Streamx transfer.

5.120 Double Buffer mode functions

Double Buffer mode functions.

Functions

- void DMA_DoubleBufferModeConfig (DMA_Stream_TypeDef ∗DMAy_Streamx, uint32_t Memory1BaseAddr, uint32_t DMA_CurrentMemory)

Configures, when the DMAy Streamx is disabled, the double buffer mode and the current memory target.

- void DMA_DoubleBufferModeCmd (DMA_Stream_TypeDef *DMAy_Streamx, FunctionalState NewState)

  Enables or disables the double buffer mode for the selected DMA stream.

- void DMA_MemoryTargetConfig (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t MemoryBaseAddr, uint32_t DMA_MemoryTarget)

  Configures the Memory address for the next buffer transfer in double buffer mode (for dynamic use). This function can be called when the DMA Stream is enabled and when the transfer is ongoing.

- uint32_t DMA_GetCurrentMemoryTarget (DMA_Stream_TypeDef *DMAy_Streamx)

  Returns the current memory target used by double buffer transfer.

5.120.1 Detailed Description

Double Buffer mode functions.

```
 ===================================================================================
                          Double Buffer mode functions
 ===================================================================================

  This subsection provides function allowing to configure and control the double
  buffer mode parameters.

  The Double Buffer mode can be used only when Circular mode is enabled.
  The Double Buffer mode cannot be used when transferring data from Memory to Memory.

  The Double Buffer mode allows to set two different Memory addresses from/to which
  the DMA controller will access alternatively (after completing transfer to/from target
  memory 0, it will start transfer to/from target memory 1).
  This allows to reduce software overhead for double buffering and reduce the CPU
  access time.

  Two functions must be called before calling the DMA_Init() function:
   - void DMA_DoubleBufferModeConfig(DMA_Stream_TypeDef* DMAy_Streamx, uint32_t Memory1BaseAddr,
                                     uint32_t DMA_CurrentMemory);
   - void DMA_DoubleBufferModeCmd(DMA_Stream_TypeDef* DMAy_Streamx, FunctionalState NewState);

  DMA_DoubleBufferModeConfig() is called to configure the Memory 1 base address and the first
  Memory target from/to which the transfer will start after enabling the DMA Stream.
  Then DMA_DoubleBufferModeCmd() must be called to enable the Double Buffer mode (or disable
  it when it should not be used).


  Two functions can be called dynamically when the transfer is ongoing (or when the DMA Stream is
  stopped) to modify on of the target Memories addresses or to check wich Memory target is currently
   used:
    - void DMA_MemoryTargetConfig(DMA_Stream_TypeDef* DMAy_Streamx, uint32_t MemoryBaseAddr,
                                  uint32_t DMA_MemoryTarget);
    - uint32_t DMA_GetCurrentMemoryTarget(DMA_Stream_TypeDef* DMAy_Streamx);

  DMA_MemoryTargetConfig() can be called to modify the base address of one of the two target Memories.
  The Memory of which the base address will be modified must not be currently be used by the DMA Stream
  (ie. if the DMA Stream is currently transferring from Memory 1 then you can only modify base address
  of target Memory 0 and vice versa).
  To check this condition, it is recommended to use the function DMA_GetCurrentMemoryTarget() which
  returns the index of the Memory target currently in use by the DMA Stream.
```

5.120.2 Function Documentation

5.120.2.1 DMA_DoubleBufferModeCmd()

```
void DMA_DoubleBufferModeCmd (
             DMA_Stream_TypeDef * DMAy_Streamx,
             FunctionalState NewState )
```

Enables or disables the double buffer mode for the selected DMA stream.

Note

This function can be called only when the DMA Stream is disabled.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
NewState	new state of the DMAy Streamx double buffer mode. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.120.2.2 DMA_DoubleBufferModeConfig()

```
void DMA_DoubleBufferModeConfig (
             DMA_Stream_TypeDef * DMAy_Streamx,
             uint32_t Memory1BaseAddr,
             uint32_t DMA_CurrentMemory )
```

Configures, when the DMAy Streamx is disabled, the double buffer mode and the current memory target.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
Memory1BaseAddr	the base address of the second buffer (Memory 1)
DMA_CurrentMemory	specifies which memory will be first buffer for the transactions when the Stream will be enabled. This parameter can be one of the following values: • DMA_Memory_0: Memory 0 is the current buffer. • DMA_Memory_1: Memory 1 is the current buffer.

Note

Memory0BaseAddr is set by the DMA structure configuration in DMA_Init().

Return values

None	

5.120.2.3 DMA_GetCurrentMemoryTarget()

```
uint32_t DMA_GetCurrentMemoryTarget (
            DMA_Stream_TypeDef * DMAy_Streamx )
```

Returns the current memory target used by double buffer transfer.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.

Return values

The	memory target number: 0 for Memory0 or 1 for Memory1.

5.120.2.4 DMA_MemoryTargetConfig()

```
void DMA_MemoryTargetConfig (
            DMA_Stream_TypeDef * DMAy_Streamx,
            uint32_t MemoryBaseAddr,
            uint32_t DMA_MemoryTarget )
```

Configures the Memory address for the next buffer transfer in double buffer mode (for dynamic use). This function can be called when the DMA Stream is enabled and when the transfer is ongoing.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
MemoryBaseAddr	The base address of the target memory buffer
DMA_MemoryTarget	Next memory target to be used. This parameter can be one of the following values: • DMA_Memory_0: To use the memory address 0 • DMA_Memory_1: To use the memory address 1

Note

It is not allowed to modify the Base Address of a target Memory when this target is involved in the current transfer. ie. If the DMA Stream is currently transferring to/from Memory 1, then it not possible to modify Base address of Memory 1, but it is possible to modify Base address of Memory 0. To know which Memory is

currently used, you can use the function DMA_GetCurrentMemoryTarget().

Return values

None	

5.121 Interrupts and flags management functions

Interrupts and flags management functions.

Functions

- FunctionalState DMA_GetCmdStatus (DMA_Stream_TypeDef *DMAy_Streamx)
  Returns the status of EN bit for the specified DMAy Streamx.
- uint32_t DMA_GetFIFOStatus (DMA_Stream_TypeDef *DMAy_Streamx)
  Returns the current DMAy Streamx FIFO filled level.
- FlagStatus DMA_GetFlagStatus (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_FLAG)
  Checks whether the specified DMAy Streamx flag is set or not.
- void DMA_ClearFlag (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_FLAG)
  Clears the DMAy Streamx's pending flags.
- void DMA_ITConfig (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_IT, FunctionalState NewState)
  Enables or disables the specified DMAy Streamx interrupts.
- ITStatus DMA_GetITStatus (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_IT)
  Checks whether the specified DMAy Streamx interrupt has occurred or not.
- void DMA_ClearITPendingBit (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_IT)
  Clears the DMAy Streamx's interrupt pending bits.

5.121.1 Detailed Description

Interrupts and flags management functions.

```
 ===================================================================================
                  Interrupts and flags management functions
 ===================================================================================

  This subsection provides functions allowing to
   - Check the DMA enable status
   - Check the FIFO status
   - Configure the DMA Interrupts sources and check or clear the flags or pending bits status.

 1. DMA Enable status:
   After configuring the DMA Stream (DMA_Init() function) and enabling the stream,
   it is recommended to check (or wait until) the DMA Stream is effectively enabled.
   A Stream may remain disabled if a configuration parameter is wrong.
   After disabling a DMA Stream, it is also recommended to check (or wait until) the DMA
   Stream is effectively disabled. If a Stream is disabled while a data transfer is ongoing,
   the current data will be transferred and the Stream will be effectively disabled only after
   this data transfer completion.
   To monitor this state it is possible to use the following function:
     - FunctionalState DMA_GetCmdStatus(DMA_Stream_TypeDef* DMAy_Streamx);
```

```
2. FIFO Status:
  It is possible to monitor the FIFO status when a transfer is ongoing using the following
  function:
    - uint32_t DMA_GetFIFOStatus(DMA_Stream_TypeDef* DMAy_Streamx);

3. DMA Interrupts and Flags:
 The user should identify which mode will be used in his application to manage the
 DMA controller events: Polling mode or Interrupt mode.

 Polling Mode
 =============
   Each DMA stream can be managed through 4 event Flags:
   (x : DMA Stream number )
      1. DMA_FLAG_FEIFx  : to indicate that a FIFO Mode Transfer Error event occurred.
      2. DMA_FLAG_DMEIFx : to indicate that a Direct Mode Transfer Error event occurred.
      3. DMA_FLAG_TEIFx  : to indicate that a Transfer Error event occurred.
      4. DMA_FLAG_HTIFx  : to indicate that a Half-Transfer Complete event occurred.
      5. DMA_FLAG_TCIFx  : to indicate that a Transfer Complete event occurred .

  In this Mode it is advised to use the following functions:
     - FlagStatus DMA_GetFlagStatus(DMA_Stream_TypeDef* DMAy_Streamx, uint32_t DMA_FLAG);
     - void DMA_ClearFlag(DMA_Stream_TypeDef* DMAy_Streamx, uint32_t DMA_FLAG);

 Interrupt Mode
 ===============
   Each DMA Stream can be managed through 4 Interrupts:

   Interrupt Source
   ----------------
      1. DMA_IT_FEIFx  : specifies the interrupt source for the  FIFO Mode Transfer Error event.
      2. DMA_IT_DMEIFx : specifies the interrupt source for the Direct Mode Transfer Error event.
      3. DMA_IT_TEIFx  : specifies the interrupt source for the Transfer Error event.
      4. DMA_IT_HTIFx  : specifies the interrupt source for the Half-Transfer Complete event.
      5. DMA_IT_TCIFx  : specifies the interrupt source for the a Transfer Complete event.

 In this Mode it is advised to use the following functions:
    - void DMA_ITConfig(DMA_Stream_TypeDef* DMAy_Streamx, uint32_t DMA_IT, FunctionalState NewState);
    - ITStatus DMA_GetITStatus(DMA_Stream_TypeDef* DMAy_Streamx, uint32_t DMA_IT);
    - void DMA_ClearITPendingBit(DMA_Stream_TypeDef* DMAy_Streamx, uint32_t DMA_IT);
```

5.121.2 Function Documentation

5.121.2.1 DMA_ClearFlag()

```
void DMA_ClearFlag (
             DMA_Stream_TypeDef * DMAy_Streamx,
             uint32_t DMA_FLAG )
```

Clears the DMAy Streamx's pending flags.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.

Parameters

DMA_FLAG	specifies the flag to clear. This parameter can be any combination of the following values: • DMA_FLAG_TCIFx: Streamx transfer complete flag • DMA_FLAG_HTIFx: Streamx half transfer complete flag • DMA_FLAG_TEIFx: Streamx transfer error flag • DMA_FLAG_DMEIFx: Streamx direct mode error flag • DMA_FLAG_FEIFx: Streamx FIFO error flag Where x can be 0 to 7 to select the DMA Stream.

Return values

None	

5.121.2.2 DMA_ClearITPendingBit()

```
void DMA_ClearITPendingBit (
             DMA_Stream_TypeDef * DMAy_Streamx,
             uint32_t DMA_IT )
```

Clears the DMAy Streamx's interrupt pending bits.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
DMA_IT	specifies the DMA interrupt pending bit to clear. This parameter can be any combination of the following values: • DMA_IT_TCIFx: Streamx transfer complete interrupt • DMA_IT_HTIFx: Streamx half transfer complete interrupt • DMA_IT_TEIFx: Streamx transfer error interrupt • DMA_IT_DMEIFx: Streamx direct mode error interrupt • DMA_IT_FEIFx: Streamx FIFO error interrupt Where x can be 0 to 7 to select the DMA Stream.

Return values

None	

5.121.2.3 DMA_GetCmdStatus()

```
FunctionalState DMA_GetCmdStatus (
            DMA_Stream_TypeDef * DMAy_Streamx )
```

Returns the status of EN bit for the specified DMAy Streamx.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.

Note

After configuring the DMA Stream (DMA_Init() function) and enabling the stream, it is recommended to check (or wait until) the DMA Stream is effectively enabled. A Stream may remain disabled if a configuration parameter is wrong. After disabling a DMA Stream, it is also recommended to check (or wait until) the DMA Stream is effectively disabled. If a Stream is disabled while a data transfer is ongoing, the current data will be transferred and the Stream will be effectively disabled only after the transfer of this single data is finished.

Return values

Current	state of the DMAy Streamx (ENABLE or DISABLE).

5.121.2.4 DMA_GetFIFOStatus()

```
uint32_t DMA_GetFIFOStatus (
            DMA_Stream_TypeDef * DMAy_Streamx )
```

Returns the current DMAy Streamx FIFO filled level.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.

Return values

The	FIFO filling state. • DMA_FIFOStatus_Less1QuarterFull: when FIFO is less than 1 quarter-full and not empty. • DMA_FIFOStatus_1QuarterFull: if more than 1 quarter-full. • DMA_FIFOStatus_HalfFull: if more than 1 half-full. • DMA_FIFOStatus_3QuartersFull: if more than 3 quarters-full. • DMA_FIFOStatus_Empty: when FIFO is empty • DMA_FIFOStatus_Full: when FIFO is full

5.121.2.5 DMA_GetFlagStatus()

```
FlagStatus DMA_GetFlagStatus (
            DMA_Stream_TypeDef * DMAy_Streamx,
            uint32_t DMA_FLAG )
```

Checks whether the specified DMAy Streamx flag is set or not.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
DMA_FLAG	specifies the flag to check. This parameter can be one of the following values: • DMA_FLAG_TCIFx: Streamx transfer complete flag • DMA_FLAG_HTIFx: Streamx half transfer complete flag • DMA_FLAG_TEIFx: Streamx transfer error flag • DMA_FLAG_DMEIFx: Streamx direct mode error flag • DMA_FLAG_FEIFx: Streamx FIFO error flag Where x can be 0 to 7 to select the DMA Stream.

Return values

The	new state of DMA_FLAG (SET or RESET).

5.121.2.6 DMA_GetITStatus()

```
ITStatus DMA_GetITStatus (
            DMA_Stream_TypeDef * DMAy_Streamx,
            uint32_t DMA_IT )
```

Checks whether the specified DMAy Streamx interrupt has occurred or not.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
DMA_IT	specifies the DMA interrupt source to check. This parameter can be one of the following values: • DMA_IT_TCIFx: Streamx transfer complete interrupt • DMA_IT_HTIFx: Streamx half transfer complete interrupt • DMA_IT_TEIFx: Streamx transfer error interrupt • DMA_IT_DMEIFx: Streamx direct mode error interrupt • DMA_IT_FEIFx: Streamx FIFO error interrupt Where x can be 0 to 7 to select the DMA Stream.

Return values

The	new state of DMA_IT (SET or RESET).

5.121.2.7 DMA_ITConfig()

```
void DMA_ITConfig (
            DMA_Stream_TypeDef * DMAy_Streamx,
            uint32_t DMA_IT,
            FunctionalState NewState )
```

Enables or disables the specified DMAy Streamx interrupts.

Parameters

DMAy_Streamx	where y can be 1 or 2 to select the DMA and x can be 0 to 7 to select the DMA Stream.
DMA_IT	specifies the DMA interrupt sources to be enabled or disabled. This parameter can be any combination of the following values: • DMA_IT_TC: Transfer complete interrupt mask • DMA_IT_HT: Half transfer complete interrupt mask • DMA_IT_TE: Transfer error interrupt mask • DMA_IT_FE: FIFO error interrupt mask
NewState	new state of the specified DMA interrupts. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.122 GPIO

GPIO driver modules.

Modules

- GPIO_Exported_Constants
- GPIO_Private_Functions

Data Structures

- struct GPIO_InitTypeDef

  GPIO Init structure definition

Macros

- #define IS_GPIO_ALL_PERIPH(PERIPH)
- #define IS_GPIO_MODE(MODE)
- #define **IS_GPIO_OTYPE**(OTYPE) (((OTYPE) == GPIO_OType_PP) || ((OTYPE) == GPIO_OType_OD))
- #define IS_GPIO_SPEED(SPEED)
- #define IS_GPIO_PUPD(PUPD)
- #define **IS_GPIO_BIT_ACTION**(ACTION) (((ACTION) == Bit_RESET) || ((ACTION) == Bit_SET))

Enumerations

- enum GPIOMode_TypeDef { GPIO_Mode_IN = 0x00 , GPIO_Mode_OUT = 0x01 , GPIO_Mode_AF = 0x02 , GPIO_Mode_AN = 0x03 }

  GPIO Configuration Mode enumeration.
- enum GPIOOType_TypeDef { **GPIO_OType_PP** = 0x00 , **GPIO_OType_OD** = 0x01 }

  GPIO Output type enumeration.
- enum GPIOSpeed_TypeDef { GPIO_Speed_2MHz = 0x00 , GPIO_Speed_25MHz = 0x01 , GPIO_Speed_50MHz = 0x02 , GPIO_Speed_100MHz = 0x03 }

  GPIO Output Maximum frequency enumeration.
- enum GPIOPuPd_TypeDef { **GPIO_PuPd_NOPULL** = 0x00 , **GPIO_PuPd_UP** = 0x01 , **GPIO_PuPd_DOWN** = 0x02 }

  GPIO Configuration PullUp PullDown enumeration.
- enum BitAction { **Bit_RESET** = 0 , **Bit_SET** }

  GPIO Bit SET and Bit RESET enumeration.

Functions

- void GPIO_DeInit (GPIO_TypeDef *GPIOx)

  Deinitializes the GPIOx peripheral registers to their default reset values.
- void GPIO_Init (GPIO_TypeDef *GPIOx, GPIO_InitTypeDef *GPIO_InitStruct)

  Initializes the GPIOx peripheral according to the specified parameters in the GPIO_InitStruct.
- void GPIO_StructInit (GPIO_InitTypeDef *GPIO_InitStruct)

  Fills each GPIO_InitStruct member with its default value.
- void GPIO_PinLockConfig (GPIO_TypeDef *GPIOx, uint16_t GPIO_Pin)

  Locks GPIO Pins configuration registers.
- uint8_t GPIO_ReadInputDataBit (GPIO_TypeDef *GPIOx, uint16_t GPIO_Pin)

  Reads the specified input port pin.
- uint16_t GPIO_ReadInputData (GPIO_TypeDef *GPIOx)

  Reads the specified GPIO input data port.
- uint8_t GPIO_ReadOutputDataBit (GPIO_TypeDef *GPIOx, uint16_t GPIO_Pin)

  Reads the specified output data port bit.
- uint16_t GPIO_ReadOutputData (GPIO_TypeDef *GPIOx)

  Reads the specified GPIO output data port.
- void GPIO_SetBits (GPIO_TypeDef *GPIOx, uint16_t GPIO_Pin)

  Sets the selected data port bits.
- void GPIO_ResetBits (GPIO_TypeDef *GPIOx, uint16_t GPIO_Pin)

  Clears the selected data port bits.
- void GPIO_WriteBit (GPIO_TypeDef *GPIOx, uint16_t GPIO_Pin, BitAction BitVal)

  Sets or clears the selected data port bit.
- void GPIO_Write (GPIO_TypeDef *GPIOx, uint16_t PortVal)

  Writes data to the specified GPIO data port.
- void GPIO_ToggleBits (GPIO_TypeDef *GPIOx, uint16_t GPIO_Pin)

  Toggles the specified GPIO pins..
- void GPIO_PinAFConfig (GPIO_TypeDef *GPIOx, uint16_t GPIO_PinSource, uint8_t GPIO_AF)

  Changes the mapping of the specified pin.

5.122.1 Detailed Description

GPIO driver modules.

5.122.2 Macro Definition Documentation

5.122.2.1 IS_GPIO_ALL_PERIPH

```
#define IS_GPIO_ALL_PERIPH(
              PERIPH )
```

Value:

```
(((PERIPH) == GPIOA) || \
((PERIPH) == GPIOB) || \
((PERIPH) == GPIOC) || \
((PERIPH) == GPIOD) || \
((PERIPH) == GPIOE) || \
((PERIPH) == GPIOF) || \
((PERIPH) == GPIOG) || \
((PERIPH) == GPIOH) || \
((PERIPH) == GPIOI))
```

5.122.2.2 IS_GPIO_MODE

```
#define IS_GPIO_MODE(
              MODE )
```

Value:

```
(((MODE) == GPIO_Mode_IN)  || ((MODE) == GPIO_Mode_OUT) || \
((MODE) == GPIO_Mode_AF)|| ((MODE) == GPIO_Mode_AN))
```

5.122.2.3 IS_GPIO_PUPD

```
#define IS_GPIO_PUPD(
              PUPD )
```

Value:

```
(((PUPD) == GPIO_PuPd_NOPULL) || ((PUPD) == GPIO_PuPd_UP) || \
((PUPD) == GPIO_PuPd_DOWN))
```

5.122.2.4 IS_GPIO_SPEED

```
#define IS_GPIO_SPEED(
              SPEED )
```

Value:

```
(((SPEED) == GPIO_Speed_2MHz) || ((SPEED) == GPIO_Speed_25MHz) || \
((SPEED) == GPIO_Speed_50MHz)||  ((SPEED) == GPIO_Speed_100MHz))
```

5.122.3 Enumeration Type Documentation

5.122.3.1 GPIOMode_TypeDef

```
enum GPIOMode_TypeDef
```

GPIO Configuration Mode enumeration.

Enumerator

GPIO_Mode_IN	GPIO Input Mode
GPIO_Mode_OUT	GPIO Output Mode
GPIO_Mode_AF	GPIO Alternate function Mode
GPIO_Mode_AN	GPIO Analog Mode

5.122.3.2 GPIOSpeed_TypeDef

```
enum GPIOSpeed_TypeDef
```

GPIO Output Maximum frequency enumeration.

Enumerator

GPIO_Speed_2MHz	Low speed
GPIO_Speed_25MHz	Medium speed
GPIO_Speed_50MHz	Fast speed
GPIO_Speed_100MHz	High speed on 30 pF (80 MHz Output max speed on 15 pF)

5.122.4 Function Documentation

5.122.4.1 GPIO_DeInit()

```
void GPIO_DeInit (
            GPIO_TypeDef * GPIOx )
```

Deinitializes the GPIOx peripheral registers to their default reset values.

Note

By default, The GPIO pins are configured in input floating mode (except JTAG pins).

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.

Return values

None	

5.122.4.2 GPIO_Init()

```
void GPIO_Init (
            GPIO_TypeDef * GPIOx,
            GPIO_InitTypeDef * GPIO_InitStruct )
```

Initializes the GPIOx peripheral according to the specified parameters in the GPIO_InitStruct.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.
GPIO_InitStruct	pointer to a GPIO_InitTypeDef structure that contains the configuration information for the specified GPIO peripheral.

Return values

None	

5.122.4.3 GPIO_PinAFConfig()

```
void GPIO_PinAFConfig (
            GPIO_TypeDef * GPIOx,
            uint16_t GPIO_PinSource,
            uint8_t GPIO_AF )
```

Changes the mapping of the specified pin.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.
GPIO_PinSource	specifies the pin for the Alternate function. This parameter can be GPIO_PinSourcex where x can be (0..15).

Parameters

GPIO_AFSelection	selects the pin to used as Alternate function. This parameter can be one of the following values: • GPIO_AF_RTC_50Hz: Connect RTC_50Hz pin to AF0 (default after reset) • GPIO_AF_MCO: Connect MCO pin (MCO1 and MCO2) to AF0 (default after reset) • GPIO_AF_TAMPER: Connect TAMPER pins (TAMPER_1 and TAMPER_2) to AF0 (default after reset) • GPIO_AF_SWJ: Connect SWJ pins (SWD and JTAG)to AF0 (default after reset) • GPIO_AF_TRACE: Connect TRACE pins to AF0 (default after reset) • GPIO_AF_TIM1: Connect TIM1 pins to AF1 • GPIO_AF_TIM2: Connect TIM2 pins to AF1 • GPIO_AF_TIM3: Connect TIM3 pins to AF2 • GPIO_AF_TIM4: Connect TIM4 pins to AF2 • GPIO_AF_TIM5: Connect TIM5 pins to AF2 • GPIO_AF_TIM8: Connect TIM8 pins to AF3 • GPIO_AF_TIM9: Connect TIM9 pins to AF3 • GPIO_AF_TIM10: Connect TIM10 pins to AF3 • GPIO_AF_TIM11: Connect TIM11 pins to AF3 • GPIO_AF_I2C1: Connect I2C1 pins to AF4 • GPIO_AF_I2C2: Connect I2C2 pins to AF4 • GPIO_AF_I2C3: Connect I2C3 pins to AF4 • GPIO_AF_SPI1: Connect SPI1 pins to AF5 • GPIO_AF_SPI2: Connect SPI2/I2S2 pins to AF5 • GPIO_AF_SPI3: Connect SPI3/I2S3 pins to AF6 • GPIO_AF_I2S3ext: Connect I2S3ext pins to AF7 • GPIO_AF_USART1: Connect USART1 pins to AF7 • GPIO_AF_USART2: Connect USART2 pins to AF7 • GPIO_AF_USART3: Connect USART3 pins to AF7 • GPIO_AF_UART4: Connect UART4 pins to AF8 • GPIO_AF_UART5: Connect UART5 pins to AF8 • GPIO_AF_USART6: Connect USART6 pins to AF8 • GPIO_AF_CAN1: Connect CAN1 pins to AF9 • GPIO_AF_CAN2: Connect CAN2 pins to AF9 • GPIO_AF_TIM12: Connect TIM12 pins to AF9 • GPIO_AF_TIM13: Connect TIM13 pins to AF9 • GPIO_AF_TIM14: Connect TIM14 pins to AF9 • GPIO_AF_OTG_FS: Connect OTG_FS pins to AF10
Generated by Doxygen	• GPIO_AF_OTG_HS: Connect OTG_HS pins to AF10 • GPIO_AF_ETH: Connect ETHERNET pins to AF11 • GPIO_AF_FSMC: Connect FSMC pins to AF12

Parameters

Return values

None	

5.122.4.4 GPIO_PinLockConfig()

```
void GPIO_PinLockConfig (
            GPIO_TypeDef * GPIOx,
            uint16_t GPIO_Pin )
```

Locks GPIO Pins configuration registers.

Note

The locked registers are GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR, GPIOx_PUPDR, GPIOx_↩ AFRL and GPIOx_AFRH.

The configuration of the locked GPIO pins can no longer be modified until the next reset.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.
GPIO_Pin	specifies the port bit to be locked. This parameter can be any combination of GPIO_Pin_x where x can be (0..15).

Return values

None	

5.122.4.5 GPIO_ReadInputData()

```
uint16_t GPIO_ReadInputData (
            GPIO_TypeDef * GPIOx )
```

Reads the specified GPIO input data port.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.

Return values

GPIO	input data port value.

5.122.4.6 GPIO_ReadInputDataBit()

```
uint8_t GPIO_ReadInputDataBit (
            GPIO_TypeDef * GPIOx,
            uint16_t GPIO_Pin )
```

Reads the specified input port pin.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.
GPIO_Pin	specifies the port bit to read. This parameter can be GPIO_Pin_x where x can be (0..15).

Return values

The	input port pin value.

5.122.4.7 GPIO_ReadOutputData()

```
uint16_t GPIO_ReadOutputData (
            GPIO_TypeDef * GPIOx )
```

Reads the specified GPIO output data port.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.

Return values

GPIO	output data port value.

5.122.4.8 GPIO_ReadOutputDataBit()

```
uint8_t GPIO_ReadOutputDataBit (
            GPIO_TypeDef * GPIOx,
            uint16_t GPIO_Pin )
```

Reads the specified output data port bit.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.
GPIO_Pin	specifies the port bit to read. This parameter can be GPIO_Pin_x where x can be (0..15).

Return values

The	output port pin value.

5.122.4.9 GPIO_ResetBits()

```
void GPIO_ResetBits (
            GPIO_TypeDef * GPIOx,
            uint16_t GPIO_Pin )
```

Clears the selected data port bits.

Note

This functions uses GPIOx_BSRR register to allow atomic read/modify accesses. In this way, there is no risk of an IRQ occurring between the read and the modify access.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.
GPIO_Pin	specifies the port bits to be written. This parameter can be any combination of GPIO_Pin_x where x can be (0..15).

Return values

None	

5.122.4.10 GPIO_SetBits()

```
void GPIO_SetBits (
            GPIO_TypeDef * GPIOx,
            uint16_t GPIO_Pin )
```

Sets the selected data port bits.

Note

This functions uses GPIOx_BSRR register to allow atomic read/modify accesses. In this way, there is no risk of an IRQ occurring between the read and the modify access.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.
GPIO_Pin	specifies the port bits to be written. This parameter can be any combination of GPIO_Pin_x where x can be (0..15).

Return values

None	

5.122.4.11 GPIO_StructInit()

```
void GPIO_StructInit (
            GPIO_InitTypeDef * GPIO_InitStruct )
```

Fills each GPIO_InitStruct member with its default value.

Parameters

GPIO_InitStruct	: pointer to a GPIO_InitTypeDef structure which will be initialized.

Return values

None	

5.122.4.12 GPIO_ToggleBits()

```
void GPIO_ToggleBits (
            GPIO_TypeDef * GPIOx,
            uint16_t GPIO_Pin )
```

Toggles the specified GPIO pins..

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.
GPIO_Pin	Specifies the pins to be toggled.

Return values

None	

5.122.4.13 GPIO_Write()

```
void GPIO_Write (
            GPIO_TypeDef * GPIOx,
            uint16_t PortVal )
```

Writes data to the specified GPIO data port.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.
PortVal	specifies the value to be written to the port output data register.

Return values

None	

5.122.4.14 GPIO_WriteBit()

```
void GPIO_WriteBit (
            GPIO_TypeDef * GPIOx,
            uint16_t GPIO_Pin,
            BitAction BitVal )
```

Sets or clears the selected data port bit.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.
GPIO_Pin	specifies the port bit to be written. This parameter can be one of GPIO_Pin_x where x can be (0..15).
BitVal	specifies the value to be written to the selected bit. This parameter can be one of the BitAction enum values: • Bit_RESET: to clear the port pin • Bit_SET: to set the port pin

Return values

None	

5.123 GPIO_Private_Functions

Modules

- Initialization and Configuration

Initialization and Configuration.

- GPIO Read and Write

  GPIO Read and Write.

- GPIO Alternate functions configuration function

  GPIO Alternate functions configuration function.

5.123.1 Detailed Description

5.124 Initialization and Configuration

Initialization and Configuration.

Functions

- void GPIO_DeInit (GPIO_TypeDef *GPIOx)

  Deinitializes the GPIOx peripheral registers to their default reset values.

- void GPIO_Init (GPIO_TypeDef *GPIOx, GPIO_InitTypeDef *GPIO_InitStruct)

  Initializes the GPIOx peripheral according to the specified parameters in the GPIO_InitStruct.

- void GPIO_StructInit (GPIO_InitTypeDef *GPIO_InitStruct)

  Fills each GPIO_InitStruct member with its default value.

- void GPIO_PinLockConfig (GPIO_TypeDef *GPIOx, uint16_t GPIO_Pin)

  Locks GPIO Pins configuration registers.

5.124.1 Detailed Description

Initialization and Configuration.

```
 ===========================================================================
                        Initialization and Configuration
 ===========================================================================
```

5.124.2 Function Documentation

5.124.2.1 GPIO_DeInit()

```
void GPIO_DeInit (
            GPIO_TypeDef * GPIOx )
```

Deinitializes the GPIOx peripheral registers to their default reset values.

Note

By default, The GPIO pins are configured in input floating mode (except JTAG pins).

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.

Return values

None	

5.124.2.2 GPIO_Init()

```
void GPIO_Init (
            GPIO_TypeDef * GPIOx,
            GPIO_InitTypeDef * GPIO_InitStruct )
```

Initializes the GPIOx peripheral according to the specified parameters in the GPIO_InitStruct.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.
GPIO_InitStruct	pointer to a GPIO_InitTypeDef structure that contains the configuration information for the specified GPIO peripheral.

Return values

None	

5.124.2.3 GPIO_PinLockConfig()

```
void GPIO_PinLockConfig (
            GPIO_TypeDef * GPIOx,
            uint16_t GPIO_Pin )
```

Locks GPIO Pins configuration registers.

Note

The locked registers are GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR, GPIOx_PUPDR, GPIOx_↩ AFRL and GPIOx_AFRH.

The configuration of the locked GPIO pins can no longer be modified until the next reset.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.
GPIO_Pin	specifies the port bit to be locked. This parameter can be any combination of GPIO_Pin_x where x can be (0..15).

Return values

None	

5.124.2.4 GPIO_StructInit()

```
void GPIO_StructInit (
            GPIO_InitTypeDef * GPIO_InitStruct )
```

Fills each GPIO_InitStruct member with its default value.

Parameters

GPIO_InitStruct	: pointer to a GPIO_InitTypeDef structure which will be initialized.

Return values

None	

5.125 GPIO Read and Write

GPIO Read and Write.

Functions

- uint8_t GPIO_ReadInputDataBit (GPIO_TypeDef ∗GPIOx, uint16_t GPIO_Pin)

  Reads the specified input port pin.
- uint16_t GPIO_ReadInputData (GPIO_TypeDef ∗GPIOx)

  Reads the specified GPIO input data port.
- uint8_t GPIO_ReadOutputDataBit (GPIO_TypeDef ∗GPIOx, uint16_t GPIO_Pin)

  Reads the specified output data port bit.
- uint16_t GPIO_ReadOutputData (GPIO_TypeDef ∗GPIOx)

  Reads the specified GPIO output data port.
- void GPIO_SetBits (GPIO_TypeDef ∗GPIOx, uint16_t GPIO_Pin)

  Sets the selected data port bits.
- void GPIO_ResetBits (GPIO_TypeDef ∗GPIOx, uint16_t GPIO_Pin)

  Clears the selected data port bits.
- void GPIO_WriteBit (GPIO_TypeDef ∗GPIOx, uint16_t GPIO_Pin, BitAction BitVal)

  Sets or clears the selected data port bit.
- void GPIO_Write (GPIO_TypeDef ∗GPIOx, uint16_t PortVal)

  Writes data to the specified GPIO data port.
- void GPIO_ToggleBits (GPIO_TypeDef ∗GPIOx, uint16_t GPIO_Pin)

  Toggles the specified GPIO pins..

5.125.1 Detailed Description

GPIO Read and Write.

```
 ===============================================================================
                              GPIO Read and Write
 ===============================================================================
```

5.125.2 Function Documentation

5.125.2.1 GPIO_ReadInputData()

```
uint16_t GPIO_ReadInputData (
             GPIO_TypeDef * GPIOx )
```

Reads the specified GPIO input data port.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.

Return values

GPIO	input data port value.

5.125.2.2 GPIO_ReadInputDataBit()

```
uint8_t GPIO_ReadInputDataBit (
             GPIO_TypeDef * GPIOx,
             uint16_t GPIO_Pin )
```

Reads the specified input port pin.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.
GPIO_Pin	specifies the port bit to read. This parameter can be GPIO_Pin_x where x can be (0..15).

Return values

The	input port pin value.

5.125.2.3 GPIO_ReadOutputData()

```
uint16_t GPIO_ReadOutputData (
            GPIO_TypeDef * GPIOx )
```

Reads the specified GPIO output data port.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.

Return values

GPIO	output data port value.

5.125.2.4 GPIO_ReadOutputDataBit()

```
uint8_t GPIO_ReadOutputDataBit (
            GPIO_TypeDef * GPIOx,
            uint16_t GPIO_Pin )
```

Reads the specified output data port bit.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.
GPIO_Pin	specifies the port bit to read. This parameter can be GPIO_Pin_x where x can be (0..15).

Return values

The	output port pin value.

5.125.2.5 GPIO_ResetBits()

```
void GPIO_ResetBits (
            GPIO_TypeDef * GPIOx,
            uint16_t GPIO_Pin )
```

Clears the selected data port bits.

Note

This functions uses GPIOx_BSRR register to allow atomic read/modify accesses. In this way, there is no risk of an IRQ occurring between the read and the modify access.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.
GPIO_Pin	specifies the port bits to be written. This parameter can be any combination of GPIO_Pin_x where x can be (0..15).

Return values

None	

5.125.2.6 GPIO_SetBits()

```
void GPIO_SetBits (
            GPIO_TypeDef * GPIOx,
            uint16_t GPIO_Pin )
```

Sets the selected data port bits.

Note

This functions uses GPIOx_BSRR register to allow atomic read/modify accesses. In this way, there is no risk of an IRQ occurring between the read and the modify access.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.
GPIO_Pin	specifies the port bits to be written. This parameter can be any combination of GPIO_Pin_x where x can be (0..15).

Return values

None	

5.125.2.7 GPIO_ToggleBits()

```
void GPIO_ToggleBits (
            GPIO_TypeDef * GPIOx,
            uint16_t GPIO_Pin )
```

Toggles the specified GPIO pins..

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.
GPIO_Pin	Specifies the pins to be toggled.

Return values

None	

5.125.2.8 GPIO_Write()

```
void GPIO_Write (
            GPIO_TypeDef * GPIOx,
            uint16_t PortVal )
```

Writes data to the specified GPIO data port.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.
PortVal	specifies the value to be written to the port output data register.

Return values

None	

5.125.2.9 GPIO_WriteBit()

```
void GPIO_WriteBit (
            GPIO_TypeDef * GPIOx,
            uint16_t GPIO_Pin,
            BitAction BitVal )
```

Sets or clears the selected data port bit.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.
GPIO_Pin	specifies the port bit to be written. This parameter can be one of GPIO_Pin_x where x can be (0..15).
BitVal	specifies the value to be written to the selected bit. This parameter can be one of the BitAction enum values: • Bit_RESET: to clear the port pin • Bit_SET: to set the port pin

Return values

None	

5.126 GPIO Alternate functions configuration function

GPIO Alternate functions configuration function.

Functions

- void GPIO_PinAFConfig (GPIO_TypeDef *GPIOx, uint16_t GPIO_PinSource, uint8_t GPIO_AF)

  Changes the mapping of the specified pin.

5.126.1 Detailed Description

GPIO Alternate functions configuration function.

```
 ===============================================================================
               GPIO Alternate functions configuration function
 ===============================================================================
```

5.126.2 Function Documentation

5.126.2.1 GPIO_PinAFConfig()

```
void GPIO_PinAFConfig (
             GPIO_TypeDef * GPIOx,
             uint16_t GPIO_PinSource,
             uint8_t GPIO_AF )
```

Changes the mapping of the specified pin.

Parameters

GPIOx	where x can be (A..I) to select the GPIO peripheral.
GPIO_PinSource	specifies the pin for the Alternate function. This parameter can be GPIO_PinSourcex where x can be (0..15).

Parameters

GPIO_AFSelection	selects the pin to used as Alternate function. This parameter can be one of the following values: • GPIO_AF_RTC_50Hz: Connect RTC_50Hz pin to AF0 (default after reset) • GPIO_AF_MCO: Connect MCO pin (MCO1 and MCO2) to AF0 (default after reset) • GPIO_AF_TAMPER: Connect TAMPER pins (TAMPER_1 and TAMPER_2) to AF0 (default after reset) • GPIO_AF_SWJ: Connect SWJ pins (SWD and JTAG)to AF0 (default after reset) • GPIO_AF_TRACE: Connect TRACE pins to AF0 (default after reset) • GPIO_AF_TIM1: Connect TIM1 pins to AF1 • GPIO_AF_TIM2: Connect TIM2 pins to AF1 • GPIO_AF_TIM3: Connect TIM3 pins to AF2 • GPIO_AF_TIM4: Connect TIM4 pins to AF2 • GPIO_AF_TIM5: Connect TIM5 pins to AF2 • GPIO_AF_TIM8: Connect TIM8 pins to AF3 • GPIO_AF_TIM9: Connect TIM9 pins to AF3 • GPIO_AF_TIM10: Connect TIM10 pins to AF3 • GPIO_AF_TIM11: Connect TIM11 pins to AF3 • GPIO_AF_I2C1: Connect I2C1 pins to AF4 • GPIO_AF_I2C2: Connect I2C2 pins to AF4 • GPIO_AF_I2C3: Connect I2C3 pins to AF4 • GPIO_AF_SPI1: Connect SPI1 pins to AF5 • GPIO_AF_SPI2: Connect SPI2/I2S2 pins to AF5 • GPIO_AF_SPI3: Connect SPI3/I2S3 pins to AF6 • GPIO_AF_I2S3ext: Connect I2S3ext pins to AF7 • GPIO_AF_USART1: Connect USART1 pins to AF7 • GPIO_AF_USART2: Connect USART2 pins to AF7 • GPIO_AF_USART3: Connect USART3 pins to AF7 • GPIO_AF_UART4: Connect UART4 pins to AF8 • GPIO_AF_UART5: Connect UART5 pins to AF8 • GPIO_AF_USART6: Connect USART6 pins to AF8 • GPIO_AF_CAN1: Connect CAN1 pins to AF9 • GPIO_AF_CAN2: Connect CAN2 pins to AF9 • GPIO_AF_TIM12: Connect TIM12 pins to AF9 • GPIO_AF_TIM13: Connect TIM13 pins to AF9 • GPIO_AF_TIM14: Connect TIM14 pins to AF9 • GPIO_AF_OTG_FS: Connect OTG_FS pins to AF10
	• GPIO_AF_OTG_HS: Connect OTG_HS pins to AF10 • GPIO_AF_ETH: Connect ETHERNET pins to AF11 • GPIO_AF_FSMC: Connect FSMC pins to AF12

Parameters

Return values

None	

5.127 RCC

RCC driver modules.

Modules

- RCC_Exported_Constants
- RCC_Private_Functions

Data Structures

- struct RCC_ClocksTypeDef

Macros

- #define **RCC_OFFSET** (RCC_BASE - PERIPH_BASE)
- #define **CR_OFFSET** (RCC_OFFSET + 0x00)
- #define **HSION_BitNumber** 0x00
- #define **CR_HSION_BB** (PERIPH_BB_BASE + (CR_OFFSET ∗ 32) + (HSION_BitNumber ∗ 4))
- #define **CSSON_BitNumber** 0x13
- #define **CR_CSSON_BB** (PERIPH_BB_BASE + (CR_OFFSET ∗ 32) + (CSSON_BitNumber ∗ 4))
- #define **PLLON_BitNumber** 0x18
- #define **CR_PLLON_BB** (PERIPH_BB_BASE + (CR_OFFSET ∗ 32) + (PLLON_BitNumber ∗ 4))
- #define **PLLI2SON_BitNumber** 0x1A
- #define **CR_PLLI2SON_BB** (PERIPH_BB_BASE + (CR_OFFSET ∗ 32) + (PLLI2SON_BitNumber ∗ 4))
- #define **CFGR_OFFSET** (RCC_OFFSET + 0x08)
- #define **I2SSRC_BitNumber** 0x17
- #define **CFGR_I2SSRC_BB** (PERIPH_BB_BASE + (CFGR_OFFSET ∗ 32) + (I2SSRC_BitNumber ∗ 4))
- #define **BDCR_OFFSET** (RCC_OFFSET + 0x70)
- #define **RTCEN_BitNumber** 0x0F
- #define **BDCR_RTCEN_BB** (PERIPH_BB_BASE + (BDCR_OFFSET ∗ 32) + (RTCEN_BitNumber ∗ 4))
- #define **BDRST_BitNumber** 0x10
- #define **BDCR_BDRST_BB** (PERIPH_BB_BASE + (BDCR_OFFSET ∗ 32) + (BDRST_BitNumber ∗ 4))
- #define **CSR_OFFSET** (RCC_OFFSET + 0x74)
- #define **LSION_BitNumber** 0x00
- #define **CSR_LSION_BB** (PERIPH_BB_BASE + (CSR_OFFSET ∗ 32) + (LSION_BitNumber ∗ 4))
- #define **CFGR_MCO2_RESET_MASK** ((uint32_t)0x07FFFFFF)
- #define **CFGR_MCO1_RESET_MASK** ((uint32_t)0xF89FFFFF)
- #define **FLAG_MASK** ((uint8_t)0x1F)
- #define **CR_BYTE3_ADDRESS** ((uint32_t)0x40023802)
- #define **CIR_BYTE2_ADDRESS** ((uint32_t)(RCC_BASE + 0x0C + 0x01))
- #define **CIR_BYTE3_ADDRESS** ((uint32_t)(RCC_BASE + 0x0C + 0x02))
- #define **BDCR_ADDRESS** (PERIPH_BASE + BDCR_OFFSET)

Functions

- void RCC_DeInit (void)

  Resets the RCC clock configuration to the default reset state.
- void RCC_HSEConfig (uint8_t RCC_HSE)

  Configures the External High Speed oscillator (HSE).
- ErrorStatus RCC_WaitForHSEStartUp (void)

  Waits for HSE start-up.
- void RCC_AdjustHSICalibrationValue (uint8_t HSICalibrationValue)

  Adjusts the Internal High Speed oscillator (HSI) calibration value.
- void RCC_HSICmd (FunctionalState NewState)

  Enables or disables the Internal High Speed oscillator (HSI).
- void RCC_LSEConfig (uint8_t RCC_LSE)

  Configures the External Low Speed oscillator (LSE).
- void RCC_LSICmd (FunctionalState NewState)

  Enables or disables the Internal Low Speed oscillator (LSI).
- void RCC_PLLConfig (uint32_t RCC_PLLSource, uint32_t PLLM, uint32_t PLLN, uint32_t PLLP, uint32_t PLLQ)

  Configures the main PLL clock source, multiplication and division factors.
- void RCC_PLLCmd (FunctionalState NewState)

  Enables or disables the main PLL.
- void RCC_PLLI2SConfig (uint32_t PLLI2SN, uint32_t PLLI2SR)

  Configures the PLLI2S clock multiplication and division factors.
- void RCC_PLLI2SCmd (FunctionalState NewState)

  Enables or disables the PLLI2S.
- void RCC_ClockSecuritySystemCmd (FunctionalState NewState)

  Enables or disables the Clock Security System.
- void RCC_MCO1Config (uint32_t RCC_MCO1Source, uint32_t RCC_MCO1Div)

  Selects the clock source to output on MCO1 pin(PA8).
- void RCC_MCO2Config (uint32_t RCC_MCO2Source, uint32_t RCC_MCO2Div)

  Selects the clock source to output on MCO2 pin(PC9).
- void RCC_SYSCLKConfig (uint32_t RCC_SYSCLKSource)

  Configures the system clock (SYSCLK).
- uint8_t RCC_GetSYSCLKSource (void)

  Returns the clock source used as system clock.
- void RCC_HCLKConfig (uint32_t RCC_SYSCLK)

  Configures the AHB clock (HCLK).
- void RCC_PCLK1Config (uint32_t RCC_HCLK)

  Configures the Low Speed APB clock (PCLK1).
- void RCC_PCLK2Config (uint32_t RCC_HCLK)

  Configures the High Speed APB clock (PCLK2).
- void RCC_GetClocksFreq (RCC_ClocksTypeDef *RCC_Clocks)

  Returns the frequencies of different on chip clocks; SYSCLK, HCLK, PCLK1 and PCLK2.
- void RCC_RTCCLKConfig (uint32_t RCC_RTCCLKSource)

  Configures the RTC clock (RTCCLK).
- void RCC_RTCCLKCmd (FunctionalState NewState)

  Enables or disables the RTC clock.
- void RCC_BackupResetCmd (FunctionalState NewState)

  Forces or releases the Backup domain reset.
- void RCC_I2SCLKConfig (uint32_t RCC_I2SCLKSource)

Configures the I2S clock source (I2SCLK).

- void RCC_AHB1PeriphClockCmd (uint32_t RCC_AHB1Periph, FunctionalState NewState)

  Enables or disables the AHB1 peripheral clock.
- void RCC_AHB2PeriphClockCmd (uint32_t RCC_AHB2Periph, FunctionalState NewState)

  Enables or disables the AHB2 peripheral clock.
- void RCC_AHB3PeriphClockCmd (uint32_t RCC_AHB3Periph, FunctionalState NewState)

  Enables or disables the AHB3 peripheral clock.
- void RCC_APB1PeriphClockCmd (uint32_t RCC_APB1Periph, FunctionalState NewState)

  Enables or disables the Low Speed APB (APB1) peripheral clock.
- void RCC_APB2PeriphClockCmd (uint32_t RCC_APB2Periph, FunctionalState NewState)

  Enables or disables the High Speed APB (APB2) peripheral clock.
- void RCC_AHB1PeriphResetCmd (uint32_t RCC_AHB1Periph, FunctionalState NewState)

  Forces or releases AHB1 peripheral reset.
- void RCC_AHB2PeriphResetCmd (uint32_t RCC_AHB2Periph, FunctionalState NewState)

  Forces or releases AHB2 peripheral reset.
- void RCC_AHB3PeriphResetCmd (uint32_t RCC_AHB3Periph, FunctionalState NewState)

  Forces or releases AHB3 peripheral reset.
- void RCC_APB1PeriphResetCmd (uint32_t RCC_APB1Periph, FunctionalState NewState)

  Forces or releases Low Speed APB (APB1) peripheral reset.
- void RCC_APB2PeriphResetCmd (uint32_t RCC_APB2Periph, FunctionalState NewState)

  Forces or releases High Speed APB (APB2) peripheral reset.
- void RCC_AHB1PeriphClockLPModeCmd (uint32_t RCC_AHB1Periph, FunctionalState NewState)

  Enables or disables the AHB1 peripheral clock during Low Power (Sleep) mode.
- void RCC_AHB2PeriphClockLPModeCmd (uint32_t RCC_AHB2Periph, FunctionalState NewState)

  Enables or disables the AHB2 peripheral clock during Low Power (Sleep) mode.
- void RCC_AHB3PeriphClockLPModeCmd (uint32_t RCC_AHB3Periph, FunctionalState NewState)

  Enables or disables the AHB3 peripheral clock during Low Power (Sleep) mode.
- void RCC_APB1PeriphClockLPModeCmd (uint32_t RCC_APB1Periph, FunctionalState NewState)

  Enables or disables the APB1 peripheral clock during Low Power (Sleep) mode.
- void RCC_APB2PeriphClockLPModeCmd (uint32_t RCC_APB2Periph, FunctionalState NewState)

  Enables or disables the APB2 peripheral clock during Low Power (Sleep) mode.
- void RCC_ITConfig (uint8_t RCC_IT, FunctionalState NewState)

  Enables or disables the specified RCC interrupts.
- FlagStatus RCC_GetFlagStatus (uint8_t RCC_FLAG)

  Checks whether the specified RCC flag is set or not.
- void RCC_ClearFlag (void)

  Clears the RCC reset flags. The reset flags are: RCC_FLAG_PINRST, RCC_FLAG_PORRST, RCC_FLAG_SFTRST, RCC_FLAG_IWDGRST, RCC_FLAG_WWDGRST, RCC_FLAG_LPWRRST.
- ITStatus RCC_GetITStatus (uint8_t RCC_IT)

  Checks whether the specified RCC interrupt has occurred or not.
- void RCC_ClearITPendingBit (uint8_t RCC_IT)

  Clears the RCC's interrupt pending bits.

5.127.1 Detailed Description

RCC driver modules.

5.127.2 Function Documentation

5.127.2.1 RCC_AdjustHSICalibrationValue()

```
void RCC_AdjustHSICalibrationValue (
            uint8_t HSICalibrationValue )
```

Adjusts the Internal High Speed oscillator (HSI) calibration value.

Note

The calibration is used to compensate for the variations in voltage and temperature that influence the frequency of the internal HSI RC.

Parameters

HSICalibrationValue	specifies the calibration trimming value. This parameter must be a number between 0 and 0x1F.

Return values

None	

5.127.2.2 RCC_AHB1PeriphClockCmd()

```
void RCC_AHB1PeriphClockCmd (
            uint32_t RCC_AHB1Periph,
            FunctionalState NewState )
```

Enables or disables the AHB1 peripheral clock.

Note

After reset, the peripheral clock (used for registers read/write access) is disabled and the application software has to enable this clock before using it.

Parameters

RCC_AHBPeriph	specifies the AHB1 peripheral to gates its clock. This parameter can be any combination of the following values: • RCC_AHB1Periph_GPIOA: GPIOA clock • RCC_AHB1Periph_GPIOB: GPIOB clock • RCC_AHB1Periph_GPIOC: GPIOC clock • RCC_AHB1Periph_GPIOD: GPIOD clock • RCC_AHB1Periph_GPIOE: GPIOE clock • RCC_AHB1Periph_GPIOF: GPIOF clock • RCC_AHB1Periph_GPIOG: GPIOG clock • RCC_AHB1Periph_GPIOG: GPIOG clock • RCC_AHB1Periph_GPIOI: GPIOI clock • RCC_AHB1Periph_CRC: CRC clock • RCC_AHB1Periph_BKPSRAM: BKPSRAM interface clock • RCC_AHB1Periph_CCMDATARAMEN CCM data RAM interface clock • RCC_AHB1Periph_DMA1: DMA1 clock • RCC_AHB1Periph_DMA2: DMA2 clock • RCC_AHB1Periph_ETH_MAC: Ethernet MAC clock • RCC_AHB1Periph_ETH_MAC_Tx: Ethernet Transmission clock • RCC_AHB1Periph_ETH_MAC_Rx: Ethernet Reception clock • RCC_AHB1Periph_ETH_MAC_PTP: Ethernet PTP clock • RCC_AHB1Periph_OTG_HS: USB OTG HS clock • RCC_AHB1Periph_OTG_HS_ULPI: USB OTG HS ULPI clock
NewState	new state of the specified peripheral clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.127.2.3 RCC_AHB1PeriphClockLPModeCmd()

```
void RCC_AHB1PeriphClockLPModeCmd (
             uint32_t RCC_AHB1Periph,
             FunctionalState NewState )
```

Enables or disables the AHB1 peripheral clock during Low Power (Sleep) mode.

Note

Peripheral clock gating in SLEEP mode can be used to further reduce power consumption.

After wakeup from SLEEP mode, the peripheral clock is enabled again.

By default, all peripheral clocks are enabled during SLEEP mode.

Parameters

RCC_AHBPeriph	specifies the AHB1 peripheral to gates its clock. This parameter can be any combination of the following values: • RCC_AHB1Periph_GPIOA: GPIOA clock • RCC_AHB1Periph_GPIOB: GPIOB clock • RCC_AHB1Periph_GPIOC: GPIOC clock • RCC_AHB1Periph_GPIOD: GPIOD clock • RCC_AHB1Periph_GPIOE: GPIOE clock • RCC_AHB1Periph_GPIOF: GPIOF clock • RCC_AHB1Periph_GPIOG: GPIOG clock • RCC_AHB1Periph_GPIOG: GPIOG clock • RCC_AHB1Periph_GPIOI: GPIOI clock • RCC_AHB1Periph_CRC: CRC clock • RCC_AHB1Periph_BKPSRAM: BKPSRAM interface clock • RCC_AHB1Periph_DMA1: DMA1 clock • RCC_AHB1Periph_DMA2: DMA2 clock • RCC_AHB1Periph_ETH_MAC: Ethernet MAC clock • RCC_AHB1Periph_ETH_MAC_Tx: Ethernet Transmission clock • RCC_AHB1Periph_ETH_MAC_Rx: Ethernet Reception clock • RCC_AHB1Periph_ETH_MAC_PTP: Ethernet PTP clock • RCC_AHB1Periph_OTG_HS: USB OTG HS clock • RCC_AHB1Periph_OTG_HS_ULPI: USB OTG HS ULPI clock
NewState	new state of the specified peripheral clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.127.2.4 RCC_AHB1PeriphResetCmd()

```
void RCC_AHB1PeriphResetCmd (
            uint32_t RCC_AHB1Periph,
            FunctionalState NewState )
```

Forces or releases AHB1 peripheral reset.

Parameters

RCC_AHB1Periph	specifies the AHB1 peripheral to reset. This parameter can be any combination of the following values: • RCC_AHB1Periph_GPIOA: GPIOA clock • RCC_AHB1Periph_GPIOB: GPIOB clock • RCC_AHB1Periph_GPIOC: GPIOC clock • RCC_AHB1Periph_GPIOD: GPIOD clock • RCC_AHB1Periph_GPIOE: GPIOE clock • RCC_AHB1Periph_GPIOF: GPIOF clock • RCC_AHB1Periph_GPIOG: GPIOG clock • RCC_AHB1Periph_GPIOG: GPIOG clock • RCC_AHB1Periph_GPIOI: GPIOI clock • RCC_AHB1Periph_CRC: CRC clock • RCC_AHB1Periph_DMA1: DMA1 clock • RCC_AHB1Periph_DMA2: DMA2 clock • RCC_AHB1Periph_ETH_MAC: Ethernet MAC clock • RCC_AHB1Periph_OTG_HS: USB OTG HS clock
NewState	new state of the specified peripheral reset. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.127.2.5 RCC_AHB2PeriphClockCmd()

```
void RCC_AHB2PeriphClockCmd (
             uint32_t RCC_AHB2Periph,
             FunctionalState NewState )
```

Enables or disables the AHB2 peripheral clock.

Note

After reset, the peripheral clock (used for registers read/write access) is disabled and the application software has to enable this clock before using it.

Parameters

RCC_AHBPeriph	specifies the AHB2 peripheral to gates its clock. This parameter can be any combination of the following values: • RCC_AHB2Periph_DCMI: DCMI clock • RCC_AHB2Periph_CRYP: CRYP clock • RCC_AHB2Periph_HASH: HASH clock • RCC_AHB2Periph_RNG: RNG clock • RCC_AHB2Periph_OTG_FS: USB OTG FS clock
NewState	new state of the specified peripheral clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.127.2.6 RCC_AHB2PeriphClockLPModeCmd()

```
void RCC_AHB2PeriphClockLPModeCmd (
             uint32_t RCC_AHB2Periph,
             FunctionalState NewState )
```

Enables or disables the AHB2 peripheral clock during Low Power (Sleep) mode.

Note

Peripheral clock gating in SLEEP mode can be used to further reduce power consumption.

After wakeup from SLEEP mode, the peripheral clock is enabled again.

By default, all peripheral clocks are enabled during SLEEP mode.

Parameters

RCC_AHBPeriph	specifies the AHB2 peripheral to gates its clock. This parameter can be any combination of the following values: • RCC_AHB2Periph_DCMI: DCMI clock • RCC_AHB2Periph_CRYP: CRYP clock • RCC_AHB2Periph_HASH: HASH clock • RCC_AHB2Periph_RNG: RNG clock • RCC_AHB2Periph_OTG_FS: USB OTG FS clock
NewState	new state of the specified peripheral clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.127.2.7 RCC_AHB2PeriphResetCmd()

```
void RCC_AHB2PeriphResetCmd (
            uint32_t RCC_AHB2Periph,
            FunctionalState NewState )
```

Forces or releases AHB2 peripheral reset.

Parameters

RCC_AHB2Periph	specifies the AHB2 peripheral to reset. This parameter can be any combination of the following values: • RCC_AHB2Periph_DCMI: DCMI clock • RCC_AHB2Periph_CRYP: CRYP clock • RCC_AHB2Periph_HASH: HASH clock • RCC_AHB2Periph_RNG: RNG clock • RCC_AHB2Periph_OTG_FS: USB OTG FS clock
NewState	new state of the specified peripheral reset. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.127.2.8 RCC_AHB3PeriphClockCmd()

```
void RCC_AHB3PeriphClockCmd (
            uint32_t RCC_AHB3Periph,
            FunctionalState NewState )
```

Enables or disables the AHB3 peripheral clock.

Note

After reset, the peripheral clock (used for registers read/write access) is disabled and the application software has to enable this clock before using it.

Parameters

RCC_AHBPeriph	specifies the AHB3 peripheral to gates its clock. This parameter must be: RCC_AHB3Periph_FSMC
NewState	new state of the specified peripheral clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.127.2.9 RCC_AHB3PeriphClockLPModeCmd()

```
void RCC_AHB3PeriphClockLPModeCmd (
             uint32_t RCC_AHB3Periph,
             FunctionalState NewState )
```

Enables or disables the AHB3 peripheral clock during Low Power (Sleep) mode.

Note

Peripheral clock gating in SLEEP mode can be used to further reduce power consumption.

After wakeup from SLEEP mode, the peripheral clock is enabled again.

By default, all peripheral clocks are enabled during SLEEP mode.

Parameters

RCC_AHBPeriph	specifies the AHB3 peripheral to gates its clock. This parameter must be: RCC_AHB3Periph_FSMC
NewState	new state of the specified peripheral clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.127.2.10 RCC_AHB3PeriphResetCmd()

```
void RCC_AHB3PeriphResetCmd (
             uint32_t RCC_AHB3Periph,
             FunctionalState NewState )
```

Forces or releases AHB3 peripheral reset.

Parameters

RCC_AHB3Periph	specifies the AHB3 peripheral to reset. This parameter must be: RCC_AHB3Periph_FSMC
NewState	new state of the specified peripheral reset. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.127.2.11 RCC_APB1PeriphClockCmd()

```
void RCC_APB1PeriphClockCmd (
             uint32_t RCC_APB1Periph,
             FunctionalState NewState )
```

Enables or disables the Low Speed APB (APB1) peripheral clock.

Note

After reset, the peripheral clock (used for registers read/write access) is disabled and the application software has to enable this clock before using it.

Parameters

RCC_APB1Periph	specifies the APB1 peripheral to gates its clock. This parameter can be any combination of the following values: • RCC_APB1Periph_TIM2: TIM2 clock • RCC_APB1Periph_TIM3: TIM3 clock • RCC_APB1Periph_TIM4: TIM4 clock • RCC_APB1Periph_TIM5: TIM5 clock • RCC_APB1Periph_TIM6: TIM6 clock • RCC_APB1Periph_TIM7: TIM7 clock • RCC_APB1Periph_TIM12: TIM12 clock • RCC_APB1Periph_TIM13: TIM13 clock • RCC_APB1Periph_TIM14: TIM14 clock • RCC_APB1Periph_WWDG: WWDG clock • RCC_APB1Periph_SPI2: SPI2 clock • RCC_APB1Periph_SPI3: SPI3 clock • RCC_APB1Periph_USART2: USART2 clock • RCC_APB1Periph_USART3: USART3 clock • RCC_APB1Periph_UART4: UART4 clock • RCC_APB1Periph_UART5: UART5 clock • RCC_APB1Periph_I2C1: I2C1 clock • RCC_APB1Periph_I2C2: I2C2 clock • RCC_APB1Periph_I2C3: I2C3 clock • RCC_APB1Periph_CAN1: CAN1 clock • RCC_APB1Periph_CAN2: CAN2 clock • RCC_APB1Periph_PWR: PWR clock • RCC_APB1Periph_DAC: DAC clock
NewState	new state of the specified peripheral clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.127.2.12 RCC_APB1PeriphClockLPModeCmd()

```
void RCC_APB1PeriphClockLPModeCmd (
            uint32_t RCC_APB1Periph,
            FunctionalState NewState )
```

Enables or disables the APB1 peripheral clock during Low Power (Sleep) mode.

Note

Peripheral clock gating in SLEEP mode can be used to further reduce power consumption.

After wakeup from SLEEP mode, the peripheral clock is enabled again.

By default, all peripheral clocks are enabled during SLEEP mode.

Parameters

RCC_APB1Periph	specifies the APB1 peripheral to gates its clock. This parameter can be any combination of the following values: • RCC_APB1Periph_TIM2: TIM2 clock • RCC_APB1Periph_TIM3: TIM3 clock • RCC_APB1Periph_TIM4: TIM4 clock • RCC_APB1Periph_TIM5: TIM5 clock • RCC_APB1Periph_TIM6: TIM6 clock • RCC_APB1Periph_TIM7: TIM7 clock • RCC_APB1Periph_TIM12: TIM12 clock • RCC_APB1Periph_TIM13: TIM13 clock • RCC_APB1Periph_TIM14: TIM14 clock • RCC_APB1Periph_WWDG: WWDG clock • RCC_APB1Periph_SPI2: SPI2 clock • RCC_APB1Periph_SPI3: SPI3 clock • RCC_APB1Periph_USART2: USART2 clock • RCC_APB1Periph_USART3: USART3 clock • RCC_APB1Periph_UART4: UART4 clock • RCC_APB1Periph_UART5: UART5 clock • RCC_APB1Periph_I2C1: I2C1 clock • RCC_APB1Periph_I2C2: I2C2 clock • RCC_APB1Periph_I2C3: I2C3 clock • RCC_APB1Periph_CAN1: CAN1 clock • RCC_APB1Periph_CAN2: CAN2 clock • RCC_APB1Periph_PWR: PWR clock • RCC_APB1Periph_DAC: DAC clock
NewState	new state of the specified peripheral clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.127.2.13 RCC_APB1PeriphResetCmd()

```
void RCC_APB1PeriphResetCmd (
            uint32_t RCC_APB1Periph,
            FunctionalState NewState )
```

Forces or releases Low Speed APB (APB1) peripheral reset.

Parameters

RCC_APB1Periph	specifies the APB1 peripheral to reset. This parameter can be any combination of the following values: • RCC_APB1Periph_TIM2: TIM2 clock • RCC_APB1Periph_TIM3: TIM3 clock • RCC_APB1Periph_TIM4: TIM4 clock • RCC_APB1Periph_TIM5: TIM5 clock • RCC_APB1Periph_TIM6: TIM6 clock • RCC_APB1Periph_TIM7: TIM7 clock • RCC_APB1Periph_TIM12: TIM12 clock • RCC_APB1Periph_TIM13: TIM13 clock • RCC_APB1Periph_TIM14: TIM14 clock • RCC_APB1Periph_WWDG: WWDG clock • RCC_APB1Periph_SPI2: SPI2 clock • RCC_APB1Periph_SPI3: SPI3 clock • RCC_APB1Periph_USART2: USART2 clock • RCC_APB1Periph_USART3: USART3 clock • RCC_APB1Periph_UART4: UART4 clock • RCC_APB1Periph_UART5: UART5 clock • RCC_APB1Periph_I2C1: I2C1 clock • RCC_APB1Periph_I2C2: I2C2 clock • RCC_APB1Periph_I2C3: I2C3 clock • RCC_APB1Periph_CAN1: CAN1 clock • RCC_APB1Periph_CAN2: CAN2 clock • RCC_APB1Periph_PWR: PWR clock • RCC_APB1Periph_DAC: DAC clock
NewState	new state of the specified peripheral reset. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.127.2.14 RCC_APB2PeriphClockCmd()

```
void RCC_APB2PeriphClockCmd (
            uint32_t RCC_APB2Periph,
            FunctionalState NewState )
```

Enables or disables the High Speed APB (APB2) peripheral clock.

Note

After reset, the peripheral clock (used for registers read/write access) is disabled and the application software has to enable this clock before using it.

Parameters

RCC_APB2Periph	specifies the APB2 peripheral to gates its clock. This parameter can be any combination of the following values: • RCC_APB2Periph_TIM1: TIM1 clock • RCC_APB2Periph_TIM8: TIM8 clock • RCC_APB2Periph_USART1: USART1 clock • RCC_APB2Periph_USART6: USART6 clock • RCC_APB2Periph_ADC1: ADC1 clock • RCC_APB2Periph_ADC2: ADC2 clock • RCC_APB2Periph_ADC3: ADC3 clock • RCC_APB2Periph_SDIO: SDIO clock • RCC_APB2Periph_SPI1: SPI1 clock • RCC_APB2Periph_SYSCFG: SYSCFG clock • RCC_APB2Periph_TIM9: TIM9 clock • RCC_APB2Periph_TIM10: TIM10 clock • RCC_APB2Periph_TIM11: TIM11 clock
NewState	new state of the specified peripheral clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.127.2.15 RCC_APB2PeriphClockLPModeCmd()

```
void RCC_APB2PeriphClockLPModeCmd (
            uint32_t RCC_APB2Periph,
            FunctionalState NewState )
```

Enables or disables the APB2 peripheral clock during Low Power (Sleep) mode.

Note

Peripheral clock gating in SLEEP mode can be used to further reduce power consumption.

After wakeup from SLEEP mode, the peripheral clock is enabled again.

By default, all peripheral clocks are enabled during SLEEP mode.

Parameters

RCC_APB2Periph	specifies the APB2 peripheral to gates its clock. This parameter can be any combination of the following values: • RCC_APB2Periph_TIM1: TIM1 clock • RCC_APB2Periph_TIM8: TIM8 clock • RCC_APB2Periph_USART1: USART1 clock • RCC_APB2Periph_USART6: USART6 clock • RCC_APB2Periph_ADC1: ADC1 clock • RCC_APB2Periph_ADC2: ADC2 clock • RCC_APB2Periph_ADC3: ADC3 clock • RCC_APB2Periph_SDIO: SDIO clock • RCC_APB2Periph_SPI1: SPI1 clock • RCC_APB2Periph_SYSCFG: SYSCFG clock • RCC_APB2Periph_TIM9: TIM9 clock • RCC_APB2Periph_TIM10: TIM10 clock • RCC_APB2Periph_TIM11: TIM11 clock
NewState	new state of the specified peripheral clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.127.2.16 RCC_APB2PeriphResetCmd()

```
void RCC_APB2PeriphResetCmd (
            uint32_t RCC_APB2Periph,
            FunctionalState NewState )
```

Forces or releases High Speed APB (APB2) peripheral reset.

Parameters

RCC_APB2Periph	specifies the APB2 peripheral to reset. This parameter can be any combination of the following values: • RCC_APB2Periph_TIM1: TIM1 clock • RCC_APB2Periph_TIM8: TIM8 clock • RCC_APB2Periph_USART1: USART1 clock • RCC_APB2Periph_USART6: USART6 clock • RCC_APB2Periph_ADC1: ADC1 clock • RCC_APB2Periph_ADC2: ADC2 clock • RCC_APB2Periph_ADC3: ADC3 clock • RCC_APB2Periph_SDIO: SDIO clock • RCC_APB2Periph_SPI1: SPI1 clock • RCC_APB2Periph_SYSCFG: SYSCFG clock • RCC_APB2Periph_TIM9: TIM9 clock • RCC_APB2Periph_TIM10: TIM10 clock • RCC_APB2Periph_TIM11: TIM11 clock
NewState	new state of the specified peripheral reset. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.127.2.17 RCC_BackupResetCmd()

```
void RCC_BackupResetCmd (
             FunctionalState NewState )
```

Forces or releases the Backup domain reset.

Note

This function resets the RTC peripheral (including the backup registers) and the RTC clock source selection in RCC_CSR register.

The BKPSRAM is not affected by this reset.

Parameters

NewState	new state of the Backup domain reset. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.127.2.18 RCC_ClearFlag()

```
void RCC_ClearFlag (
             void  )
```

Clears the RCC reset flags. The reset flags are: RCC_FLAG_PINRST, RCC_FLAG_PORRST, RCC_FLAG_↩ SFTRST, RCC_FLAG_IWDGRST, RCC_FLAG_WWDGRST, RCC_FLAG_LPWRRST.

Parameters

None	

Return values

None	

5.127.2.19 RCC_ClearITPendingBit()

```
void RCC_ClearITPendingBit (
             uint8_t RCC_IT )
```

Clears the RCC's interrupt pending bits.

Parameters

RCC↩ _IT	specifies the interrupt pending bit to clear. This parameter can be any combination of the following values: • RCC_IT_LSIRDY: LSI ready interrupt • RCC_IT_LSERDY: LSE ready interrupt • RCC_IT_HSIRDY: HSI ready interrupt • RCC_IT_HSERDY: HSE ready interrupt • RCC_IT_PLLRDY: main PLL ready interrupt • RCC_IT_PLLI2SRDY: PLLI2S ready interrupt • RCC_IT_CSS: Clock Security System interrupt

Return values

None	

5.127.2.20 RCC_ClockSecuritySystemCmd()

```
void RCC_ClockSecuritySystemCmd (
             FunctionalState NewState )
```

Enables or disables the Clock Security System.

Note

If a failure is detected on the HSE oscillator clock, this oscillator is automatically disabled and an interrupt is generated to inform the software about the failure (Clock Security System Interrupt, CSSI), allowing the MCU to perform rescue operations. The CSSI is linked to the Cortex-M4 NMI (Non-Maskable Interrupt) exception vector.

Parameters

NewState	new state of the Clock Security System. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.127.2.21 RCC_DeInit()

```
void RCC_DeInit (
             void  )
```

Resets the RCC clock configuration to the default reset state.

Note

The default reset state of the clock configuration is given below:

- HSI ON and used as system clock source
- HSE, PLL and PLLI2S OFF
- AHB, APB1 and APB2 prescaler set to 1.
- CSS, MCO1 and MCO2 OFF
- All interrupts disabled

This function doesn't modify the configuration of the

- Peripheral clocks
- LSI, LSE and RTC clocks

Parameters

None	

Return values

None	

5.127.2.22 RCC_GetClocksFreq()

```
void RCC_GetClocksFreq (
            RCC_ClocksTypeDef * RCC_Clocks )
```

Returns the frequencies of different on chip clocks; SYSCLK, HCLK, PCLK1 and PCLK2.

Note

The system frequency computed by this function is not the real frequency in the chip. It is calculated based on the predefined constant and the selected clock source:

If SYSCLK source is HSI, function returns values based on HSI_VALUE(∗)

If SYSCLK source is HSE, function returns values based on HSE_VALUE(∗∗)

If SYSCLK source is PLL, function returns values based on HSE_VALUE(∗∗) or HSI_VALUE(∗) multiplied/divided by the PLL factors.

(∗) HSI_VALUE is a constant defined in stm32f4xx.h file (default value 16 MHz) but the real value may vary depending on the variations in voltage and temperature.

(∗∗) HSE_VALUE is a constant defined in stm32f4xx.h file (default value 25 MHz), user has to ensure that HSE_VALUE is same as the real frequency of the crystal used. Otherwise, this function may have wrong result.

The result of this function could be not correct when using fractional value for HSE crystal.

Parameters

RCC_Clocks	pointer to a RCC_ClocksTypeDef structure which will hold the clocks frequencies.

Note

This function can be used by the user application to compute the baudrate for the communication peripherals or configure other parameters.

Each time SYSCLK, HCLK, PCLK1 and/or PCLK2 clock changes, this function must be called to update the structure's field. Otherwise, any configuration based on this function will be incorrect.

Return values

None	

5.127.2.23 RCC_GetFlagStatus()

```
FlagStatus RCC_GetFlagStatus (
            uint8_t RCC_FLAG )
```

Checks whether the specified RCC flag is set or not.

Parameters

RCC_FLAG	specifies the flag to check. This parameter can be one of the following values: • RCC_FLAG_HSIRDY: HSI oscillator clock ready • RCC_FLAG_HSERDY: HSE oscillator clock ready • RCC_FLAG_PLLRDY: main PLL clock ready • RCC_FLAG_PLLI2SRDY: PLLI2S clock ready • RCC_FLAG_LSERDY: LSE oscillator clock ready • RCC_FLAG_LSIRDY: LSI oscillator clock ready • RCC_FLAG_BORRST: POR/PDR or BOR reset • RCC_FLAG_PINRST: Pin reset • RCC_FLAG_PORRST: POR/PDR reset • RCC_FLAG_SFTRST: Software reset • RCC_FLAG_IWDGRST: Independent Watchdog reset • RCC_FLAG_WWDGRST: Window Watchdog reset • RCC_FLAG_LPWRRST: Low Power reset

Return values

The	new state of RCC_FLAG (SET or RESET).

5.127.2.24 RCC_GetITStatus()

```
ITStatus RCC_GetITStatus (
            uint8_t RCC_IT )
```

Checks whether the specified RCC interrupt has occurred or not.

Parameters

RCC_IT	specifies the RCC interrupt source to check. This parameter can be one of the following values: • RCC_IT_LSIRDY: LSI ready interrupt • RCC_IT_LSERDY: LSE ready interrupt • RCC_IT_HSIRDY: HSI ready interrupt • RCC_IT_HSERDY: HSE ready interrupt • RCC_IT_PLLRDY: main PLL ready interrupt • RCC_IT_PLLI2SRDY: PLLI2S ready interrupt • RCC_IT_CSS: Clock Security System interrupt

Return values

The	new state of RCC_IT (SET or RESET).

5.127.2.25 RCC_GetSYSCLKSource()

```
uint8_t RCC_GetSYSCLKSource (
             void  )
```

Returns the clock source used as system clock.

Parameters

None	

Return values

The	clock source used as system clock. The returned value can be one of the following: • 0x00: HSI used as system clock • 0x04: HSE used as system clock • 0x08: PLL used as system clock

5.127.2.26 RCC_HCLKConfig()

```
void RCC_HCLKConfig (
             uint32_t RCC_SYSCLK )
```

Configures the AHB clock (HCLK).

Note

Depending on the device voltage range, the software has to set correctly these bits to ensure that HCLK not exceed the maximum allowed frequency (for more details refer to section above "CPU, AHB and APB busses clocks configuration functions")

Parameters

RCC_SYSCLK	defines the AHB clock divider. This clock is derived from the system clock (SYSCLK). This parameter can be one of the following values: • RCC_SYSCLK_Div1: AHB clock = SYSCLK • RCC_SYSCLK_Div2: AHB clock = SYSCLK/2 • RCC_SYSCLK_Div4: AHB clock = SYSCLK/4 • RCC_SYSCLK_Div8: AHB clock = SYSCLK/8 • RCC_SYSCLK_Div16: AHB clock = SYSCLK/16 • RCC_SYSCLK_Div64: AHB clock = SYSCLK/64 • RCC_SYSCLK_Div128: AHB clock = SYSCLK/128 • RCC_SYSCLK_Div256: AHB clock = SYSCLK/256 • RCC_SYSCLK_Div512: AHB clock = SYSCLK/512

Return values

None	

5.127.2.27 RCC_HSEConfig()

```
void RCC_HSEConfig (
             uint8_t RCC_HSE )
```

Configures the External High Speed oscillator (HSE).

Note

After enabling the HSE (RCC_HSE_ON or RCC_HSE_Bypass), the application software should wait on HSERDY flag to be set indicating that HSE clock is stable and can be used to clock the PLL and/or system clock.

HSE state can not be changed if it is used directly or through the PLL as system clock. In this case, you have to select another source of the system clock then change the HSE state (ex. disable it).

The HSE is stopped by hardware when entering STOP and STANDBY modes.

This function reset the CSSON bit, so if the Clock security system(CSS) was previously enabled you have to enable it again after calling this function.

Parameters

RCC_HSE	specifies the new state of the HSE. This parameter can be one of the following values: • RCC_HSE_OFF: turn OFF the HSE oscillator, HSERDY flag goes low after 6 HSE oscillator clock cycles. • RCC_HSE_ON: turn ON the HSE oscillator • RCC_HSE_Bypass: HSE oscillator bypassed with external clock

Return values

None	

5.127.2.28 RCC_HSICmd()

```
void RCC_HSICmd (
             FunctionalState NewState )
```

Enables or disables the Internal High Speed oscillator (HSI).

Note

The HSI is stopped by hardware when entering STOP and STANDBY modes. It is used (enabled by hardware) as system clock source after startup from Reset, wakeup from STOP and STANDBY mode, or in case of failure of the HSE used directly or indirectly as system clock (if the Clock Security System CSS is enabled).

HSI can not be stopped if it is used as system clock source. In this case, you have to select another source of the system clock then stop the HSI.

After enabling the HSI, the application software should wait on HSIRDY flag to be set indicating that HSI clock is stable and can be used as system clock source.

Parameters

NewState	new state of the HSI. This parameter can be: ENABLE or DISABLE.

Note

When the HSI is stopped, HSIRDY flag goes low after 6 HSI oscillator clock cycles.

Return values

None	

5.127.2.29 RCC_I2SCLKConfig()

```
void RCC_I2SCLKConfig (
             uint32_t RCC_I2SCLKSource )
```

Configures the I2S clock source (I2SCLK).

Note

This function must be called before enabling the I2S APB clock.

Parameters

RCC_I2SCLKSource	specifies the I2S clock source. This parameter can be one of the following values: • RCC_I2S2CLKSource_PLLI2S: PLLI2S clock used as I2S clock source • RCC_I2S2CLKSource_Ext: External clock mapped on the I2S_CKIN pin used as I2S clock source

Return values

None	

5.127.2.30 RCC_ITConfig()

```
void RCC_ITConfig (
             uint8_t RCC_IT,
             FunctionalState NewState )
```

Enables or disables the specified RCC interrupts.

Parameters

RCC_IT	specifies the RCC interrupt sources to be enabled or disabled. This parameter can be any combination of the following values: • RCC_IT_LSIRDY: LSI ready interrupt • RCC_IT_LSERDY: LSE ready interrupt • RCC_IT_HSIRDY: HSI ready interrupt • RCC_IT_HSERDY: HSE ready interrupt • RCC_IT_PLLRDY: main PLL ready interrupt • RCC_IT_PLLI2SRDY: PLLI2S ready interrupt
NewState	new state of the specified RCC interrupts. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.127.2.31 RCC_LSEConfig()

```
void RCC_LSEConfig (
            uint8_t RCC_LSE )
```

Configures the External Low Speed oscillator (LSE).

Note

As the LSE is in the Backup domain and write access is denied to this domain after reset, you have to enable write access using PWR_BackupAccessCmd(ENABLE) function before to configure the LSE (to be done once after reset).

After enabling the LSE (RCC_LSE_ON or RCC_LSE_Bypass), the application software should wait on LSERDY flag to be set indicating that LSE clock is stable and can be used to clock the RTC.

Parameters

RCC_LSE	specifies the new state of the LSE. This parameter can be one of the following values: • RCC_LSE_OFF: turn OFF the LSE oscillator, LSERDY flag goes low after 6 LSE oscillator clock cycles. • RCC_LSE_ON: turn ON the LSE oscillator • RCC_LSE_Bypass: LSE oscillator bypassed with external clock

Return values

None	

5.127.2.32 RCC_LSICmd()

```
void RCC_LSICmd (
            FunctionalState NewState )
```

Enables or disables the Internal Low Speed oscillator (LSI).

Note

After enabling the LSI, the application software should wait on LSIRDY flag to be set indicating that LSI clock is stable and can be used to clock the IWDG and/or the RTC.

LSI can not be disabled if the IWDG is running.

Parameters

NewState	new state of the LSI. This parameter can be: ENABLE or DISABLE.

Note

When the LSI is stopped, LSIRDY flag goes low after 6 LSI oscillator clock cycles.

Return values

None	

5.127.2.33 RCC_MCO1Config()

```
void RCC_MCO1Config (
             uint32_t RCC_MCO1Source,
             uint32_t RCC_MCO1Div )
```

Selects the clock source to output on MCO1 pin(PA8).

Note

PA8 should be configured in alternate function mode.

Parameters

RCC_MCO1Source	specifies the clock source to output. This parameter can be one of the following values: • RCC_MCO1Source_HSI: HSI clock selected as MCO1 source • RCC_MCO1Source_LSE: LSE clock selected as MCO1 source • RCC_MCO1Source_HSE: HSE clock selected as MCO1 source • RCC_MCO1Source_PLLCLK: main PLL clock selected as MCO1 source
RCC_MCO1Div	specifies the MCO1 prescaler. This parameter can be one of the following values: • RCC_MCO1Div_1: no division applied to MCO1 clock • RCC_MCO1Div_2: division by 2 applied to MCO1 clock • RCC_MCO1Div_3: division by 3 applied to MCO1 clock • RCC_MCO1Div_4: division by 4 applied to MCO1 clock • RCC_MCO1Div_5: division by 5 applied to MCO1 clock

Return values

None	

5.127.2.34 RCC_MCO2Config()

```
void RCC_MCO2Config (
            uint32_t RCC_MCO2Source,
            uint32_t RCC_MCO2Div )
```

Selects the clock source to output on MCO2 pin(PC9).

Note

PC9 should be configured in alternate function mode.

Parameters

RCC_MCO2Source	specifies the clock source to output. This parameter can be one of the following values: • RCC_MCO2Source_SYSCLK: System clock (SYSCLK) selected as MCO2 source • RCC_MCO2Source_PLLI2SCLK: PLLI2S clock selected as MCO2 source • RCC_MCO2Source_HSE: HSE clock selected as MCO2 source • RCC_MCO2Source_PLLCLK: main PLL clock selected as MCO2 source
RCC_MCO2Div	specifies the MCO2 prescaler. This parameter can be one of the following values: • RCC_MCO2Div_1: no division applied to MCO2 clock • RCC_MCO2Div_2: division by 2 applied to MCO2 clock • RCC_MCO2Div_3: division by 3 applied to MCO2 clock • RCC_MCO2Div_4: division by 4 applied to MCO2 clock • RCC_MCO2Div_5: division by 5 applied to MCO2 clock

Return values

None	

5.127.2.35 RCC_PCLK1Config()

```
void RCC_PCLK1Config (
            uint32_t RCC_HCLK )
```

Configures the Low Speed APB clock (PCLK1).

Parameters

RCC_HCLK	defines the APB1 clock divider. This clock is derived from the AHB clock (HCLK). This parameter can be one of the following values: • RCC_HCLK_Div1: APB1 clock = HCLK • RCC_HCLK_Div2: APB1 clock = HCLK/2 • RCC_HCLK_Div4: APB1 clock = HCLK/4 • RCC_HCLK_Div8: APB1 clock = HCLK/8 • RCC_HCLK_Div16: APB1 clock = HCLK/16

Return values

None	

5.127.2.36 RCC_PCLK2Config()

```
void RCC_PCLK2Config (
             uint32_t RCC_HCLK )
```

Configures the High Speed APB clock (PCLK2).

Parameters

RCC_HCLK	defines the APB2 clock divider. This clock is derived from the AHB clock (HCLK). This parameter can be one of the following values: • RCC_HCLK_Div1: APB2 clock = HCLK • RCC_HCLK_Div2: APB2 clock = HCLK/2 • RCC_HCLK_Div4: APB2 clock = HCLK/4 • RCC_HCLK_Div8: APB2 clock = HCLK/8 • RCC_HCLK_Div16: APB2 clock = HCLK/16

Return values

None	

5.127.2.37 RCC_PLLCmd()

```
void RCC_PLLCmd (
             FunctionalState NewState )
```

Enables or disables the main PLL.

Note

After enabling the main PLL, the application software should wait on PLLRDY flag to be set indicating that PLL clock is stable and can be used as system clock source.

The main PLL can not be disabled if it is used as system clock source

The main PLL is disabled by hardware when entering STOP and STANDBY modes.

Parameters

NewState	new state of the main PLL. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.127.2.38 RCC_PLLConfig()

```
void RCC_PLLConfig (
             uint32_t RCC_PLLSource,
             uint32_t PLLM,
             uint32_t PLLN,
             uint32_t PLLP,
             uint32_t PLLQ )
```

Configures the main PLL clock source, multiplication and division factors.

Note

This function must be used only when the main PLL is disabled.

Parameters

RCC_PLLSource	specifies the PLL entry clock source. This parameter can be one of the following values: • RCC_PLLSource_HSI: HSI oscillator clock selected as PLL clock entry • RCC_PLLSource_HSE: HSE oscillator clock selected as PLL clock entry

Note

This clock source (RCC_PLLSource) is common for the main PLL and PLLI2S.

Parameters

PLLM	specifies the division factor for PLL VCO input clock This parameter must be a number between 0 and 63.

Note

You have to set the PLLM parameter correctly to ensure that the VCO input frequency ranges from 1 to 2 MHz. It is recommended to select a frequency of 2 MHz to limit PLL jitter.

Parameters

PLLN	specifies the multiplication factor for PLL VCO output clock This parameter must be a number between 192 and 432.

Note

You have to set the PLLN parameter correctly to ensure that the VCO output frequency is between 192 and 432 MHz.

Parameters

PLLP	specifies the division factor for main system clock (SYSCLK) This parameter must be a number in the range {2, 4, 6, or 8}.

Note

You have to set the PLLP parameter correctly to not exceed 168 MHz on the System clock frequency.

Parameters

PLLQ	specifies the division factor for OTG FS, SDIO and RNG clocks This parameter must be a number between 4 and 15.

Note

If the USB OTG FS is used in your application, you have to set the PLLQ parameter correctly to have 48 MHz clock for the USB. However, the SDIO and RNG need a frequency lower than or equal to 48 MHz to work correctly.

Return values

None	

5.127.2.39 RCC_PLLI2SCmd()

```
void RCC_PLLI2SCmd (
             FunctionalState NewState )
```

Enables or disables the PLLI2S.

Note

The PLLI2S is disabled by hardware when entering STOP and STANDBY modes.

Parameters

NewState	new state of the PLLI2S. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.127.2.40 RCC_PLLI2SConfig()

```
void RCC_PLLI2SConfig (
             uint32_t PLLI2SN,
             uint32_t PLLI2SR )
```

Configures the PLLI2S clock multiplication and division factors.

Note

This function must be used only when the PLLI2S is disabled.

PLLI2S clock source is common with the main PLL (configured in RCC_PLLConfig function)

Parameters

PLLI2SN	specifies the multiplication factor for PLLI2S VCO output clock This parameter must be a number between 192 and 432.

Note

You have to set the PLLI2SN parameter correctly to ensure that the VCO output frequency is between 192 and 432 MHz.

Parameters

PLLI2SR	specifies the division factor for I2S clock This parameter must be a number between 2 and 7.

Note

You have to set the PLLI2SR parameter correctly to not exceed 192 MHz on the I2S clock frequency.

Return values

None	

5.127.2.41 RCC_RTCCLKCmd()

```
void RCC_RTCCLKCmd (
             FunctionalState NewState )
```

Enables or disables the RTC clock.

Note

This function must be used only after the RTC clock source was selected using the RCC_RTCCLKConfig function.

Parameters

NewState	new state of the RTC clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.127.2.42 RCC_RTCCLKConfig()

```
void RCC_RTCCLKConfig (
             uint32_t RCC_RTCCLKSource )
```

Configures the RTC clock (RTCCLK).

Note

As the RTC clock configuration bits are in the Backup domain and write access is denied to this domain after reset, you have to enable write access using PWR_BackupAccessCmd(ENABLE) function before to configure the RTC clock source (to be done once after reset).

Once the RTC clock is configured it can't be changed unless the
Backup domain is reset using RCC_BackupResetCmd() function, or by a Power On Reset (POR).

Parameters

RCC_RTCCLKSource	specifies the RTC clock source. This parameter can be one of the following values: • RCC_RTCCLKSource_LSE: LSE selected as RTC clock • RCC_RTCCLKSource_LSI: LSI selected as RTC clock • RCC_RTCCLKSource_HSE_Divx: HSE clock divided by x selected as RTC clock, where x:[2,31]

Note

If the LSE or LSI is used as RTC clock source, the RTC continues to work in STOP and STANDBY modes, and can be used as wakeup source. However, when the HSE clock is used as RTC clock source, the RTC cannot be used in STOP and STANDBY modes.

The maximum input clock frequency for RTC is 1MHz (when using HSE as RTC clock source).

Return values

None	

5.127.2.43 RCC_SYSCLKConfig()

```
void RCC_SYSCLKConfig (
            uint32_t RCC_SYSCLKSource )
```

Configures the system clock (SYSCLK).

Note

The HSI is used (enabled by hardware) as system clock source after startup from Reset, wake-up from STOP and STANDBY mode, or in case of failure of the HSE used directly or indirectly as system clock (if the Clock Security System CSS is enabled).

A switch from one clock source to another occurs only if the target clock source is ready (clock stable after startup delay or PLL locked). If a clock source which is not yet ready is selected, the switch will occur when the clock source will be ready. You can use RCC_GetSYSCLKSource() function to know which clock is currently used as system clock source.

Parameters

RCC_SYSCLKSource	specifies the clock source used as system clock. This parameter can be one of the following values: • RCC_SYSCLKSource_HSI: HSI selected as system clock source • RCC_SYSCLKSource_HSE: HSE selected as system clock source • RCC_SYSCLKSource_PLLCLK: PLL selected as system clock source

Return values

None	

5.127.2.44 RCC_WaitForHSEStartUp()

```
ErrorStatus RCC_WaitForHSEStartUp (
```

```
            void  )
```

Waits for HSE start-up.

Note

This functions waits on HSERDY flag to be set and return SUCCESS if this flag is set, otherwise returns ERROR if the timeout is reached and this flag is not set. The timeout value is defined by the constant HSE↩ _STARTUP_TIMEOUT in stm32f4xx.h file. You can tailor it depending on the HSE crystal used in your application.

Parameters

None	

Return values

An	ErrorStatus enumeration value: • SUCCESS: HSE oscillator is stable and ready to use • ERROR: HSE oscillator not yet ready

5.128 RCC_Private_Functions

Modules

- Internal and external clocks, PLL, CSS and MCO configuration functions

  Internal and external clocks, PLL, CSS and MCO configuration functions.
- System AHB and APB busses clocks configuration functions

  System, AHB and APB busses clocks configuration functions.
- Peripheral clocks configuration functions

  Peripheral clocks configuration functions.
- Interrupts and flags management functions

  Interrupts and flags management functions.

5.128.1 Detailed Description

5.129 Internal and external clocks, PLL, CSS and MCO configuration functions

Internal and external clocks, PLL, CSS and MCO configuration functions.

Functions

- void RCC_DeInit (void)

  Resets the RCC clock configuration to the default reset state.
- void RCC_HSEConfig (uint8_t RCC_HSE)

  Configures the External High Speed oscillator (HSE).
- ErrorStatus RCC_WaitForHSEStartUp (void)

  Waits for HSE start-up.
- void RCC_AdjustHSICalibrationValue (uint8_t HSICalibrationValue)

  Adjusts the Internal High Speed oscillator (HSI) calibration value.
- void RCC_HSICmd (FunctionalState NewState)

  Enables or disables the Internal High Speed oscillator (HSI).
- void RCC_LSEConfig (uint8_t RCC_LSE)

  Configures the External Low Speed oscillator (LSE).
- void RCC_LSICmd (FunctionalState NewState)

  Enables or disables the Internal Low Speed oscillator (LSI).
- void RCC_PLLConfig (uint32_t RCC_PLLSource, uint32_t PLLM, uint32_t PLLN, uint32_t PLLP, uint32_t PLLQ)

  Configures the main PLL clock source, multiplication and division factors.
- void RCC_PLLCmd (FunctionalState NewState)

  Enables or disables the main PLL.
- void RCC_PLLI2SConfig (uint32_t PLLI2SN, uint32_t PLLI2SR)

  Configures the PLLI2S clock multiplication and division factors.
- void RCC_PLLI2SCmd (FunctionalState NewState)

  Enables or disables the PLLI2S.
- void RCC_ClockSecuritySystemCmd (FunctionalState NewState)

  Enables or disables the Clock Security System.
- void RCC_MCO1Config (uint32_t RCC_MCO1Source, uint32_t RCC_MCO1Div)

  Selects the clock source to output on MCO1 pin(PA8).
- void RCC_MCO2Config (uint32_t RCC_MCO2Source, uint32_t RCC_MCO2Div)

  Selects the clock source to output on MCO2 pin(PC9).

5.129.1 Detailed Description

Internal and external clocks, PLL, CSS and MCO configuration functions.

```
 ================================================================================
      Internal/external clocks, PLL, CSS and MCO configuration functions
 ================================================================================

  This section provide functions allowing to configure the internal/external clocks,
  PLLs, CSS and MCO pins.

  1. HSI (high-speed internal), 16 MHz factory-trimmed RC used directly or through
     the PLL as System clock source.

  2. LSI (low-speed internal), 32 KHz low consumption RC used as IWDG and/or RTC
     clock source.

  3. HSE (high-speed external), 4 to 26 MHz crystal oscillator used directly or
     through the PLL as System clock source. Can be used also as RTC clock source.

  4. LSE (low-speed external), 32 KHz oscillator used as RTC clock source.

  5. PLL (clocked by HSI or HSE), featuring two different output clocks:
      - The first output is used to generate the high speed system clock (up to 168 MHz)
```

```
     - The second output is used to generate the clock for the USB OTG FS (48 MHz),
       the random analog generator (<=48 MHz) and the SDIO (<= 48 MHz).

  6. PLLI2S (clocked by HSI or HSE), used to generate an accurate clock to achieve
     high-quality audio performance on the I2S interface.

  7. CSS (Clock security system), once enable and if a HSE clock failure occurs
     (HSE used directly or through PLL as System clock source), the System clock
     is automatically switched to HSI and an interrupt is generated if enabled.
     The interrupt is linked to the Cortex-M4 NMI (Non-Maskable Interrupt)
     exception vector.

  8. MCO1 (microcontroller clock output), used to output HSI, LSE, HSE or PLL
     clock (through a configurable prescaler) on PA8 pin.

  9. MCO2 (microcontroller clock output), used to output HSE, PLL, SYSCLK or PLLI2S
     clock (through a configurable prescaler) on PC9 pin.
```

5.129.2 Function Documentation

5.129.2.1 RCC_AdjustHSICalibrationValue()

```
void RCC_AdjustHSICalibrationValue (
             uint8_t HSICalibrationValue )
```

Adjusts the Internal High Speed oscillator (HSI) calibration value.

Note

The calibration is used to compensate for the variations in voltage and temperature that influence the frequency of the internal HSI RC.

Parameters

HSICalibrationValue	specifies the calibration trimming value. This parameter must be a number between 0 and 0x1F.

Return values

None	

5.129.2.2 RCC_ClockSecuritySystemCmd()

```
void RCC_ClockSecuritySystemCmd (
             FunctionalState NewState )
```

Enables or disables the Clock Security System.

Note

If a failure is detected on the HSE oscillator clock, this oscillator is automatically disabled and an interrupt is generated to inform the software about the failure (Clock Security System Interrupt, CSSI), allowing the MCU to perform rescue operations. The CSSI is linked to the Cortex-M4 NMI (Non-Maskable Interrupt) exception vector.

Parameters

NewState	new state of the Clock Security System. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.129.2.3 RCC_DeInit()

```
void RCC_DeInit (
             void  )
```

Resets the RCC clock configuration to the default reset state.

Note

The default reset state of the clock configuration is given below:

- HSI ON and used as system clock source
- HSE, PLL and PLLI2S OFF
- AHB, APB1 and APB2 prescaler set to 1.
- CSS, MCO1 and MCO2 OFF
- All interrupts disabled

This function doesn't modify the configuration of the

- Peripheral clocks
- LSI, LSE and RTC clocks

Parameters

None	

Return values

None	

5.129.2.4 RCC_HSEConfig()

```
void RCC_HSEConfig (
            uint8_t RCC_HSE )
```

Configures the External High Speed oscillator (HSE).

Note

After enabling the HSE (RCC_HSE_ON or RCC_HSE_Bypass), the application software should wait on HSERDY flag to be set indicating that HSE clock is stable and can be used to clock the PLL and/or system clock.

HSE state can not be changed if it is used directly or through the PLL as system clock. In this case, you have to select another source of the system clock then change the HSE state (ex. disable it).

The HSE is stopped by hardware when entering STOP and STANDBY modes.

This function reset the CSSON bit, so if the Clock security system(CSS) was previously enabled you have to enable it again after calling this function.

Parameters

RCC_HSE	specifies the new state of the HSE. This parameter can be one of the following values: • RCC_HSE_OFF: turn OFF the HSE oscillator, HSERDY flag goes low after 6 HSE oscillator clock cycles. • RCC_HSE_ON: turn ON the HSE oscillator • RCC_HSE_Bypass: HSE oscillator bypassed with external clock

Return values

None	

5.129.2.5 RCC_HSICmd()

```
void RCC_HSICmd (
            FunctionalState NewState )
```

Enables or disables the Internal High Speed oscillator (HSI).

Note

The HSI is stopped by hardware when entering STOP and STANDBY modes. It is used (enabled by hardware) as system clock source after startup from Reset, wakeup from STOP and STANDBY mode, or in case of failure of the HSE used directly or indirectly as system clock (if the Clock Security System CSS is enabled).

HSI can not be stopped if it is used as system clock source. In this case, you have to select another source of the system clock then stop the HSI.

After enabling the HSI, the application software should wait on HSIRDY flag to be set indicating that HSI clock is stable and can be used as system clock source.

Parameters

NewState	new state of the HSI. This parameter can be: ENABLE or DISABLE.

Note

When the HSI is stopped, HSIRDY flag goes low after 6 HSI oscillator clock cycles.

Return values

None	

5.129.2.6 RCC_LSEConfig()

```
void RCC_LSEConfig (
            uint8_t RCC_LSE )
```

Configures the External Low Speed oscillator (LSE).

Note

As the LSE is in the Backup domain and write access is denied to this domain after reset, you have to enable write access using PWR_BackupAccessCmd(ENABLE) function before to configure the LSE (to be done once after reset).

After enabling the LSE (RCC_LSE_ON or RCC_LSE_Bypass), the application software should wait on LSERDY flag to be set indicating that LSE clock is stable and can be used to clock the RTC.

Parameters

RCC_LSE	specifies the new state of the LSE. This parameter can be one of the following values: • RCC_LSE_OFF: turn OFF the LSE oscillator, LSERDY flag goes low after 6 LSE oscillator clock cycles. • RCC_LSE_ON: turn ON the LSE oscillator • RCC_LSE_Bypass: LSE oscillator bypassed with external clock

Return values

None	

5.129.2.7 RCC_LSICmd()

```
void RCC_LSICmd (
            FunctionalState NewState )
```

Enables or disables the Internal Low Speed oscillator (LSI).

Note

After enabling the LSI, the application software should wait on LSIRDY flag to be set indicating that LSI clock is stable and can be used to clock the IWDG and/or the RTC.

LSI can not be disabled if the IWDG is running.

Parameters

NewState	new state of the LSI. This parameter can be: ENABLE or DISABLE.

Note

When the LSI is stopped, LSIRDY flag goes low after 6 LSI oscillator clock cycles.

Return values

None	

5.129.2.8 RCC_MCO1Config()

```
void RCC_MCO1Config (
            uint32_t RCC_MCO1Source,
            uint32_t RCC_MCO1Div )
```

Selects the clock source to output on MCO1 pin(PA8).

Note

PA8 should be configured in alternate function mode.

Parameters

RCC_MCO1Source	specifies the clock source to output. This parameter can be one of the following values: • RCC_MCO1Source_HSI: HSI clock selected as MCO1 source • RCC_MCO1Source_LSE: LSE clock selected as MCO1 source • RCC_MCO1Source_HSE: HSE clock selected as MCO1 source • RCC_MCO1Source_PLLCLK: main PLL clock selected as MCO1 source
RCC_MCO1Div	specifies the MCO1 prescaler. This parameter can be one of the following values: • RCC_MCO1Div_1: no division applied to MCO1 clock • RCC_MCO1Div_2: division by 2 applied to MCO1 clock • RCC_MCO1Div_3: division by 3 applied to MCO1 clock • RCC_MCO1Div_4: division by 4 applied to MCO1 clock • RCC_MCO1Div_5: division by 5 applied to MCO1 clock

Return values

None	

5.129.2.9 RCC_MCO2Config()

```
void RCC_MCO2Config (
             uint32_t RCC_MCO2Source,
             uint32_t RCC_MCO2Div )
```

Selects the clock source to output on MCO2 pin(PC9).

Note

PC9 should be configured in alternate function mode.

Parameters

RCC_MCO2Source	specifies the clock source to output. This parameter can be one of the following values: • RCC_MCO2Source_SYSCLK: System clock (SYSCLK) selected as MCO2 source • RCC_MCO2Source_PLLI2SCLK: PLLI2S clock selected as MCO2 source • RCC_MCO2Source_HSE: HSE clock selected as MCO2 source • RCC_MCO2Source_PLLCLK: main PLL clock selected as MCO2 source

Parameters

RCC_MCO2Div	specifies the MCO2 prescaler. This parameter can be one of the following values: • RCC_MCO2Div_1: no division applied to MCO2 clock • RCC_MCO2Div_2: division by 2 applied to MCO2 clock • RCC_MCO2Div_3: division by 3 applied to MCO2 clock • RCC_MCO2Div_4: division by 4 applied to MCO2 clock • RCC_MCO2Div_5: division by 5 applied to MCO2 clock

Return values

None	

5.129.2.10 RCC_PLLCmd()

```
void RCC_PLLCmd (
             FunctionalState NewState )
```

Enables or disables the main PLL.

Note

After enabling the main PLL, the application software should wait on PLLRDY flag to be set indicating that PLL clock is stable and can be used as system clock source.

The main PLL can not be disabled if it is used as system clock source

The main PLL is disabled by hardware when entering STOP and STANDBY modes.

Parameters

NewState	new state of the main PLL. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.129.2.11 RCC_PLLConfig()

```
void RCC_PLLConfig (
             uint32_t RCC_PLLSource,
             uint32_t PLLM,
```

```
uint32_t PLLN,
uint32_t PLLP,
uint32_t PLLQ )
```

Configures the main PLL clock source, multiplication and division factors.

Note

This function must be used only when the main PLL is disabled.

Parameters

RCC_PLLSource	specifies the PLL entry clock source. This parameter can be one of the following values: • RCC_PLLSource_HSI: HSI oscillator clock selected as PLL clock entry • RCC_PLLSource_HSE: HSE oscillator clock selected as PLL clock entry

Note

This clock source (RCC_PLLSource) is common for the main PLL and PLLI2S.

Parameters

PLLM	specifies the division factor for PLL VCO input clock This parameter must be a number between 0 and 63.

Note

You have to set the PLLM parameter correctly to ensure that the VCO input frequency ranges from 1 to 2 MHz. It is recommended to select a frequency of 2 MHz to limit PLL jitter.

Parameters

PLLN	specifies the multiplication factor for PLL VCO output clock This parameter must be a number between 192 and 432.

Note

You have to set the PLLN parameter correctly to ensure that the VCO output frequency is between 192 and 432 MHz.

Parameters

PLLP	specifies the division factor for main system clock (SYSCLK) This parameter must be a number in the range {2, 4, 6, or 8}.

Note

You have to set the PLLP parameter correctly to not exceed 168 MHz on the System clock frequency.

Parameters

PLLQ	specifies the division factor for OTG FS, SDIO and RNG clocks This parameter must be a number between 4 and 15.

Note

If the USB OTG FS is used in your application, you have to set the PLLQ parameter correctly to have 48 MHz clock for the USB. However, the SDIO and RNG need a frequency lower than or equal to 48 MHz to work correctly.

Return values

None	

5.129.2.12 RCC_PLLI2SCmd()

```
void RCC_PLLI2SCmd (
             FunctionalState NewState )
```

Enables or disables the PLLI2S.

Note

The PLLI2S is disabled by hardware when entering STOP and STANDBY modes.

Parameters

NewState	new state of the PLLI2S. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.129.2.13 RCC_PLLI2SConfig()

```
void RCC_PLLI2SConfig (
             uint32_t PLLI2SN,
             uint32_t PLLI2SR )
```

Configures the PLLI2S clock multiplication and division factors.

Note

This function must be used only when the PLLI2S is disabled.

PLLI2S clock source is common with the main PLL (configured in RCC_PLLConfig function)

Parameters

PLLI2SN	specifies the multiplication factor for PLLI2S VCO output clock This parameter must be a number between 192 and 432.

Note

You have to set the PLLI2SN parameter correctly to ensure that the VCO output frequency is between 192 and 432 MHz.

Parameters

PLLI2SR	specifies the division factor for I2S clock This parameter must be a number between 2 and 7.

Note

You have to set the PLLI2SR parameter correctly to not exceed 192 MHz on the I2S clock frequency.

Return values

None	

5.129.2.14 RCC_WaitForHSEStartUp()

```
ErrorStatus RCC_WaitForHSEStartUp (
             void  )
```

Waits for HSE start-up.

Note

This functions waits on HSERDY flag to be set and return SUCCESS if this flag is set, otherwise returns ERROR if the timeout is reached and this flag is not set. The timeout value is defined by the constant HSE↩ _STARTUP_TIMEOUT in stm32f4xx.h file. You can tailor it depending on the HSE crystal used in your application.

Parameters

None	

Return values

An	ErrorStatus enumeration value: • SUCCESS: HSE oscillator is stable and ready to use • ERROR: HSE oscillator not yet ready

5.130 System AHB and APB busses clocks configuration functions

System, AHB and APB busses clocks configuration functions.

Functions

- void RCC_SYSCLKConfig (uint32_t RCC_SYSCLKSource)

  Configures the system clock (SYSCLK).
- uint8_t RCC_GetSYSCLKSource (void)

  Returns the clock source used as system clock.
- void RCC_HCLKConfig (uint32_t RCC_SYSCLK)

  Configures the AHB clock (HCLK).
- void RCC_PCLK1Config (uint32_t RCC_HCLK)

  Configures the Low Speed APB clock (PCLK1).
- void RCC_PCLK2Config (uint32_t RCC_HCLK)

  Configures the High Speed APB clock (PCLK2).
- void RCC_GetClocksFreq (RCC_ClocksTypeDef *RCC_Clocks)

  Returns the frequencies of different on chip clocks; SYSCLK, HCLK, PCLK1 and PCLK2.

5.130.1 Detailed Description

System, AHB and APB busses clocks configuration functions.

```
 ===============================================================================
             System, AHB and APB busses clocks configuration functions
 ===============================================================================

  This section provide functions allowing to configure the System, AHB, APB1 and
  APB2 busses clocks.

  1. Several clock sources can be used to drive the System clock (SYSCLK): HSI,
     HSE and PLL.
     The AHB clock (HCLK) is derived from System clock through configurable prescaler
     and used to clock the CPU, memory and peripherals mapped on AHB bus (DMA, GPIO...).
     APB1 (PCLK1) and APB2 (PCLK2) clocks are derived from AHB clock through
     configurable prescalers and used to clock the peripherals mapped on these busses.
     You can use "RCC_GetClocksFreq()" function to retrieve the frequencies of these clocks.

@note All the peripheral clocks are derived from the System clock (SYSCLK) except:
       - I2S: the I2S clock can be derived either from a specific PLL (PLLI2S) or
          from an external clock mapped on the I2S_CKIN pin.
          You have to use RCC_I2SCLKConfig() function to configure this clock.
       - RTC: the RTC clock can be derived either from the LSI, LSE or HSE clock
          divided by 2 to 31. You have to use RCC_RTCCLKConfig() and RCC_RTCCLKCmd()
```

```
      functions to configure this clock.
    - USB OTG FS, SDIO and RTC: USB OTG FS require a frequency equal to 48 MHz
      to work correctly, while the SDIO require a frequency equal or lower than
      to 48. This clock is derived of the main PLL through PLLQ divider.
    - IWDG clock which is always the LSI clock.

 2. The maximum frequency of the SYSCLK and HCLK is 168 MHz, PCLK2 82 MHz and PCLK1 42 MHz.
    Depending on the device voltage range, the maximum frequency should be
    adapted accordingly:
+-------------------------------------------------------------------------------------+
| Latency       |                HCLK clock frequency (MHz)                           |
|               |-----------------------------------------------------------------------|
|               | voltage range  | voltage range  | voltage range   | voltage range   |
|               | 2.7 V - 3.6 V  | 2.4 V - 2.7 V  | 2.1 V - 2.4 V   | 1.8 V - 2.1 V   |
|---------------|----------------|----------------|-----------------|-----------------|
|0WS(1CPU cycle)|0 < HCLK <= 30  |0 < HCLK <= 24  |0 < HCLK <= 18   |0 < HCLK <= 16   |
|---------------|----------------|----------------|-----------------|-----------------|
|1WS(2CPU cycle)|30 < HCLK <= 60 |24 < HCLK <= 48 |18 < HCLK <= 36  |16 < HCLK <= 32  |
|---------------|----------------|----------------|-----------------|-----------------|
|2WS(3CPU cycle)|60 < HCLK <= 90 |48 < HCLK <= 72 |36 < HCLK <= 54  |32 < HCLK <= 48  |
|---------------|----------------|----------------|-----------------|-----------------|
|3WS(4CPU cycle)|90 < HCLK <= 120|72 < HCLK <= 96 |54 < HCLK <= 72  |48 < HCLK <= 64  |
|---------------|----------------|----------------|-----------------|-----------------|
|4WS(5CPU cycle)|120< HCLK <= 150|96 < HCLK <= 120|72 < HCLK <= 90  |64 < HCLK <= 80  |
|---------------|----------------|----------------|-----------------|-----------------|
|5WS(6CPU cycle)|120< HCLK <= 168|120< HCLK <= 144|90 < HCLK <= 108 |80 < HCLK <= 96  |
|---------------|----------------|----------------|-----------------|-----------------|
|6WS(7CPU cycle)|      NA        |144< HCLK <= 168|108 < HCLK <= 120|96 < HCLK <= 112 |
|---------------|----------------|----------------|-----------------|-----------------|
|7WS(8CPU cycle)|      NA        |      NA        |120 < HCLK <= 138|112 < HCLK <= 120|
+-------------------------------------------------------------------------------------+
  @note When VOS bit (in PWR_CR register) is reset to '0, the maximum value of HCLK is 144 MHz.
        You can use PWR_MainRegulatorModeConfig() function to set or reset this bit.
```

5.130.2 Function Documentation

5.130.2.1 RCC_GetClocksFreq()

```
void RCC_GetClocksFreq (
            RCC_ClocksTypeDef * RCC_Clocks )
```

Returns the frequencies of different on chip clocks; SYSCLK, HCLK, PCLK1 and PCLK2.

Note

The system frequency computed by this function is not the real frequency in the chip. It is calculated based on the predefined constant and the selected clock source:

If SYSCLK source is HSI, function returns values based on HSI_VALUE(∗)

If SYSCLK source is HSE, function returns values based on HSE_VALUE(∗∗)

If SYSCLK source is PLL, function returns values based on HSE_VALUE(∗∗) or HSI_VALUE(∗) multiplied/divided by the PLL factors.

(∗) HSI_VALUE is a constant defined in stm32f4xx.h file (default value 16 MHz) but the real value may vary depending on the variations in voltage and temperature.

(∗∗) HSE_VALUE is a constant defined in stm32f4xx.h file (default value 25 MHz), user has to ensure that HSE_VALUE is same as the real frequency of the crystal used. Otherwise, this function may have wrong result.

The result of this function could be not correct when using fractional value for HSE crystal.

Parameters

RCC_Clocks	pointer to a RCC_ClocksTypeDef structure which will hold the clocks frequencies.

Note

This function can be used by the user application to compute the baudrate for the communication peripherals or configure other parameters.

Each time SYSCLK, HCLK, PCLK1 and/or PCLK2 clock changes, this function must be called to update the structure's field. Otherwise, any configuration based on this function will be incorrect.

Return values

None	

5.130.2.2 RCC_GetSYSCLKSource()

```
uint8_t RCC_GetSYSCLKSource (
            void  )
```

Returns the clock source used as system clock.

Parameters

None	

Return values

The	clock source used as system clock. The returned value can be one of the following: • 0x00: HSI used as system clock • 0x04: HSE used as system clock • 0x08: PLL used as system clock

5.130.2.3 RCC_HCLKConfig()

```
void RCC_HCLKConfig (
            uint32_t RCC_SYSCLK )
```

Configures the AHB clock (HCLK).

Note

Depending on the device voltage range, the software has to set correctly these bits to ensure that HCLK not exceed the maximum allowed frequency (for more details refer to section above "CPU, AHB and APB busses clocks configuration functions")

Parameters

RCC_SYSCLK	defines the AHB clock divider. This clock is derived from the system clock (SYSCLK). This parameter can be one of the following values: • RCC_SYSCLK_Div1: AHB clock = SYSCLK • RCC_SYSCLK_Div2: AHB clock = SYSCLK/2 • RCC_SYSCLK_Div4: AHB clock = SYSCLK/4 • RCC_SYSCLK_Div8: AHB clock = SYSCLK/8 • RCC_SYSCLK_Div16: AHB clock = SYSCLK/16 • RCC_SYSCLK_Div64: AHB clock = SYSCLK/64 • RCC_SYSCLK_Div128: AHB clock = SYSCLK/128 • RCC_SYSCLK_Div256: AHB clock = SYSCLK/256 • RCC_SYSCLK_Div512: AHB clock = SYSCLK/512

Return values

None	

5.130.2.4 RCC_PCLK1Config()

```
void RCC_PCLK1Config (
             uint32_t RCC_HCLK )
```

Configures the Low Speed APB clock (PCLK1).

Parameters

RCC_HCLK	defines the APB1 clock divider. This clock is derived from the AHB clock (HCLK). This parameter can be one of the following values: • RCC_HCLK_Div1: APB1 clock = HCLK • RCC_HCLK_Div2: APB1 clock = HCLK/2 • RCC_HCLK_Div4: APB1 clock = HCLK/4 • RCC_HCLK_Div8: APB1 clock = HCLK/8 • RCC_HCLK_Div16: APB1 clock = HCLK/16

Return values

None	

5.130.2.5 RCC_PCLK2Config()

```
void RCC_PCLK2Config (
            uint32_t RCC_HCLK )
```

Configures the High Speed APB clock (PCLK2).

Parameters

RCC_HCLK	defines the APB2 clock divider. This clock is derived from the AHB clock (HCLK). This parameter can be one of the following values: • RCC_HCLK_Div1: APB2 clock = HCLK • RCC_HCLK_Div2: APB2 clock = HCLK/2 • RCC_HCLK_Div4: APB2 clock = HCLK/4 • RCC_HCLK_Div8: APB2 clock = HCLK/8 • RCC_HCLK_Div16: APB2 clock = HCLK/16

Return values

None	

5.130.2.6 RCC_SYSCLKConfig()

```
void RCC_SYSCLKConfig (
            uint32_t RCC_SYSCLKSource )
```

Configures the system clock (SYSCLK).

Note

The HSI is used (enabled by hardware) as system clock source after startup from Reset, wake-up from STOP and STANDBY mode, or in case of failure of the HSE used directly or indirectly as system clock (if the Clock Security System CSS is enabled).

A switch from one clock source to another occurs only if the target clock source is ready (clock stable after startup delay or PLL locked). If a clock source which is not yet ready is selected, the switch will occur when the clock source will be ready. You can use RCC_GetSYSCLKSource() function to know which clock is currently used as system clock source.

Parameters

RCC_SYSCLKSource	specifies the clock source used as system clock. This parameter can be one of the following values: • RCC_SYSCLKSource_HSI: HSI selected as system clock source • RCC_SYSCLKSource_HSE: HSE selected as system clock source • RCC_SYSCLKSource_PLLCLK: PLL selected as system clock source

Return values

None	

5.131 Peripheral clocks configuration functions

Peripheral clocks configuration functions.

Functions

- void RCC_RTCCLKConfig (uint32_t RCC_RTCCLKSource)

  Configures the RTC clock (RTCCLK).
- void RCC_RTCCLKCmd (FunctionalState NewState)

  Enables or disables the RTC clock.
- void RCC_BackupResetCmd (FunctionalState NewState)

  Forces or releases the Backup domain reset.
- void RCC_I2SCLKConfig (uint32_t RCC_I2SCLKSource)

  Configures the I2S clock source (I2SCLK).
- void RCC_AHB1PeriphClockCmd (uint32_t RCC_AHB1Periph, FunctionalState NewState)

  Enables or disables the AHB1 peripheral clock.
- void RCC_AHB2PeriphClockCmd (uint32_t RCC_AHB2Periph, FunctionalState NewState)

  Enables or disables the AHB2 peripheral clock.
- void RCC_AHB3PeriphClockCmd (uint32_t RCC_AHB3Periph, FunctionalState NewState)

  Enables or disables the AHB3 peripheral clock.
- void RCC_APB1PeriphClockCmd (uint32_t RCC_APB1Periph, FunctionalState NewState)

  Enables or disables the Low Speed APB (APB1) peripheral clock.
- void RCC_APB2PeriphClockCmd (uint32_t RCC_APB2Periph, FunctionalState NewState)

  Enables or disables the High Speed APB (APB2) peripheral clock.
- void RCC_AHB1PeriphResetCmd (uint32_t RCC_AHB1Periph, FunctionalState NewState)

  Forces or releases AHB1 peripheral reset.
- void RCC_AHB2PeriphResetCmd (uint32_t RCC_AHB2Periph, FunctionalState NewState)

  Forces or releases AHB2 peripheral reset.
- void RCC_AHB3PeriphResetCmd (uint32_t RCC_AHB3Periph, FunctionalState NewState)

  Forces or releases AHB3 peripheral reset.
- void RCC_APB1PeriphResetCmd (uint32_t RCC_APB1Periph, FunctionalState NewState)

  Forces or releases Low Speed APB (APB1) peripheral reset.
- void RCC_APB2PeriphResetCmd (uint32_t RCC_APB2Periph, FunctionalState NewState)

Forces or releases High Speed APB (APB2) peripheral reset.

- void RCC_AHB1PeriphClockLPModeCmd (uint32_t RCC_AHB1Periph, FunctionalState NewState)

  Enables or disables the AHB1 peripheral clock during Low Power (Sleep) mode.
- void RCC_AHB2PeriphClockLPModeCmd (uint32_t RCC_AHB2Periph, FunctionalState NewState)

  Enables or disables the AHB2 peripheral clock during Low Power (Sleep) mode.
- void RCC_AHB3PeriphClockLPModeCmd (uint32_t RCC_AHB3Periph, FunctionalState NewState)

  Enables or disables the AHB3 peripheral clock during Low Power (Sleep) mode.
- void RCC_APB1PeriphClockLPModeCmd (uint32_t RCC_APB1Periph, FunctionalState NewState)

  Enables or disables the APB1 peripheral clock during Low Power (Sleep) mode.
- void RCC_APB2PeriphClockLPModeCmd (uint32_t RCC_APB2Periph, FunctionalState NewState)

  Enables or disables the APB2 peripheral clock during Low Power (Sleep) mode.

5.131.1 Detailed Description

Peripheral clocks configuration functions.

```
 ===============================================================================
                   Peripheral clocks configuration functions
 ===============================================================================

  This section provide functions allowing to configure the Peripheral clocks.

  1. The RTC clock which is derived from the LSI, LSE or HSE clock divided by 2 to 31.

  2. After restart from Reset or wakeup from STANDBY, all peripherals are off
     except internal SRAM, Flash and JTAG. Before to start using a peripheral you
     have to enable its interface clock. You can do this using RCC_AHBPeriphClockCmd()
     , RCC_APB2PeriphClockCmd() and RCC_APB1PeriphClockCmd() functions.

  3. To reset the peripherals configuration (to the default state after device reset)
     you can use RCC_AHBPeriphResetCmd(), RCC_APB2PeriphResetCmd() and
     RCC_APB1PeriphResetCmd() functions.

  4. To further reduce power consumption in SLEEP mode the peripheral clocks can
     be disabled prior to executing the WFI or WFE instructions. You can do this
     using RCC_AHBPeriphClockLPModeCmd(), RCC_APB2PeriphClockLPModeCmd() and
     RCC_APB1PeriphClockLPModeCmd() functions.
```

5.131.2 Function Documentation

5.131.2.1 RCC_AHB1PeriphClockCmd()

```
void RCC_AHB1PeriphClockCmd (
             uint32_t RCC_AHB1Periph,
             FunctionalState NewState )
```

Enables or disables the AHB1 peripheral clock.

Note

After reset, the peripheral clock (used for registers read/write access) is disabled and the application software has to enable this clock before using it.

Parameters

RCC_AHBPeriph	specifies the AHB1 peripheral to gates its clock. This parameter can be any combination of the following values: • RCC_AHB1Periph_GPIOA: GPIOA clock • RCC_AHB1Periph_GPIOB: GPIOB clock • RCC_AHB1Periph_GPIOC: GPIOC clock • RCC_AHB1Periph_GPIOD: GPIOD clock • RCC_AHB1Periph_GPIOE: GPIOE clock • RCC_AHB1Periph_GPIOF: GPIOF clock • RCC_AHB1Periph_GPIOG: GPIOG clock • RCC_AHB1Periph_GPIOG: GPIOG clock • RCC_AHB1Periph_GPIOI: GPIOI clock • RCC_AHB1Periph_CRC: CRC clock • RCC_AHB1Periph_BKPSRAM: BKPSRAM interface clock • RCC_AHB1Periph_CCMDATARAMEN CCM data RAM interface clock • RCC_AHB1Periph_DMA1: DMA1 clock • RCC_AHB1Periph_DMA2: DMA2 clock • RCC_AHB1Periph_ETH_MAC: Ethernet MAC clock • RCC_AHB1Periph_ETH_MAC_Tx: Ethernet Transmission clock • RCC_AHB1Periph_ETH_MAC_Rx: Ethernet Reception clock • RCC_AHB1Periph_ETH_MAC_PTP: Ethernet PTP clock • RCC_AHB1Periph_OTG_HS: USB OTG HS clock • RCC_AHB1Periph_OTG_HS_ULPI: USB OTG HS ULPI clock
NewState	new state of the specified peripheral clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.131.2.2 RCC_AHB1PeriphClockLPModeCmd()

```
void RCC_AHB1PeriphClockLPModeCmd (
             uint32_t RCC_AHB1Periph,
             FunctionalState NewState )
```

Enables or disables the AHB1 peripheral clock during Low Power (Sleep) mode.

Note

Peripheral clock gating in SLEEP mode can be used to further reduce power consumption.

After wakeup from SLEEP mode, the peripheral clock is enabled again.

By default, all peripheral clocks are enabled during SLEEP mode.

Parameters

RCC_AHBPeriph	specifies the AHB1 peripheral to gates its clock. This parameter can be any combination of the following values: • RCC_AHB1Periph_GPIOA: GPIOA clock • RCC_AHB1Periph_GPIOB: GPIOB clock • RCC_AHB1Periph_GPIOC: GPIOC clock • RCC_AHB1Periph_GPIOD: GPIOD clock • RCC_AHB1Periph_GPIOE: GPIOE clock • RCC_AHB1Periph_GPIOF: GPIOF clock • RCC_AHB1Periph_GPIOG: GPIOG clock • RCC_AHB1Periph_GPIOG: GPIOG clock • RCC_AHB1Periph_GPIOI: GPIOI clock • RCC_AHB1Periph_CRC: CRC clock • RCC_AHB1Periph_BKPSRAM: BKPSRAM interface clock • RCC_AHB1Periph_DMA1: DMA1 clock • RCC_AHB1Periph_DMA2: DMA2 clock • RCC_AHB1Periph_ETH_MAC: Ethernet MAC clock • RCC_AHB1Periph_ETH_MAC_Tx: Ethernet Transmission clock • RCC_AHB1Periph_ETH_MAC_Rx: Ethernet Reception clock • RCC_AHB1Periph_ETH_MAC_PTP: Ethernet PTP clock • RCC_AHB1Periph_OTG_HS: USB OTG HS clock • RCC_AHB1Periph_OTG_HS_ULPI: USB OTG HS ULPI clock
NewState	new state of the specified peripheral clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.131.2.3 RCC_AHB1PeriphResetCmd()

```
void RCC_AHB1PeriphResetCmd (
            uint32_t RCC_AHB1Periph,
            FunctionalState NewState )
```

Forces or releases AHB1 peripheral reset.

Parameters

RCC_AHB1Periph	specifies the AHB1 peripheral to reset. This parameter can be any combination of the following values: • RCC_AHB1Periph_GPIOA: GPIOA clock • RCC_AHB1Periph_GPIOB: GPIOB clock • RCC_AHB1Periph_GPIOC: GPIOC clock • RCC_AHB1Periph_GPIOD: GPIOD clock • RCC_AHB1Periph_GPIOE: GPIOE clock • RCC_AHB1Periph_GPIOF: GPIOF clock • RCC_AHB1Periph_GPIOG: GPIOG clock • RCC_AHB1Periph_GPIOG: GPIOG clock • RCC_AHB1Periph_GPIOI: GPIOI clock • RCC_AHB1Periph_CRC: CRC clock • RCC_AHB1Periph_DMA1: DMA1 clock • RCC_AHB1Periph_DMA2: DMA2 clock • RCC_AHB1Periph_ETH_MAC: Ethernet MAC clock • RCC_AHB1Periph_OTG_HS: USB OTG HS clock
NewState	new state of the specified peripheral reset. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.131.2.4 RCC_AHB2PeriphClockCmd()

```
void RCC_AHB2PeriphClockCmd (
             uint32_t RCC_AHB2Periph,
             FunctionalState NewState )
```

Enables or disables the AHB2 peripheral clock.

Note

After reset, the peripheral clock (used for registers read/write access) is disabled and the application software has to enable this clock before using it.

Parameters

RCC_AHBPeriph	specifies the AHB2 peripheral to gates its clock. This parameter can be any combination of the following values: • RCC_AHB2Periph_DCMI: DCMI clock • RCC_AHB2Periph_CRYP: CRYP clock • RCC_AHB2Periph_HASH: HASH clock • RCC_AHB2Periph_RNG: RNG clock • RCC_AHB2Periph_OTG_FS: USB OTG FS clock
NewState	new state of the specified peripheral clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.131.2.5 RCC_AHB2PeriphClockLPModeCmd()

```
void RCC_AHB2PeriphClockLPModeCmd (
             uint32_t RCC_AHB2Periph,
             FunctionalState NewState )
```

Enables or disables the AHB2 peripheral clock during Low Power (Sleep) mode.

Note

Peripheral clock gating in SLEEP mode can be used to further reduce power consumption.

After wakeup from SLEEP mode, the peripheral clock is enabled again.

By default, all peripheral clocks are enabled during SLEEP mode.

Parameters

RCC_AHBPeriph	specifies the AHB2 peripheral to gates its clock. This parameter can be any combination of the following values: • RCC_AHB2Periph_DCMI: DCMI clock • RCC_AHB2Periph_CRYP: CRYP clock • RCC_AHB2Periph_HASH: HASH clock • RCC_AHB2Periph_RNG: RNG clock • RCC_AHB2Periph_OTG_FS: USB OTG FS clock
NewState	new state of the specified peripheral clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.131.2.6 RCC_AHB2PeriphResetCmd()

```
void RCC_AHB2PeriphResetCmd (
            uint32_t RCC_AHB2Periph,
            FunctionalState NewState )
```

Forces or releases AHB2 peripheral reset.

Parameters

RCC_AHB2Periph	specifies the AHB2 peripheral to reset. This parameter can be any combination of the following values: • RCC_AHB2Periph_DCMI: DCMI clock • RCC_AHB2Periph_CRYP: CRYP clock • RCC_AHB2Periph_HASH: HASH clock • RCC_AHB2Periph_RNG: RNG clock • RCC_AHB2Periph_OTG_FS: USB OTG FS clock
NewState	new state of the specified peripheral reset. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.131.2.7 RCC_AHB3PeriphClockCmd()

```
void RCC_AHB3PeriphClockCmd (
            uint32_t RCC_AHB3Periph,
            FunctionalState NewState )
```

Enables or disables the AHB3 peripheral clock.

Note

After reset, the peripheral clock (used for registers read/write access) is disabled and the application software has to enable this clock before using it.

Parameters

RCC_AHBPeriph	specifies the AHB3 peripheral to gates its clock. This parameter must be: RCC_AHB3Periph_FSMC
NewState	new state of the specified peripheral clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.131.2.8 RCC_AHB3PeriphClockLPModeCmd()

```
void RCC_AHB3PeriphClockLPModeCmd (
             uint32_t RCC_AHB3Periph,
             FunctionalState NewState )
```

Enables or disables the AHB3 peripheral clock during Low Power (Sleep) mode.

Note

Peripheral clock gating in SLEEP mode can be used to further reduce power consumption.

After wakeup from SLEEP mode, the peripheral clock is enabled again.

By default, all peripheral clocks are enabled during SLEEP mode.

Parameters

RCC_AHBPeriph	specifies the AHB3 peripheral to gates its clock. This parameter must be: RCC_AHB3Periph_FSMC
NewState	new state of the specified peripheral clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.131.2.9 RCC_AHB3PeriphResetCmd()

```
void RCC_AHB3PeriphResetCmd (
             uint32_t RCC_AHB3Periph,
             FunctionalState NewState )
```

Forces or releases AHB3 peripheral reset.

Parameters

RCC_AHB3Periph	specifies the AHB3 peripheral to reset. This parameter must be: RCC_AHB3Periph_FSMC
NewState	new state of the specified peripheral reset. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.131.2.10 RCC_APB1PeriphClockCmd()

```
void RCC_APB1PeriphClockCmd (
             uint32_t RCC_APB1Periph,
             FunctionalState NewState )
```

Enables or disables the Low Speed APB (APB1) peripheral clock.

Note

After reset, the peripheral clock (used for registers read/write access) is disabled and the application software has to enable this clock before using it.

Parameters

RCC_APB1Periph	specifies the APB1 peripheral to gates its clock. This parameter can be any combination of the following values: • RCC_APB1Periph_TIM2: TIM2 clock • RCC_APB1Periph_TIM3: TIM3 clock • RCC_APB1Periph_TIM4: TIM4 clock • RCC_APB1Periph_TIM5: TIM5 clock • RCC_APB1Periph_TIM6: TIM6 clock • RCC_APB1Periph_TIM7: TIM7 clock • RCC_APB1Periph_TIM12: TIM12 clock • RCC_APB1Periph_TIM13: TIM13 clock • RCC_APB1Periph_TIM14: TIM14 clock • RCC_APB1Periph_WWDG: WWDG clock • RCC_APB1Periph_SPI2: SPI2 clock • RCC_APB1Periph_SPI3: SPI3 clock • RCC_APB1Periph_USART2: USART2 clock • RCC_APB1Periph_USART3: USART3 clock • RCC_APB1Periph_UART4: UART4 clock • RCC_APB1Periph_UART5: UART5 clock • RCC_APB1Periph_I2C1: I2C1 clock • RCC_APB1Periph_I2C2: I2C2 clock • RCC_APB1Periph_I2C3: I2C3 clock • RCC_APB1Periph_CAN1: CAN1 clock • RCC_APB1Periph_CAN2: CAN2 clock • RCC_APB1Periph_PWR: PWR clock • RCC_APB1Periph_DAC: DAC clock
NewState	new state of the specified peripheral clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.131.2.11 RCC_APB1PeriphClockLPModeCmd()

```
void RCC_APB1PeriphClockLPModeCmd (
            uint32_t RCC_APB1Periph,
            FunctionalState NewState )
```

Enables or disables the APB1 peripheral clock during Low Power (Sleep) mode.

Note

Peripheral clock gating in SLEEP mode can be used to further reduce power consumption.

After wakeup from SLEEP mode, the peripheral clock is enabled again.

By default, all peripheral clocks are enabled during SLEEP mode.

Parameters

RCC_APB1Periph	specifies the APB1 peripheral to gates its clock. This parameter can be any combination of the following values: • RCC_APB1Periph_TIM2: TIM2 clock • RCC_APB1Periph_TIM3: TIM3 clock • RCC_APB1Periph_TIM4: TIM4 clock • RCC_APB1Periph_TIM5: TIM5 clock • RCC_APB1Periph_TIM6: TIM6 clock • RCC_APB1Periph_TIM7: TIM7 clock • RCC_APB1Periph_TIM12: TIM12 clock • RCC_APB1Periph_TIM13: TIM13 clock • RCC_APB1Periph_TIM14: TIM14 clock • RCC_APB1Periph_WWDG: WWDG clock • RCC_APB1Periph_SPI2: SPI2 clock • RCC_APB1Periph_SPI3: SPI3 clock • RCC_APB1Periph_USART2: USART2 clock • RCC_APB1Periph_USART3: USART3 clock • RCC_APB1Periph_UART4: UART4 clock • RCC_APB1Periph_UART5: UART5 clock • RCC_APB1Periph_I2C1: I2C1 clock • RCC_APB1Periph_I2C2: I2C2 clock • RCC_APB1Periph_I2C3: I2C3 clock • RCC_APB1Periph_CAN1: CAN1 clock • RCC_APB1Periph_CAN2: CAN2 clock • RCC_APB1Periph_PWR: PWR clock • RCC_APB1Periph_DAC: DAC clock
NewState	new state of the specified peripheral clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.131.2.12 RCC_APB1PeriphResetCmd()

```
void RCC_APB1PeriphResetCmd (
            uint32_t RCC_APB1Periph,
            FunctionalState NewState )
```

Forces or releases Low Speed APB (APB1) peripheral reset.

Parameters

RCC_APB1Periph	specifies the APB1 peripheral to reset. This parameter can be any combination of the following values: • RCC_APB1Periph_TIM2: TIM2 clock • RCC_APB1Periph_TIM3: TIM3 clock • RCC_APB1Periph_TIM4: TIM4 clock • RCC_APB1Periph_TIM5: TIM5 clock • RCC_APB1Periph_TIM6: TIM6 clock • RCC_APB1Periph_TIM7: TIM7 clock • RCC_APB1Periph_TIM12: TIM12 clock • RCC_APB1Periph_TIM13: TIM13 clock • RCC_APB1Periph_TIM14: TIM14 clock • RCC_APB1Periph_WWDG: WWDG clock • RCC_APB1Periph_SPI2: SPI2 clock • RCC_APB1Periph_SPI3: SPI3 clock • RCC_APB1Periph_USART2: USART2 clock • RCC_APB1Periph_USART3: USART3 clock • RCC_APB1Periph_UART4: UART4 clock • RCC_APB1Periph_UART5: UART5 clock • RCC_APB1Periph_I2C1: I2C1 clock • RCC_APB1Periph_I2C2: I2C2 clock • RCC_APB1Periph_I2C3: I2C3 clock • RCC_APB1Periph_CAN1: CAN1 clock • RCC_APB1Periph_CAN2: CAN2 clock • RCC_APB1Periph_PWR: PWR clock • RCC_APB1Periph_DAC: DAC clock
NewState	new state of the specified peripheral reset. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.131.2.13 RCC_APB2PeriphClockCmd()

```
void RCC_APB2PeriphClockCmd (
            uint32_t RCC_APB2Periph,
            FunctionalState NewState )
```

Enables or disables the High Speed APB (APB2) peripheral clock.

Note

After reset, the peripheral clock (used for registers read/write access) is disabled and the application software has to enable this clock before using it.

Parameters

RCC_APB2Periph	specifies the APB2 peripheral to gates its clock. This parameter can be any combination of the following values: • RCC_APB2Periph_TIM1: TIM1 clock • RCC_APB2Periph_TIM8: TIM8 clock • RCC_APB2Periph_USART1: USART1 clock • RCC_APB2Periph_USART6: USART6 clock • RCC_APB2Periph_ADC1: ADC1 clock • RCC_APB2Periph_ADC2: ADC2 clock • RCC_APB2Periph_ADC3: ADC3 clock • RCC_APB2Periph_SDIO: SDIO clock • RCC_APB2Periph_SPI1: SPI1 clock • RCC_APB2Periph_SYSCFG: SYSCFG clock • RCC_APB2Periph_TIM9: TIM9 clock • RCC_APB2Periph_TIM10: TIM10 clock • RCC_APB2Periph_TIM11: TIM11 clock
NewState	new state of the specified peripheral clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.131.2.14 RCC_APB2PeriphClockLPModeCmd()

```
void RCC_APB2PeriphClockLPModeCmd (
            uint32_t RCC_APB2Periph,
            FunctionalState NewState )
```

Enables or disables the APB2 peripheral clock during Low Power (Sleep) mode.

Note

Peripheral clock gating in SLEEP mode can be used to further reduce power consumption.

After wakeup from SLEEP mode, the peripheral clock is enabled again.

By default, all peripheral clocks are enabled during SLEEP mode.

Parameters

RCC_APB2Periph	specifies the APB2 peripheral to gates its clock. This parameter can be any combination of the following values: • RCC_APB2Periph_TIM1: TIM1 clock • RCC_APB2Periph_TIM8: TIM8 clock • RCC_APB2Periph_USART1: USART1 clock • RCC_APB2Periph_USART6: USART6 clock • RCC_APB2Periph_ADC1: ADC1 clock • RCC_APB2Periph_ADC2: ADC2 clock • RCC_APB2Periph_ADC3: ADC3 clock • RCC_APB2Periph_SDIO: SDIO clock • RCC_APB2Periph_SPI1: SPI1 clock • RCC_APB2Periph_SYSCFG: SYSCFG clock • RCC_APB2Periph_TIM9: TIM9 clock • RCC_APB2Periph_TIM10: TIM10 clock • RCC_APB2Periph_TIM11: TIM11 clock
NewState	new state of the specified peripheral clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.131.2.15 RCC_APB2PeriphResetCmd()

```
void RCC_APB2PeriphResetCmd (
            uint32_t RCC_APB2Periph,
            FunctionalState NewState )
```

Forces or releases High Speed APB (APB2) peripheral reset.

Parameters

RCC_APB2Periph	specifies the APB2 peripheral to reset. This parameter can be any combination of the following values: • RCC_APB2Periph_TIM1: TIM1 clock • RCC_APB2Periph_TIM8: TIM8 clock • RCC_APB2Periph_USART1: USART1 clock • RCC_APB2Periph_USART6: USART6 clock • RCC_APB2Periph_ADC1: ADC1 clock • RCC_APB2Periph_ADC2: ADC2 clock • RCC_APB2Periph_ADC3: ADC3 clock • RCC_APB2Periph_SDIO: SDIO clock • RCC_APB2Periph_SPI1: SPI1 clock • RCC_APB2Periph_SYSCFG: SYSCFG clock • RCC_APB2Periph_TIM9: TIM9 clock • RCC_APB2Periph_TIM10: TIM10 clock • RCC_APB2Periph_TIM11: TIM11 clock
NewState	new state of the specified peripheral reset. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.131.2.16 RCC_BackupResetCmd()

```
void RCC_BackupResetCmd (
             FunctionalState NewState )
```

Forces or releases the Backup domain reset.

Note

This function resets the RTC peripheral (including the backup registers) and the RTC clock source selection in RCC_CSR register.

The BKPSRAM is not affected by this reset.

Parameters

NewState	new state of the Backup domain reset. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.131.2.17 RCC_I2SCLKConfig()

```
void RCC_I2SCLKConfig (
            uint32_t RCC_I2SCLKSource )
```

Configures the I2S clock source (I2SCLK).

Note

This function must be called before enabling the I2S APB clock.

Parameters

RCC_I2SCLKSource	specifies the I2S clock source. This parameter can be one of the following values: • RCC_I2S2CLKSource_PLLI2S: PLLI2S clock used as I2S clock source • RCC_I2S2CLKSource_Ext: External clock mapped on the I2S_CKIN pin used as I2S clock source

Return values

None	

5.131.2.18 RCC_RTCCLKCmd()

```
void RCC_RTCCLKCmd (
            FunctionalState NewState )
```

Enables or disables the RTC clock.

Note

This function must be used only after the RTC clock source was selected using the RCC_RTCCLKConfig function.

Parameters

NewState	new state of the RTC clock. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.131.2.19 RCC_RTCCLKConfig()

```
void RCC_RTCCLKConfig (
            uint32_t RCC_RTCCLKSource )
```

Configures the RTC clock (RTCCLK).

Note

As the RTC clock configuration bits are in the Backup domain and write access is denied to this domain after reset, you have to enable write access using PWR_BackupAccessCmd(ENABLE) function before to configure the RTC clock source (to be done once after reset).

Once the RTC clock is configured it can't be changed unless the
Backup domain is reset using RCC_BackupResetCmd() function, or by a Power On Reset (POR).

Parameters

RCC_RTCCLKSource	specifies the RTC clock source. This parameter can be one of the following values: • RCC_RTCCLKSource_LSE: LSE selected as RTC clock • RCC_RTCCLKSource_LSI: LSI selected as RTC clock • RCC_RTCCLKSource_HSE_Divx: HSE clock divided by x selected as RTC clock, where x:[2,31]

Note

If the LSE or LSI is used as RTC clock source, the RTC continues to work in STOP and STANDBY modes, and can be used as wakeup source. However, when the HSE clock is used as RTC clock source, the RTC cannot be used in STOP and STANDBY modes.

The maximum input clock frequency for RTC is 1MHz (when using HSE as RTC clock source).

Return values

None	

5.132 Interrupts and flags management functions

Interrupts and flags management functions.

Functions

- void RCC_ITConfig (uint8_t RCC_IT, FunctionalState NewState)

  Enables or disables the specified RCC interrupts.
- FlagStatus RCC_GetFlagStatus (uint8_t RCC_FLAG)

  Checks whether the specified RCC flag is set or not.
- void RCC_ClearFlag (void)

  Clears the RCC reset flags. The reset flags are: RCC_FLAG_PINRST, RCC_FLAG_PORRST, RCC_FLAG_SFTRST, RCC_FLAG_IWDGRST, RCC_FLAG_WWDGRST, RCC_FLAG_LPWRRST.
- ITStatus RCC_GetITStatus (uint8_t RCC_IT)

  Checks whether the specified RCC interrupt has occurred or not.
- void RCC_ClearITPendingBit (uint8_t RCC_IT)

  Clears the RCC's interrupt pending bits.

5.132.1 Detailed Description

Interrupts and flags management functions.

```
 ===============================================================================
                   Interrupts and flags management functions
 ===============================================================================
```

5.132.2 Function Documentation

5.132.2.1 RCC_ClearFlag()

```
void RCC_ClearFlag (
             void  )
```

Clears the RCC reset flags. The reset flags are: RCC_FLAG_PINRST, RCC_FLAG_PORRST, RCC_FLAG_↩ SFTRST, RCC_FLAG_IWDGRST, RCC_FLAG_WWDGRST, RCC_FLAG_LPWRRST.

Parameters

None	

Return values

None	

5.132.2.2 RCC_ClearITPendingBit()

```
void RCC_ClearITPendingBit (
             uint8_t RCC_IT )
```

Clears the RCC's interrupt pending bits.

Parameters

RCC_IT	specifies the interrupt pending bit to clear. This parameter can be any combination of the following values: • RCC_IT_LSIRDY: LSI ready interrupt • RCC_IT_LSERDY: LSE ready interrupt • RCC_IT_HSIRDY: HSI ready interrupt • RCC_IT_HSERDY: HSE ready interrupt • RCC_IT_PLLRDY: main PLL ready interrupt • RCC_IT_PLLI2SRDY: PLLI2S ready interrupt • RCC_IT_CSS: Clock Security System interrupt

Return values

None	

5.132.2.3 RCC_GetFlagStatus()

```
FlagStatus RCC_GetFlagStatus (
            uint8_t RCC_FLAG )
```

Checks whether the specified RCC flag is set or not.

Parameters

RCC_FLAG	specifies the flag to check. This parameter can be one of the following values: • RCC_FLAG_HSIRDY: HSI oscillator clock ready • RCC_FLAG_HSERDY: HSE oscillator clock ready • RCC_FLAG_PLLRDY: main PLL clock ready • RCC_FLAG_PLLI2SRDY: PLLI2S clock ready • RCC_FLAG_LSERDY: LSE oscillator clock ready • RCC_FLAG_LSIRDY: LSI oscillator clock ready • RCC_FLAG_BORRST: POR/PDR or BOR reset • RCC_FLAG_PINRST: Pin reset • RCC_FLAG_PORRST: POR/PDR reset • RCC_FLAG_SFTRST: Software reset • RCC_FLAG_IWDGRST: Independent Watchdog reset • RCC_FLAG_WWDGRST: Window Watchdog reset • RCC_FLAG_LPWRRST: Low Power reset

Return values

The	new state of RCC_FLAG (SET or RESET).

5.132.2.4 RCC_GetITStatus()

```
ITStatus RCC_GetITStatus (
             uint8_t RCC_IT )
```

Checks whether the specified RCC interrupt has occurred or not.

Parameters

RCC_IT	specifies the RCC interrupt source to check. This parameter can be one of the following values: • RCC_IT_LSIRDY: LSI ready interrupt • RCC_IT_LSERDY: LSE ready interrupt • RCC_IT_HSIRDY: HSI ready interrupt • RCC_IT_HSERDY: HSE ready interrupt • RCC_IT_PLLRDY: main PLL ready interrupt • RCC_IT_PLLI2SRDY: PLLI2S ready interrupt • RCC_IT_CSS: Clock Security System interrupt

Return values

The	new state of RCC_IT (SET or RESET).

5.132.2.5 RCC_ITConfig()

```
void RCC_ITConfig (
             uint8_t RCC_IT,
             FunctionalState NewState )
```

Enables or disables the specified RCC interrupts.

Parameters

RCC_IT	specifies the RCC interrupt sources to be enabled or disabled. This parameter can be any combination of the following values: • RCC_IT_LSIRDY: LSI ready interrupt • RCC_IT_LSERDY: LSE ready interrupt • RCC_IT_HSIRDY: HSI ready interrupt • RCC_IT_HSERDY: HSE ready interrupt • RCC_IT_PLLRDY: main PLL ready interrupt • RCC_IT_PLLI2SRDY: PLLI2S ready interrupt
NewState	new state of the specified RCC interrupts. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.133 TIM

TIM driver modules.

Modules

- TIM_Exported_constants
- TIM_Private_Functions

Data Structures

- struct TIM_TimeBaseInitTypeDef

  TIM Time Base Init structure definition

- struct TIM_OCInitTypeDef

  TIM Output Compare Init structure definition

- struct TIM_ICInitTypeDef

  TIM Input Capture Init structure definition

- struct TIM_BDTRInitTypeDef

  BDTR structure definition.

Macros

- #define **SMCR_ETR_MASK** ((uint16_t)0x00FF)
- #define **CCMR_OFFSET** ((uint16_t)0x0018)
- #define **CCER_CCE_SET** ((uint16_t)0x0001)
- #define **CCER_CCNE_SET** ((uint16_t)0x0004)
- #define **CCMR_OC13M_MASK** ((uint16_t)0xFF8F)
- #define **CCMR_OC24M_MASK** ((uint16_t)0x8FFF)

Functions

- void TIM_DeInit (TIM_TypeDef ∗TIMx)

  Deinitializes the TIMx peripheral registers to their default reset values.
- void TIM_TimeBaseInit (TIM_TypeDef ∗TIMx, TIM_TimeBaseInitTypeDef ∗TIM_TimeBaseInitStruct)

  Initializes the TIMx Time Base Unit peripheral according to the specified parameters in the TIM_TimeBaseInitStruct.
- void TIM_TimeBaseStructInit (TIM_TimeBaseInitTypeDef ∗TIM_TimeBaseInitStruct)

  Fills each TIM_TimeBaseInitStruct member with its default value.
- void TIM_PrescalerConfig (TIM_TypeDef ∗TIMx, uint16_t Prescaler, uint16_t TIM_PSCReloadMode)

  Configures the TIMx Prescaler.
- void TIM_CounterModeConfig (TIM_TypeDef ∗TIMx, uint16_t TIM_CounterMode)

  Specifies the TIMx Counter Mode to be used.
- void TIM_SetCounter (TIM_TypeDef ∗TIMx, uint32_t Counter)

  Sets the TIMx Counter Register value.
- void TIM_SetAutoreload (TIM_TypeDef ∗TIMx, uint32_t Autoreload)

  Sets the TIMx Autoreload Register value.
- uint32_t TIM_GetCounter (TIM_TypeDef ∗TIMx)

Gets the TIMx Counter value.

- uint16_t TIM_GetPrescaler (TIM_TypeDef *TIMx)

  Gets the TIMx Prescaler value.
- void TIM_UpdateDisableConfig (TIM_TypeDef *TIMx, FunctionalState NewState)

  Enables or Disables the TIMx Update event.
- void TIM_UpdateRequestConfig (TIM_TypeDef *TIMx, uint16_t TIM_UpdateSource)

  Configures the TIMx Update Request Interrupt source.
- void TIM_ARRPreloadConfig (TIM_TypeDef *TIMx, FunctionalState NewState)

  Enables or disables TIMx peripheral Preload register on ARR.
- void TIM_SelectOnePulseMode (TIM_TypeDef *TIMx, uint16_t TIM_OPMode)

  Selects the TIMx's One Pulse Mode.
- void TIM_SetClockDivision (TIM_TypeDef *TIMx, uint16_t TIM_CKD)

  Sets the TIMx Clock Division value.
- void TIM_Cmd (TIM_TypeDef *TIMx, FunctionalState NewState)

  Enables or disables the specified TIM peripheral.
- void TIM_OC1Init (TIM_TypeDef *TIMx, TIM_OCInitTypeDef *TIM_OCInitStruct)

  Initializes the TIMx Channel1 according to the specified parameters in the TIM_OCInitStruct.
- void TIM_OC2Init (TIM_TypeDef *TIMx, TIM_OCInitTypeDef *TIM_OCInitStruct)

  Initializes the TIMx Channel2 according to the specified parameters in the TIM_OCInitStruct.
- void TIM_OC3Init (TIM_TypeDef *TIMx, TIM_OCInitTypeDef *TIM_OCInitStruct)

  Initializes the TIMx Channel3 according to the specified parameters in the TIM_OCInitStruct.
- void TIM_OC4Init (TIM_TypeDef *TIMx, TIM_OCInitTypeDef *TIM_OCInitStruct)

  Initializes the TIMx Channel4 according to the specified parameters in the TIM_OCInitStruct.
- void TIM_OCStructInit (TIM_OCInitTypeDef *TIM_OCInitStruct)

  Fills each TIM_OCInitStruct member with its default value.
- void TIM_SelectOCxM (TIM_TypeDef *TIMx, uint16_t TIM_Channel, uint16_t TIM_OCMode)

  Selects the TIM Output Compare Mode.
- void TIM_SetCompare1 (TIM_TypeDef *TIMx, uint32_t Compare1)

  Sets the TIMx Capture Compare1 Register value.
- void TIM_SetCompare2 (TIM_TypeDef *TIMx, uint32_t Compare2)

  Sets the TIMx Capture Compare2 Register value.
- void TIM_SetCompare3 (TIM_TypeDef *TIMx, uint32_t Compare3)

  Sets the TIMx Capture Compare3 Register value.
- void TIM_SetCompare4 (TIM_TypeDef *TIMx, uint32_t Compare4)

  Sets the TIMx Capture Compare4 Register value.
- void TIM_ForcedOC1Config (TIM_TypeDef *TIMx, uint16_t TIM_ForcedAction)

  Forces the TIMx output 1 waveform to active or inactive level.
- void TIM_ForcedOC2Config (TIM_TypeDef *TIMx, uint16_t TIM_ForcedAction)

  Forces the TIMx output 2 waveform to active or inactive level.
- void TIM_ForcedOC3Config (TIM_TypeDef *TIMx, uint16_t TIM_ForcedAction)

  Forces the TIMx output 3 waveform to active or inactive level.
- void TIM_ForcedOC4Config (TIM_TypeDef *TIMx, uint16_t TIM_ForcedAction)

  Forces the TIMx output 4 waveform to active or inactive level.
- void TIM_OC1PreloadConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPreload)

  Enables or disables the TIMx peripheral Preload register on CCR1.
- void TIM_OC2PreloadConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPreload)

  Enables or disables the TIMx peripheral Preload register on CCR2.
- void TIM_OC3PreloadConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPreload)

  Enables or disables the TIMx peripheral Preload register on CCR3.
- void TIM_OC4PreloadConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPreload)

  Enables or disables the TIMx peripheral Preload register on CCR4.

- void TIM_OC1FastConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCFast)

  Configures the TIMx Output Compare 1 Fast feature.
- void TIM_OC2FastConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCFast)

  Configures the TIMx Output Compare 2 Fast feature.
- void TIM_OC3FastConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCFast)

  Configures the TIMx Output Compare 3 Fast feature.
- void TIM_OC4FastConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCFast)

  Configures the TIMx Output Compare 4 Fast feature.
- void TIM_ClearOC1Ref (TIM_TypeDef *TIMx, uint16_t TIM_OCClear)

  Clears or safeguards the OCREF1 signal on an external event.
- void TIM_ClearOC2Ref (TIM_TypeDef *TIMx, uint16_t TIM_OCClear)

  Clears or safeguards the OCREF2 signal on an external event.
- void TIM_ClearOC3Ref (TIM_TypeDef *TIMx, uint16_t TIM_OCClear)

  Clears or safeguards the OCREF3 signal on an external event.
- void TIM_ClearOC4Ref (TIM_TypeDef *TIMx, uint16_t TIM_OCClear)

  Clears or safeguards the OCREF4 signal on an external event.
- void TIM_OC1PolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPolarity)

  Configures the TIMx channel 1 polarity.
- void TIM_OC1NPolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCNPolarity)

  Configures the TIMx Channel 1N polarity.
- void TIM_OC2PolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPolarity)

  Configures the TIMx channel 2 polarity.
- void TIM_OC2NPolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCNPolarity)

  Configures the TIMx Channel 2N polarity.
- void TIM_OC3PolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPolarity)

  Configures the TIMx channel 3 polarity.
- void TIM_OC3NPolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCNPolarity)

  Configures the TIMx Channel 3N polarity.
- void TIM_OC4PolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPolarity)

  Configures the TIMx channel 4 polarity.
- void TIM_CCxCmd (TIM_TypeDef *TIMx, uint16_t TIM_Channel, uint16_t TIM_CCx)

  Enables or disables the TIM Capture Compare Channel x.
- void TIM_CCxNCmd (TIM_TypeDef *TIMx, uint16_t TIM_Channel, uint16_t TIM_CCxN)

  Enables or disables the TIM Capture Compare Channel xN.
- void TIM_ICInit (TIM_TypeDef *TIMx, TIM_ICInitTypeDef *TIM_ICInitStruct)

  Initializes the TIM peripheral according to the specified parameters in the TIM_ICInitStruct.
- void TIM_ICStructInit (TIM_ICInitTypeDef *TIM_ICInitStruct)

  Fills each TIM_ICInitStruct member with its default value.
- void TIM_PWMIConfig (TIM_TypeDef *TIMx, TIM_ICInitTypeDef *TIM_ICInitStruct)

  Configures the TIM peripheral according to the specified parameters in the TIM_ICInitStruct to measure an external PWM signal.
- uint32_t TIM_GetCapture1 (TIM_TypeDef *TIMx)

  Gets the TIMx Input Capture 1 value.
- uint32_t TIM_GetCapture2 (TIM_TypeDef *TIMx)

  Gets the TIMx Input Capture 2 value.
- uint32_t TIM_GetCapture3 (TIM_TypeDef *TIMx)

  Gets the TIMx Input Capture 3 value.
- uint32_t TIM_GetCapture4 (TIM_TypeDef *TIMx)

  Gets the TIMx Input Capture 4 value.
- void TIM_SetIC1Prescaler (TIM_TypeDef *TIMx, uint16_t TIM_ICPSC)

  Sets the TIMx Input Capture 1 prescaler.

- void TIM_SetIC2Prescaler (TIM_TypeDef *TIMx, uint16_t TIM_ICPSC)

  Sets the TIMx Input Capture 2 prescaler.
- void TIM_SetIC3Prescaler (TIM_TypeDef *TIMx, uint16_t TIM_ICPSC)

  Sets the TIMx Input Capture 3 prescaler.
- void TIM_SetIC4Prescaler (TIM_TypeDef *TIMx, uint16_t TIM_ICPSC)

  Sets the TIMx Input Capture 4 prescaler.
- void TIM_BDTRConfig (TIM_TypeDef *TIMx, TIM_BDTRInitTypeDef *TIM_BDTRInitStruct)

  Configures the Break feature, dead time, Lock level, OSSI/OSSR State and the AOE(automatic output enable).
- void TIM_BDTRStructInit (TIM_BDTRInitTypeDef *TIM_BDTRInitStruct)

  Fills each TIM_BDTRInitStruct member with its default value.
- void TIM_CtrlPWMOutputs (TIM_TypeDef *TIMx, FunctionalState NewState)

  Enables or disables the TIM peripheral Main Outputs.
- void TIM_SelectCOM (TIM_TypeDef *TIMx, FunctionalState NewState)

  Selects the TIM peripheral Commutation event.
- void TIM_CCPreloadControl (TIM_TypeDef *TIMx, FunctionalState NewState)

  Sets or Resets the TIM peripheral Capture Compare Preload Control bit.
- void TIM_ITConfig (TIM_TypeDef *TIMx, uint16_t TIM_IT, FunctionalState NewState)

  Enables or disables the specified TIM interrupts.
- void TIM_GenerateEvent (TIM_TypeDef *TIMx, uint16_t TIM_EventSource)

  Configures the TIMx event to be generate by software.
- FlagStatus TIM_GetFlagStatus (TIM_TypeDef *TIMx, uint16_t TIM_FLAG)

  Checks whether the specified TIM flag is set or not.
- void TIM_ClearFlag (TIM_TypeDef *TIMx, uint16_t TIM_FLAG)

  Clears the TIMx's pending flags.
- ITStatus TIM_GetITStatus (TIM_TypeDef *TIMx, uint16_t TIM_IT)

  Checks whether the TIM interrupt has occurred or not.
- void TIM_ClearITPendingBit (TIM_TypeDef *TIMx, uint16_t TIM_IT)

  Clears the TIMx's interrupt pending bits.
- void TIM_DMAConfig (TIM_TypeDef *TIMx, uint16_t TIM_DMABase, uint16_t TIM_DMABurstLength)

  Configures the TIMx's DMA interface.
- void TIM_DMACmd (TIM_TypeDef *TIMx, uint16_t TIM_DMASource, FunctionalState NewState)

  Enables or disables the TIMx's DMA Requests.
- void TIM_SelectCCDMA (TIM_TypeDef *TIMx, FunctionalState NewState)

  Selects the TIMx peripheral Capture Compare DMA source.
- void TIM_InternalClockConfig (TIM_TypeDef *TIMx)

  Configures the TIMx internal Clock.
- void TIM_ITRxExternalClockConfig (TIM_TypeDef *TIMx, uint16_t TIM_InputTriggerSource)

  Configures the TIMx Internal Trigger as External Clock.
- void TIM_TIxExternalClockConfig (TIM_TypeDef *TIMx, uint16_t TIM_TIxExternalCLKSource, uint16_t TIM_ICPolarity, uint16_t ICFilter)

  Configures the TIMx Trigger as External Clock.
- void TIM_ETRClockMode1Config (TIM_TypeDef *TIMx, uint16_t TIM_ExtTRGPrescaler, uint16_t TIM_ExtTRGPolarity, uint16_t ExtTRGFilter)

  Configures the External clock Mode1.
- void TIM_ETRClockMode2Config (TIM_TypeDef *TIMx, uint16_t TIM_ExtTRGPrescaler, uint16_t TIM_ExtTRGPolarity, uint16_t ExtTRGFilter)

  Configures the External clock Mode2.
- void TIM_SelectInputTrigger (TIM_TypeDef *TIMx, uint16_t TIM_InputTriggerSource)

  Selects the Input Trigger source.
- void TIM_SelectOutputTrigger (TIM_TypeDef *TIMx, uint16_t TIM_TRGOSource)

  Selects the TIMx Trigger Output Mode.

- void TIM_SelectSlaveMode (TIM_TypeDef *TIMx, uint16_t TIM_SlaveMode)

  Selects the TIMx Slave Mode.
- void TIM_SelectMasterSlaveMode (TIM_TypeDef *TIMx, uint16_t TIM_MasterSlaveMode)

  Sets or Resets the TIMx Master/Slave Mode.
- void TIM_ETRConfig (TIM_TypeDef *TIMx, uint16_t TIM_ExtTRGPrescaler, uint16_t TIM_ExtTRGPolarity, uint16_t ExtTRGFilter)

  Configures the TIMx External Trigger (ETR).
- void TIM_EncoderInterfaceConfig (TIM_TypeDef *TIMx, uint16_t TIM_EncoderMode, uint16_t TIM_IC1↩ Polarity, uint16_t TIM_IC2Polarity)

  Configures the TIMx Encoder Interface.
- void TIM_SelectHallSensor (TIM_TypeDef *TIMx, FunctionalState NewState)

  Enables or disables the TIMx's Hall sensor interface.
- void TIM_RemapConfig (TIM_TypeDef *TIMx, uint16_t TIM_Remap)

  Configures the TIM2, TIM5 and TIM11 Remapping input capabilities.

5.133.1 Detailed Description

TIM driver modules.

5.133.2 Function Documentation

5.133.2.1 TIM_ARRPreloadConfig()

```
void TIM_ARRPreloadConfig (
            TIM_TypeDef * TIMx,
            FunctionalState NewState )
```

Enables or disables TIMx peripheral Preload register on ARR.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
NewState	new state of the TIMx peripheral Preload register This parameter can be: ENABLE or DISABLE.

Return values

None	

5.133.2.2 TIM_BDTRConfig()

```
void TIM_BDTRConfig (
            TIM_TypeDef * TIMx,
            TIM_BDTRInitTypeDef * TIM_BDTRInitStruct )
```

Configures the Break feature, dead time, Lock level, OSSI/OSSR State and the AOE(automatic output enable).

Parameters

TIMx	where x can be 1 or 8 to select the TIM
TIM_BDTRInitStruct	pointer to a TIM_BDTRInitTypeDef structure that contains the BDTR Register configuration information for the TIM peripheral.

Return values

None	

5.133.2.3 TIM_BDTRStructInit()

```
void TIM_BDTRStructInit (
            TIM_BDTRInitTypeDef * TIM_BDTRInitStruct )
```

Fills each TIM_BDTRInitStruct member with its default value.

Parameters

TIM_BDTRInitStruct	pointer to a TIM_BDTRInitTypeDef structure which will be initialized.

Return values

None	

5.133.2.4 TIM_CCPreloadControl()

```
void TIM_CCPreloadControl (
            TIM_TypeDef * TIMx,
            FunctionalState NewState )
```

Sets or Resets the TIM peripheral Capture Compare Preload Control bit.

Parameters

TIMx	where x can be 1 or 8 to select the TIMx peripheral
NewState	new state of the Capture Compare Preload Control bit This parameter can be: ENABLE or DISABLE.

Return values

None	

5.133.2.5 TIM_CCxCmd()

```
void TIM_CCxCmd (
             TIM_TypeDef * TIMx,
             uint16_t TIM_Channel,
             uint16_t TIM_CCx )
```

Enables or disables the TIM Capture Compare Channel x.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
TIM_Channel	specifies the TIM Channel This parameter can be one of the following values: • TIM_Channel_1: TIM Channel 1 • TIM_Channel_2: TIM Channel 2 • TIM_Channel_3: TIM Channel 3 • TIM_Channel_4: TIM Channel 4
TIM_CCx	specifies the TIM Channel CCxE bit new state. This parameter can be: TIM_CCx_Enable or TIM_CCx_Disable.

Return values

None	

5.133.2.6 TIM_CCxNCmd()

```
void TIM_CCxNCmd (
             TIM_TypeDef * TIMx,
             uint16_t TIM_Channel,
             uint16_t TIM_CCxN )
```

Enables or disables the TIM Capture Compare Channel xN.

Parameters

TIMx	where x can be 1 or 8 to select the TIM peripheral.
TIM_Channel	specifies the TIM Channel This parameter can be one of the following values: • TIM_Channel_1: TIM Channel 1 • TIM_Channel_2: TIM Channel 2 • TIM_Channel_3: TIM Channel 3
TIM_CCxN	specifies the TIM Channel CCxNE bit new state. This parameter can be: TIM_CCxN_Enable or TIM_CCxN_Disable.

Return values

None	

5.133.2.7 TIM_ClearFlag()

```
void TIM_ClearFlag (
            TIM_TypeDef * TIMx,
            uint16_t TIM_FLAG )
```

Clears the TIMx's pending flags.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
TIM_FLAG	specifies the flag bit to clear. This parameter can be any combination of the following values: • TIM_FLAG_Update: TIM update Flag • TIM_FLAG_CC1: TIM Capture Compare 1 Flag • TIM_FLAG_CC2: TIM Capture Compare 2 Flag • TIM_FLAG_CC3: TIM Capture Compare 3 Flag • TIM_FLAG_CC4: TIM Capture Compare 4 Flag • TIM_FLAG_COM: TIM Commutation Flag • TIM_FLAG_Trigger: TIM Trigger Flag • TIM_FLAG_Break: TIM Break Flag • TIM_FLAG_CC1OF: TIM Capture Compare 1 over capture Flag • TIM_FLAG_CC2OF: TIM Capture Compare 2 over capture Flag • TIM_FLAG_CC3OF: TIM Capture Compare 3 over capture Flag • TIM_FLAG_CC4OF: TIM Capture Compare 4 over capture Flag

Note

TIM6 and TIM7 can have only one update flag.

TIM_FLAG_COM and TIM_FLAG_Break are used only with TIM1 and TIM8.

Return values

None	

5.133.2.8 TIM_ClearITPendingBit()

```
void TIM_ClearITPendingBit (
             TIM_TypeDef * TIMx,
             uint16_t TIM_IT )
```

Clears the TIMx's interrupt pending bits.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
TIM_IT	specifies the pending bit to clear. This parameter can be any combination of the following values: • TIM_IT_Update: TIM1 update Interrupt source • TIM_IT_CC1: TIM Capture Compare 1 Interrupt source • TIM_IT_CC2: TIM Capture Compare 2 Interrupt source • TIM_IT_CC3: TIM Capture Compare 3 Interrupt source • TIM_IT_CC4: TIM Capture Compare 4 Interrupt source • TIM_IT_COM: TIM Commutation Interrupt source • TIM_IT_Trigger: TIM Trigger Interrupt source • TIM_IT_Break: TIM Break Interrupt source

Note

TIM6 and TIM7 can generate only an update interrupt.

TIM_IT_COM and TIM_IT_Break are used only with TIM1 and TIM8.

Return values

None	

5.133.2.9 TIM_ClearOC1Ref()

```
void TIM_ClearOC1Ref (
             TIM_TypeDef * TIMx,
             uint16_t TIM_OCClear )
```

Clears or safeguards the OCREF1 signal on an external event.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
TIM_OCClear	new state of the Output Compare Clear Enable Bit. This parameter can be one of the following values: • TIM_OCClear_Enable: TIM Output clear enable
	• TIM_OCClear_Disable: TIM Output clear disable

Return values

None	

5.133.2.10 TIM_ClearOC2Ref()

```
void TIM_ClearOC2Ref (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCClear )
```

Clears or safeguards the OCREF2 signal on an external event.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
TIM_OCClear	new state of the Output Compare Clear Enable Bit. This parameter can be one of the following values: • TIM_OCClear_Enable: TIM Output clear enable • TIM_OCClear_Disable: TIM Output clear disable

Return values

None	

5.133.2.11 TIM_ClearOC3Ref()

```
void TIM_ClearOC3Ref (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCClear )
```

Clears or safeguards the OCREF3 signal on an external event.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_OCClear	new state of the Output Compare Clear Enable Bit. This parameter can be one of the following values: • TIM_OCClear_Enable: TIM Output clear enable • TIM_OCClear_Disable: TIM Output clear disable

Return values

None	

5.133.2.12 TIM_ClearOC4Ref()

```
void TIM_ClearOC4Ref (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCClear )
```

Clears or safeguards the OCREF4 signal on an external event.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_OCClear	new state of the Output Compare Clear Enable Bit. This parameter can be one of the following values: • TIM_OCClear_Enable: TIM Output clear enable • TIM_OCClear_Disable: TIM Output clear disable

Return values

None	

5.133.2.13 TIM_Cmd()

```
void TIM_Cmd (
            TIM_TypeDef * TIMx,
            FunctionalState NewState )
```

Enables or disables the specified TIM peripheral.

Parameters

TIMx	where x can be 1 to 14 to select the TIMx peripheral.
NewState	new state of the TIMx peripheral. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.133.2.14 TIM_CounterModeConfig()

```
void TIM_CounterModeConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_CounterMode )
```

Specifies the TIMx Counter Mode to be used.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_CounterMode	specifies the Counter Mode to be used This parameter can be one of the following values: • TIM_CounterMode_Up: TIM Up Counting Mode • TIM_CounterMode_Down: TIM Down Counting Mode • TIM_CounterMode_CenterAligned1: TIM Center Aligned Mode1 • TIM_CounterMode_CenterAligned2: TIM Center Aligned Mode2 • TIM_CounterMode_CenterAligned3: TIM Center Aligned Mode3

Return values

None	

5.133.2.15 TIM_CtrlPWMOutputs()

```
void TIM_CtrlPWMOutputs (
            TIM_TypeDef * TIMx,
            FunctionalState NewState )
```

Enables or disables the TIM peripheral Main Outputs.

Parameters

TIMx	where x can be 1 or 8 to select the TIMx peripheral.
NewState	new state of the TIM peripheral Main Outputs. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.133.2.16 TIM_DeInit()

```
void TIM_DeInit (
            TIM_TypeDef * TIMx )
```

Deinitializes the TIMx peripheral registers to their default reset values.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.

Return values

None	

5.133.2.17 TIM_DMACmd()

```
void TIM_DMACmd (
              TIM_TypeDef * TIMx,
              uint16_t TIM_DMASource,
              FunctionalState NewState )
```

Enables or disables the TIMx's DMA Requests.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 6, 7 or 8 to select the TIM peripheral.
TIM_DMASource	specifies the DMA Request sources. This parameter can be any combination of the following values: • TIM_DMA_Update: TIM update Interrupt source • TIM_DMA_CC1: TIM Capture Compare 1 DMA source • TIM_DMA_CC2: TIM Capture Compare 2 DMA source • TIM_DMA_CC3: TIM Capture Compare 3 DMA source • TIM_DMA_CC4: TIM Capture Compare 4 DMA source • TIM_DMA_COM: TIM Commutation DMA source • TIM_DMA_Trigger: TIM Trigger DMA source
NewState	new state of the DMA Request sources. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.133.2.18 TIM_DMAConfig()

```
void TIM_DMAConfig (
              TIM_TypeDef * TIMx,
```

```
            uint16_t TIM_DMABase,
            uint16_t TIM_DMABurstLength )
```

Configures the TIMx's DMA interface.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_DMABase	DMA Base address. This parameter can be one of the following values: • TIM_DMABase_CR1 • TIM_DMABase_CR2 • TIM_DMABase_SMCR • TIM_DMABase_DIER • TIM1_DMABase_SR • TIM_DMABase_EGR • TIM_DMABase_CCMR1 • TIM_DMABase_CCMR2 • TIM_DMABase_CCER • TIM_DMABase_CNT • TIM_DMABase_PSC • TIM_DMABase_ARR • TIM_DMABase_RCR • TIM_DMABase_CCR1 • TIM_DMABase_CCR2 • TIM_DMABase_CCR3 • TIM_DMABase_CCR4 • TIM_DMABase_BDTR • TIM_DMABase_DCR
TIM_DMABurstLength	DMA Burst length. This parameter can be one value between: TIM_DMABurstLength_1Transfer and TIM_DMABurstLength_18Transfers.

Return values

None	

5.133.2.19 TIM_EncoderInterfaceConfig()

```
void TIM_EncoderInterfaceConfig (
             TIM_TypeDef * TIMx,
             uint16_t TIM_EncoderMode,
             uint16_t TIM_IC1Polarity,
             uint16_t TIM_IC2Polarity )
```

Configures the TIMx Encoder Interface.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
TIM_EncoderMode	specifies the TIMx Encoder Mode. This parameter can be one of the following values: • TIM_EncoderMode_TI1: Counter counts on TI1FP1 edge depending on TI2FP2 level. • TIM_EncoderMode_TI2: Counter counts on TI2FP2 edge depending on TI1FP1 level. • TIM_EncoderMode_TI12: Counter counts on both TI1FP1 and TI2FP2 edges depending on the level of the other input.
TIM_IC1Polarity	specifies the IC1 Polarity This parameter can be one of the following values: • TIM_ICPolarity_Falling: IC Falling edge. • TIM_ICPolarity_Rising: IC Rising edge.
TIM_IC2Polarity	specifies the IC2 Polarity This parameter can be one of the following values: • TIM_ICPolarity_Falling: IC Falling edge. • TIM_ICPolarity_Rising: IC Rising edge.

Return values

None	

5.133.2.20 TIM_ETRClockMode1Config()

```
void TIM_ETRClockMode1Config (
             TIM_TypeDef * TIMx,
             uint16_t TIM_ExtTRGPrescaler,
             uint16_t TIM_ExtTRGPolarity,
             uint16_t ExtTRGFilter )
```

Configures the External clock Mode1.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.

Parameters

TIM_ExtTRGPrescaler	The external Trigger Prescaler. This parameter can be one of the following values: • TIM_ExtTRGPSC_OFF: ETRP Prescaler OFF. • TIM_ExtTRGPSC_DIV2: ETRP frequency divided by 2. • TIM_ExtTRGPSC_DIV4: ETRP frequency divided by 4. • TIM_ExtTRGPSC_DIV8: ETRP frequency divided by 8.
TIM_ExtTRGPolarity	The external Trigger Polarity. This parameter can be one of the following values: • TIM_ExtTRGPolarity_Inverted: active low or falling edge active. • TIM_ExtTRGPolarity_NonInverted: active high or rising edge active.
ExtTRGFilter	External Trigger Filter. This parameter must be a value between 0x00 and 0x0F

Return values

None	

5.133.2.21 TIM_ETRClockMode2Config()

```
void TIM_ETRClockMode2Config (
             TIM_TypeDef * TIMx,
             uint16_t TIM_ExtTRGPrescaler,
             uint16_t TIM_ExtTRGPolarity,
             uint16_t ExtTRGFilter )
```

Configures the External clock Mode2.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_ExtTRGPrescaler	The external Trigger Prescaler. This parameter can be one of the following values: • TIM_ExtTRGPSC_OFF: ETRP Prescaler OFF. • TIM_ExtTRGPSC_DIV2: ETRP frequency divided by 2. • TIM_ExtTRGPSC_DIV4: ETRP frequency divided by 4. • TIM_ExtTRGPSC_DIV8: ETRP frequency divided by 8.
TIM_ExtTRGPolarity	The external Trigger Polarity. This parameter can be one of the following values: • TIM_ExtTRGPolarity_Inverted: active low or falling edge active. • TIM_ExtTRGPolarity_NonInverted: active high or rising edge active.
ExtTRGFilter	External Trigger Filter. This parameter must be a value between 0x00 and 0x0F

Return values

None	

5.133.2.22 TIM_ETRConfig()

```
void TIM_ETRConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_ExtTRGPrescaler,
            uint16_t TIM_ExtTRGPolarity,
            uint16_t ExtTRGFilter )
```

Configures the TIMx External Trigger (ETR).

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_ExtTRGPrescaler	The external Trigger Prescaler. This parameter can be one of the following values: • TIM_ExtTRGPSC_OFF: ETRP Prescaler OFF. • TIM_ExtTRGPSC_DIV2: ETRP frequency divided by 2. • TIM_ExtTRGPSC_DIV4: ETRP frequency divided by 4. • TIM_ExtTRGPSC_DIV8: ETRP frequency divided by 8.
TIM_ExtTRGPolarity	The external Trigger Polarity. This parameter can be one of the following values: • TIM_ExtTRGPolarity_Inverted: active low or falling edge active. • TIM_ExtTRGPolarity_NonInverted: active high or rising edge active.
ExtTRGFilter	External Trigger Filter. This parameter must be a value between 0x00 and 0x0F

Return values

None	

5.133.2.23 TIM_ForcedOC1Config()

```
void TIM_ForcedOC1Config (
            TIM_TypeDef * TIMx,
            uint16_t TIM_ForcedAction )
```

Forces the TIMx output 1 waveform to active or inactive level.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
TIM_ForcedAction	specifies the forced Action to be set to the output waveform. This parameter can be one of the following values: • TIM_ForcedAction_Active: Force active level on OC1REF • TIM_ForcedAction_InActive: Force inactive level on OC1REF.

Return values

None	

5.133.2.24 TIM_ForcedOC2Config()

```
void TIM_ForcedOC2Config (
             TIM_TypeDef * TIMx,
             uint16_t TIM_ForcedAction )
```

Forces the TIMx output 2 waveform to active or inactive level.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
TIM_ForcedAction	specifies the forced Action to be set to the output waveform. This parameter can be one of the following values: • TIM_ForcedAction_Active: Force active level on OC2REF • TIM_ForcedAction_InActive: Force inactive level on OC2REF.

Return values

None	

5.133.2.25 TIM_ForcedOC3Config()

```
void TIM_ForcedOC3Config (
             TIM_TypeDef * TIMx,
             uint16_t TIM_ForcedAction )
```

Forces the TIMx output 3 waveform to active or inactive level.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_ForcedAction	specifies the forced Action to be set to the output waveform. This parameter can be one of the following values: • TIM_ForcedAction_Active: Force active level on OC3REF • TIM_ForcedAction_InActive: Force inactive level on OC3REF.

Return values

None	

5.133.2.26 TIM_ForcedOC4Config()

```
void TIM_ForcedOC4Config (
             TIM_TypeDef * TIMx,
             uint16_t TIM_ForcedAction )
```

Forces the TIMx output 4 waveform to active or inactive level.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_ForcedAction	specifies the forced Action to be set to the output waveform. This parameter can be one of the following values: • TIM_ForcedAction_Active: Force active level on OC4REF • TIM_ForcedAction_InActive: Force inactive level on OC4REF.

Return values

None	

5.133.2.27 TIM_GenerateEvent()

```
void TIM_GenerateEvent (
             TIM_TypeDef * TIMx,
             uint16_t TIM_EventSource )
```

Configures the TIMx event to be generate by software.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
TIM_EventSource	specifies the event source. This parameter can be one or more of the following values: • TIM_EventSource_Update: Timer update Event source • TIM_EventSource_CC1: Timer Capture Compare 1 Event source • TIM_EventSource_CC2: Timer Capture Compare 2 Event source • TIM_EventSource_CC3: Timer Capture Compare 3 Event source • TIM_EventSource_CC4: Timer Capture Compare 4 Event source • TIM_EventSource_COM: Timer COM event source • TIM_EventSource_Trigger: Timer Trigger Event source • TIM_EventSource_Break: Timer Break event source

Note

TIM6 and TIM7 can only generate an update event.

TIM_EventSource_COM and TIM_EventSource_Break are used only with TIM1 and TIM8.

Return values

None	

5.133.2.28 TIM_GetCapture1()

```
uint32_t TIM_GetCapture1 (
             TIM_TypeDef * TIMx )
```

Gets the TIMx Input Capture 1 value.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.

Return values

Capture	Compare 1 Register value.

5.133.2.29 TIM_GetCapture2()

```
uint32_t TIM_GetCapture2 (
            TIM_TypeDef * TIMx )
```

Gets the TIMx Input Capture 2 value.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.

Return values

Capture	Compare 2 Register value.

5.133.2.30 TIM_GetCapture3()

```
uint32_t TIM_GetCapture3 (
            TIM_TypeDef * TIMx )
```

Gets the TIMx Input Capture 3 value.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.

Return values

Capture	Compare 3 Register value.

5.133.2.31 TIM_GetCapture4()

```
uint32_t TIM_GetCapture4 (
            TIM_TypeDef * TIMx )
```

Gets the TIMx Input Capture 4 value.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.

Return values

Capture	Compare 4 Register value.

5.133.2.32 TIM_GetCounter()

```
uint32_t TIM_GetCounter (
            TIM_TypeDef * TIMx )
```

Gets the TIMx Counter value.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.

Return values

Counter	Register value

5.133.2.33 TIM_GetFlagStatus()

```
FlagStatus TIM_GetFlagStatus (
            TIM_TypeDef * TIMx,
            uint16_t TIM_FLAG )
```

Checks whether the specified TIM flag is set or not.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
TIM_FLAG	specifies the flag to check. This parameter can be one of the following values: • TIM_FLAG_Update: TIM update Flag • TIM_FLAG_CC1: TIM Capture Compare 1 Flag • TIM_FLAG_CC2: TIM Capture Compare 2 Flag • TIM_FLAG_CC3: TIM Capture Compare 3 Flag • TIM_FLAG_CC4: TIM Capture Compare 4 Flag • TIM_FLAG_COM: TIM Commutation Flag • TIM_FLAG_Trigger: TIM Trigger Flag • TIM_FLAG_Break: TIM Break Flag • TIM_FLAG_CC1OF: TIM Capture Compare 1 over capture Flag • TIM_FLAG_CC2OF: TIM Capture Compare 2 over capture Flag • TIM_FLAG_CC3OF: TIM Capture Compare 3 over capture Flag • TIM_FLAG_CC4OF: TIM Capture Compare 4 over capture Flag

Note

TIM6 and TIM7 can have only one update flag.

TIM_FLAG_COM and TIM_FLAG_Break are used only with TIM1 and TIM8.

Return values

The	new state of TIM_FLAG (SET or RESET).

5.133.2.34 TIM_GetITStatus()

```
ITStatus TIM_GetITStatus (
             TIM_TypeDef * TIMx,
             uint16_t TIM_IT )
```

Checks whether the TIM interrupt has occurred or not.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
TIM_IT	specifies the TIM interrupt source to check. This parameter can be one of the following values: • TIM_IT_Update: TIM update Interrupt source • TIM_IT_CC1: TIM Capture Compare 1 Interrupt source • TIM_IT_CC2: TIM Capture Compare 2 Interrupt source • TIM_IT_CC3: TIM Capture Compare 3 Interrupt source • TIM_IT_CC4: TIM Capture Compare 4 Interrupt source • TIM_IT_COM: TIM Commutation Interrupt source • TIM_IT_Trigger: TIM Trigger Interrupt source • TIM_IT_Break: TIM Break Interrupt source

Note

TIM6 and TIM7 can generate only an update interrupt.

TIM_IT_COM and TIM_IT_Break are used only with TIM1 and TIM8.

Return values

The	new state of the TIM_IT(SET or RESET).

5.133.2.35 TIM_GetPrescaler()

```
uint16_t TIM_GetPrescaler (
            TIM_TypeDef * TIMx )
```

Gets the TIMx Prescaler value.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.

Return values

Prescaler	Register value.

5.133.2.36 TIM_ICInit()

```
void TIM_ICInit (
            TIM_TypeDef * TIMx,
            TIM_ICInitTypeDef * TIM_ICInitStruct )
```

Initializes the TIM peripheral according to the specified parameters in the TIM_ICInitStruct.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
TIM_ICInitStruct	pointer to a TIM_ICInitTypeDef structure that contains the configuration information for the specified TIM peripheral.

Return values

None	

5.133.2.37 TIM_ICStructInit()

```
void TIM_ICStructInit (
            TIM_ICInitTypeDef * TIM_ICInitStruct )
```

Fills each TIM_ICInitStruct member with its default value.

Parameters

TIM_ICInitStruct	pointer to a TIM_ICInitTypeDef structure which will be initialized.

Return values

None	

5.133.2.38 TIM_InternalClockConfig()

```
void TIM_InternalClockConfig (
            TIM_TypeDef * TIMx )
```

Configures the TIMx internal Clock.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.

Return values

None	

5.133.2.39 TIM_ITConfig()

```
void TIM_ITConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_IT,
            FunctionalState NewState )
```

Enables or disables the specified TIM interrupts.

Parameters

TIMx	where x can be 1 to 14 to select the TIMx peripheral.
TIM_IT	specifies the TIM interrupts sources to be enabled or disabled. This parameter can be any combination of the following values: • TIM_IT_Update: TIM update Interrupt source • TIM_IT_CC1: TIM Capture Compare 1 Interrupt source • TIM_IT_CC2: TIM Capture Compare 2 Interrupt source • TIM_IT_CC3: TIM Capture Compare 3 Interrupt source • TIM_IT_CC4: TIM Capture Compare 4 Interrupt source • TIM_IT_COM: TIM Commutation Interrupt source • TIM_IT_Trigger: TIM Trigger Interrupt source • TIM_IT_Break: TIM Break Interrupt source

Note

For TIM6 and TIM7 only the parameter TIM_IT_Update can be used

For TIM9 and TIM12 only one of the following parameters can be used: TIM_IT_Update, TIM_IT_CC1, TIM↩ _IT_CC2 or TIM_IT_Trigger.

For TIM10, TIM11, TIM13 and TIM14 only one of the following parameters can be used: TIM_IT_Update or TIM_IT_CC1

TIM_IT_COM and TIM_IT_Break can be used only with TIM1 and TIM8

Parameters

NewState	new state of the TIM interrupts. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.133.2.40 TIM_ITRxExternalClockConfig()

```
void TIM_ITRxExternalClockConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_InputTriggerSource )
```

Configures the TIMx Internal Trigger as External Clock.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
TIM_InputTriggerSource	Trigger source. This parameter can be one of the following values: • TIM_TS_ITR0: Internal Trigger 0 • TIM_TS_ITR1: Internal Trigger 1 • TIM_TS_ITR2: Internal Trigger 2 • TIM_TS_ITR3: Internal Trigger 3

Return values

None	

5.133.2.41 TIM_OC1FastConfig()

```
void TIM_OC1FastConfig (
```

```
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCFast )
```

Configures the TIMx Output Compare 1 Fast feature.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
TIM_OCFast	new state of the Output Compare Fast Enable Bit. This parameter can be one of the following values: • TIM_OCFast_Enable: TIM output compare fast enable • TIM_OCFast_Disable: TIM output compare fast disable

Return values

None	

5.133.2.42 TIM_OC1Init()

```
void TIM_OC1Init (
            TIM_TypeDef * TIMx,
            TIM_OCInitTypeDef * TIM_OCInitStruct )
```

Initializes the TIMx Channel1 according to the specified parameters in the TIM_OCInitStruct.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
TIM_OCInitStruct	pointer to a TIM_OCInitTypeDef structure that contains the configuration information for the specified TIM peripheral.

Return values

None	

5.133.2.43 TIM_OC1NPolarityConfig()

```
void TIM_OC1NPolarityConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCNPolarity )
```

Configures the TIMx Channel 1N polarity.

Parameters

TIMx	where x can be 1 or 8 to select the TIM peripheral.
TIM_OCNPolarity	specifies the OC1N Polarity This parameter can be one of the following values: • TIM_OCNPolarity_High: Output Compare active high • TIM_OCNPolarity_Low: Output Compare active low

Return values

None	

5.133.2.44 TIM_OC1PolarityConfig()

```
void TIM_OC1PolarityConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCPolarity )
```

Configures the TIMx channel 1 polarity.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
TIM_OCPolarity	specifies the OC1 Polarity This parameter can be one of the following values: • TIM_OCPolarity_High: Output Compare active high • TIM_OCPolarity_Low: Output Compare active low

Return values

None	

5.133.2.45 TIM_OC1PreloadConfig()

```
void TIM_OC1PreloadConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCPreload )
```

Enables or disables the TIMx peripheral Preload register on CCR1.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.

Parameters

TIM_OCPreload	new state of the TIMx peripheral Preload register This parameter can be one of the following values: • TIM_OCPreload_Enable • TIM_OCPreload_Disable

Return values

None	

5.133.2.46 TIM_OC2FastConfig()

```
void TIM_OC2FastConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCFast )
```

Configures the TIMx Output Compare 2 Fast feature.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
TIM_OCFast	new state of the Output Compare Fast Enable Bit. This parameter can be one of the following values: • TIM_OCFast_Enable: TIM output compare fast enable • TIM_OCFast_Disable: TIM output compare fast disable

Return values

None	

5.133.2.47 TIM_OC2Init()

```
void TIM_OC2Init (
            TIM_TypeDef * TIMx,
            TIM_OCInitTypeDef * TIM_OCInitStruct )
```

Initializes the TIMx Channel2 according to the specified parameters in the TIM_OCInitStruct.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
TIM_OCInitStruct	pointer to a TIM_OCInitTypeDef structure that contains the configuration information for the specified TIM peripheral.

Return values

None	

5.133.2.48 TIM_OC2NPolarityConfig()

```
void TIM_OC2NPolarityConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCNPolarity )
```

Configures the TIMx Channel 2N polarity.

Parameters

TIMx	where x can be 1 or 8 to select the TIM peripheral.
TIM_OCNPolarity	specifies the OC2N Polarity This parameter can be one of the following values: • TIM_OCNPolarity_High: Output Compare active high • TIM_OCNPolarity_Low: Output Compare active low

Return values

None	

5.133.2.49 TIM_OC2PolarityConfig()

```
void TIM_OC2PolarityConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCPolarity )
```

Configures the TIMx channel 2 polarity.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
TIM_OCPolarity	specifies the OC2 Polarity This parameter can be one of the following values: • TIM_OCPolarity_High: Output Compare active high • TIM_OCPolarity_Low: Output Compare active low

Return values

None	

5.133.2.50 TIM_OC2PreloadConfig()

```
void TIM_OC2PreloadConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCPreload )
```

Enables or disables the TIMx peripheral Preload register on CCR2.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
TIM_OCPreload	new state of the TIMx peripheral Preload register This parameter can be one of the following values: • TIM_OCPreload_Enable • TIM_OCPreload_Disable

Return values

None	

5.133.2.51 TIM_OC3FastConfig()

```
void TIM_OC3FastConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCFast )
```

Configures the TIMx Output Compare 3 Fast feature.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_OCFast	new state of the Output Compare Fast Enable Bit. This parameter can be one of the following values: • TIM_OCFast_Enable: TIM output compare fast enable • TIM_OCFast_Disable: TIM output compare fast disable

Return values

None	

5.133.2.52 TIM_OC3Init()

```
void TIM_OC3Init (
            TIM_TypeDef * TIMx,
            TIM_OCInitTypeDef * TIM_OCInitStruct )
```

Initializes the TIMx Channel3 according to the specified parameters in the TIM_OCInitStruct.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_OCInitStruct	pointer to a TIM_OCInitTypeDef structure that contains the configuration information for the specified TIM peripheral.

Return values

None	

5.133.2.53 TIM_OC3NPolarityConfig()

```
void TIM_OC3NPolarityConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCNPolarity )
```

Configures the TIMx Channel 3N polarity.

Parameters

TIMx	where x can be 1 or 8 to select the TIM peripheral.
TIM_OCNPolarity	specifies the OC3N Polarity This parameter can be one of the following values: • TIM_OCNPolarity_High: Output Compare active high • TIM_OCNPolarity_Low: Output Compare active low

Return values

None	

5.133.2.54 TIM_OC3PolarityConfig()

```
void TIM_OC3PolarityConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCPolarity )
```

Configures the TIMx channel 3 polarity.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_OCPolarity	specifies the OC3 Polarity This parameter can be one of the following values: • TIM_OCPolarity_High: Output Compare active high • TIM_OCPolarity_Low: Output Compare active low

Return values

None	

5.133.2.55 TIM_OC3PreloadConfig()

```
void TIM_OC3PreloadConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCPreload )
```

Enables or disables the TIMx peripheral Preload register on CCR3.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_OCPreload	new state of the TIMx peripheral Preload register This parameter can be one of the following values: • TIM_OCPreload_Enable • TIM_OCPreload_Disable

Return values

None	

5.133.2.56 TIM_OC4FastConfig()

```
void TIM_OC4FastConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCFast )
```

Configures the TIMx Output Compare 4 Fast feature.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.

Parameters

TIM_OCFast	new state of the Output Compare Fast Enable Bit. This parameter can be one of the following values: • TIM_OCFast_Enable: TIM output compare fast enable • TIM_OCFast_Disable: TIM output compare fast disable

Return values

None	

5.133.2.57 TIM_OC4Init()

```
void TIM_OC4Init (
            TIM_TypeDef * TIMx,
            TIM_OCInitTypeDef * TIM_OCInitStruct )
```

Initializes the TIMx Channel4 according to the specified parameters in the TIM_OCInitStruct.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_OCInitStruct	pointer to a TIM_OCInitTypeDef structure that contains the configuration information for the specified TIM peripheral.

Return values

None	

5.133.2.58 TIM_OC4PolarityConfig()

```
void TIM_OC4PolarityConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCPolarity )
```

Configures the TIMx channel 4 polarity.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_OCPolarity	specifies the OC4 Polarity This parameter can be one of the following values: • TIM_OCPolarity_High: Output Compare active high • TIM_OCPolarity_Low: Output Compare active low

Return values

None	

5.133.2.59 TIM_OC4PreloadConfig()

```
void TIM_OC4PreloadConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCPreload )
```

Enables or disables the TIMx peripheral Preload register on CCR4.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_OCPreload	new state of the TIMx peripheral Preload register This parameter can be one of the following values: • TIM_OCPreload_Enable • TIM_OCPreload_Disable

Return values

None	

5.133.2.60 TIM_OCStructInit()

```
void TIM_OCStructInit (
            TIM_OCInitTypeDef * TIM_OCInitStruct )
```

Fills each TIM_OCInitStruct member with its default value.

Parameters

TIM_OCInitStruct	pointer to a TIM_OCInitTypeDef structure which will be initialized.

Return values

None	

5.133.2.61 TIM_PrescalerConfig()

```
void TIM_PrescalerConfig (
             TIM_TypeDef * TIMx,
             uint16_t Prescaler,
             uint16_t TIM_PSCReloadMode )
```

Configures the TIMx Prescaler.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
Prescaler	specifies the Prescaler Register value
TIM_PSCReloadMode	specifies the TIM Prescaler Reload mode This parameter can be one of the following values: • TIM_PSCReloadMode_Update: The Prescaler is loaded at the update event. • TIM_PSCReloadMode_Immediate: The Prescaler is loaded immediatly.

Return values

None	

5.133.2.62 TIM_PWMIConfig()

```
void TIM_PWMIConfig (
             TIM_TypeDef * TIMx,
             TIM_ICInitTypeDef * TIM_ICInitStruct )
```

Configures the TIM peripheral according to the specified parameters in the TIM_ICInitStruct to measure an external PWM signal.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5,8, 9 or 12 to select the TIM peripheral.
TIM_ICInitStruct	pointer to a TIM_ICInitTypeDef structure that contains the configuration information for the specified TIM peripheral.

Return values

None	

5.133.2.63 TIM_RemapConfig()

```
void TIM_RemapConfig (
```

```
                TIM_TypeDef * TIMx,
                uint16_t TIM_Remap )
```

Configures the TIM2, TIM5 and TIM11 Remapping input capabilities.

Parameters

TIMx	where x can be 2, 5 or 11 to select the TIM peripheral.
TIM_Remap	specifies the TIM input remapping source. This parameter can be one of the following values: • TIM2_TIM8_TRGO: TIM2 ITR1 input is connected to TIM8 Trigger output(default) • TIM2_ETH_PTP: TIM2 ITR1 input is connected to ETH PTP trogger output. • TIM2_USBFS_SOF: TIM2 ITR1 input is connected to USB FS SOF. • TIM2_USBHS_SOF: TIM2 ITR1 input is connected to USB HS SOF. • TIM5_GPIO: TIM5 CH4 input is connected to dedicated Timer pin(default) • TIM5_LSI: TIM5 CH4 input is connected to LSI clock. • TIM5_LSE: TIM5 CH4 input is connected to LSE clock. • TIM5_RTC: TIM5 CH4 input is connected to RTC Output event. • TIM11_GPIO: TIM11 CH4 input is connected to dedicated Timer pin(default) • TIM11_HSE: TIM11 CH4 input is connected to HSE_RTC clock (HSE divided by a programmable prescaler)

Return values

None	

5.133.2.64 TIM_SelectCCDMA()

```
void TIM_SelectCCDMA (
                TIM_TypeDef * TIMx,
                FunctionalState NewState )
```

Selects the TIMx peripheral Capture Compare DMA source.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
NewState	new state of the Capture Compare DMA source This parameter can be: ENABLE or DISABLE.

Return values

None	

5.133.2.65 TIM_SelectCOM()

```
void TIM_SelectCOM (
            TIM_TypeDef * TIMx,
            FunctionalState NewState )
```

Selects the TIM peripheral Commutation event.

Parameters

TIMx	where x can be 1 or 8 to select the TIMx peripheral
NewState	new state of the Commutation event. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.133.2.66 TIM_SelectHallSensor()

```
void TIM_SelectHallSensor (
            TIM_TypeDef * TIMx,
            FunctionalState NewState )
```

Enables or disables the TIMx's Hall sensor interface.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
NewState	new state of the TIMx Hall sensor interface. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.133.2.67 TIM_SelectInputTrigger()

```
void TIM_SelectInputTrigger (
            TIM_TypeDef * TIMx,
            uint16_t TIM_InputTriggerSource )
```

Selects the Input Trigger source.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9, 10, 11, 12, 13 or 14 to select the TIM peripheral.
TIM_InputTriggerSource	The Input Trigger source. This parameter can be one of the following values: • TIM_TS_ITR0: Internal Trigger 0 • TIM_TS_ITR1: Internal Trigger 1 • TIM_TS_ITR2: Internal Trigger 2 • TIM_TS_ITR3: Internal Trigger 3 • TIM_TS_TI1F_ED: TI1 Edge Detector • TIM_TS_TI1FP1: Filtered Timer Input 1 • TIM_TS_TI2FP2: Filtered Timer Input 2 • TIM_TS_ETRF: External Trigger input

Return values

None	

5.133.2.68 TIM_SelectMasterSlaveMode()

```
void TIM_SelectMasterSlaveMode (
             TIM_TypeDef * TIMx,
             uint16_t TIM_MasterSlaveMode )
```

Sets or Resets the TIMx Master/Slave Mode.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
TIM_MasterSlaveMode	specifies the Timer Master Slave Mode. This parameter can be one of the following values: • TIM_MasterSlaveMode_Enable: synchronization between the current timer and its slaves (through TRGO) • TIM_MasterSlaveMode_Disable: No action

Return values

None	

5.133.2.69 TIM_SelectOCxM()

```
void TIM_SelectOCxM (
             TIM_TypeDef * TIMx,
             uint16_t TIM_Channel,
             uint16_t TIM_OCMode )
```

Selects the TIM Output Compare Mode.

Note

This function disables the selected channel before changing the Output Compare Mode. If needed, user has to enable this channel using TIM_CCxCmd() and TIM_CCxNCmd() functions.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
TIM_Channel	specifies the TIM Channel This parameter can be one of the following values: • TIM_Channel_1: TIM Channel 1 • TIM_Channel_2: TIM Channel 2 • TIM_Channel_3: TIM Channel 3 • TIM_Channel_4: TIM Channel 4
TIM_OCMode	specifies the TIM Output Compare Mode. This parameter can be one of the following values: • TIM_OCMode_Timing • TIM_OCMode_Active • TIM_OCMode_Toggle • TIM_OCMode_PWM1 • TIM_OCMode_PWM2 • TIM_ForcedAction_Active • TIM_ForcedAction_InActive

Return values

None	

5.133.2.70 TIM_SelectOnePulseMode()

```
void TIM_SelectOnePulseMode (
             TIM_TypeDef * TIMx,
             uint16_t TIM_OPMode )
```

Selects the TIMx's One Pulse Mode.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
TIM_OPMode	specifies the OPM Mode to be used. This parameter can be one of the following values: • TIM_OPMode_Single • TIM_OPMode_Repetitive

Return values

None	

5.133.2.71 TIM_SelectOutputTrigger()

```
void TIM_SelectOutputTrigger (
             TIM_TypeDef * TIMx,
             uint16_t TIM_TRGOSource )
```

Selects the TIMx Trigger Output Mode.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 6, 7 or 8 to select the TIM peripheral.
TIM_TRGOSource	specifies the Trigger Output source. This parameter can be one of the following values:

- For all TIMx
  - TIM_TRGOSource_Reset: The UG bit in the TIM_EGR register is used as the trigger output(TRGO)
  - TIM_TRGOSource_Enable: The Counter Enable CEN is used as the trigger output(TRGO)
  - TIM_TRGOSource_Update: The update event is selected as the trigger output(TRGO)
- For all TIMx except TIM6 and TIM7
  - TIM_TRGOSource_OC1: The trigger output sends a positive pulse when the CC1IF flag is to be set, as soon as a capture or compare match occurs(TRGO)
  - TIM_TRGOSource_OC1Ref: OC1REF signal is used as the trigger output(TRGO)
  - TIM_TRGOSource_OC2Ref: OC2REF signal is used as the trigger output(TRGO)
  - TIM_TRGOSource_OC3Ref: OC3REF signal is used as the trigger output(TRGO)
  - TIM_TRGOSource_OC4Ref: OC4REF signal is used as the trigger output(TRGO)

Return values

None	

5.133.2.72 TIM_SelectSlaveMode()

```
void TIM_SelectSlaveMode (
             TIM_TypeDef * TIMx,
             uint16_t TIM_SlaveMode )
```

Selects the TIMx Slave Mode.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
TIM_SlaveMode	specifies the Timer Slave Mode. This parameter can be one of the following values: • TIM_SlaveMode_Reset: Rising edge of the selected trigger signal(TRGI) reinitialize the counter and triggers an update of the registers • TIM_SlaveMode_Gated: The counter clock is enabled when the trigger signal (TRGI) is high • TIM_SlaveMode_Trigger: The counter starts at a rising edge of the trigger TRGI • TIM_SlaveMode_External1: Rising edges of the selected trigger (TRGI) clock the counter

Return values

None	

5.133.2.73 TIM_SetAutoreload()

```
void TIM_SetAutoreload (
             TIM_TypeDef * TIMx,
             uint32_t Autoreload )
```

Sets the TIMx Autoreload Register value.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
Autoreload	specifies the Autoreload register new value.

Return values

None	

5.133.2.74 TIM_SetClockDivision()

```
void TIM_SetClockDivision (
            TIM_TypeDef * TIMx,
            uint16_t TIM_CKD )
```

Sets the TIMx Clock Division value.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
TIM_CKD	specifies the clock division value. This parameter can be one of the following value: • TIM_CKD_DIV1: TDTS = Tck_tim • TIM_CKD_DIV2: TDTS = 2∗Tck_tim • TIM_CKD_DIV4: TDTS = 4∗Tck_tim

Return values

None	

5.133.2.75 TIM_SetCompare1()

```
void TIM_SetCompare1 (
            TIM_TypeDef * TIMx,
            uint32_t Compare1 )
```

Sets the TIMx Capture Compare1 Register value.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
Compare1	specifies the Capture Compare1 register new value.

Return values

None	

5.133.2.76 TIM_SetCompare2()

```
void TIM_SetCompare2 (
            TIM_TypeDef * TIMx,
            uint32_t Compare2 )
```

Sets the TIMx Capture Compare2 Register value.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
Compare2	specifies the Capture Compare2 register new value.

Return values

None	

5.133.2.77 TIM_SetCompare3()

```
void TIM_SetCompare3 (
             TIM_TypeDef * TIMx,
             uint32_t Compare3 )
```

Sets the TIMx Capture Compare3 Register value.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
Compare3	specifies the Capture Compare3 register new value.

Return values

None	

5.133.2.78 TIM_SetCompare4()

```
void TIM_SetCompare4 (
             TIM_TypeDef * TIMx,
             uint32_t Compare4 )
```

Sets the TIMx Capture Compare4 Register value.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
Compare4	specifies the Capture Compare4 register new value.

Return values

None	

5.133.2.79 TIM_SetCounter()

```
void TIM_SetCounter (
            TIM_TypeDef * TIMx,
            uint32_t Counter )
```

Sets the TIMx Counter Register value.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
Counter	specifies the Counter register new value.

Return values

None	

5.133.2.80 TIM_SetIC1Prescaler()

```
void TIM_SetIC1Prescaler (
            TIM_TypeDef * TIMx,
            uint16_t TIM_ICPSC )
```

Sets the TIMx Input Capture 1 prescaler.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
TIM_ICPSC	specifies the Input Capture1 prescaler new value. This parameter can be one of the following values: • TIM_ICPSC_DIV1: no prescaler • TIM_ICPSC_DIV2: capture is done once every 2 events • TIM_ICPSC_DIV4: capture is done once every 4 events • TIM_ICPSC_DIV8: capture is done once every 8 events

Return values

None	

5.133.2.81 TIM_SetIC2Prescaler()

```
void TIM_SetIC2Prescaler (
            TIM_TypeDef * TIMx,
            uint16_t TIM_ICPSC )
```

Sets the TIMx Input Capture 2 prescaler.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
TIM_ICPSC	specifies the Input Capture2 prescaler new value. This parameter can be one of the following values: • TIM_ICPSC_DIV1: no prescaler • TIM_ICPSC_DIV2: capture is done once every 2 events • TIM_ICPSC_DIV4: capture is done once every 4 events • TIM_ICPSC_DIV8: capture is done once every 8 events

Return values

None	

5.133.2.82 TIM_SetIC3Prescaler()

```
void TIM_SetIC3Prescaler (
            TIM_TypeDef * TIMx,
            uint16_t TIM_ICPSC )
```

Sets the TIMx Input Capture 3 prescaler.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_ICPSC	specifies the Input Capture3 prescaler new value. This parameter can be one of the following values: • TIM_ICPSC_DIV1: no prescaler • TIM_ICPSC_DIV2: capture is done once every 2 events • TIM_ICPSC_DIV4: capture is done once every 4 events • TIM_ICPSC_DIV8: capture is done once every 8 events

Return values

None	

5.133.2.83 TIM_SetIC4Prescaler()

```
void TIM_SetIC4Prescaler (
            TIM_TypeDef * TIMx,
            uint16_t TIM_ICPSC )
```

Sets the TIMx Input Capture 4 prescaler.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_ICPSC	specifies the Input Capture4 prescaler new value. This parameter can be one of the following values: • TIM_ICPSC_DIV1: no prescaler • TIM_ICPSC_DIV2: capture is done once every 2 events • TIM_ICPSC_DIV4: capture is done once every 4 events • TIM_ICPSC_DIV8: capture is done once every 8 events

Return values

None	

5.133.2.84 TIM_TimeBaseInit()

```
void TIM_TimeBaseInit (
            TIM_TypeDef * TIMx,
            TIM_TimeBaseInitTypeDef * TIM_TimeBaseInitStruct )
```

Initializes the TIMx Time Base Unit peripheral according to the specified parameters in the TIM_TimeBaseInitStruct.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
TIM_TimeBaseInitStruct	pointer to a TIM_TimeBaseInitTypeDef structure that contains the configuration information for the specified TIM peripheral.

Return values

None	

5.133.2.85 TIM_TimeBaseStructInit()

```
void TIM_TimeBaseStructInit (
            TIM_TimeBaseInitTypeDef * TIM_TimeBaseInitStruct )
```

Fills each TIM_TimeBaseInitStruct member with its default value.

Parameters

TIM_TimeBaseInitStruct	: pointer to a TIM_TimeBaseInitTypeDef structure which will be initialized.

Return values

None	

5.133.2.86 TIM_TIxExternalClockConfig()

```
void TIM_TIxExternalClockConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_TIxExternalCLKSource,
            uint16_t TIM_ICPolarity,
            uint16_t ICFilter )
```

Configures the TIMx Trigger as External Clock.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9, 10, 11, 12, 13 or 14 to select the TIM peripheral.
TIM_TIxExternalCLKSource	Trigger source. This parameter can be one of the following values: • TIM_TIxExternalCLK1Source_TI1ED: TI1 Edge Detector • TIM_TIxExternalCLK1Source_TI1: Filtered Timer Input 1 • TIM_TIxExternalCLK1Source_TI2: Filtered Timer Input 2
TIM_ICPolarity	specifies the TIx Polarity. This parameter can be one of the following values: • TIM_ICPolarity_Rising • TIM_ICPolarity_Falling
ICFilter	specifies the filter value. This parameter must be a value between 0x0 and 0xF.

Return values

None	

5.133.2.87 TIM_UpdateDisableConfig()

```
void TIM_UpdateDisableConfig (
             TIM_TypeDef * TIMx,
             FunctionalState NewState )
```

Enables or Disables the TIMx Update event.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
NewState	new state of the TIMx UDIS bit This parameter can be: ENABLE or DISABLE.

Return values

None	

5.133.2.88 TIM_UpdateRequestConfig()

```
void TIM_UpdateRequestConfig (
             TIM_TypeDef * TIMx,
             uint16_t TIM_UpdateSource )
```

Configures the TIMx Update Request Interrupt source.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
TIM_UpdateSource	specifies the Update source. This parameter can be one of the following values: • TIM_UpdateSource_Global: Source of update is the counter overflow/underflow or the setting of UG bit, or an update generation through the slave mode controller. • TIM_UpdateSource_Regular: Source of update is counter overflow/underflow.

Return values

None	

5.134 TIM_Private_Functions

Modules

- TimeBase management functions

  TimeBase management functions.

- Output Compare management functions

  Output Compare management functions.
- Input Capture management functions

  Input Capture management functions.
- Advanced-control timers (TIM1 and TIM8) specific features

  Advanced-control timers (TIM1 and TIM8) specific features.
- Interrupts DMA and flags management functions

  Interrupts, DMA and flags management functions.
- Clocks management functions

  Clocks management functions.
- Synchronization management functions

  Synchronization management functions.
- Specific interface management functions

  Specific interface management functions.
- Specific remapping management function

  Specific remapping management function.

5.134.1 Detailed Description

5.135 TimeBase management functions

TimeBase management functions.

Functions

- void TIM_DeInit (TIM_TypeDef ∗TIMx)

  Deinitializes the TIMx peripheral registers to their default reset values.
- void TIM_TimeBaseInit (TIM_TypeDef ∗TIMx, TIM_TimeBaseInitTypeDef ∗TIM_TimeBaseInitStruct)

  Initializes the TIMx Time Base Unit peripheral according to the specified parameters in the TIM_TimeBaseInitStruct.
- void TIM_TimeBaseStructInit (TIM_TimeBaseInitTypeDef ∗TIM_TimeBaseInitStruct)

  Fills each TIM_TimeBaseInitStruct member with its default value.
- void TIM_PrescalerConfig (TIM_TypeDef ∗TIMx, uint16_t Prescaler, uint16_t TIM_PSCReloadMode)

  Configures the TIMx Prescaler.
- void TIM_CounterModeConfig (TIM_TypeDef ∗TIMx, uint16_t TIM_CounterMode)

  Specifies the TIMx Counter Mode to be used.
- void TIM_SetCounter (TIM_TypeDef ∗TIMx, uint32_t Counter)

  Sets the TIMx Counter Register value.
- void TIM_SetAutoreload (TIM_TypeDef ∗TIMx, uint32_t Autoreload)

  Sets the TIMx Autoreload Register value.
- uint32_t TIM_GetCounter (TIM_TypeDef ∗TIMx)

  Gets the TIMx Counter value.
- uint16_t TIM_GetPrescaler (TIM_TypeDef ∗TIMx)

  Gets the TIMx Prescaler value.
- void TIM_UpdateDisableConfig (TIM_TypeDef ∗TIMx, FunctionalState NewState)

  Enables or Disables the TIMx Update event.
- void TIM_UpdateRequestConfig (TIM_TypeDef ∗TIMx, uint16_t TIM_UpdateSource)

  Configures the TIMx Update Request Interrupt source.
- void TIM_ARRPreloadConfig (TIM_TypeDef ∗TIMx, FunctionalState NewState)

Enables or disables TIMx peripheral Preload register on ARR.

- void TIM_SelectOnePulseMode (TIM_TypeDef *TIMx, uint16_t TIM_OPMode)

  Selects the TIMx's One Pulse Mode.
- void TIM_SetClockDivision (TIM_TypeDef *TIMx, uint16_t TIM_CKD)

  Sets the TIMx Clock Division value.
- void TIM_Cmd (TIM_TypeDef *TIMx, FunctionalState NewState)

  Enables or disables the specified TIM peripheral.

5.135.1 Detailed Description

TimeBase management functions.

```
 ===================================================================================
                        TimeBase management functions
 ===================================================================================

       ===================================================================
              TIM Driver: how to use it in Timing(Time base) Mode
       ===================================================================
       To use the Timer in Timing(Time base) mode, the following steps are mandatory:

       1. Enable TIM clock using RCC_APBxPeriphClockCmd(RCC_APBxPeriph_TIMx, ENABLE) function

       2. Fill the TIM_TimeBaseInitStruct with the desired parameters.

       3. Call TIM_TimeBaseInit(TIMx, &TIM_TimeBaseInitStruct) to configure the Time Base unit
          with the corresponding configuration

       4. Enable the NVIC if you need to generate the update interrupt.

       5. Enable the corresponding interrupt using the function TIM_ITConfig(TIMx, TIM_IT_Update)

       6. Call the TIM_Cmd(ENABLE) function to enable the TIM counter.

       Note1: All other functions can be used separately to modify, if needed,
          a specific feature of the Timer.
```

5.135.2 Function Documentation

5.135.2.1 TIM_ARRPreloadConfig()

```
void TIM_ARRPreloadConfig (
             TIM_TypeDef * TIMx,
             FunctionalState NewState )
```

Enables or disables TIMx peripheral Preload register on ARR.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
NewState	new state of the TIMx peripheral Preload register This parameter can be: ENABLE or DISABLE.

Return values

None	

5.135.2.2 TIM_Cmd()

```
void TIM_Cmd (
              TIM_TypeDef * TIMx,
              FunctionalState NewState )
```

Enables or disables the specified TIM peripheral.

Parameters

TIMx	where x can be 1 to 14 to select the TIMx peripheral.
NewState	new state of the TIMx peripheral. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.135.2.3 TIM_CounterModeConfig()

```
void TIM_CounterModeConfig (
              TIM_TypeDef * TIMx,
              uint16_t TIM_CounterMode )
```

Specifies the TIMx Counter Mode to be used.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_CounterMode	specifies the Counter Mode to be used This parameter can be one of the following values: • TIM_CounterMode_Up: TIM Up Counting Mode • TIM_CounterMode_Down: TIM Down Counting Mode • TIM_CounterMode_CenterAligned1: TIM Center Aligned Mode1 • TIM_CounterMode_CenterAligned2: TIM Center Aligned Mode2 • TIM_CounterMode_CenterAligned3: TIM Center Aligned Mode3

Return values

None	

5.135.2.4 TIM_DeInit()

```
void TIM_DeInit (
            TIM_TypeDef * TIMx )
```

Deinitializes the TIMx peripheral registers to their default reset values.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.

Return values

None	

5.135.2.5 TIM_GetCounter()

```
uint32_t TIM_GetCounter (
            TIM_TypeDef * TIMx )
```

Gets the TIMx Counter value.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.

Return values

Counter	Register value

5.135.2.6 TIM_GetPrescaler()

```
uint16_t TIM_GetPrescaler (
            TIM_TypeDef * TIMx )
```

Gets the TIMx Prescaler value.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.

Return values

Prescaler	Register value.

5.135.2.7 TIM_PrescalerConfig()

```
void TIM_PrescalerConfig (
             TIM_TypeDef * TIMx,
             uint16_t Prescaler,
             uint16_t TIM_PSCReloadMode )
```

Configures the TIMx Prescaler.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
Prescaler	specifies the Prescaler Register value
TIM_PSCReloadMode	specifies the TIM Prescaler Reload mode This parameter can be one of the following values: • TIM_PSCReloadMode_Update: The Prescaler is loaded at the update event. • TIM_PSCReloadMode_Immediate: The Prescaler is loaded immediatly.

Return values

None	

5.135.2.8 TIM_SelectOnePulseMode()

```
void TIM_SelectOnePulseMode (
             TIM_TypeDef * TIMx,
             uint16_t TIM_OPMode )
```

Selects the TIMx's One Pulse Mode.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
TIM_OPMode	specifies the OPM Mode to be used. This parameter can be one of the following values: • TIM_OPMode_Single • TIM_OPMode_Repetitive

Return values

None	

5.135.2.9 TIM_SetAutoreload()

```
void TIM_SetAutoreload (
            TIM_TypeDef * TIMx,
            uint32_t Autoreload )
```

Sets the TIMx Autoreload Register value.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
Autoreload	specifies the Autoreload register new value.

Return values

None	

5.135.2.10 TIM_SetClockDivision()

```
void TIM_SetClockDivision (
            TIM_TypeDef * TIMx,
            uint16_t TIM_CKD )
```

Sets the TIMx Clock Division value.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
TIM_CKD	specifies the clock division value. This parameter can be one of the following value: • TIM_CKD_DIV1: TDTS = Tck_tim • TIM_CKD_DIV2: TDTS = 2∗Tck_tim • TIM_CKD_DIV4: TDTS = 4∗Tck_tim

Return values

None	

5.135.2.11 TIM_SetCounter()

```
void TIM_SetCounter (
             TIM_TypeDef * TIMx,
             uint32_t Counter )
```

Sets the TIMx Counter Register value.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
Counter	specifies the Counter register new value.

Return values

None	

5.135.2.12 TIM_TimeBaseInit()

```
void TIM_TimeBaseInit (
             TIM_TypeDef * TIMx,
             TIM_TimeBaseInitTypeDef * TIM_TimeBaseInitStruct )
```

Initializes the TIMx Time Base Unit peripheral according to the specified parameters in the TIM_TimeBaseInitStruct.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
TIM_TimeBaseInitStruct	pointer to a TIM_TimeBaseInitTypeDef structure that contains the configuration information for the specified TIM peripheral.

Return values

None	

5.135.2.13 TIM_TimeBaseStructInit()

```
void TIM_TimeBaseStructInit (
             TIM_TimeBaseInitTypeDef * TIM_TimeBaseInitStruct )
```

Fills each TIM_TimeBaseInitStruct member with its default value.

Parameters

TIM_TimeBaseInitStruct	: pointer to a TIM_TimeBaseInitTypeDef structure which will be initialized.

Return values

None	

5.135.2.14 TIM_UpdateDisableConfig()

```
void TIM_UpdateDisableConfig (
            TIM_TypeDef * TIMx,
            FunctionalState NewState )
```

Enables or Disables the TIMx Update event.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
NewState	new state of the TIMx UDIS bit This parameter can be: ENABLE or DISABLE.

Return values

None	

5.135.2.15 TIM_UpdateRequestConfig()

```
void TIM_UpdateRequestConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_UpdateSource )
```

Configures the TIMx Update Request Interrupt source.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
TIM_UpdateSource	specifies the Update source. This parameter can be one of the following values: • TIM_UpdateSource_Global: Source of update is the counter overflow/underflow or the setting of UG bit, or an update generation through the slave mode controller. • TIM_UpdateSource_Regular: Source of update is counter overflow/underflow.

Return values

None	

5.136 Output Compare management functions

Output Compare management functions.

Functions

- void TIM_OC1Init (TIM_TypeDef ∗TIMx, TIM_OCInitTypeDef ∗TIM_OCInitStruct)

  Initializes the TIMx Channel1 according to the specified parameters in the TIM_OCInitStruct.
- void TIM_OC2Init (TIM_TypeDef ∗TIMx, TIM_OCInitTypeDef ∗TIM_OCInitStruct)

  Initializes the TIMx Channel2 according to the specified parameters in the TIM_OCInitStruct.
- void TIM_OC3Init (TIM_TypeDef ∗TIMx, TIM_OCInitTypeDef ∗TIM_OCInitStruct)

  Initializes the TIMx Channel3 according to the specified parameters in the TIM_OCInitStruct.
- void TIM_OC4Init (TIM_TypeDef ∗TIMx, TIM_OCInitTypeDef ∗TIM_OCInitStruct)

  Initializes the TIMx Channel4 according to the specified parameters in the TIM_OCInitStruct.
- void TIM_OCStructInit (TIM_OCInitTypeDef ∗TIM_OCInitStruct)

  Fills each TIM_OCInitStruct member with its default value.
- void TIM_SelectOCxM (TIM_TypeDef ∗TIMx, uint16_t TIM_Channel, uint16_t TIM_OCMode)

  Selects the TIM Output Compare Mode.
- void TIM_SetCompare1 (TIM_TypeDef ∗TIMx, uint32_t Compare1)

  Sets the TIMx Capture Compare1 Register value.
- void TIM_SetCompare2 (TIM_TypeDef ∗TIMx, uint32_t Compare2)

  Sets the TIMx Capture Compare2 Register value.
- void TIM_SetCompare3 (TIM_TypeDef ∗TIMx, uint32_t Compare3)

  Sets the TIMx Capture Compare3 Register value.
- void TIM_SetCompare4 (TIM_TypeDef ∗TIMx, uint32_t Compare4)

  Sets the TIMx Capture Compare4 Register value.
- void TIM_ForcedOC1Config (TIM_TypeDef ∗TIMx, uint16_t TIM_ForcedAction)

  Forces the TIMx output 1 waveform to active or inactive level.
- void TIM_ForcedOC2Config (TIM_TypeDef ∗TIMx, uint16_t TIM_ForcedAction)

  Forces the TIMx output 2 waveform to active or inactive level.
- void TIM_ForcedOC3Config (TIM_TypeDef ∗TIMx, uint16_t TIM_ForcedAction)

  Forces the TIMx output 3 waveform to active or inactive level.
- void TIM_ForcedOC4Config (TIM_TypeDef ∗TIMx, uint16_t TIM_ForcedAction)

  Forces the TIMx output 4 waveform to active or inactive level.
- void TIM_OC1PreloadConfig (TIM_TypeDef ∗TIMx, uint16_t TIM_OCPreload)

  Enables or disables the TIMx peripheral Preload register on CCR1.
- void TIM_OC2PreloadConfig (TIM_TypeDef ∗TIMx, uint16_t TIM_OCPreload)

  Enables or disables the TIMx peripheral Preload register on CCR2.
- void TIM_OC3PreloadConfig (TIM_TypeDef ∗TIMx, uint16_t TIM_OCPreload)

  Enables or disables the TIMx peripheral Preload register on CCR3.
- void TIM_OC4PreloadConfig (TIM_TypeDef ∗TIMx, uint16_t TIM_OCPreload)

  Enables or disables the TIMx peripheral Preload register on CCR4.
- void TIM_OC1FastConfig (TIM_TypeDef ∗TIMx, uint16_t TIM_OCFast)

  Configures the TIMx Output Compare 1 Fast feature.
- void TIM_OC2FastConfig (TIM_TypeDef ∗TIMx, uint16_t TIM_OCFast)

  Configures the TIMx Output Compare 2 Fast feature.
- void TIM_OC3FastConfig (TIM_TypeDef ∗TIMx, uint16_t TIM_OCFast)

  Configures the TIMx Output Compare 3 Fast feature.
- void TIM_OC4FastConfig (TIM_TypeDef ∗TIMx, uint16_t TIM_OCFast)

Configures the TIMx Output Compare 4 Fast feature.

- void TIM_ClearOC1Ref (TIM_TypeDef *TIMx, uint16_t TIM_OCClear)

  Clears or safeguards the OCREF1 signal on an external event.
- void TIM_ClearOC2Ref (TIM_TypeDef *TIMx, uint16_t TIM_OCClear)

  Clears or safeguards the OCREF2 signal on an external event.
- void TIM_ClearOC3Ref (TIM_TypeDef *TIMx, uint16_t TIM_OCClear)

  Clears or safeguards the OCREF3 signal on an external event.
- void TIM_ClearOC4Ref (TIM_TypeDef *TIMx, uint16_t TIM_OCClear)

  Clears or safeguards the OCREF4 signal on an external event.
- void TIM_OC1PolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPolarity)

  Configures the TIMx channel 1 polarity.
- void TIM_OC1NPolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCNPolarity)

  Configures the TIMx Channel 1N polarity.
- void TIM_OC2PolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPolarity)

  Configures the TIMx channel 2 polarity.
- void TIM_OC2NPolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCNPolarity)

  Configures the TIMx Channel 2N polarity.
- void TIM_OC3PolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPolarity)

  Configures the TIMx channel 3 polarity.
- void TIM_OC3NPolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCNPolarity)

  Configures the TIMx Channel 3N polarity.
- void TIM_OC4PolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPolarity)

  Configures the TIMx channel 4 polarity.
- void TIM_CCxCmd (TIM_TypeDef *TIMx, uint16_t TIM_Channel, uint16_t TIM_CCx)

  Enables or disables the TIM Capture Compare Channel x.
- void TIM_CCxNCmd (TIM_TypeDef *TIMx, uint16_t TIM_Channel, uint16_t TIM_CCxN)

  Enables or disables the TIM Capture Compare Channel xN.

5.136.1 Detailed Description

Output Compare management functions.

```
 ===============================================================================
                        Output Compare management functions
 ===============================================================================

       ===================================================================
              TIM Driver: how to use it in Output Compare Mode
       ===================================================================
       To use the Timer in Output Compare mode, the following steps are mandatory:

       1. Enable TIM clock using RCC_APBxPeriphClockCmd(RCC_APBxPeriph_TIMx, ENABLE) function

       2. Configure the TIM pins by configuring the corresponding GPIO pins

       2. Configure the Time base unit as described in the first part of this driver,
          if needed, else the Timer will run with the default configuration:
          - Autoreload value = 0xFFFF
          - Prescaler value = 0x0000
          - Counter mode = Up counting
          - Clock Division = TIM_CKD_DIV1

       3. Fill the TIM_OCInitStruct with the desired parameters including:
          - The TIM Output Compare mode: TIM_OCMode
          - TIM Output State: TIM_OutputState
          - TIM Pulse value: TIM_Pulse
          - TIM Output Compare Polarity : TIM_OCPolarity
```

```
4. Call TIM_OCxInit(TIMx, &TIM_OCInitStruct) to configure the desired channel with the
   corresponding configuration

5. Call the TIM_Cmd(ENABLE) function to enable the TIM counter.

Note1: All other functions can be used separately to modify, if needed,
       a specific feature of the Timer.

Note2: In case of PWM mode, this function is mandatory:
       TIM_OCxPreloadConfig(TIMx, TIM_OCPreload_ENABLE);

Note3: If the corresponding interrupt or DMA request are needed, the user should:
         1. Enable the NVIC (or the DMA) to use the TIM interrupts (or DMA requests).
         2. Enable the corresponding interrupt (or DMA request) using the function
            TIM_ITConfig(TIMx, TIM_IT_CCx) (or TIM_DMA_Cmd(TIMx, TIM_DMA_CCx))
```

5.136.2 Function Documentation

5.136.2.1 TIM_CCxCmd()

```
void TIM_CCxCmd (
             TIM_TypeDef * TIMx,
             uint16_t TIM_Channel,
             uint16_t TIM_CCx )
```

Enables or disables the TIM Capture Compare Channel x.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
TIM_Channel	specifies the TIM Channel This parameter can be one of the following values: • TIM_Channel_1: TIM Channel 1 • TIM_Channel_2: TIM Channel 2 • TIM_Channel_3: TIM Channel 3 • TIM_Channel_4: TIM Channel 4
TIM_CCx	specifies the TIM Channel CCxE bit new state. This parameter can be: TIM_CCx_Enable or TIM_CCx_Disable.

Return values

None	

5.136.2.2 TIM_CCxNCmd()

```
void TIM_CCxNCmd (
             TIM_TypeDef * TIMx,
```

```
            uint16_t TIM_Channel,
            uint16_t TIM_CCxN )
```

Enables or disables the TIM Capture Compare Channel xN.

Parameters

TIMx	where x can be 1 or 8 to select the TIM peripheral.
TIM_Channel	specifies the TIM Channel This parameter can be one of the following values: • TIM_Channel_1: TIM Channel 1 • TIM_Channel_2: TIM Channel 2 • TIM_Channel_3: TIM Channel 3
TIM_CCxN	specifies the TIM Channel CCxNE bit new state. This parameter can be: TIM_CCxN_Enable or TIM_CCxN_Disable.

Return values

None	

5.136.2.3 TIM_ClearOC1Ref()

```
void TIM_ClearOC1Ref (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCClear )
```

Clears or safeguards the OCREF1 signal on an external event.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
TIM_OCClear	new state of the Output Compare Clear Enable Bit. This parameter can be one of the following values: • TIM_OCClear_Enable: TIM Output clear enable • TIM_OCClear_Disable: TIM Output clear disable

Return values

None	

5.136.2.4 TIM_ClearOC2Ref()

```
void TIM_ClearOC2Ref (
```

```
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCClear )
```

Clears or safeguards the OCREF2 signal on an external event.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
TIM_OCClear	new state of the Output Compare Clear Enable Bit. This parameter can be one of the following values: • TIM_OCClear_Enable: TIM Output clear enable • TIM_OCClear_Disable: TIM Output clear disable

Return values

None	

5.136.2.5 TIM_ClearOC3Ref()

```
void TIM_ClearOC3Ref (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCClear )
```

Clears or safeguards the OCREF3 signal on an external event.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_OCClear	new state of the Output Compare Clear Enable Bit. This parameter can be one of the following values: • TIM_OCClear_Enable: TIM Output clear enable • TIM_OCClear_Disable: TIM Output clear disable

Return values

None	

5.136.2.6 TIM_ClearOC4Ref()

```
void TIM_ClearOC4Ref (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCClear )
```

Clears or safeguards the OCREF4 signal on an external event.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_OCClear	new state of the Output Compare Clear Enable Bit. This parameter can be one of the following values: • TIM_OCClear_Enable: TIM Output clear enable • TIM_OCClear_Disable: TIM Output clear disable

Return values

None	

5.136.2.7 TIM_ForcedOC1Config()

```
void TIM_ForcedOC1Config (
             TIM_TypeDef * TIMx,
             uint16_t TIM_ForcedAction )
```

Forces the TIMx output 1 waveform to active or inactive level.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
TIM_ForcedAction	specifies the forced Action to be set to the output waveform. This parameter can be one of the following values: • TIM_ForcedAction_Active: Force active level on OC1REF • TIM_ForcedAction_InActive: Force inactive level on OC1REF.

Return values

None	

5.136.2.8 TIM_ForcedOC2Config()

```
void TIM_ForcedOC2Config (
             TIM_TypeDef * TIMx,
             uint16_t TIM_ForcedAction )
```

Forces the TIMx output 2 waveform to active or inactive level.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
TIM_ForcedAction	specifies the forced Action to be set to the output waveform. This parameter can be one of the following values: • TIM_ForcedAction_Active: Force active level on OC2REF • TIM_ForcedAction_InActive: Force inactive level on OC2REF.

Return values

None	

5.136.2.9 TIM_ForcedOC3Config()

```
void TIM_ForcedOC3Config (
             TIM_TypeDef * TIMx,
             uint16_t TIM_ForcedAction )
```

Forces the TIMx output 3 waveform to active or inactive level.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_ForcedAction	specifies the forced Action to be set to the output waveform. This parameter can be one of the following values: • TIM_ForcedAction_Active: Force active level on OC3REF • TIM_ForcedAction_InActive: Force inactive level on OC3REF.

Return values

None	

5.136.2.10 TIM_ForcedOC4Config()

```
void TIM_ForcedOC4Config (
             TIM_TypeDef * TIMx,
             uint16_t TIM_ForcedAction )
```

Forces the TIMx output 4 waveform to active or inactive level.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_ForcedAction	specifies the forced Action to be set to the output waveform. This parameter can be one of the following values: • TIM_ForcedAction_Active: Force active level on OC4REF • TIM_ForcedAction_InActive: Force inactive level on OC4REF.

Return values

None	

5.136.2.11 TIM_OC1FastConfig()

```
void TIM_OC1FastConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCFast )
```

Configures the TIMx Output Compare 1 Fast feature.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
TIM_OCFast	new state of the Output Compare Fast Enable Bit. This parameter can be one of the following values: • TIM_OCFast_Enable: TIM output compare fast enable • TIM_OCFast_Disable: TIM output compare fast disable

Return values

None	

5.136.2.12 TIM_OC1Init()

```
void TIM_OC1Init (
            TIM_TypeDef * TIMx,
            TIM_OCInitTypeDef * TIM_OCInitStruct )
```

Initializes the TIMx Channel1 according to the specified parameters in the TIM_OCInitStruct.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
TIM_OCInitStruct	pointer to a TIM_OCInitTypeDef structure that contains the configuration information for the specified TIM peripheral.

Return values

None	

5.136.2.13 TIM_OC1NPolarityConfig()

```
void TIM_OC1NPolarityConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCNPolarity )
```

Configures the TIMx Channel 1N polarity.

Parameters

TIMx	where x can be 1 or 8 to select the TIM peripheral.
TIM_OCNPolarity	specifies the OC1N Polarity This parameter can be one of the following values: • TIM_OCNPolarity_High: Output Compare active high • TIM_OCNPolarity_Low: Output Compare active low

Return values

None	

5.136.2.14 TIM_OC1PolarityConfig()

```
void TIM_OC1PolarityConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCPolarity )
```

Configures the TIMx channel 1 polarity.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
TIM_OCPolarity	specifies the OC1 Polarity This parameter can be one of the following values: • TIM_OCPolarity_High: Output Compare active high • TIM_OCPolarity_Low: Output Compare active low

Return values

None	

5.136.2.15 TIM_OC1PreloadConfig()

```
void TIM_OC1PreloadConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCPreload )
```

Enables or disables the TIMx peripheral Preload register on CCR1.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
TIM_OCPreload	new state of the TIMx peripheral Preload register This parameter can be one of the following values: • TIM_OCPreload_Enable • TIM_OCPreload_Disable

Return values

None	

5.136.2.16 TIM_OC2FastConfig()

```
void TIM_OC2FastConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCFast )
```

Configures the TIMx Output Compare 2 Fast feature.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
TIM_OCFast	new state of the Output Compare Fast Enable Bit. This parameter can be one of the following values: • TIM_OCFast_Enable: TIM output compare fast enable • TIM_OCFast_Disable: TIM output compare fast disable

Return values

None	

5.136.2.17 TIM_OC2Init()

```
void TIM_OC2Init (
            TIM_TypeDef * TIMx,
            TIM_OCInitTypeDef * TIM_OCInitStruct )
```

Initializes the TIMx Channel2 according to the specified parameters in the TIM_OCInitStruct.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
TIM_OCInitStruct	pointer to a TIM_OCInitTypeDef structure that contains the configuration information for the specified TIM peripheral.

Return values

None	

5.136.2.18 TIM_OC2NPolarityConfig()

```
void TIM_OC2NPolarityConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCNPolarity )
```

Configures the TIMx Channel 2N polarity.

Parameters

TIMx	where x can be 1 or 8 to select the TIM peripheral.
TIM_OCNPolarity	specifies the OC2N Polarity This parameter can be one of the following values: • TIM_OCNPolarity_High: Output Compare active high • TIM_OCNPolarity_Low: Output Compare active low

Return values

None	

5.136.2.19 TIM_OC2PolarityConfig()

```
void TIM_OC2PolarityConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCPolarity )
```

Configures the TIMx channel 2 polarity.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
TIM_OCPolarity	specifies the OC2 Polarity This parameter can be one of the following values: • TIM_OCPolarity_High: Output Compare active high • TIM_OCPolarity_Low: Output Compare active low

Return values

None	

5.136.2.20 TIM_OC2PreloadConfig()

```
void TIM_OC2PreloadConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCPreload )
```

Enables or disables the TIMx peripheral Preload register on CCR2.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
TIM_OCPreload	new state of the TIMx peripheral Preload register This parameter can be one of the following values: • TIM_OCPreload_Enable • TIM_OCPreload_Disable

Return values

None	

5.136.2.21 TIM_OC3FastConfig()

```
void TIM_OC3FastConfig (
```

```
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCFast )
```

Configures the TIMx Output Compare 3 Fast feature.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_OCFast	new state of the Output Compare Fast Enable Bit. This parameter can be one of the following values: • TIM_OCFast_Enable: TIM output compare fast enable • TIM_OCFast_Disable: TIM output compare fast disable

Return values

None	

5.136.2.22 TIM_OC3Init()

```
void TIM_OC3Init (
            TIM_TypeDef * TIMx,
            TIM_OCInitTypeDef * TIM_OCInitStruct )
```

Initializes the TIMx Channel3 according to the specified parameters in the TIM_OCInitStruct.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_OCInitStruct	pointer to a TIM_OCInitTypeDef structure that contains the configuration information for the specified TIM peripheral.

Return values

None	

5.136.2.23 TIM_OC3NPolarityConfig()

```
void TIM_OC3NPolarityConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCNPolarity )
```

Configures the TIMx Channel 3N polarity.

Parameters

TIMx	where x can be 1 or 8 to select the TIM peripheral.
TIM_OCNPolarity	specifies the OC3N Polarity This parameter can be one of the following values: • TIM_OCNPolarity_High: Output Compare active high • TIM_OCNPolarity_Low: Output Compare active low

Return values

None	

5.136.2.24 TIM_OC3PolarityConfig()

```
void TIM_OC3PolarityConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCPolarity )
```

Configures the TIMx channel 3 polarity.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_OCPolarity	specifies the OC3 Polarity This parameter can be one of the following values: • TIM_OCPolarity_High: Output Compare active high • TIM_OCPolarity_Low: Output Compare active low

Return values

None	

5.136.2.25 TIM_OC3PreloadConfig()

```
void TIM_OC3PreloadConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCPreload )
```

Enables or disables the TIMx peripheral Preload register on CCR3.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.

Parameters

TIM_OCPreload	new state of the TIMx peripheral Preload register This parameter can be one of the following values: • TIM_OCPreload_Enable • TIM_OCPreload_Disable

Return values

None	

5.136.2.26 TIM_OC4FastConfig()

```
void TIM_OC4FastConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCFast )
```

Configures the TIMx Output Compare 4 Fast feature.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_OCFast	new state of the Output Compare Fast Enable Bit. This parameter can be one of the following values: • TIM_OCFast_Enable: TIM output compare fast enable • TIM_OCFast_Disable: TIM output compare fast disable

Return values

None	

5.136.2.27 TIM_OC4Init()

```
void TIM_OC4Init (
            TIM_TypeDef * TIMx,
            TIM_OCInitTypeDef * TIM_OCInitStruct )
```

Initializes the TIMx Channel4 according to the specified parameters in the TIM_OCInitStruct.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_OCInitStruct	pointer to a TIM_OCInitTypeDef structure that contains the configuration information for the specified TIM peripheral.

Return values

None	

5.136.2.28 TIM_OC4PolarityConfig()

```
void TIM_OC4PolarityConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCPolarity )
```

Configures the TIMx channel 4 polarity.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_OCPolarity	specifies the OC4 Polarity This parameter can be one of the following values: • TIM_OCPolarity_High: Output Compare active high • TIM_OCPolarity_Low: Output Compare active low

Return values

None	

5.136.2.29 TIM_OC4PreloadConfig()

```
void TIM_OC4PreloadConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_OCPreload )
```

Enables or disables the TIMx peripheral Preload register on CCR4.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_OCPreload	new state of the TIMx peripheral Preload register This parameter can be one of the following values: • TIM_OCPreload_Enable • TIM_OCPreload_Disable

Return values

None	

5.136.2.30 TIM_OCStructInit()

```
void TIM_OCStructInit (
            TIM_OCInitTypeDef * TIM_OCInitStruct )
```

Fills each TIM_OCInitStruct member with its default value.

Parameters

TIM_OCInitStruct	pointer to a TIM_OCInitTypeDef structure which will be initialized.

Return values

None	

5.136.2.31 TIM_SelectOCxM()

```
void TIM_SelectOCxM (
            TIM_TypeDef * TIMx,
            uint16_t TIM_Channel,
            uint16_t TIM_OCMode )
```

Selects the TIM Output Compare Mode.

Note

This function disables the selected channel before changing the Output Compare Mode. If needed, user has to enable this channel using TIM_CCxCmd() and TIM_CCxNCmd() functions.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
TIM_Channel	specifies the TIM Channel This parameter can be one of the following values: • TIM_Channel_1: TIM Channel 1 • TIM_Channel_2: TIM Channel 2 • TIM_Channel_3: TIM Channel 3 • TIM_Channel_4: TIM Channel 4

Parameters

TIM_OCMode	specifies the TIM Output Compare Mode. This parameter can be one of the following values: • TIM_OCMode_Timing • TIM_OCMode_Active • TIM_OCMode_Toggle • TIM_OCMode_PWM1 • TIM_OCMode_PWM2 • TIM_ForcedAction_Active • TIM_ForcedAction_InActive

Return values

None	

5.136.2.32 TIM_SetCompare1()

```
void TIM_SetCompare1 (
             TIM_TypeDef * TIMx,
             uint32_t Compare1 )
```

Sets the TIMx Capture Compare1 Register value.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
Compare1	specifies the Capture Compare1 register new value.

Return values

None	

5.136.2.33 TIM_SetCompare2()

```
void TIM_SetCompare2 (
             TIM_TypeDef * TIMx,
             uint32_t Compare2 )
```

Sets the TIMx Capture Compare2 Register value.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
Compare2	specifies the Capture Compare2 register new value.

Return values

None	

5.136.2.34 TIM_SetCompare3()

```
void TIM_SetCompare3 (
             TIM_TypeDef * TIMx,
             uint32_t Compare3 )
```

Sets the TIMx Capture Compare3 Register value.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
Compare3	specifies the Capture Compare3 register new value.

Return values

None	

5.136.2.35 TIM_SetCompare4()

```
void TIM_SetCompare4 (
             TIM_TypeDef * TIMx,
             uint32_t Compare4 )
```

Sets the TIMx Capture Compare4 Register value.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
Compare4	specifies the Capture Compare4 register new value.

Return values

None	

5.137 Input Capture management functions

Input Capture management functions.

Functions

- void TIM_ICInit (TIM_TypeDef ∗TIMx, TIM_ICInitTypeDef ∗TIM_ICInitStruct)

  Initializes the TIM peripheral according to the specified parameters in the TIM_ICInitStruct.
- void TIM_ICStructInit (TIM_ICInitTypeDef ∗TIM_ICInitStruct)

  Fills each TIM_ICInitStruct member with its default value.
- void TIM_PWMIConfig (TIM_TypeDef ∗TIMx, TIM_ICInitTypeDef ∗TIM_ICInitStruct)

  Configures the TIM peripheral according to the specified parameters in the TIM_ICInitStruct to measure an external PWM signal.
- uint32_t TIM_GetCapture1 (TIM_TypeDef ∗TIMx)

  Gets the TIMx Input Capture 1 value.
- uint32_t TIM_GetCapture2 (TIM_TypeDef ∗TIMx)

  Gets the TIMx Input Capture 2 value.
- uint32_t TIM_GetCapture3 (TIM_TypeDef ∗TIMx)

  Gets the TIMx Input Capture 3 value.
- uint32_t TIM_GetCapture4 (TIM_TypeDef ∗TIMx)

  Gets the TIMx Input Capture 4 value.
- void TIM_SetIC1Prescaler (TIM_TypeDef ∗TIMx, uint16_t TIM_ICPSC)

  Sets the TIMx Input Capture 1 prescaler.
- void TIM_SetIC2Prescaler (TIM_TypeDef ∗TIMx, uint16_t TIM_ICPSC)

  Sets the TIMx Input Capture 2 prescaler.
- void TIM_SetIC3Prescaler (TIM_TypeDef ∗TIMx, uint16_t TIM_ICPSC)

  Sets the TIMx Input Capture 3 prescaler.
- void TIM_SetIC4Prescaler (TIM_TypeDef ∗TIMx, uint16_t TIM_ICPSC)

  Sets the TIMx Input Capture 4 prescaler.

5.137.1 Detailed Description

Input Capture management functions.

```
 ===================================================================================
                      Input Capture management functions
 ===================================================================================

       ===================================================================
              TIM Driver: how to use it in Input Capture Mode
       ===================================================================
       To use the Timer in Input Capture mode, the following steps are mandatory:

       1. Enable TIM clock using RCC_APBxPeriphClockCmd(RCC_APBxPeriph_TIMx, ENABLE) function

       2. Configure the TIM pins by configuring the corresponding GPIO pins

       2. Configure the Time base unit as described in the first part of this driver,
          if needed, else the Timer will run with the default configuration:
          - Autoreload value = 0xFFFF
          - Prescaler value = 0x0000
          - Counter mode = Up counting
          - Clock Division = TIM_CKD_DIV1

       3. Fill the TIM_ICInitStruct with the desired parameters including:
```

```
   - TIM Channel: TIM_Channel
   - TIM Input Capture polarity: TIM_ICPolarity
   - TIM Input Capture selection: TIM_ICSelection
   - TIM Input Capture Prescaler: TIM_ICPrescaler
   - TIM Input CApture filter value: TIM_ICFilter

4. Call TIM_ICInit(TIMx, &TIM_ICInitStruct) to configure the desired channel with the
   corresponding configuration and to measure only frequency or duty cycle of the input signal,
   or,
   Call TIM_PWMIConfig(TIMx, &TIM_ICInitStruct) to configure the desired channels with the
   corresponding configuration and to measure the frequency and the duty cycle of the input signal

5. Enable the NVIC or the DMA to read the measured frequency.

6. Enable the corresponding interrupt (or DMA request) to read the Captured value,
   using the function TIM_ITConfig(TIMx, TIM_IT_CCx) (or TIM_DMA_Cmd(TIMx, TIM_DMA_CCx))

7. Call the TIM_Cmd(ENABLE) function to enable the TIM counter.

8. Use TIM_GetCapturex(TIMx); to read the captured value.

Note1: All other functions can be used separately to modify, if needed,
       a specific feature of the Timer.
```

5.137.2 Function Documentation

5.137.2.1 TIM_GetCapture1()

```
uint32_t TIM_GetCapture1 (
             TIM_TypeDef * TIMx )
```

Gets the TIMx Input Capture 1 value.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.

Return values

Capture	Compare 1 Register value.

5.137.2.2 TIM_GetCapture2()

```
uint32_t TIM_GetCapture2 (
             TIM_TypeDef * TIMx )
```

Gets the TIMx Input Capture 2 value.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.

Return values

Capture	Compare 2 Register value.

5.137.2.3 TIM_GetCapture3()

```
uint32_t TIM_GetCapture3 (
            TIM_TypeDef * TIMx )
```

Gets the TIMx Input Capture 3 value.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.

Return values

Capture	Compare 3 Register value.

5.137.2.4 TIM_GetCapture4()

```
uint32_t TIM_GetCapture4 (
            TIM_TypeDef * TIMx )
```

Gets the TIMx Input Capture 4 value.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.

Return values

Capture	Compare 4 Register value.

5.137.2.5 TIM_ICInit()

```
void TIM_ICInit (
            TIM_TypeDef * TIMx,
            TIM_ICInitTypeDef * TIM_ICInitStruct )
```

Initializes the TIM peripheral according to the specified parameters in the TIM_ICInitStruct.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
TIM_ICInitStruct	pointer to a TIM_ICInitTypeDef structure that contains the configuration information for the specified TIM peripheral.

Return values

None	

5.137.2.6 TIM_ICStructInit()

```
void TIM_ICStructInit (
             TIM_ICInitTypeDef * TIM_ICInitStruct )
```

Fills each TIM_ICInitStruct member with its default value.

Parameters

TIM_ICInitStruct	pointer to a TIM_ICInitTypeDef structure which will be initialized.

Return values

None	

5.137.2.7 TIM_PWMIConfig()

```
void TIM_PWMIConfig (
             TIM_TypeDef * TIMx,
             TIM_ICInitTypeDef * TIM_ICInitStruct )
```

Configures the TIM peripheral according to the specified parameters in the TIM_ICInitStruct to measure an external PWM signal.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5,8, 9 or 12 to select the TIM peripheral.
TIM_ICInitStruct	pointer to a TIM_ICInitTypeDef structure that contains the configuration information for the specified TIM peripheral.

Return values

None	

5.137.2.8 TIM_SetIC1Prescaler()

```
void TIM_SetIC1Prescaler (
            TIM_TypeDef * TIMx,
            uint16_t TIM_ICPSC )
```

Sets the TIMx Input Capture 1 prescaler.

Parameters

TIMx	where x can be 1 to 14 except 6 and 7, to select the TIM peripheral.
TIM_ICPSC	specifies the Input Capture1 prescaler new value. This parameter can be one of the following values: • TIM_ICPSC_DIV1: no prescaler • TIM_ICPSC_DIV2: capture is done once every 2 events • TIM_ICPSC_DIV4: capture is done once every 4 events • TIM_ICPSC_DIV8: capture is done once every 8 events

Return values

None	

5.137.2.9 TIM_SetIC2Prescaler()

```
void TIM_SetIC2Prescaler (
            TIM_TypeDef * TIMx,
            uint16_t TIM_ICPSC )
```

Sets the TIMx Input Capture 2 prescaler.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
TIM_ICPSC	specifies the Input Capture2 prescaler new value. This parameter can be one of the following values: • TIM_ICPSC_DIV1: no prescaler • TIM_ICPSC_DIV2: capture is done once every 2 events • TIM_ICPSC_DIV4: capture is done once every 4 events • TIM_ICPSC_DIV8: capture is done once every 8 events

Return values

None	

5.137.2.10 TIM_SetIC3Prescaler()

```
void TIM_SetIC3Prescaler (
            TIM_TypeDef * TIMx,
            uint16_t TIM_ICPSC )
```

Sets the TIMx Input Capture 3 prescaler.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_ICPSC	specifies the Input Capture3 prescaler new value. This parameter can be one of the following values: • TIM_ICPSC_DIV1: no prescaler • TIM_ICPSC_DIV2: capture is done once every 2 events • TIM_ICPSC_DIV4: capture is done once every 4 events • TIM_ICPSC_DIV8: capture is done once every 8 events

Return values

None	

5.137.2.11 TIM_SetIC4Prescaler()

```
void TIM_SetIC4Prescaler (
            TIM_TypeDef * TIMx,
            uint16_t TIM_ICPSC )
```

Sets the TIMx Input Capture 4 prescaler.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_ICPSC	specifies the Input Capture4 prescaler new value. This parameter can be one of the following values: • TIM_ICPSC_DIV1: no prescaler • TIM_ICPSC_DIV2: capture is done once every 2 events • TIM_ICPSC_DIV4: capture is done once every 4 events
	• TIM_ICPSC_DIV8: capture is done once every 8 events

Return values

None	

5.138 Advanced-control timers (TIM1 and TIM8) specific features

Advanced-control timers (TIM1 and TIM8) specific features.

Functions

- void TIM_BDTRConfig (TIM_TypeDef ∗TIMx, TIM_BDTRInitTypeDef ∗TIM_BDTRInitStruct)

  Configures the Break feature, dead time, Lock level, OSSI/OSSR State and the AOE(automatic output enable).
- void TIM_BDTRStructInit (TIM_BDTRInitTypeDef ∗TIM_BDTRInitStruct)

  Fills each TIM_BDTRInitStruct member with its default value.
- void TIM_CtrlPWMOutputs (TIM_TypeDef ∗TIMx, FunctionalState NewState)

  Enables or disables the TIM peripheral Main Outputs.
- void TIM_SelectCOM (TIM_TypeDef ∗TIMx, FunctionalState NewState)

  Selects the TIM peripheral Commutation event.
- void TIM_CCPreloadControl (TIM_TypeDef ∗TIMx, FunctionalState NewState)

  Sets or Resets the TIM peripheral Capture Compare Preload Control bit.

5.138.1 Detailed Description

Advanced-control timers (TIM1 and TIM8) specific features.

```
 ===================================================================================
           Advanced-control timers (TIM1 and TIM8) specific features
 ===================================================================================

        ===================================================================
               TIM Driver: how to use the Break feature
        ===================================================================
        After configuring the Timer channel(s) in the appropriate Output Compare mode:

        1. Fill the TIM_BDTRInitStruct with the desired parameters for the Timer
           Break Polarity, dead time, Lock level, the OSSI/OSSR State and the
           AOE(automatic output enable).

        2. Call TIM_BDTRConfig(TIMx, &TIM_BDTRInitStruct) to configure the Timer

        3. Enable the Main Output using TIM_CtrlPWMOutputs(TIM1, ENABLE)

        4. Once the break even occurs, the Timer's output signals are put in reset
           state or in a known state (according to the configuration made in
           TIM_BDTRConfig() function).
```

5.138.2 Function Documentation

5.138.2.1 TIM_BDTRConfig()

```
void TIM_BDTRConfig (
            TIM_TypeDef * TIMx,
            TIM_BDTRInitTypeDef * TIM_BDTRInitStruct )
```

Configures the Break feature, dead time, Lock level, OSSI/OSSR State and the AOE(automatic output enable).

Parameters

TIMx	where x can be 1 or 8 to select the TIM
TIM_BDTRInitStruct	pointer to a TIM_BDTRInitTypeDef structure that contains the BDTR Register configuration information for the TIM peripheral.

Return values

None	

5.138.2.2 TIM_BDTRStructInit()

```
void TIM_BDTRStructInit (
             TIM_BDTRInitTypeDef * TIM_BDTRInitStruct )
```

Fills each TIM_BDTRInitStruct member with its default value.

Parameters

TIM_BDTRInitStruct	pointer to a TIM_BDTRInitTypeDef structure which will be initialized.

Return values

None	

5.138.2.3 TIM_CCPreloadControl()

```
void TIM_CCPreloadControl (
             TIM_TypeDef * TIMx,
             FunctionalState NewState )
```

Sets or Resets the TIM peripheral Capture Compare Preload Control bit.

Parameters

TIMx	where x can be 1 or 8 to select the TIMx peripheral
NewState	new state of the Capture Compare Preload Control bit This parameter can be: ENABLE or DISABLE.

Return values

None	

5.138.2.4 TIM_CtrlPWMOutputs()

```
void TIM_CtrlPWMOutputs (
             TIM_TypeDef * TIMx,
             FunctionalState NewState )
```

Enables or disables the TIM peripheral Main Outputs.

Parameters

TIMx	where x can be 1 or 8 to select the TIMx peripheral.
NewState	new state of the TIM peripheral Main Outputs. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.138.2.5 TIM_SelectCOM()

```
void TIM_SelectCOM (
             TIM_TypeDef * TIMx,
             FunctionalState NewState )
```

Selects the TIM peripheral Commutation event.

Parameters

TIMx	where x can be 1 or 8 to select the TIMx peripheral
NewState	new state of the Commutation event. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.139 Interrupts DMA and flags management functions

Interrupts, DMA and flags management functions.

Functions

- void TIM_ITConfig (TIM_TypeDef *TIMx, uint16_t TIM_IT, FunctionalState NewState)

  Enables or disables the specified TIM interrupts.
- void TIM_GenerateEvent (TIM_TypeDef *TIMx, uint16_t TIM_EventSource)

  Configures the TIMx event to be generate by software.

- FlagStatus TIM_GetFlagStatus (TIM_TypeDef ∗TIMx, uint16_t TIM_FLAG)

  Checks whether the specified TIM flag is set or not.
- void TIM_ClearFlag (TIM_TypeDef ∗TIMx, uint16_t TIM_FLAG)

  Clears the TIMx's pending flags.
- ITStatus TIM_GetITStatus (TIM_TypeDef ∗TIMx, uint16_t TIM_IT)

  Checks whether the TIM interrupt has occurred or not.
- void TIM_ClearITPendingBit (TIM_TypeDef ∗TIMx, uint16_t TIM_IT)

  Clears the TIMx's interrupt pending bits.
- void TIM_DMAConfig (TIM_TypeDef ∗TIMx, uint16_t TIM_DMABase, uint16_t TIM_DMABurstLength)

  Configures the TIMx's DMA interface.
- void TIM_DMACmd (TIM_TypeDef ∗TIMx, uint16_t TIM_DMASource, FunctionalState NewState)

  Enables or disables the TIMx's DMA Requests.
- void TIM_SelectCCDMA (TIM_TypeDef ∗TIMx, FunctionalState NewState)

  Selects the TIMx peripheral Capture Compare DMA source.

5.139.1 Detailed Description

Interrupts, DMA and flags management functions.

```
 ===============================================================================
                 Interrupts, DMA and flags management functions
 ===============================================================================
```

5.139.2 Function Documentation

5.139.2.1 TIM_ClearFlag()

```
void TIM_ClearFlag (
             TIM_TypeDef * TIMx,
             uint16_t TIM_FLAG )
```

Clears the TIMx's pending flags.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.

Parameters

TIM_FLAG	specifies the flag bit to clear. This parameter can be any combination of the following values: • TIM_FLAG_Update: TIM update Flag • TIM_FLAG_CC1: TIM Capture Compare 1 Flag • TIM_FLAG_CC2: TIM Capture Compare 2 Flag • TIM_FLAG_CC3: TIM Capture Compare 3 Flag • TIM_FLAG_CC4: TIM Capture Compare 4 Flag • TIM_FLAG_COM: TIM Commutation Flag • TIM_FLAG_Trigger: TIM Trigger Flag • TIM_FLAG_Break: TIM Break Flag • TIM_FLAG_CC1OF: TIM Capture Compare 1 over capture Flag • TIM_FLAG_CC2OF: TIM Capture Compare 2 over capture Flag • TIM_FLAG_CC3OF: TIM Capture Compare 3 over capture Flag • TIM_FLAG_CC4OF: TIM Capture Compare 4 over capture Flag

Note

TIM6 and TIM7 can have only one update flag.

TIM_FLAG_COM and TIM_FLAG_Break are used only with TIM1 and TIM8.

Return values

None	

5.139.2.2 TIM_ClearITPendingBit()

```
void TIM_ClearITPendingBit (
             TIM_TypeDef * TIMx,
             uint16_t TIM_IT )
```

Clears the TIMx's interrupt pending bits.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.

Parameters

TIM_IT	specifies the pending bit to clear. This parameter can be any combination of the following values: • TIM_IT_Update: TIM1 update Interrupt source • TIM_IT_CC1: TIM Capture Compare 1 Interrupt source • TIM_IT_CC2: TIM Capture Compare 2 Interrupt source • TIM_IT_CC3: TIM Capture Compare 3 Interrupt source • TIM_IT_CC4: TIM Capture Compare 4 Interrupt source • TIM_IT_COM: TIM Commutation Interrupt source • TIM_IT_Trigger: TIM Trigger Interrupt source • TIM_IT_Break: TIM Break Interrupt source

Note

TIM6 and TIM7 can generate only an update interrupt.

TIM_IT_COM and TIM_IT_Break are used only with TIM1 and TIM8.

Return values

None	

5.139.2.3 TIM_DMACmd()

```
void TIM_DMACmd (
              TIM_TypeDef * TIMx,
              uint16_t TIM_DMASource,
              FunctionalState NewState )
```

Enables or disables the TIMx's DMA Requests.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 6, 7 or 8 to select the TIM peripheral.
TIM_DMASource	specifies the DMA Request sources. This parameter can be any combination of the following values: • TIM_DMA_Update: TIM update Interrupt source • TIM_DMA_CC1: TIM Capture Compare 1 DMA source • TIM_DMA_CC2: TIM Capture Compare 2 DMA source • TIM_DMA_CC3: TIM Capture Compare 3 DMA source • TIM_DMA_CC4: TIM Capture Compare 4 DMA source • TIM_DMA_COM: TIM Commutation DMA source • TIM_DMA_Trigger: TIM Trigger DMA source

Parameters

NewState	new state of the DMA Request sources. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.139.2.4 TIM_DMAConfig()

```
void TIM_DMAConfig (
             TIM_TypeDef * TIMx,
             uint16_t TIM_DMABase,
             uint16_t TIM_DMABurstLength )
```

Configures the TIMx's DMA interface.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.

Parameters

TIM_DMABase	DMA Base address. This parameter can be one of the following values: • TIM_DMABase_CR1 • TIM_DMABase_CR2 • TIM_DMABase_SMCR • TIM_DMABase_DIER • TIM1_DMABase_SR • TIM_DMABase_EGR • TIM_DMABase_CCMR1 • TIM_DMABase_CCMR2 • TIM_DMABase_CCER • TIM_DMABase_CNT • TIM_DMABase_PSC • TIM_DMABase_ARR • TIM_DMABase_RCR • TIM_DMABase_CCR1 • TIM_DMABase_CCR2 • TIM_DMABase_CCR3 • TIM_DMABase_CCR4 • TIM_DMABase_BDTR • TIM_DMABase_DCR
TIM_DMABurstLength	DMA Burst length. This parameter can be one value between: TIM_DMABurstLength_1Transfer and TIM_DMABurstLength_18Transfers.

Return values

None	

5.139.2.5 TIM_GenerateEvent()

```
void TIM_GenerateEvent (
            TIM_TypeDef * TIMx,
            uint16_t TIM_EventSource )
```

Configures the TIMx event to be generate by software.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.
TIM_EventSource	specifies the event source. This parameter can be one or more of the following values: • TIM_EventSource_Update: Timer update Event source • TIM_EventSource_CC1: Timer Capture Compare 1 Event source • TIM_EventSource_CC2: Timer Capture Compare 2 Event source • TIM_EventSource_CC3: Timer Capture Compare 3 Event source • TIM_EventSource_CC4: Timer Capture Compare 4 Event source • TIM_EventSource_COM: Timer COM event source • TIM_EventSource_Trigger: Timer Trigger Event source • TIM_EventSource_Break: Timer Break event source

Note

TIM6 and TIM7 can only generate an update event.

TIM_EventSource_COM and TIM_EventSource_Break are used only with TIM1 and TIM8.

Return values

None	

5.139.2.6 TIM_GetFlagStatus()

```
FlagStatus TIM_GetFlagStatus (
             TIM_TypeDef * TIMx,
             uint16_t TIM_FLAG )
```

Checks whether the specified TIM flag is set or not.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.

Parameters

TIM_FLAG	specifies the flag to check. This parameter can be one of the following values: • TIM_FLAG_Update: TIM update Flag • TIM_FLAG_CC1: TIM Capture Compare 1 Flag • TIM_FLAG_CC2: TIM Capture Compare 2 Flag • TIM_FLAG_CC3: TIM Capture Compare 3 Flag • TIM_FLAG_CC4: TIM Capture Compare 4 Flag • TIM_FLAG_COM: TIM Commutation Flag • TIM_FLAG_Trigger: TIM Trigger Flag • TIM_FLAG_Break: TIM Break Flag • TIM_FLAG_CC1OF: TIM Capture Compare 1 over capture Flag • TIM_FLAG_CC2OF: TIM Capture Compare 2 over capture Flag • TIM_FLAG_CC3OF: TIM Capture Compare 3 over capture Flag • TIM_FLAG_CC4OF: TIM Capture Compare 4 over capture Flag

Note

TIM6 and TIM7 can have only one update flag.

TIM_FLAG_COM and TIM_FLAG_Break are used only with TIM1 and TIM8.

Return values

The	new state of TIM_FLAG (SET or RESET).

5.139.2.7 TIM_GetITStatus()

```
ITStatus TIM_GetITStatus (
            TIM_TypeDef * TIMx,
            uint16_t TIM_IT )
```

Checks whether the TIM interrupt has occurred or not.

Parameters

TIMx	where x can be 1 to 14 to select the TIM peripheral.

Parameters

TIM_IT	specifies the TIM interrupt source to check. This parameter can be one of the following values: • TIM_IT_Update: TIM update Interrupt source • TIM_IT_CC1: TIM Capture Compare 1 Interrupt source • TIM_IT_CC2: TIM Capture Compare 2 Interrupt source • TIM_IT_CC3: TIM Capture Compare 3 Interrupt source • TIM_IT_CC4: TIM Capture Compare 4 Interrupt source • TIM_IT_COM: TIM Commutation Interrupt source • TIM_IT_Trigger: TIM Trigger Interrupt source • TIM_IT_Break: TIM Break Interrupt source

Note

TIM6 and TIM7 can generate only an update interrupt.

TIM_IT_COM and TIM_IT_Break are used only with TIM1 and TIM8.

Return values

The	new state of the TIM_IT(SET or RESET).

5.139.2.8 TIM_ITConfig()

```
void TIM_ITConfig (
              TIM_TypeDef * TIMx,
              uint16_t TIM_IT,
              FunctionalState NewState )
```

Enables or disables the specified TIM interrupts.

Parameters

TIMx	where x can be 1 to 14 to select the TIMx peripheral.

Parameters

TIM_IT	specifies the TIM interrupts sources to be enabled or disabled. This parameter can be any combination of the following values: • TIM_IT_Update: TIM update Interrupt source • TIM_IT_CC1: TIM Capture Compare 1 Interrupt source • TIM_IT_CC2: TIM Capture Compare 2 Interrupt source • TIM_IT_CC3: TIM Capture Compare 3 Interrupt source • TIM_IT_CC4: TIM Capture Compare 4 Interrupt source • TIM_IT_COM: TIM Commutation Interrupt source • TIM_IT_Trigger: TIM Trigger Interrupt source • TIM_IT_Break: TIM Break Interrupt source

Note

For TIM6 and TIM7 only the parameter TIM_IT_Update can be used

For TIM9 and TIM12 only one of the following parameters can be used: TIM_IT_Update, TIM_IT_CC1, TIM_IT_CC2 or TIM_IT_Trigger.

For TIM10, TIM11, TIM13 and TIM14 only one of the following parameters can be used: TIM_IT_Update or TIM_IT_CC1

TIM_IT_COM and TIM_IT_Break can be used only with TIM1 and TIM8

Parameters

NewState	new state of the TIM interrupts. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.139.2.9 TIM_SelectCCDMA()

```
void TIM_SelectCCDMA (
            TIM_TypeDef * TIMx,
            FunctionalState NewState )
```

Selects the TIMx peripheral Capture Compare DMA source.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
NewState	new state of the Capture Compare DMA source This parameter can be: ENABLE or DISABLE.

Return values

None	

5.140 Clocks management functions

Clocks management functions.

Functions

- void TIM_InternalClockConfig (TIM_TypeDef *TIMx)

  Configures the TIMx internal Clock.
- void TIM_ITRxExternalClockConfig (TIM_TypeDef *TIMx, uint16_t TIM_InputTriggerSource)

  Configures the TIMx Internal Trigger as External Clock.
- void TIM_TIxExternalClockConfig (TIM_TypeDef *TIMx, uint16_t TIM_TIxExternalCLKSource, uint16_t TIM_ICPolarity, uint16_t ICFilter)

  Configures the TIMx Trigger as External Clock.
- void TIM_ETRClockMode1Config (TIM_TypeDef *TIMx, uint16_t TIM_ExtTRGPrescaler, uint16_t TIM_ExtTRGPolarity, uint16_t ExtTRGFilter)

  Configures the External clock Mode1.
- void TIM_ETRClockMode2Config (TIM_TypeDef *TIMx, uint16_t TIM_ExtTRGPrescaler, uint16_t TIM_ExtTRGPolarity, uint16_t ExtTRGFilter)

  Configures the External clock Mode2.

5.140.1 Detailed Description

Clocks management functions.

```
 ==============================================================================
                        Clocks management functions
 ==============================================================================
```

5.140.2 Function Documentation

5.140.2.1 TIM_ETRClockMode1Config()

```
void TIM_ETRClockMode1Config (
             TIM_TypeDef * TIMx,
             uint16_t TIM_ExtTRGPrescaler,
             uint16_t TIM_ExtTRGPolarity,
             uint16_t ExtTRGFilter )
```

Configures the External clock Mode1.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_ExtTRGPrescaler	The external Trigger Prescaler. This parameter can be one of the following values: • TIM_ExtTRGPSC_OFF: ETRP Prescaler OFF. • TIM_ExtTRGPSC_DIV2: ETRP frequency divided by 2. • TIM_ExtTRGPSC_DIV4: ETRP frequency divided by 4. • TIM_ExtTRGPSC_DIV8: ETRP frequency divided by 8.
TIM_ExtTRGPolarity	The external Trigger Polarity. This parameter can be one of the following values: • TIM_ExtTRGPolarity_Inverted: active low or falling edge active. • TIM_ExtTRGPolarity_NonInverted: active high or rising edge active.
ExtTRGFilter	External Trigger Filter. This parameter must be a value between 0x00 and 0x0F

Return values

None	

5.140.2.2 TIM_ETRClockMode2Config()

```
void TIM_ETRClockMode2Config (
             TIM_TypeDef * TIMx,
             uint16_t TIM_ExtTRGPrescaler,
             uint16_t TIM_ExtTRGPolarity,
             uint16_t ExtTRGFilter )
```

Configures the External clock Mode2.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_ExtTRGPrescaler	The external Trigger Prescaler. This parameter can be one of the following values: • TIM_ExtTRGPSC_OFF: ETRP Prescaler OFF. • TIM_ExtTRGPSC_DIV2: ETRP frequency divided by 2. • TIM_ExtTRGPSC_DIV4: ETRP frequency divided by 4. • TIM_ExtTRGPSC_DIV8: ETRP frequency divided by 8.
TIM_ExtTRGPolarity	The external Trigger Polarity. This parameter can be one of the following values: • TIM_ExtTRGPolarity_Inverted: active low or falling edge active. • TIM_ExtTRGPolarity_NonInverted: active high or rising edge active.
ExtTRGFilter	External Trigger Filter. This parameter must be a value between 0x00 and 0x0F

Return values

None	

5.140.2.3 TIM_InternalClockConfig()

```
void TIM_InternalClockConfig (
            TIM_TypeDef * TIMx )
```

Configures the TIMx internal Clock.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.

Return values

None	

5.140.2.4 TIM_ITRxExternalClockConfig()

```
void TIM_ITRxExternalClockConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_InputTriggerSource )
```

Configures the TIMx Internal Trigger as External Clock.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
TIM_InputTriggerSource	Trigger source. This parameter can be one of the following values: • TIM_TS_ITR0: Internal Trigger 0 • TIM_TS_ITR1: Internal Trigger 1 • TIM_TS_ITR2: Internal Trigger 2 • TIM_TS_ITR3: Internal Trigger 3

Return values

None	

5.140.2.5 TIM_TIxExternalClockConfig()

```
void TIM_TIxExternalClockConfig (
            TIM_TypeDef * TIMx,
            uint16_t TIM_TIxExternalCLKSource,
            uint16_t TIM_ICPolarity,
            uint16_t ICFilter )
```

Configures the TIMx Trigger as External Clock.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9, 10, 11, 12, 13 or 14 to select the TIM peripheral.
TIM_TIxExternalCLKSource	Trigger source. This parameter can be one of the following values: • TIM_TIxExternalCLK1Source_TI1ED: TI1 Edge Detector • TIM_TIxExternalCLK1Source_TI1: Filtered Timer Input 1 • TIM_TIxExternalCLK1Source_TI2: Filtered Timer Input 2
TIM_ICPolarity	specifies the TIx Polarity. This parameter can be one of the following values: • TIM_ICPolarity_Rising • TIM_ICPolarity_Falling
ICFilter	specifies the filter value. This parameter must be a value between 0x0 and 0xF.

Return values

None	

5.141 Synchronization management functions

Synchronization management functions.

Functions

- void TIM_SelectInputTrigger (TIM_TypeDef ∗TIMx, uint16_t TIM_InputTriggerSource)

  Selects the Input Trigger source.
- void TIM_SelectOutputTrigger (TIM_TypeDef ∗TIMx, uint16_t TIM_TRGOSource)

  Selects the TIMx Trigger Output Mode.
- void TIM_SelectSlaveMode (TIM_TypeDef ∗TIMx, uint16_t TIM_SlaveMode)

  Selects the TIMx Slave Mode.
- void TIM_SelectMasterSlaveMode (TIM_TypeDef ∗TIMx, uint16_t TIM_MasterSlaveMode)

  Sets or Resets the TIMx Master/Slave Mode.
- void TIM_ETRConfig (TIM_TypeDef ∗TIMx, uint16_t TIM_ExtTRGPrescaler, uint16_t TIM_ExtTRGPolarity, uint16_t ExtTRGFilter)

  Configures the TIMx External Trigger (ETR).

5.141.1 Detailed Description

Synchronization management functions.

```
 ===================================================================================
                        Synchronization management functions
 ===================================================================================

       ===================================================================
              TIM Driver: how to use it in synchronization Mode
       ===================================================================
       Case of two/several Timers
       **************************
       1. Configure the Master Timers using the following functions:
          - void TIM_SelectOutputTrigger(TIM_TypeDef* TIMx, uint16_t TIM_TRGOSource);
          - void TIM_SelectMasterSlaveMode(TIM_TypeDef* TIMx, uint16_t TIM_MasterSlaveMode);
       2. Configure the Slave Timers using the following functions:
          - void TIM_SelectInputTrigger(TIM_TypeDef* TIMx, uint16_t TIM_InputTriggerSource);
          - void TIM_SelectSlaveMode(TIM_TypeDef* TIMx, uint16_t TIM_SlaveMode);

       Case of Timers and external trigger(ETR pin)
       ********************************************
       1. Configure the External trigger using this function:
          - void TIM_ETRConfig(TIM_TypeDef* TIMx, uint16_t TIM_ExtTRGPrescaler, uint16_t TIM_ExtTRGPolarity,
                               uint16_t ExtTRGFilter);
       2. Configure the Slave Timers using the following functions:
          - void TIM_SelectInputTrigger(TIM_TypeDef* TIMx, uint16_t TIM_InputTriggerSource);
          - void TIM_SelectSlaveMode(TIM_TypeDef* TIMx, uint16_t TIM_SlaveMode);
```

5.141.2 Function Documentation

5.141.2.1 TIM_ETRConfig()

```
void TIM_ETRConfig (
             TIM_TypeDef * TIMx,
             uint16_t TIM_ExtTRGPrescaler,
             uint16_t TIM_ExtTRGPolarity,
             uint16_t ExtTRGFilter )
```

Configures the TIMx External Trigger (ETR).

Parameters

TIMx	where x can be 1, 2, 3, 4, 5 or 8 to select the TIM peripheral.
TIM_ExtTRGPrescaler	The external Trigger Prescaler. This parameter can be one of the following values: • TIM_ExtTRGPSC_OFF: ETRP Prescaler OFF. • TIM_ExtTRGPSC_DIV2: ETRP frequency divided by 2. • TIM_ExtTRGPSC_DIV4: ETRP frequency divided by 4. • TIM_ExtTRGPSC_DIV8: ETRP frequency divided by 8.
TIM_ExtTRGPolarity	The external Trigger Polarity. This parameter can be one of the following values: • TIM_ExtTRGPolarity_Inverted: active low or falling edge active. • TIM_ExtTRGPolarity_NonInverted: active high or rising edge active.
ExtTRGFilter	External Trigger Filter. This parameter must be a value between 0x00 and 0x0F

Return values

None	

5.141.2.2 TIM_SelectInputTrigger()

```
void TIM_SelectInputTrigger (
            TIM_TypeDef * TIMx,
            uint16_t TIM_InputTriggerSource )
```

Selects the Input Trigger source.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9, 10, 11, 12, 13 or 14 to select the TIM peripheral.
TIM_InputTriggerSource	The Input Trigger source. This parameter can be one of the following values: • TIM_TS_ITR0: Internal Trigger 0 • TIM_TS_ITR1: Internal Trigger 1 • TIM_TS_ITR2: Internal Trigger 2 • TIM_TS_ITR3: Internal Trigger 3 • TIM_TS_TI1F_ED: TI1 Edge Detector • TIM_TS_TI1FP1: Filtered Timer Input 1 • TIM_TS_TI2FP2: Filtered Timer Input 2 • TIM_TS_ETRF: External Trigger input

Return values

None	

5.141.2.3 TIM_SelectMasterSlaveMode()

```
void TIM_SelectMasterSlaveMode (
            TIM_TypeDef * TIMx,
            uint16_t TIM_MasterSlaveMode )
```

Sets or Resets the TIMx Master/Slave Mode.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.

Parameters

TIM_MasterSlaveMode	specifies the Timer Master Slave Mode. This parameter can be one of the following values: • TIM_MasterSlaveMode_Enable: synchronization between the current timer and its slaves (through TRGO) • TIM_MasterSlaveMode_Disable: No action

Return values

None	

5.141.2.4 TIM_SelectOutputTrigger()

```
void TIM_SelectOutputTrigger (
             TIM_TypeDef * TIMx,
             uint16_t TIM_TRGOSource )
```

Selects the TIMx Trigger Output Mode.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 6, 7 or 8 to select the TIM peripheral.
TIM_TRGOSource	specifies the Trigger Output source. This parameter can be one of the following values:

- For all TIMx
  - TIM_TRGOSource_Reset: The UG bit in the TIM_EGR register is used as the trigger output(TRGO)
  - TIM_TRGOSource_Enable: The Counter Enable CEN is used as the trigger output(TRGO)
  - TIM_TRGOSource_Update: The update event is selected as the trigger output(TRGO)
- For all TIMx except TIM6 and TIM7
  - TIM_TRGOSource_OC1: The trigger output sends a positive pulse when the CC1IF flag is to be set, as soon as a capture or compare match occurs(TRGO)
  - TIM_TRGOSource_OC1Ref: OC1REF signal is used as the trigger output(TRGO)
  - TIM_TRGOSource_OC2Ref: OC2REF signal is used as the trigger output(TRGO)
  - TIM_TRGOSource_OC3Ref: OC3REF signal is used as the trigger output(TRGO)
  - TIM_TRGOSource_OC4Ref: OC4REF signal is used as the trigger output(TRGO)

Return values

None	

5.141.2.5 TIM_SelectSlaveMode()

```
void TIM_SelectSlaveMode (
             TIM_TypeDef * TIMx,
             uint16_t TIM_SlaveMode )
```

Selects the TIMx Slave Mode.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
TIM_SlaveMode	specifies the Timer Slave Mode. This parameter can be one of the following values: • TIM_SlaveMode_Reset: Rising edge of the selected trigger signal(TRGI) reinitialize the counter and triggers an update of the registers • TIM_SlaveMode_Gated: The counter clock is enabled when the trigger signal (TRGI) is high • TIM_SlaveMode_Trigger: The counter starts at a rising edge of the trigger TRGI • TIM_SlaveMode_External1: Rising edges of the selected trigger (TRGI) clock the counter

Return values

None	

5.142 Specific interface management functions

Specific interface management functions.

Functions

- void TIM_EncoderInterfaceConfig (TIM_TypeDef ∗TIMx, uint16_t TIM_EncoderMode, uint16_t TIM_IC1↩ Polarity, uint16_t TIM_IC2Polarity)

  Configures the TIMx Encoder Interface.
- void TIM_SelectHallSensor (TIM_TypeDef ∗TIMx, FunctionalState NewState)

  Enables or disables the TIMx's Hall sensor interface.

5.142.1 Detailed Description

Specific interface management functions.

```
 ============================================================================
                    Specific interface management functions
 ============================================================================
```

5.142.2 Function Documentation

5.142.2.1 TIM_EncoderInterfaceConfig()

```
void TIM_EncoderInterfaceConfig (
             TIM_TypeDef * TIMx,
             uint16_t TIM_EncoderMode,
             uint16_t TIM_IC1Polarity,
             uint16_t TIM_IC2Polarity )
```

Configures the TIMx Encoder Interface.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
TIM_EncoderMode	specifies the TIMx Encoder Mode. This parameter can be one of the following values: • TIM_EncoderMode_TI1: Counter counts on TI1FP1 edge depending on TI2FP2 level. • TIM_EncoderMode_TI2: Counter counts on TI2FP2 edge depending on TI1FP1 level. • TIM_EncoderMode_TI12: Counter counts on both TI1FP1 and TI2FP2 edges depending on the level of the other input.
TIM_IC1Polarity	specifies the IC1 Polarity This parameter can be one of the following values: • TIM_ICPolarity_Falling: IC Falling edge. • TIM_ICPolarity_Rising: IC Rising edge.
TIM_IC2Polarity	specifies the IC2 Polarity This parameter can be one of the following values: • TIM_ICPolarity_Falling: IC Falling edge. • TIM_ICPolarity_Rising: IC Rising edge.

Return values

None	

5.142.2.2 TIM_SelectHallSensor()

```
void TIM_SelectHallSensor (
             TIM_TypeDef * TIMx,
             FunctionalState NewState )
```

Enables or disables the TIMx's Hall sensor interface.

Parameters

TIMx	where x can be 1, 2, 3, 4, 5, 8, 9 or 12 to select the TIM peripheral.
NewState	new state of the TIMx Hall sensor interface. This parameter can be: ENABLE or DISABLE.

Return values

None	

5.143 Specific remapping management function

Specific remapping management function.

Functions

- void TIM_RemapConfig (TIM_TypeDef *TIMx, uint16_t TIM_Remap)
  Configures the TIM2, TIM5 and TIM11 Remapping input capabilities.

5.143.1 Detailed Description

Specific remapping management function.

```
 ==============================================================================
                     Specific remapping management function
 ==============================================================================
```

5.143.2 Function Documentation

5.143.2.1 TIM_RemapConfig()

```
void TIM_RemapConfig (
             TIM_TypeDef * TIMx,
             uint16_t TIM_Remap )
```

Configures the TIM2, TIM5 and TIM11 Remapping input capabilities.

Parameters

TIMx	where x can be 2, 5 or 11 to select the TIM peripheral.

Parameters

TIM_Remap	specifies the TIM input remapping source. This parameter can be one of the following values: • TIM2_TIM8_TRGO: TIM2 ITR1 input is connected to TIM8 Trigger output(default) • TIM2_ETH_PTP: TIM2 ITR1 input is connected to ETH PTP trogger output. • TIM2_USBFS_SOF: TIM2 ITR1 input is connected to USB FS SOF. • TIM2_USBHS_SOF: TIM2 ITR1 input is connected to USB HS SOF. • TIM5_GPIO: TIM5 CH4 input is connected to dedicated Timer pin(default) • TIM5_LSI: TIM5 CH4 input is connected to LSI clock. • TIM5_LSE: TIM5 CH4 input is connected to LSE clock. • TIM5_RTC: TIM5 CH4 input is connected to RTC Output event. • TIM11_GPIO: TIM11 CH4 input is connected to dedicated Timer pin(default) • TIM11_HSE: TIM11 CH4 input is connected to HSE_RTC clock (HSE divided by a programmable prescaler)

Return values

None	

5.144 STM32F4xx_StdPeriph_Driver

Modules

- MISC
  MISC driver modules.
- DMA
  DMA driver modules.
- GPIO
  GPIO driver modules.
- RCC
  RCC driver modules.
- TIM
  TIM driver modules.

5.144.1 Detailed Description

5.145 CMSIS

**

Modules

- Stm32f4xx
- Stm32f4xx_system

**

5.145.1 Detailed Description

**

**

5.146 Stm32f4xx

Modules

- Library_configuration_section
- Configuration_section_for_CMSIS
- Exported_types
- Peripheral_registers_structures
- Peripheral_memory_map
- Peripheral_declaration
- Exported_constants
- Exported_macro

5.146.1 Detailed Description

5.147 Library_configuration_section

Macros

- #define **STM32F4XX**
- #define HSE_VALUE ((uint32_t)8000000)

  Comment the line below if you will not use the peripherals drivers. In this case, these drivers will not be included and the application code will be based on direct access to peripherals registers.
- #define HSE_STARTUP_TIMEOUT ((uint16_t)0x0500)

  In the following line adjust the External High Speed oscillator (HSE) Startup Timeout value.
- #define HSI_VALUE ((uint32_t)16000000)
- #define __STM32F4XX_STDPERIPH_VERSION_MAIN (0x01)

  STM32F4XX Standard Peripherals Library version number V1.0.0.
- #define __STM32F4XX_STDPERIPH_VERSION_SUB1 (0x00)
- #define __STM32F4XX_STDPERIPH_VERSION_SUB2 (0x00)
- #define __STM32F4XX_STDPERIPH_VERSION_RC (0x00)
- #define __STM32F4XX_STDPERIPH_VERSION

5.147.1 Detailed Description

5.147.2 Macro Definition Documentation

5.147.2.1 __STM32F4XX_STDPERIPH_VERSION

```
#define __STM32F4XX_STDPERIPH_VERSION
```

Value:

```
((__STM32F4XX_STDPERIPH_VERSION_MAIN << 24)\
|(__STM32F4XX_STDPERIPH_VERSION_SUB1 << 16)\
|(__STM32F4XX_STDPERIPH_VERSION_SUB2 << 8)\
|(__STM32F4XX_STDPERIPH_VERSION_RC))
```

5.147.2.2 __STM32F4XX_STDPERIPH_VERSION_MAIN

```
#define __STM32F4XX_STDPERIPH_VERSION_MAIN (0x01)
```

STM32F4XX Standard Peripherals Library version number V1.0.0.

[31:24] main version

5.147.2.3 __STM32F4XX_STDPERIPH_VERSION_RC

```
#define __STM32F4XX_STDPERIPH_VERSION_RC (0x00)
```

[7:0] release candidate

5.147.2.4 __STM32F4XX_STDPERIPH_VERSION_SUB1

```
#define __STM32F4XX_STDPERIPH_VERSION_SUB1 (0x00)
```

[23:16] sub1 version

5.147.2.5 __STM32F4XX_STDPERIPH_VERSION_SUB2

```
#define __STM32F4XX_STDPERIPH_VERSION_SUB2 (0x00)
```

[15:8] sub2 version

5.147.2.6 HSE_STARTUP_TIMEOUT

```
#define HSE_STARTUP_TIMEOUT ((uint16_t)0x0500)
```

In the following line adjust the External High Speed oscillator (HSE) Startup Timeout value.

Time out for HSE start up

5.147.2.7 HSE_VALUE

```
#define HSE_VALUE ((uint32_t)8000000)
```

Comment the line below if you will not use the peripherals drivers. In this case, these drivers will not be included and the application code will be based on direct access to peripherals registers.

In the following line adjust the value of External High Speed oscillator (HSE) used in your application

Tip: To avoid modifying this file each time you need to use different HSE, you can define the HSE value in your toolchain compiler preprocessor. Value of the External oscillator in Hz

5.147.2.8 HSI_VALUE

```
#define HSI_VALUE ((uint32_t)16000000)
```

Value of the Internal oscillator in Hz

5.148 Configuration_section_for_CMSIS

Macros

- #define __CM4_REV 0x0001

  Configuration of the Cortex-M4 Processor and Core Peripherals.
- #define __MPU_PRESENT 1
- #define __NVIC_PRIO_BITS 4
- #define __Vendor_SysTickConfig 0
- #define __FPU_PRESENT 1

Typedefs

- typedef enum IRQn **IRQn_Type**

  STM32F4XX Interrupt Number Definition, according to the selected device in Library_configuration_section.

Enumerations

- enum IRQn {
  NonMaskableInt_IRQn = -14 , MemoryManagement_IRQn = -12 , BusFault_IRQn = -11 , UsageFault_IRQn = -10 ,
  SVCall_IRQn = -5 , DebugMonitor_IRQn = -4 , PendSV_IRQn = -2 , SysTick_IRQn = -1 ,
  WWDG_IRQn = 0 , PVD_IRQn = 1 , TAMP_STAMP_IRQn = 2 , RTC_WKUP_IRQn = 3 ,
  FLASH_IRQn = 4 , RCC_IRQn = 5 , EXTI0_IRQn = 6 , EXTI1_IRQn = 7 ,
  EXTI2_IRQn = 8 , EXTI3_IRQn = 9 , EXTI4_IRQn = 10 , DMA1_Stream0_IRQn = 11 ,
  DMA1_Stream1_IRQn = 12 , DMA1_Stream2_IRQn = 13 , DMA1_Stream3_IRQn = 14 , DMA1_Stream4_IRQn = 15 ,
  DMA1_Stream5_IRQn = 16 , DMA1_Stream6_IRQn = 17 , ADC_IRQn = 18 , CAN1_TX_IRQn = 19 ,
  CAN1_RX0_IRQn = 20 , CAN1_RX1_IRQn = 21 , CAN1_SCE_IRQn = 22 , EXTI9_5_IRQn = 23 ,
  TIM1_BRK_TIM9_IRQn = 24 , TIM1_UP_TIM10_IRQn = 25 , TIM1_TRG_COM_TIM11_IRQn = 26 ,
  TIM1_CC_IRQn = 27 ,
  TIM2_IRQn = 28 , TIM3_IRQn = 29 , TIM4_IRQn = 30 , I2C1_EV_IRQn = 31 ,

I2C1_ER_IRQn = 32 , I2C2_EV_IRQn = 33 , I2C2_ER_IRQn = 34 , SPI1_IRQn = 35 ,
SPI2_IRQn = 36 , USART1_IRQn = 37 , USART2_IRQn = 38 , USART3_IRQn = 39 ,
EXTI15_10_IRQn = 40 , RTC_Alarm_IRQn = 41 , OTG_FS_WKUP_IRQn = 42 , TIM8_BRK_TIM12_IRQn = 43 ,
TIM8_UP_TIM13_IRQn = 44 , TIM8_TRG_COM_TIM14_IRQn = 45 , TIM8_CC_IRQn = 46 , DMA1_Stream7_IRQn = 47 ,
FSMC_IRQn = 48 , SDIO_IRQn = 49 , TIM5_IRQn = 50 , SPI3_IRQn = 51 ,
UART4_IRQn = 52 , UART5_IRQn = 53 , TIM6_DAC_IRQn = 54 , TIM7_IRQn = 55 ,
DMA2_Stream0_IRQn = 56 , DMA2_Stream1_IRQn = 57 , DMA2_Stream2_IRQn = 58 , DMA2_Stream3_IRQn = 59 ,
DMA2_Stream4_IRQn = 60 , ETH_IRQn = 61 , ETH_WKUP_IRQn = 62 , CAN2_TX_IRQn = 63 ,
CAN2_RX0_IRQn = 64 , CAN2_RX1_IRQn = 65 , CAN2_SCE_IRQn = 66 , OTG_FS_IRQn = 67 ,
DMA2_Stream5_IRQn = 68 , DMA2_Stream6_IRQn = 69 , DMA2_Stream7_IRQn = 70 , USART6_IRQn = 71 ,
I2C3_EV_IRQn = 72 , I2C3_ER_IRQn = 73 , OTG_HS_EP1_OUT_IRQn = 74 , OTG_HS_EP1_IN_IRQn = 75 ,
OTG_HS_WKUP_IRQn = 76 , OTG_HS_IRQn = 77 , DCMI_IRQn = 78 , CRYP_IRQn = 79 ,
HASH_RNG_IRQn = 80 , FPU_IRQn = 81 }

STM32F4XX Interrupt Number Definition, according to the selected device in Library_configuration_section.

5.148.1 Detailed Description

5.148.2 Macro Definition Documentation

5.148.2.1 __CM4_REV

```
#define __CM4_REV 0x0001
```

Configuration of the Cortex-M4 Processor and Core Peripherals.

Core revision r0p1

5.148.2.2 __FPU_PRESENT

```
#define __FPU_PRESENT 1
```

FPU present

5.148.2.3 __MPU_PRESENT

```
#define __MPU_PRESENT 1
```

STM32F4XX provides an MPU

5.148.2.4 __NVIC_PRIO_BITS

```
#define __NVIC_PRIO_BITS 4
```

STM32F4XX uses 4 Bits for the Priority Levels

5.148.2.5 __Vendor_SysTickConfig

```
#define __Vendor_SysTickConfig 0
```

Set to 1 if different SysTick Config is used

5.148.3 Enumeration Type Documentation

5.148.3.1 IRQn

```
enum IRQn
```

STM32F4XX Interrupt Number Definition, according to the selected device in Library_configuration_section.

Enumerator

NonMaskableInt_IRQn	2 Non Maskable Interrupt
MemoryManagement_IRQn	4 Cortex-M4 Memory Management Interrupt
BusFault_IRQn	5 Cortex-M4 Bus Fault Interrupt
UsageFault_IRQn	6 Cortex-M4 Usage Fault Interrupt
SVCall_IRQn	11 Cortex-M4 SV Call Interrupt
DebugMonitor_IRQn	12 Cortex-M4 Debug Monitor Interrupt
PendSV_IRQn	14 Cortex-M4 Pend SV Interrupt
SysTick_IRQn	15 Cortex-M4 System Tick Interrupt
WWDG_IRQn	Window WatchDog Interrupt
PVD_IRQn	PVD through EXTI Line detection Interrupt
TAMP_STAMP_IRQn	Tamper and TimeStamp interrupts through the EXTI line
RTC_WKUP_IRQn	RTC Wakeup interrupt through the EXTI line
FLASH_IRQn	FLASH global Interrupt

Enumerator

RCC_IRQn	RCC global Interrupt
EXTI0_IRQn	EXTI Line0 Interrupt
EXTI1_IRQn	EXTI Line1 Interrupt
EXTI2_IRQn	EXTI Line2 Interrupt
EXTI3_IRQn	EXTI Line3 Interrupt
EXTI4_IRQn	EXTI Line4 Interrupt
DMA1_Stream0_IRQn	DMA1 Stream 0 global Interrupt
DMA1_Stream1_IRQn	DMA1 Stream 1 global Interrupt
DMA1_Stream2_IRQn	DMA1 Stream 2 global Interrupt
DMA1_Stream3_IRQn	DMA1 Stream 3 global Interrupt
DMA1_Stream4_IRQn	DMA1 Stream 4 global Interrupt
DMA1_Stream5_IRQn	DMA1 Stream 5 global Interrupt
DMA1_Stream6_IRQn	DMA1 Stream 6 global Interrupt
ADC_IRQn	ADC1, ADC2 and ADC3 global Interrupts
CAN1_TX_IRQn	CAN1 TX Interrupt
CAN1_RX0_IRQn	CAN1 RX0 Interrupt
CAN1_RX1_IRQn	CAN1 RX1 Interrupt
CAN1_SCE_IRQn	CAN1 SCE Interrupt
EXTI9_5_IRQn	External Line[9:5] Interrupts
TIM1_BRK_TIM9_IRQn	TIM1 Break interrupt and TIM9 global interrupt
TIM1_UP_TIM10_IRQn	TIM1 Update Interrupt and TIM10 global interrupt
TIM1_TRG_COM_TIM11_IRQn	TIM1 Trigger and Commutation Interrupt and TIM11 global interrupt
TIM1_CC_IRQn	TIM1 Capture Compare Interrupt
TIM2_IRQn	TIM2 global Interrupt
TIM3_IRQn	TIM3 global Interrupt
TIM4_IRQn	TIM4 global Interrupt
I2C1_EV_IRQn	I2C1 Event Interrupt
I2C1_ER_IRQn	I2C1 Error Interrupt

Enumerator

I2C2_EV_IRQn	I2C2 Event Interrupt
I2C2_ER_IRQn	I2C2 Error Interrupt
SPI1_IRQn	SPI1 global Interrupt
SPI2_IRQn	SPI2 global Interrupt
USART1_IRQn	USART1 global Interrupt
USART2_IRQn	USART2 global Interrupt
USART3_IRQn	USART3 global Interrupt
EXTI15_10_IRQn	External Line[15:10] Interrupts
RTC_Alarm_IRQn	RTC Alarm (A and B) through EXTI Line Interrupt
OTG_FS_WKUP_IRQn	USB OTG FS Wakeup through EXTI line interrupt
TIM8_BRK_TIM12_IRQn	TIM8 Break Interrupt and TIM12 global interrupt
TIM8_UP_TIM13_IRQn	TIM8 Update Interrupt and TIM13 global interrupt
TIM8_TRG_COM_TIM14_IRQn	TIM8 Trigger and Commutation Interrupt and TIM14 global interrupt
TIM8_CC_IRQn	TIM8 Capture Compare Interrupt
DMA1_Stream7_IRQn	DMA1 Stream7 Interrupt
FSMC_IRQn	FSMC global Interrupt
SDIO_IRQn	SDIO global Interrupt
TIM5_IRQn	TIM5 global Interrupt
SPI3_IRQn	SPI3 global Interrupt
UART4_IRQn	UART4 global Interrupt
UART5_IRQn	UART5 global Interrupt
TIM6_DAC_IRQn	TIM6 global and DAC1&2 underrun error interrupts
TIM7_IRQn	TIM7 global interrupt
DMA2_Stream0_IRQn	DMA2 Stream 0 global Interrupt
DMA2_Stream1_IRQn	DMA2 Stream 1 global Interrupt
DMA2_Stream2_IRQn	DMA2 Stream 2 global Interrupt
DMA2_Stream3_IRQn	DMA2 Stream 3 global Interrupt
DMA2_Stream4_IRQn	DMA2 Stream 4 global Interrupt

Enumerator

ETH_IRQn	Ethernet global Interrupt
ETH_WKUP_IRQn	Ethernet Wakeup through EXTI line Interrupt
CAN2_TX_IRQn	CAN2 TX Interrupt
CAN2_RX0_IRQn	CAN2 RX0 Interrupt
CAN2_RX1_IRQn	CAN2 RX1 Interrupt
CAN2_SCE_IRQn	CAN2 SCE Interrupt
OTG_FS_IRQn	USB OTG FS global Interrupt
DMA2_Stream5_IRQn	DMA2 Stream 5 global interrupt
DMA2_Stream6_IRQn	DMA2 Stream 6 global interrupt
DMA2_Stream7_IRQn	DMA2 Stream 7 global interrupt
USART6_IRQn	USART6 global interrupt
I2C3_EV_IRQn	I2C3 event interrupt
I2C3_ER_IRQn	I2C3 error interrupt
OTG_HS_EP1_OUT_IRQn	USB OTG HS End Point 1 Out global interrupt
OTG_HS_EP1_IN_IRQn	USB OTG HS End Point 1 In global interrupt
OTG_HS_WKUP_IRQn	USB OTG HS Wakeup through EXTI interrupt
OTG_HS_IRQn	USB OTG HS global interrupt
DCMI_IRQn	DCMI global interrupt
CRYP_IRQn	CRYP crypto global interrupt
HASH_RNG_IRQn	Hash and Rng global interrupt
FPU_IRQn	FPU global interrupt

5.149 Exported_types

Macros

- #define **IS_FUNCTIONAL_STATE**(STATE) (((STATE) == DISABLE) || ((STATE) == ENABLE))

Typedefs

- typedef int32_t s32
- typedef int16_t **s16**
- typedef int8_t **s8**
- typedef const int32_t sc32
- typedef const int16_t sc16
- typedef const int8_t sc8
- typedef __IO int32_t **vs32**
- typedef __IO int16_t **vs16**
- typedef __IO int8_t **vs8**
- typedef __I int32_t vsc32
- typedef __I int16_t vsc16
- typedef __I int8_t vsc8
- typedef uint32_t **u32**
- typedef uint16_t **u16**
- typedef uint8_t **u8**
- typedef const uint32_t uc32
- typedef const uint16_t uc16
- typedef const uint8_t uc8
- typedef __IO uint32_t **vu32**
- typedef __IO uint16_t **vu16**
- typedef __IO uint8_t **vu8**
- typedef __I uint32_t vuc32
- typedef __I uint16_t vuc16
- typedef __I uint8_t vuc8
- typedef enum FlagStatus **ITStatus**

Enumerations

- enum **FlagStatus** { **RESET** = 0 , **SET** = !RESET }
- enum **FunctionalState** { **DISABLE** = 0 , **ENABLE** = !DISABLE }
- enum **ErrorStatus** { **ERROR** = 0 , **SUCCESS** = !ERROR }

5.149.1 Detailed Description

5.149.2 Typedef Documentation

5.149.2.1 s32

```
typedef int32_t s32
```

< STM32F10x Standard Peripheral Library old types (maintained for legacy purpose)

5.149.2.2 sc16

```
typedef const int16_t sc16
```

Read Only

5.149.2.3 sc32

```
typedef const int32_t sc32
```

Read Only

5.149.2.4 sc8

```
typedef const int8_t sc8
```

Read Only

5.149.2.5 uc16

```
typedef const uint16_t uc16
```

Read Only

5.149.2.6 uc32

```
typedef const uint32_t uc32
```

Read Only

5.149.2.7 uc8

```
typedef const uint8_t uc8
```

Read Only

5.149.2.8 vsc16

```
typedef __I int16_t vsc16
```

Read Only

5.149.2.9 vsc32

```
typedef __I int32_t vsc32
```

Read Only

5.149.2.10 vsc8

```
typedef __I int8_t vsc8
```

Read Only

5.149.2.11 vuc16

```
typedef __I uint16_t vuc16
```

Read Only

5.149.2.12 vuc32

```
typedef __I uint32_t vuc32
```

Read Only

5.149.2.13 vuc8

```
typedef __I uint8_t vuc8
```

Read Only

5.150 Peripheral_registers_structures

Data Structures

- struct ADC_TypeDef
  Analog to Digital Converter
- struct ADC_Common_TypeDef
- struct CAN_TxMailBox_TypeDef
  Controller Area Network TxMailBox.
- struct CAN_FIFOMailBox_TypeDef
  Controller Area Network FIFOMailBox.
- struct CAN_FilterRegister_TypeDef
  Controller Area Network FilterRegister.
- struct CAN_TypeDef
  Controller Area Network.
- struct CRC_TypeDef
  CRC calculation unit.
- struct DAC_TypeDef
  Digital to Analog Converter.
- struct DBGMCU_TypeDef
  Debug MCU.
- struct DCMI_TypeDef
  DCMI.
- struct DMA_Stream_TypeDef
  DMA Controller.
- struct DMA_TypeDef
- struct ETH_TypeDef
  Ethernet MAC.
- struct EXTI_TypeDef

External Interrupt/Event Controller.

- struct FLASH_TypeDef

  FLASH Registers.
- struct FSMC_Bank1_TypeDef

  Flexible Static Memory Controller.
- struct FSMC_Bank1E_TypeDef

  Flexible Static Memory Controller Bank1E.
- struct FSMC_Bank2_TypeDef

  Flexible Static Memory Controller Bank2.
- struct FSMC_Bank3_TypeDef

  Flexible Static Memory Controller Bank3.
- struct FSMC_Bank4_TypeDef

  Flexible Static Memory Controller Bank4.
- struct GPIO_TypeDef

  General Purpose I/O.
- struct SYSCFG_TypeDef

  System configuration controller.
- struct I2C_TypeDef

  Inter-integrated Circuit Interface.
- struct IWDG_TypeDef

  Independent WATCHDOG.
- struct PWR_TypeDef

  Power Control.
- struct RCC_TypeDef

  Reset and Clock Control.
- struct RTC_TypeDef

  Real-Time Clock.
- struct SDIO_TypeDef

  SD host Interface.
- struct SPI_TypeDef

  Serial Peripheral Interface.
- struct TIM_TypeDef

  TIM.
- struct USART_TypeDef

  Universal Synchronous Asynchronous Receiver Transmitter.
- struct WWDG_TypeDef

  Window WATCHDOG.
- struct CRYP_TypeDef

  Crypto Processor.
- struct HASH_TypeDef

  HASH.
- struct RNG_TypeDef

  HASH.

5.150.1 Detailed Description

5.151 Peripheral_memory_map

Macros

- #define FLASH_BASE ((uint32_t)0x08000000)
- #define CCMDATARAM_BASE ((uint32_t)0x10000000)
- #define SRAM1_BASE ((uint32_t)0x20000000)
- #define SRAM2_BASE ((uint32_t)0x2001C000)
- #define PERIPH_BASE ((uint32_t)0x40000000)
- #define BKPSRAM_BASE ((uint32_t)0x40024000)
- #define FSMC_R_BASE ((uint32_t)0xA0000000)
- #define CCMDATARAM_BB_BASE ((uint32_t)0x12000000)
- #define SRAM1_BB_BASE ((uint32_t)0x22000000)
- #define SRAM2_BB_BASE ((uint32_t)0x2201C000)
- #define PERIPH_BB_BASE ((uint32_t)0x42000000)
- #define BKPSRAM_BB_BASE ((uint32_t)0x42024000)
- #define **SRAM_BASE** SRAM1_BASE
- #define SRAM_BB_BASE SRAM1_BB_BASE
- #define **APB1PERIPH_BASE** PERIPH_BASE
- #define **APB2PERIPH_BASE** (PERIPH_BASE + 0x00010000)
- #define **AHB1PERIPH_BASE** (PERIPH_BASE + 0x00020000)
- #define AHB2PERIPH_BASE (PERIPH_BASE + 0x10000000)
- #define **TIM2_BASE** (APB1PERIPH_BASE + 0x0000)
- #define **TIM3_BASE** (APB1PERIPH_BASE + 0x0400)
- #define **TIM4_BASE** (APB1PERIPH_BASE + 0x0800)
- #define **TIM5_BASE** (APB1PERIPH_BASE + 0x0C00)
- #define **TIM6_BASE** (APB1PERIPH_BASE + 0x1000)
- #define **TIM7_BASE** (APB1PERIPH_BASE + 0x1400)
- #define **TIM12_BASE** (APB1PERIPH_BASE + 0x1800)
- #define **TIM13_BASE** (APB1PERIPH_BASE + 0x1C00)
- #define **TIM14_BASE** (APB1PERIPH_BASE + 0x2000)
- #define **RTC_BASE** (APB1PERIPH_BASE + 0x2800)
- #define **WWDG_BASE** (APB1PERIPH_BASE + 0x2C00)
- #define **IWDG_BASE** (APB1PERIPH_BASE + 0x3000)
- #define **I2S2ext_BASE** (APB1PERIPH_BASE + 0x3400)
- #define **SPI2_BASE** (APB1PERIPH_BASE + 0x3800)
- #define **SPI3_BASE** (APB1PERIPH_BASE + 0x3C00)
- #define **I2S3ext_BASE** (APB1PERIPH_BASE + 0x4000)
- #define **USART2_BASE** (APB1PERIPH_BASE + 0x4400)
- #define **USART3_BASE** (APB1PERIPH_BASE + 0x4800)
- #define **UART4_BASE** (APB1PERIPH_BASE + 0x4C00)
- #define **UART5_BASE** (APB1PERIPH_BASE + 0x5000)
- #define **I2C1_BASE** (APB1PERIPH_BASE + 0x5400)
- #define **I2C2_BASE** (APB1PERIPH_BASE + 0x5800)
- #define **I2C3_BASE** (APB1PERIPH_BASE + 0x5C00)
- #define **CAN1_BASE** (APB1PERIPH_BASE + 0x6400)
- #define **CAN2_BASE** (APB1PERIPH_BASE + 0x6800)
- #define **PWR_BASE** (APB1PERIPH_BASE + 0x7000)
- #define DAC_BASE (APB1PERIPH_BASE + 0x7400)
- #define **TIM1_BASE** (APB2PERIPH_BASE + 0x0000)
- #define **TIM8_BASE** (APB2PERIPH_BASE + 0x0400)

- #define **USART1_BASE** (APB2PERIPH_BASE + 0x1000)
- #define **USART6_BASE** (APB2PERIPH_BASE + 0x1400)
- #define **ADC1_BASE** (APB2PERIPH_BASE + 0x2000)
- #define **ADC2_BASE** (APB2PERIPH_BASE + 0x2100)
- #define **ADC3_BASE** (APB2PERIPH_BASE + 0x2200)
- #define **ADC_BASE** (APB2PERIPH_BASE + 0x2300)
- #define **SDIO_BASE** (APB2PERIPH_BASE + 0x2C00)
- #define **SPI1_BASE** (APB2PERIPH_BASE + 0x3000)
- #define **SYSCFG_BASE** (APB2PERIPH_BASE + 0x3800)
- #define **EXTI_BASE** (APB2PERIPH_BASE + 0x3C00)
- #define **TIM9_BASE** (APB2PERIPH_BASE + 0x4000)
- #define **TIM10_BASE** (APB2PERIPH_BASE + 0x4400)
- #define TIM11_BASE (APB2PERIPH_BASE + 0x4800)
- #define **GPIOA_BASE** (AHB1PERIPH_BASE + 0x0000)
- #define **GPIOB_BASE** (AHB1PERIPH_BASE + 0x0400)
- #define **GPIOC_BASE** (AHB1PERIPH_BASE + 0x0800)
- #define **GPIOD_BASE** (AHB1PERIPH_BASE + 0x0C00)
- #define **GPIOE_BASE** (AHB1PERIPH_BASE + 0x1000)
- #define **GPIOF_BASE** (AHB1PERIPH_BASE + 0x1400)
- #define **GPIOG_BASE** (AHB1PERIPH_BASE + 0x1800)
- #define **GPIOH_BASE** (AHB1PERIPH_BASE + 0x1C00)
- #define **GPIOI_BASE** (AHB1PERIPH_BASE + 0x2000)
- #define **CRC_BASE** (AHB1PERIPH_BASE + 0x3000)
- #define **RCC_BASE** (AHB1PERIPH_BASE + 0x3800)
- #define **FLASH_R_BASE** (AHB1PERIPH_BASE + 0x3C00)
- #define **DMA1_BASE** (AHB1PERIPH_BASE + 0x6000)
- #define **DMA1_Stream0_BASE** (DMA1_BASE + 0x010)
- #define **DMA1_Stream1_BASE** (DMA1_BASE + 0x028)
- #define **DMA1_Stream2_BASE** (DMA1_BASE + 0x040)
- #define **DMA1_Stream3_BASE** (DMA1_BASE + 0x058)
- #define **DMA1_Stream4_BASE** (DMA1_BASE + 0x070)
- #define **DMA1_Stream5_BASE** (DMA1_BASE + 0x088)
- #define **DMA1_Stream6_BASE** (DMA1_BASE + 0x0A0)
- #define **DMA1_Stream7_BASE** (DMA1_BASE + 0x0B8)
- #define **DMA2_BASE** (AHB1PERIPH_BASE + 0x6400)
- #define **DMA2_Stream0_BASE** (DMA2_BASE + 0x010)
- #define **DMA2_Stream1_BASE** (DMA2_BASE + 0x028)
- #define **DMA2_Stream2_BASE** (DMA2_BASE + 0x040)
- #define **DMA2_Stream3_BASE** (DMA2_BASE + 0x058)
- #define **DMA2_Stream4_BASE** (DMA2_BASE + 0x070)
- #define **DMA2_Stream5_BASE** (DMA2_BASE + 0x088)
- #define **DMA2_Stream6_BASE** (DMA2_BASE + 0x0A0)
- #define **DMA2_Stream7_BASE** (DMA2_BASE + 0x0B8)
- #define **ETH_BASE** (AHB1PERIPH_BASE + 0x8000)
- #define **ETH_MAC_BASE** (ETH_BASE)
- #define **ETH_MMC_BASE** (ETH_BASE + 0x0100)
- #define **ETH_PTP_BASE** (ETH_BASE + 0x0700)
- #define ETH_DMA_BASE (ETH_BASE + 0x1000)
- #define **DCMI_BASE** (AHB2PERIPH_BASE + 0x50000)
- #define **CRYP_BASE** (AHB2PERIPH_BASE + 0x60000)
- #define **HASH_BASE** (AHB2PERIPH_BASE + 0x60400)
- #define RNG_BASE (AHB2PERIPH_BASE + 0x60800)
- #define **FSMC_Bank1_R_BASE** (FSMC_R_BASE + 0x0000)
- #define **FSMC_Bank1E_R_BASE** (FSMC_R_BASE + 0x0104)
- #define **FSMC_Bank2_R_BASE** (FSMC_R_BASE + 0x0060)
- #define **FSMC_Bank3_R_BASE** (FSMC_R_BASE + 0x0080)
- #define **FSMC_Bank4_R_BASE** (FSMC_R_BASE + 0x00A0)
- #define **DBGMCU_BASE** ((uint32_t)0xE0042000)

5.151.1 Detailed Description

5.151.2 Macro Definition Documentation

5.151.2.1 AHB2PERIPH_BASE

```
#define AHB2PERIPH_BASE (PERIPH_BASE + 0x10000000)
```

APB1 peripherals

5.151.2.2 BKPSRAM_BASE

```
#define BKPSRAM_BASE ((uint32_t)0x40024000)
```

Backup SRAM(4 KB) base address in the alias region

5.151.2.3 BKPSRAM_BB_BASE

```
#define BKPSRAM_BB_BASE ((uint32_t)0x42024000)
```

Backup SRAM(4 KB) base address in the bit-band region

5.151.2.4 CCMDATARAM_BASE

```
#define CCMDATARAM_BASE ((uint32_t)0x10000000)
```

CCM(core coupled memory) data RAM(64 KB) base address in the alias region

5.151.2.5 CCMDATARAM_BB_BASE

```
#define CCMDATARAM_BB_BASE ((uint32_t)0x12000000)
```

CCM(core coupled memory) data RAM(64 KB) base address in the bit-band region

5.151.2.6 DAC_BASE

```
#define DAC_BASE (APB1PERIPH_BASE + 0x7400)
```

APB2 peripherals

5.151.2.7 ETH_DMA_BASE

```
#define ETH_DMA_BASE (ETH_BASE + 0x1000)
```

AHB2 peripherals

5.151.2.8 FLASH_BASE

```
#define FLASH_BASE ((uint32_t)0x08000000)
```

FLASH(up to 1 MB) base address in the alias region

5.151.2.9 FSMC_R_BASE

```
#define FSMC_R_BASE ((uint32_t)0xA0000000)
```

FSMC registers base address

5.151.2.10 PERIPH_BASE

```
#define PERIPH_BASE ((uint32_t)0x40000000)
```

Peripheral base address in the alias region

5.151.2.11 PERIPH_BB_BASE

```
#define PERIPH_BB_BASE ((uint32_t)0x42000000)
```

Peripheral base address in the bit-band region

5.151.2.12 RNG_BASE

```
#define RNG_BASE (AHB2PERIPH_BASE + 0x60800)
```

FSMC Bankx registers base address

5.151.2.13 SRAM1_BASE

```
#define SRAM1_BASE ((uint32_t)0x20000000)
```

SRAM1(112 KB) base address in the alias region

5.151.2.14 SRAM1_BB_BASE

```
#define SRAM1_BB_BASE ((uint32_t)0x22000000)
```

SRAM1(112 KB) base address in the bit-band region

5.151.2.15 SRAM2_BASE

```
#define SRAM2_BASE ((uint32_t)0x2001C000)
```

SRAM2(16 KB) base address in the alias region

5.151.2.16 SRAM2_BB_BASE

```
#define SRAM2_BB_BASE ((uint32_t)0x2201C000)
```

SRAM2(16 KB) base address in the bit-band region

5.151.2.17 SRAM_BB_BASE

```
#define SRAM_BB_BASE SRAM1_BB_BASE
```

Peripheral memory map

5.151.2.18 TIM11_BASE

```
#define TIM11_BASE (APB2PERIPH_BASE + 0x4800)
```

AHB1 peripherals

5.152 Peripheral_declaration

Macros

- #define **TIM2** ((TIM_TypeDef *) TIM2_BASE)
- #define **TIM3** ((TIM_TypeDef *) TIM3_BASE)
- #define **TIM4** ((TIM_TypeDef *) TIM4_BASE)
- #define **TIM5** ((TIM_TypeDef *) TIM5_BASE)
- #define **TIM6** ((TIM_TypeDef *) TIM6_BASE)
- #define **TIM7** ((TIM_TypeDef *) TIM7_BASE)
- #define **TIM12** ((TIM_TypeDef *) TIM12_BASE)
- #define **TIM13** ((TIM_TypeDef *) TIM13_BASE)
- #define **TIM14** ((TIM_TypeDef *) TIM14_BASE)
- #define **RTC** ((RTC_TypeDef *) RTC_BASE)

- #define **WWDG** ((WWDG_TypeDef *) WWDG_BASE)
- #define **IWDG** ((IWDG_TypeDef *) IWDG_BASE)
- #define **I2S2ext** ((SPI_TypeDef *) I2S2ext_BASE)
- #define **SPI2** ((SPI_TypeDef *) SPI2_BASE)
- #define **SPI3** ((SPI_TypeDef *) SPI3_BASE)
- #define **I2S3ext** ((SPI_TypeDef *) I2S3ext_BASE)
- #define **USART2** ((USART_TypeDef *) USART2_BASE)
- #define **USART3** ((USART_TypeDef *) USART3_BASE)
- #define **UART4** ((USART_TypeDef *) UART4_BASE)
- #define **UART5** ((USART_TypeDef *) UART5_BASE)
- #define **I2C1** ((I2C_TypeDef *) I2C1_BASE)
- #define **I2C2** ((I2C_TypeDef *) I2C2_BASE)
- #define **I2C3** ((I2C_TypeDef *) I2C3_BASE)
- #define **CAN1** ((CAN_TypeDef *) CAN1_BASE)
- #define **CAN2** ((CAN_TypeDef *) CAN2_BASE)
- #define **PWR** ((PWR_TypeDef *) PWR_BASE)
- #define **DAC** ((DAC_TypeDef *) DAC_BASE)
- #define **TIM1** ((TIM_TypeDef *) TIM1_BASE)
- #define **TIM8** ((TIM_TypeDef *) TIM8_BASE)
- #define **USART1** ((USART_TypeDef *) USART1_BASE)
- #define **USART6** ((USART_TypeDef *) USART6_BASE)
- #define **ADC** ((ADC_Common_TypeDef *) ADC_BASE)
- #define **ADC1** ((ADC_TypeDef *) ADC1_BASE)
- #define **ADC2** ((ADC_TypeDef *) ADC2_BASE)
- #define **ADC3** ((ADC_TypeDef *) ADC3_BASE)
- #define **SDIO** ((SDIO_TypeDef *) SDIO_BASE)
- #define **SPI1** ((SPI_TypeDef *) SPI1_BASE)
- #define **SYSCFG** ((SYSCFG_TypeDef *) SYSCFG_BASE)
- #define **EXTI** ((EXTI_TypeDef *) EXTI_BASE)
- #define **TIM9** ((TIM_TypeDef *) TIM9_BASE)
- #define **TIM10** ((TIM_TypeDef *) TIM10_BASE)
- #define **TIM11** ((TIM_TypeDef *) TIM11_BASE)
- #define **GPIOA** ((GPIO_TypeDef *) GPIOA_BASE)
- #define **GPIOB** ((GPIO_TypeDef *) GPIOB_BASE)
- #define **GPIOC** ((GPIO_TypeDef *) GPIOC_BASE)
- #define **GPIOD** ((GPIO_TypeDef *) GPIOD_BASE)
- #define **GPIOE** ((GPIO_TypeDef *) GPIOE_BASE)
- #define **GPIOF** ((GPIO_TypeDef *) GPIOF_BASE)
- #define **GPIOG** ((GPIO_TypeDef *) GPIOG_BASE)
- #define **GPIOH** ((GPIO_TypeDef *) GPIOH_BASE)
- #define **GPIOI** ((GPIO_TypeDef *) GPIOI_BASE)
- #define **CRC** ((CRC_TypeDef *) CRC_BASE)
- #define **RCC** ((RCC_TypeDef *) RCC_BASE)
- #define **FLASH** ((FLASH_TypeDef *) FLASH_R_BASE)
- #define **DMA1** ((DMA_TypeDef *) DMA1_BASE)
- #define **DMA1_Stream0** ((DMA_Stream_TypeDef *) DMA1_Stream0_BASE)
- #define **DMA1_Stream1** ((DMA_Stream_TypeDef *) DMA1_Stream1_BASE)
- #define **DMA1_Stream2** ((DMA_Stream_TypeDef *) DMA1_Stream2_BASE)
- #define **DMA1_Stream3** ((DMA_Stream_TypeDef *) DMA1_Stream3_BASE)
- #define **DMA1_Stream4** ((DMA_Stream_TypeDef *) DMA1_Stream4_BASE)
- #define **DMA1_Stream5** ((DMA_Stream_TypeDef *) DMA1_Stream5_BASE)
- #define **DMA1_Stream6** ((DMA_Stream_TypeDef *) DMA1_Stream6_BASE)
- #define **DMA1_Stream7** ((DMA_Stream_TypeDef *) DMA1_Stream7_BASE)
- #define **DMA2** ((DMA_TypeDef *) DMA2_BASE)
- #define **DMA2_Stream0** ((DMA_Stream_TypeDef *) DMA2_Stream0_BASE)

- #define **DMA2_Stream1** ((DMA_Stream_TypeDef *) DMA2_Stream1_BASE)
- #define **DMA2_Stream2** ((DMA_Stream_TypeDef *) DMA2_Stream2_BASE)
- #define **DMA2_Stream3** ((DMA_Stream_TypeDef *) DMA2_Stream3_BASE)
- #define **DMA2_Stream4** ((DMA_Stream_TypeDef *) DMA2_Stream4_BASE)
- #define **DMA2_Stream5** ((DMA_Stream_TypeDef *) DMA2_Stream5_BASE)
- #define **DMA2_Stream6** ((DMA_Stream_TypeDef *) DMA2_Stream6_BASE)
- #define **DMA2_Stream7** ((DMA_Stream_TypeDef *) DMA2_Stream7_BASE)
- #define **ETH** ((ETH_TypeDef *) ETH_BASE)
- #define **DCMI** ((DCMI_TypeDef *) DCMI_BASE)
- #define **CRYP** ((CRYP_TypeDef *) CRYP_BASE)
- #define **HASH** ((HASH_TypeDef *) HASH_BASE)
- #define **RNG** ((RNG_TypeDef *) RNG_BASE)
- #define **FSMC_Bank1** ((FSMC_Bank1_TypeDef *) FSMC_Bank1_R_BASE)
- #define **FSMC_Bank1E** ((FSMC_Bank1E_TypeDef *) FSMC_Bank1E_R_BASE)
- #define **FSMC_Bank2** ((FSMC_Bank2_TypeDef *) FSMC_Bank2_R_BASE)
- #define **FSMC_Bank3** ((FSMC_Bank3_TypeDef *) FSMC_Bank3_R_BASE)
- #define **FSMC_Bank4** ((FSMC_Bank4_TypeDef *) FSMC_Bank4_R_BASE)
- #define **DBGMCU** ((DBGMCU_TypeDef *) DBGMCU_BASE)

5.152.1 Detailed Description

5.153 Exported_constants

Modules

- Peripheral_Registers_Bits_Definition

5.153.1 Detailed Description

5.154 Peripheral_Registers_Bits_Definition

Macros

- #define ADC_SR_AWD ((uint8_t)0x01)
- #define ADC_SR_EOC ((uint8_t)0x02)
- #define ADC_SR_JEOC ((uint8_t)0x04)
- #define ADC_SR_JSTRT ((uint8_t)0x08)
- #define ADC_SR_STRT ((uint8_t)0x10)
- #define ADC_SR_OVR ((uint8_t)0x20)
- #define ADC_CR1_AWDCH ((uint32_t)0x0000001F)
- #define ADC_CR1_AWDCH_0 ((uint32_t)0x00000001)
- #define ADC_CR1_AWDCH_1 ((uint32_t)0x00000002)
- #define ADC_CR1_AWDCH_2 ((uint32_t)0x00000004)
- #define ADC_CR1_AWDCH_3 ((uint32_t)0x00000008)
- #define ADC_CR1_AWDCH_4 ((uint32_t)0x00000010)
- #define ADC_CR1_EOCIE ((uint32_t)0x00000020)
- #define ADC_CR1_AWDIE ((uint32_t)0x00000040)
- #define ADC_CR1_JEOCIE ((uint32_t)0x00000080)
- #define ADC_CR1_SCAN ((uint32_t)0x00000100)

- #define ADC_CR1_AWDSGL ((uint32_t)0x00000200)
- #define ADC_CR1_JAUTO ((uint32_t)0x00000400)
- #define ADC_CR1_DISCEN ((uint32_t)0x00000800)
- #define ADC_CR1_JDISCEN ((uint32_t)0x00001000)
- #define ADC_CR1_DISCNUM ((uint32_t)0x0000E000)
- #define ADC_CR1_DISCNUM_0 ((uint32_t)0x00002000)
- #define ADC_CR1_DISCNUM_1 ((uint32_t)0x00004000)
- #define ADC_CR1_DISCNUM_2 ((uint32_t)0x00008000)
- #define ADC_CR1_JAWDEN ((uint32_t)0x00400000)
- #define ADC_CR1_AWDEN ((uint32_t)0x00800000)
- #define ADC_CR1_RES ((uint32_t)0x03000000)
- #define ADC_CR1_RES_0 ((uint32_t)0x01000000)
- #define ADC_CR1_RES_1 ((uint32_t)0x02000000)
- #define ADC_CR1_OVRIE ((uint32_t)0x04000000)
- #define ADC_CR2_ADON ((uint32_t)0x00000001)
- #define ADC_CR2_CONT ((uint32_t)0x00000002)
- #define ADC_CR2_DMA ((uint32_t)0x00000100)
- #define ADC_CR2_DDS ((uint32_t)0x00000200)
- #define ADC_CR2_EOCS ((uint32_t)0x00000400)
- #define ADC_CR2_ALIGN ((uint32_t)0x00000800)
- #define ADC_CR2_JEXTSEL ((uint32_t)0x000F0000)
- #define ADC_CR2_JEXTSEL_0 ((uint32_t)0x00010000)
- #define ADC_CR2_JEXTSEL_1 ((uint32_t)0x00020000)
- #define ADC_CR2_JEXTSEL_2 ((uint32_t)0x00040000)
- #define ADC_CR2_JEXTSEL_3 ((uint32_t)0x00080000)
- #define ADC_CR2_JEXTEN ((uint32_t)0x00300000)
- #define ADC_CR2_JEXTEN_0 ((uint32_t)0x00100000)
- #define ADC_CR2_JEXTEN_1 ((uint32_t)0x00200000)
- #define ADC_CR2_JSWSTART ((uint32_t)0x00400000)
- #define ADC_CR2_EXTSEL ((uint32_t)0x0F000000)
- #define ADC_CR2_EXTSEL_0 ((uint32_t)0x01000000)
- #define ADC_CR2_EXTSEL_1 ((uint32_t)0x02000000)
- #define ADC_CR2_EXTSEL_2 ((uint32_t)0x04000000)
- #define ADC_CR2_EXTSEL_3 ((uint32_t)0x08000000)
- #define ADC_CR2_EXTEN ((uint32_t)0x30000000)
- #define ADC_CR2_EXTEN_0 ((uint32_t)0x10000000)
- #define ADC_CR2_EXTEN_1 ((uint32_t)0x20000000)
- #define ADC_CR2_SWSTART ((uint32_t)0x40000000)
- #define ADC_SMPR1_SMP10 ((uint32_t)0x00000007)
- #define ADC_SMPR1_SMP10_0 ((uint32_t)0x00000001)
- #define ADC_SMPR1_SMP10_1 ((uint32_t)0x00000002)
- #define ADC_SMPR1_SMP10_2 ((uint32_t)0x00000004)
- #define ADC_SMPR1_SMP11 ((uint32_t)0x00000038)
- #define ADC_SMPR1_SMP11_0 ((uint32_t)0x00000008)
- #define ADC_SMPR1_SMP11_1 ((uint32_t)0x00000010)
- #define ADC_SMPR1_SMP11_2 ((uint32_t)0x00000020)
- #define ADC_SMPR1_SMP12 ((uint32_t)0x000001C0)
- #define ADC_SMPR1_SMP12_0 ((uint32_t)0x00000040)
- #define ADC_SMPR1_SMP12_1 ((uint32_t)0x00000080)
- #define ADC_SMPR1_SMP12_2 ((uint32_t)0x00000100)
- #define ADC_SMPR1_SMP13 ((uint32_t)0x00000E00)
- #define ADC_SMPR1_SMP13_0 ((uint32_t)0x00000200)
- #define ADC_SMPR1_SMP13_1 ((uint32_t)0x00000400)
- #define ADC_SMPR1_SMP13_2 ((uint32_t)0x00000800)
- #define ADC_SMPR1_SMP14 ((uint32_t)0x00007000)

- #define ADC_SMPR1_SMP14_0 ((uint32_t)0x00001000)
- #define ADC_SMPR1_SMP14_1 ((uint32_t)0x00002000)
- #define ADC_SMPR1_SMP14_2 ((uint32_t)0x00004000)
- #define ADC_SMPR1_SMP15 ((uint32_t)0x00038000)
- #define ADC_SMPR1_SMP15_0 ((uint32_t)0x00008000)
- #define ADC_SMPR1_SMP15_1 ((uint32_t)0x00010000)
- #define ADC_SMPR1_SMP15_2 ((uint32_t)0x00020000)
- #define ADC_SMPR1_SMP16 ((uint32_t)0x001C0000)
- #define ADC_SMPR1_SMP16_0 ((uint32_t)0x00040000)
- #define ADC_SMPR1_SMP16_1 ((uint32_t)0x00080000)
- #define ADC_SMPR1_SMP16_2 ((uint32_t)0x00100000)
- #define ADC_SMPR1_SMP17 ((uint32_t)0x00E00000)
- #define ADC_SMPR1_SMP17_0 ((uint32_t)0x00200000)
- #define ADC_SMPR1_SMP17_1 ((uint32_t)0x00400000)
- #define ADC_SMPR1_SMP17_2 ((uint32_t)0x00800000)
- #define ADC_SMPR1_SMP18 ((uint32_t)0x07000000)
- #define ADC_SMPR1_SMP18_0 ((uint32_t)0x01000000)
- #define ADC_SMPR1_SMP18_1 ((uint32_t)0x02000000)
- #define ADC_SMPR1_SMP18_2 ((uint32_t)0x04000000)
- #define ADC_SMPR2_SMP0 ((uint32_t)0x00000007)
- #define ADC_SMPR2_SMP0_0 ((uint32_t)0x00000001)
- #define ADC_SMPR2_SMP0_1 ((uint32_t)0x00000002)
- #define ADC_SMPR2_SMP0_2 ((uint32_t)0x00000004)
- #define ADC_SMPR2_SMP1 ((uint32_t)0x00000038)
- #define ADC_SMPR2_SMP1_0 ((uint32_t)0x00000008)
- #define ADC_SMPR2_SMP1_1 ((uint32_t)0x00000010)
- #define ADC_SMPR2_SMP1_2 ((uint32_t)0x00000020)
- #define ADC_SMPR2_SMP2 ((uint32_t)0x000001C0)
- #define ADC_SMPR2_SMP2_0 ((uint32_t)0x00000040)
- #define ADC_SMPR2_SMP2_1 ((uint32_t)0x00000080)
- #define ADC_SMPR2_SMP2_2 ((uint32_t)0x00000100)
- #define ADC_SMPR2_SMP3 ((uint32_t)0x00000E00)
- #define ADC_SMPR2_SMP3_0 ((uint32_t)0x00000200)
- #define ADC_SMPR2_SMP3_1 ((uint32_t)0x00000400)
- #define ADC_SMPR2_SMP3_2 ((uint32_t)0x00000800)
- #define ADC_SMPR2_SMP4 ((uint32_t)0x00007000)
- #define ADC_SMPR2_SMP4_0 ((uint32_t)0x00001000)
- #define ADC_SMPR2_SMP4_1 ((uint32_t)0x00002000)
- #define ADC_SMPR2_SMP4_2 ((uint32_t)0x00004000)
- #define ADC_SMPR2_SMP5 ((uint32_t)0x00038000)
- #define ADC_SMPR2_SMP5_0 ((uint32_t)0x00008000)
- #define ADC_SMPR2_SMP5_1 ((uint32_t)0x00010000)
- #define ADC_SMPR2_SMP5_2 ((uint32_t)0x00020000)
- #define ADC_SMPR2_SMP6 ((uint32_t)0x001C0000)
- #define ADC_SMPR2_SMP6_0 ((uint32_t)0x00040000)
- #define ADC_SMPR2_SMP6_1 ((uint32_t)0x00080000)
- #define ADC_SMPR2_SMP6_2 ((uint32_t)0x00100000)
- #define ADC_SMPR2_SMP7 ((uint32_t)0x00E00000)
- #define ADC_SMPR2_SMP7_0 ((uint32_t)0x00200000)
- #define ADC_SMPR2_SMP7_1 ((uint32_t)0x00400000)
- #define ADC_SMPR2_SMP7_2 ((uint32_t)0x00800000)
- #define ADC_SMPR2_SMP8 ((uint32_t)0x07000000)
- #define ADC_SMPR2_SMP8_0 ((uint32_t)0x01000000)
- #define ADC_SMPR2_SMP8_1 ((uint32_t)0x02000000)
- #define ADC_SMPR2_SMP8_2 ((uint32_t)0x04000000)

- #define ADC_SMPR2_SMP9 ((uint32_t)0x38000000)
- #define ADC_SMPR2_SMP9_0 ((uint32_t)0x08000000)
- #define ADC_SMPR2_SMP9_1 ((uint32_t)0x10000000)
- #define ADC_SMPR2_SMP9_2 ((uint32_t)0x20000000)
- #define ADC_JOFR1_JOFFSET1 ((uint16_t)0x0FFF)
- #define ADC_JOFR2_JOFFSET2 ((uint16_t)0x0FFF)
- #define ADC_JOFR3_JOFFSET3 ((uint16_t)0x0FFF)
- #define ADC_JOFR4_JOFFSET4 ((uint16_t)0x0FFF)
- #define ADC_HTR_HT ((uint16_t)0x0FFF)
- #define ADC_LTR_LT ((uint16_t)0x0FFF)
- #define ADC_SQR1_SQ13 ((uint32_t)0x0000001F)
- #define ADC_SQR1_SQ13_0 ((uint32_t)0x00000001)
- #define ADC_SQR1_SQ13_1 ((uint32_t)0x00000002)
- #define ADC_SQR1_SQ13_2 ((uint32_t)0x00000004)
- #define ADC_SQR1_SQ13_3 ((uint32_t)0x00000008)
- #define ADC_SQR1_SQ13_4 ((uint32_t)0x00000010)
- #define ADC_SQR1_SQ14 ((uint32_t)0x000003E0)
- #define ADC_SQR1_SQ14_0 ((uint32_t)0x00000020)
- #define ADC_SQR1_SQ14_1 ((uint32_t)0x00000040)
- #define ADC_SQR1_SQ14_2 ((uint32_t)0x00000080)
- #define ADC_SQR1_SQ14_3 ((uint32_t)0x00000100)
- #define ADC_SQR1_SQ14_4 ((uint32_t)0x00000200)
- #define ADC_SQR1_SQ15 ((uint32_t)0x00007C00)
- #define ADC_SQR1_SQ15_0 ((uint32_t)0x00000400)
- #define ADC_SQR1_SQ15_1 ((uint32_t)0x00000800)
- #define ADC_SQR1_SQ15_2 ((uint32_t)0x00001000)
- #define ADC_SQR1_SQ15_3 ((uint32_t)0x00002000)
- #define ADC_SQR1_SQ15_4 ((uint32_t)0x00004000)
- #define ADC_SQR1_SQ16 ((uint32_t)0x000F8000)
- #define ADC_SQR1_SQ16_0 ((uint32_t)0x00008000)
- #define ADC_SQR1_SQ16_1 ((uint32_t)0x00010000)
- #define ADC_SQR1_SQ16_2 ((uint32_t)0x00020000)
- #define ADC_SQR1_SQ16_3 ((uint32_t)0x00040000)
- #define ADC_SQR1_SQ16_4 ((uint32_t)0x00080000)
- #define ADC_SQR1_L ((uint32_t)0x00F00000)
- #define ADC_SQR1_L_0 ((uint32_t)0x00100000)
- #define ADC_SQR1_L_1 ((uint32_t)0x00200000)
- #define ADC_SQR1_L_2 ((uint32_t)0x00400000)
- #define ADC_SQR1_L_3 ((uint32_t)0x00800000)
- #define ADC_SQR2_SQ7 ((uint32_t)0x0000001F)
- #define ADC_SQR2_SQ7_0 ((uint32_t)0x00000001)
- #define ADC_SQR2_SQ7_1 ((uint32_t)0x00000002)
- #define ADC_SQR2_SQ7_2 ((uint32_t)0x00000004)
- #define ADC_SQR2_SQ7_3 ((uint32_t)0x00000008)
- #define ADC_SQR2_SQ7_4 ((uint32_t)0x00000010)
- #define ADC_SQR2_SQ8 ((uint32_t)0x000003E0)
- #define ADC_SQR2_SQ8_0 ((uint32_t)0x00000020)
- #define ADC_SQR2_SQ8_1 ((uint32_t)0x00000040)
- #define ADC_SQR2_SQ8_2 ((uint32_t)0x00000080)
- #define ADC_SQR2_SQ8_3 ((uint32_t)0x00000100)
- #define ADC_SQR2_SQ8_4 ((uint32_t)0x00000200)
- #define ADC_SQR2_SQ9 ((uint32_t)0x00007C00)
- #define ADC_SQR2_SQ9_0 ((uint32_t)0x00000400)
- #define ADC_SQR2_SQ9_1 ((uint32_t)0x00000800)
- #define ADC_SQR2_SQ9_2 ((uint32_t)0x00001000)

- #define ADC_SQR2_SQ9_3 ((uint32_t)0x00002000)
- #define ADC_SQR2_SQ9_4 ((uint32_t)0x00004000)
- #define ADC_SQR2_SQ10 ((uint32_t)0x000F8000)
- #define ADC_SQR2_SQ10_0 ((uint32_t)0x00008000)
- #define ADC_SQR2_SQ10_1 ((uint32_t)0x00010000)
- #define ADC_SQR2_SQ10_2 ((uint32_t)0x00020000)
- #define ADC_SQR2_SQ10_3 ((uint32_t)0x00040000)
- #define ADC_SQR2_SQ10_4 ((uint32_t)0x00080000)
- #define ADC_SQR2_SQ11 ((uint32_t)0x01F00000)
- #define ADC_SQR2_SQ11_0 ((uint32_t)0x00100000)
- #define ADC_SQR2_SQ11_1 ((uint32_t)0x00200000)
- #define ADC_SQR2_SQ11_2 ((uint32_t)0x00400000)
- #define ADC_SQR2_SQ11_3 ((uint32_t)0x00800000)
- #define ADC_SQR2_SQ11_4 ((uint32_t)0x01000000)
- #define ADC_SQR2_SQ12 ((uint32_t)0x3E000000)
- #define ADC_SQR2_SQ12_0 ((uint32_t)0x02000000)
- #define ADC_SQR2_SQ12_1 ((uint32_t)0x04000000)
- #define ADC_SQR2_SQ12_2 ((uint32_t)0x08000000)
- #define ADC_SQR2_SQ12_3 ((uint32_t)0x10000000)
- #define ADC_SQR2_SQ12_4 ((uint32_t)0x20000000)
- #define ADC_SQR3_SQ1 ((uint32_t)0x0000001F)
- #define ADC_SQR3_SQ1_0 ((uint32_t)0x00000001)
- #define ADC_SQR3_SQ1_1 ((uint32_t)0x00000002)
- #define ADC_SQR3_SQ1_2 ((uint32_t)0x00000004)
- #define ADC_SQR3_SQ1_3 ((uint32_t)0x00000008)
- #define ADC_SQR3_SQ1_4 ((uint32_t)0x00000010)
- #define ADC_SQR3_SQ2 ((uint32_t)0x000003E0)
- #define ADC_SQR3_SQ2_0 ((uint32_t)0x00000020)
- #define ADC_SQR3_SQ2_1 ((uint32_t)0x00000040)
- #define ADC_SQR3_SQ2_2 ((uint32_t)0x00000080)
- #define ADC_SQR3_SQ2_3 ((uint32_t)0x00000100)
- #define ADC_SQR3_SQ2_4 ((uint32_t)0x00000200)
- #define ADC_SQR3_SQ3 ((uint32_t)0x00007C00)
- #define ADC_SQR3_SQ3_0 ((uint32_t)0x00000400)
- #define ADC_SQR3_SQ3_1 ((uint32_t)0x00000800)
- #define ADC_SQR3_SQ3_2 ((uint32_t)0x00001000)
- #define ADC_SQR3_SQ3_3 ((uint32_t)0x00002000)
- #define ADC_SQR3_SQ3_4 ((uint32_t)0x00004000)
- #define ADC_SQR3_SQ4 ((uint32_t)0x000F8000)
- #define ADC_SQR3_SQ4_0 ((uint32_t)0x00008000)
- #define ADC_SQR3_SQ4_1 ((uint32_t)0x00010000)
- #define ADC_SQR3_SQ4_2 ((uint32_t)0x00020000)
- #define ADC_SQR3_SQ4_3 ((uint32_t)0x00040000)
- #define ADC_SQR3_SQ4_4 ((uint32_t)0x00080000)
- #define ADC_SQR3_SQ5 ((uint32_t)0x01F00000)
- #define ADC_SQR3_SQ5_0 ((uint32_t)0x00100000)
- #define ADC_SQR3_SQ5_1 ((uint32_t)0x00200000)
- #define ADC_SQR3_SQ5_2 ((uint32_t)0x00400000)
- #define ADC_SQR3_SQ5_3 ((uint32_t)0x00800000)
- #define ADC_SQR3_SQ5_4 ((uint32_t)0x01000000)
- #define ADC_SQR3_SQ6 ((uint32_t)0x3E000000)
- #define ADC_SQR3_SQ6_0 ((uint32_t)0x02000000)
- #define ADC_SQR3_SQ6_1 ((uint32_t)0x04000000)
- #define ADC_SQR3_SQ6_2 ((uint32_t)0x08000000)
- #define ADC_SQR3_SQ6_3 ((uint32_t)0x10000000)

- #define ADC_SQR3_SQ6_4 ((uint32_t)0x20000000)
- #define ADC_JSQR_JSQ1 ((uint32_t)0x0000001F)
- #define ADC_JSQR_JSQ1_0 ((uint32_t)0x00000001)
- #define ADC_JSQR_JSQ1_1 ((uint32_t)0x00000002)
- #define ADC_JSQR_JSQ1_2 ((uint32_t)0x00000004)
- #define ADC_JSQR_JSQ1_3 ((uint32_t)0x00000008)
- #define ADC_JSQR_JSQ1_4 ((uint32_t)0x00000010)
- #define ADC_JSQR_JSQ2 ((uint32_t)0x000003E0)
- #define ADC_JSQR_JSQ2_0 ((uint32_t)0x00000020)
- #define ADC_JSQR_JSQ2_1 ((uint32_t)0x00000040)
- #define ADC_JSQR_JSQ2_2 ((uint32_t)0x00000080)
- #define ADC_JSQR_JSQ2_3 ((uint32_t)0x00000100)
- #define ADC_JSQR_JSQ2_4 ((uint32_t)0x00000200)
- #define ADC_JSQR_JSQ3 ((uint32_t)0x00007C00)
- #define ADC_JSQR_JSQ3_0 ((uint32_t)0x00000400)
- #define ADC_JSQR_JSQ3_1 ((uint32_t)0x00000800)
- #define ADC_JSQR_JSQ3_2 ((uint32_t)0x00001000)
- #define ADC_JSQR_JSQ3_3 ((uint32_t)0x00002000)
- #define ADC_JSQR_JSQ3_4 ((uint32_t)0x00004000)
- #define ADC_JSQR_JSQ4 ((uint32_t)0x000F8000)
- #define ADC_JSQR_JSQ4_0 ((uint32_t)0x00008000)
- #define ADC_JSQR_JSQ4_1 ((uint32_t)0x00010000)
- #define ADC_JSQR_JSQ4_2 ((uint32_t)0x00020000)
- #define ADC_JSQR_JSQ4_3 ((uint32_t)0x00040000)
- #define ADC_JSQR_JSQ4_4 ((uint32_t)0x00080000)
- #define ADC_JSQR_JL ((uint32_t)0x00300000)
- #define ADC_JSQR_JL_0 ((uint32_t)0x00100000)
- #define ADC_JSQR_JL_1 ((uint32_t)0x00200000)
- #define ADC_JDR1_JDATA ((uint16_t)0xFFFF)
- #define ADC_JDR2_JDATA ((uint16_t)0xFFFF)
- #define ADC_JDR3_JDATA ((uint16_t)0xFFFF)
- #define ADC_JDR4_JDATA ((uint16_t)0xFFFF)
- #define ADC_DR_DATA ((uint32_t)0x0000FFFF)
- #define ADC_DR_ADC2DATA ((uint32_t)0xFFFF0000)
- #define ADC_CSR_AWD1 ((uint32_t)0x00000001)
- #define ADC_CSR_EOC1 ((uint32_t)0x00000002)
- #define ADC_CSR_JEOC1 ((uint32_t)0x00000004)
- #define ADC_CSR_JSTRT1 ((uint32_t)0x00000008)
- #define ADC_CSR_STRT1 ((uint32_t)0x00000010)
- #define ADC_CSR_DOVR1 ((uint32_t)0x00000020)
- #define ADC_CSR_AWD2 ((uint32_t)0x00000100)
- #define ADC_CSR_EOC2 ((uint32_t)0x00000200)
- #define ADC_CSR_JEOC2 ((uint32_t)0x00000400)
- #define ADC_CSR_JSTRT2 ((uint32_t)0x00000800)
- #define ADC_CSR_STRT2 ((uint32_t)0x00001000)
- #define ADC_CSR_DOVR2 ((uint32_t)0x00002000)
- #define ADC_CSR_AWD3 ((uint32_t)0x00010000)
- #define ADC_CSR_EOC3 ((uint32_t)0x00020000)
- #define ADC_CSR_JEOC3 ((uint32_t)0x00040000)
- #define ADC_CSR_JSTRT3 ((uint32_t)0x00080000)
- #define ADC_CSR_STRT3 ((uint32_t)0x00100000)
- #define ADC_CSR_DOVR3 ((uint32_t)0x00200000)
- #define ADC_CCR_MULTI ((uint32_t)0x0000001F)
- #define ADC_CCR_MULTI_0 ((uint32_t)0x00000001)
- #define ADC_CCR_MULTI_1 ((uint32_t)0x00000002)

- #define ADC_CCR_MULTI_2 ((uint32_t)0x00000004)
- #define ADC_CCR_MULTI_3 ((uint32_t)0x00000008)
- #define ADC_CCR_MULTI_4 ((uint32_t)0x00000010)
- #define ADC_CCR_DELAY ((uint32_t)0x00000F00)
- #define ADC_CCR_DELAY_0 ((uint32_t)0x00000100)
- #define ADC_CCR_DELAY_1 ((uint32_t)0x00000200)
- #define ADC_CCR_DELAY_2 ((uint32_t)0x00000400)
- #define ADC_CCR_DELAY_3 ((uint32_t)0x00000800)
- #define ADC_CCR_DDS ((uint32_t)0x00002000)
- #define ADC_CCR_DMA ((uint32_t)0x0000C000)
- #define ADC_CCR_DMA_0 ((uint32_t)0x00004000)
- #define ADC_CCR_DMA_1 ((uint32_t)0x00008000)
- #define ADC_CCR_ADCPRE ((uint32_t)0x00030000)
- #define ADC_CCR_ADCPRE_0 ((uint32_t)0x00010000)
- #define ADC_CCR_ADCPRE_1 ((uint32_t)0x00020000)
- #define ADC_CCR_VBATE ((uint32_t)0x00400000)
- #define ADC_CCR_TSVREFE ((uint32_t)0x00800000)
- #define ADC_CDR_DATA1 ((uint32_t)0x0000FFFF)
- #define ADC_CDR_DATA2 ((uint32_t)0xFFFF0000)
- #define CAN_MCR_INRQ ((uint16_t)0x0001)
- #define CAN_MCR_SLEEP ((uint16_t)0x0002)
- #define CAN_MCR_TXFP ((uint16_t)0x0004)
- #define CAN_MCR_RFLM ((uint16_t)0x0008)
- #define CAN_MCR_NART ((uint16_t)0x0010)
- #define CAN_MCR_AWUM ((uint16_t)0x0020)
- #define CAN_MCR_ABOM ((uint16_t)0x0040)
- #define CAN_MCR_TTCM ((uint16_t)0x0080)
- #define CAN_MCR_RESET ((uint16_t)0x8000)
- #define CAN_MSR_INAK ((uint16_t)0x0001)
- #define CAN_MSR_SLAK ((uint16_t)0x0002)
- #define CAN_MSR_ERRI ((uint16_t)0x0004)
- #define CAN_MSR_WKUI ((uint16_t)0x0008)
- #define CAN_MSR_SLAKI ((uint16_t)0x0010)
- #define CAN_MSR_TXM ((uint16_t)0x0100)
- #define CAN_MSR_RXM ((uint16_t)0x0200)
- #define CAN_MSR_SAMP ((uint16_t)0x0400)
- #define CAN_MSR_RX ((uint16_t)0x0800)
- #define CAN_TSR_RQCP0 ((uint32_t)0x00000001)
- #define CAN_TSR_TXOK0 ((uint32_t)0x00000002)
- #define CAN_TSR_ALST0 ((uint32_t)0x00000004)
- #define CAN_TSR_TERR0 ((uint32_t)0x00000008)
- #define CAN_TSR_ABRQ0 ((uint32_t)0x00000080)
- #define CAN_TSR_RQCP1 ((uint32_t)0x00000100)
- #define CAN_TSR_TXOK1 ((uint32_t)0x00000200)
- #define CAN_TSR_ALST1 ((uint32_t)0x00000400)
- #define CAN_TSR_TERR1 ((uint32_t)0x00000800)
- #define CAN_TSR_ABRQ1 ((uint32_t)0x00008000)
- #define CAN_TSR_RQCP2 ((uint32_t)0x00010000)
- #define CAN_TSR_TXOK2 ((uint32_t)0x00020000)
- #define CAN_TSR_ALST2 ((uint32_t)0x00040000)
- #define CAN_TSR_TERR2 ((uint32_t)0x00080000)
- #define CAN_TSR_ABRQ2 ((uint32_t)0x00800000)
- #define CAN_TSR_CODE ((uint32_t)0x03000000)
- #define CAN_TSR_TME ((uint32_t)0x1C000000)
- #define CAN_TSR_TME0 ((uint32_t)0x04000000)

- #define CAN_TSR_TME1 ((uint32_t)0x08000000)
- #define CAN_TSR_TME2 ((uint32_t)0x10000000)
- #define CAN_TSR_LOW ((uint32_t)0xE0000000)
- #define CAN_TSR_LOW0 ((uint32_t)0x20000000)
- #define CAN_TSR_LOW1 ((uint32_t)0x40000000)
- #define CAN_TSR_LOW2 ((uint32_t)0x80000000)
- #define CAN_RF0R_FMP0 ((uint8_t)0x03)
- #define CAN_RF0R_FULL0 ((uint8_t)0x08)
- #define CAN_RF0R_FOVR0 ((uint8_t)0x10)
- #define CAN_RF0R_RFOM0 ((uint8_t)0x20)
- #define CAN_RF1R_FMP1 ((uint8_t)0x03)
- #define CAN_RF1R_FULL1 ((uint8_t)0x08)
- #define CAN_RF1R_FOVR1 ((uint8_t)0x10)
- #define CAN_RF1R_RFOM1 ((uint8_t)0x20)
- #define CAN_IER_TMEIE ((uint32_t)0x00000001)
- #define CAN_IER_FMPIE0 ((uint32_t)0x00000002)
- #define CAN_IER_FFIE0 ((uint32_t)0x00000004)
- #define CAN_IER_FOVIE0 ((uint32_t)0x00000008)
- #define CAN_IER_FMPIE1 ((uint32_t)0x00000010)
- #define CAN_IER_FFIE1 ((uint32_t)0x00000020)
- #define CAN_IER_FOVIE1 ((uint32_t)0x00000040)
- #define CAN_IER_EWGIE ((uint32_t)0x00000100)
- #define CAN_IER_EPVIE ((uint32_t)0x00000200)
- #define CAN_IER_BOFIE ((uint32_t)0x00000400)
- #define CAN_IER_LECIE ((uint32_t)0x00000800)
- #define CAN_IER_ERRIE ((uint32_t)0x00008000)
- #define CAN_IER_WKUIE ((uint32_t)0x00010000)
- #define CAN_IER_SLKIE ((uint32_t)0x00020000)
- #define CAN_ESR_EWGF ((uint32_t)0x00000001)
- #define CAN_ESR_EPVF ((uint32_t)0x00000002)
- #define CAN_ESR_BOFF ((uint32_t)0x00000004)
- #define CAN_ESR_LEC ((uint32_t)0x00000070)
- #define CAN_ESR_LEC_0 ((uint32_t)0x00000010)
- #define CAN_ESR_LEC_1 ((uint32_t)0x00000020)
- #define CAN_ESR_LEC_2 ((uint32_t)0x00000040)
- #define CAN_ESR_TEC ((uint32_t)0x00FF0000)
- #define CAN_ESR_REC ((uint32_t)0xFF000000)
- #define CAN_BTR_BRP ((uint32_t)0x000003FF)
- #define CAN_BTR_TS1 ((uint32_t)0x000F0000)
- #define CAN_BTR_TS2 ((uint32_t)0x00700000)
- #define CAN_BTR_SJW ((uint32_t)0x03000000)
- #define CAN_BTR_LBKM ((uint32_t)0x40000000)
- #define CAN_BTR_SILM ((uint32_t)0x80000000)
- #define CAN_TI0R_TXRQ ((uint32_t)0x00000001)
- #define CAN_TI0R_RTR ((uint32_t)0x00000002)
- #define CAN_TI0R_IDE ((uint32_t)0x00000004)
- #define CAN_TI0R_EXID ((uint32_t)0x001FFFF8)
- #define CAN_TI0R_STID ((uint32_t)0xFFE00000)
- #define CAN_TDT0R_DLC ((uint32_t)0x0000000F)
- #define CAN_TDT0R_TGT ((uint32_t)0x00000100)
- #define CAN_TDT0R_TIME ((uint32_t)0xFFFF0000)
- #define CAN_TDL0R_DATA0 ((uint32_t)0x000000FF)
- #define CAN_TDL0R_DATA1 ((uint32_t)0x0000FF00)
- #define CAN_TDL0R_DATA2 ((uint32_t)0x00FF0000)
- #define CAN_TDL0R_DATA3 ((uint32_t)0xFF000000)

- #define CAN_TDH0R_DATA4 ((uint32_t)0x000000FF)
- #define CAN_TDH0R_DATA5 ((uint32_t)0x0000FF00)
- #define CAN_TDH0R_DATA6 ((uint32_t)0x00FF0000)
- #define CAN_TDH0R_DATA7 ((uint32_t)0xFF000000)
- #define CAN_TI1R_TXRQ ((uint32_t)0x00000001)
- #define CAN_TI1R_RTR ((uint32_t)0x00000002)
- #define CAN_TI1R_IDE ((uint32_t)0x00000004)
- #define CAN_TI1R_EXID ((uint32_t)0x001FFFF8)
- #define CAN_TI1R_STID ((uint32_t)0xFFE00000)
- #define CAN_TDT1R_DLC ((uint32_t)0x0000000F)
- #define CAN_TDT1R_TGT ((uint32_t)0x00000100)
- #define CAN_TDT1R_TIME ((uint32_t)0xFFFF0000)
- #define CAN_TDL1R_DATA0 ((uint32_t)0x000000FF)
- #define CAN_TDL1R_DATA1 ((uint32_t)0x0000FF00)
- #define CAN_TDL1R_DATA2 ((uint32_t)0x00FF0000)
- #define CAN_TDL1R_DATA3 ((uint32_t)0xFF000000)
- #define CAN_TDH1R_DATA4 ((uint32_t)0x000000FF)
- #define CAN_TDH1R_DATA5 ((uint32_t)0x0000FF00)
- #define CAN_TDH1R_DATA6 ((uint32_t)0x00FF0000)
- #define CAN_TDH1R_DATA7 ((uint32_t)0xFF000000)
- #define CAN_TI2R_TXRQ ((uint32_t)0x00000001)
- #define CAN_TI2R_RTR ((uint32_t)0x00000002)
- #define CAN_TI2R_IDE ((uint32_t)0x00000004)
- #define CAN_TI2R_EXID ((uint32_t)0x001FFFF8)
- #define CAN_TI2R_STID ((uint32_t)0xFFE00000)
- #define CAN_TDT2R_DLC ((uint32_t)0x0000000F)
- #define CAN_TDT2R_TGT ((uint32_t)0x00000100)
- #define CAN_TDT2R_TIME ((uint32_t)0xFFFF0000)
- #define CAN_TDL2R_DATA0 ((uint32_t)0x000000FF)
- #define CAN_TDL2R_DATA1 ((uint32_t)0x0000FF00)
- #define CAN_TDL2R_DATA2 ((uint32_t)0x00FF0000)
- #define CAN_TDL2R_DATA3 ((uint32_t)0xFF000000)
- #define CAN_TDH2R_DATA4 ((uint32_t)0x000000FF)
- #define CAN_TDH2R_DATA5 ((uint32_t)0x0000FF00)
- #define CAN_TDH2R_DATA6 ((uint32_t)0x00FF0000)
- #define CAN_TDH2R_DATA7 ((uint32_t)0xFF000000)
- #define CAN_RI0R_RTR ((uint32_t)0x00000002)
- #define CAN_RI0R_IDE ((uint32_t)0x00000004)
- #define CAN_RI0R_EXID ((uint32_t)0x001FFFF8)
- #define CAN_RI0R_STID ((uint32_t)0xFFE00000)
- #define CAN_RDT0R_DLC ((uint32_t)0x0000000F)
- #define CAN_RDT0R_FMI ((uint32_t)0x0000FF00)
- #define CAN_RDT0R_TIME ((uint32_t)0xFFFF0000)
- #define CAN_RDL0R_DATA0 ((uint32_t)0x000000FF)
- #define CAN_RDL0R_DATA1 ((uint32_t)0x0000FF00)
- #define CAN_RDL0R_DATA2 ((uint32_t)0x00FF0000)
- #define CAN_RDL0R_DATA3 ((uint32_t)0xFF000000)
- #define CAN_RDH0R_DATA4 ((uint32_t)0x000000FF)
- #define CAN_RDH0R_DATA5 ((uint32_t)0x0000FF00)
- #define CAN_RDH0R_DATA6 ((uint32_t)0x00FF0000)
- #define CAN_RDH0R_DATA7 ((uint32_t)0xFF000000)
- #define CAN_RI1R_RTR ((uint32_t)0x00000002)
- #define CAN_RI1R_IDE ((uint32_t)0x00000004)
- #define CAN_RI1R_EXID ((uint32_t)0x001FFFF8)
- #define CAN_RI1R_STID ((uint32_t)0xFFE00000)

- #define CAN_RDT1R_DLC ((uint32_t)0x0000000F)
- #define CAN_RDT1R_FMI ((uint32_t)0x0000FF00)
- #define CAN_RDT1R_TIME ((uint32_t)0xFFFF0000)
- #define CAN_RDL1R_DATA0 ((uint32_t)0x000000FF)
- #define CAN_RDL1R_DATA1 ((uint32_t)0x0000FF00)
- #define CAN_RDL1R_DATA2 ((uint32_t)0x00FF0000)
- #define CAN_RDL1R_DATA3 ((uint32_t)0xFF000000)
- #define CAN_RDH1R_DATA4 ((uint32_t)0x000000FF)
- #define CAN_RDH1R_DATA5 ((uint32_t)0x0000FF00)
- #define CAN_RDH1R_DATA6 ((uint32_t)0x00FF0000)
- #define CAN_RDH1R_DATA7 ((uint32_t)0xFF000000)
- #define CAN_FMR_FINIT ((uint8_t)0x01)
- #define CAN_FM1R_FBM ((uint16_t)0x3FFF)
- #define CAN_FM1R_FBM0 ((uint16_t)0x0001)
- #define CAN_FM1R_FBM1 ((uint16_t)0x0002)
- #define CAN_FM1R_FBM2 ((uint16_t)0x0004)
- #define CAN_FM1R_FBM3 ((uint16_t)0x0008)
- #define CAN_FM1R_FBM4 ((uint16_t)0x0010)
- #define CAN_FM1R_FBM5 ((uint16_t)0x0020)
- #define CAN_FM1R_FBM6 ((uint16_t)0x0040)
- #define CAN_FM1R_FBM7 ((uint16_t)0x0080)
- #define CAN_FM1R_FBM8 ((uint16_t)0x0100)
- #define CAN_FM1R_FBM9 ((uint16_t)0x0200)
- #define CAN_FM1R_FBM10 ((uint16_t)0x0400)
- #define CAN_FM1R_FBM11 ((uint16_t)0x0800)
- #define CAN_FM1R_FBM12 ((uint16_t)0x1000)
- #define CAN_FM1R_FBM13 ((uint16_t)0x2000)
- #define CAN_FS1R_FSC ((uint16_t)0x3FFF)
- #define CAN_FS1R_FSC0 ((uint16_t)0x0001)
- #define CAN_FS1R_FSC1 ((uint16_t)0x0002)
- #define CAN_FS1R_FSC2 ((uint16_t)0x0004)
- #define CAN_FS1R_FSC3 ((uint16_t)0x0008)
- #define CAN_FS1R_FSC4 ((uint16_t)0x0010)
- #define CAN_FS1R_FSC5 ((uint16_t)0x0020)
- #define CAN_FS1R_FSC6 ((uint16_t)0x0040)
- #define CAN_FS1R_FSC7 ((uint16_t)0x0080)
- #define CAN_FS1R_FSC8 ((uint16_t)0x0100)
- #define CAN_FS1R_FSC9 ((uint16_t)0x0200)
- #define CAN_FS1R_FSC10 ((uint16_t)0x0400)
- #define CAN_FS1R_FSC11 ((uint16_t)0x0800)
- #define CAN_FS1R_FSC12 ((uint16_t)0x1000)
- #define CAN_FS1R_FSC13 ((uint16_t)0x2000)
- #define CAN_FFA1R_FFA ((uint16_t)0x3FFF)
- #define CAN_FFA1R_FFA0 ((uint16_t)0x0001)
- #define CAN_FFA1R_FFA1 ((uint16_t)0x0002)
- #define CAN_FFA1R_FFA2 ((uint16_t)0x0004)
- #define CAN_FFA1R_FFA3 ((uint16_t)0x0008)
- #define CAN_FFA1R_FFA4 ((uint16_t)0x0010)
- #define CAN_FFA1R_FFA5 ((uint16_t)0x0020)
- #define CAN_FFA1R_FFA6 ((uint16_t)0x0040)
- #define CAN_FFA1R_FFA7 ((uint16_t)0x0080)
- #define CAN_FFA1R_FFA8 ((uint16_t)0x0100)
- #define CAN_FFA1R_FFA9 ((uint16_t)0x0200)
- #define CAN_FFA1R_FFA10 ((uint16_t)0x0400)
- #define CAN_FFA1R_FFA11 ((uint16_t)0x0800)

- #define CAN_FFA1R_FFA12 ((uint16_t)0x1000)
- #define CAN_FFA1R_FFA13 ((uint16_t)0x2000)
- #define CAN_FA1R_FACT ((uint16_t)0x3FFF)
- #define CAN_FA1R_FACT0 ((uint16_t)0x0001)
- #define CAN_FA1R_FACT1 ((uint16_t)0x0002)
- #define CAN_FA1R_FACT2 ((uint16_t)0x0004)
- #define CAN_FA1R_FACT3 ((uint16_t)0x0008)
- #define CAN_FA1R_FACT4 ((uint16_t)0x0010)
- #define CAN_FA1R_FACT5 ((uint16_t)0x0020)
- #define CAN_FA1R_FACT6 ((uint16_t)0x0040)
- #define CAN_FA1R_FACT7 ((uint16_t)0x0080)
- #define CAN_FA1R_FACT8 ((uint16_t)0x0100)
- #define CAN_FA1R_FACT9 ((uint16_t)0x0200)
- #define CAN_FA1R_FACT10 ((uint16_t)0x0400)
- #define CAN_FA1R_FACT11 ((uint16_t)0x0800)
- #define CAN_FA1R_FACT12 ((uint16_t)0x1000)
- #define CAN_FA1R_FACT13 ((uint16_t)0x2000)
- #define CAN_F0R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F0R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F0R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F0R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F0R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F0R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F0R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F0R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F0R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F0R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F0R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F0R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F0R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F0R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F0R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F0R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F0R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F0R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F0R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F0R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F0R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F0R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F0R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F0R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F0R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F0R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F0R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F0R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F0R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F0R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F0R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F0R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F1R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F1R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F1R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F1R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F1R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F1R1_FB5 ((uint32_t)0x00000020)

- #define CAN_F1R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F1R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F1R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F1R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F1R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F1R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F1R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F1R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F1R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F1R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F1R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F1R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F1R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F1R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F1R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F1R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F1R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F1R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F1R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F1R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F1R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F1R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F1R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F1R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F1R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F1R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F2R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F2R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F2R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F2R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F2R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F2R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F2R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F2R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F2R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F2R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F2R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F2R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F2R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F2R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F2R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F2R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F2R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F2R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F2R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F2R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F2R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F2R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F2R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F2R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F2R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F2R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F2R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F2R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F2R1_FB28 ((uint32_t)0x10000000)

- #define CAN_F2R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F2R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F2R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F3R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F3R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F3R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F3R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F3R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F3R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F3R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F3R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F3R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F3R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F3R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F3R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F3R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F3R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F3R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F3R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F3R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F3R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F3R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F3R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F3R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F3R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F3R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F3R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F3R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F3R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F3R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F3R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F3R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F3R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F3R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F3R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F4R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F4R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F4R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F4R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F4R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F4R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F4R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F4R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F4R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F4R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F4R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F4R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F4R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F4R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F4R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F4R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F4R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F4R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F4R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F4R1_FB19 ((uint32_t)0x00080000)

- #define CAN_F4R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F4R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F4R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F4R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F4R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F4R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F4R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F4R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F4R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F4R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F4R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F4R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F5R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F5R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F5R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F5R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F5R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F5R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F5R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F5R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F5R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F5R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F5R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F5R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F5R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F5R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F5R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F5R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F5R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F5R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F5R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F5R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F5R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F5R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F5R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F5R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F5R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F5R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F5R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F5R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F5R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F5R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F5R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F5R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F6R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F6R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F6R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F6R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F6R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F6R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F6R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F6R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F6R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F6R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F6R1_FB10 ((uint32_t)0x00000400)

- #define CAN_F6R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F6R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F6R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F6R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F6R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F6R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F6R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F6R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F6R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F6R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F6R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F6R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F6R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F6R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F6R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F6R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F6R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F6R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F6R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F6R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F6R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F7R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F7R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F7R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F7R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F7R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F7R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F7R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F7R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F7R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F7R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F7R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F7R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F7R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F7R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F7R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F7R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F7R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F7R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F7R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F7R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F7R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F7R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F7R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F7R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F7R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F7R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F7R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F7R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F7R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F7R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F7R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F7R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F8R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F8R1_FB1 ((uint32_t)0x00000002)

- #define CAN_F8R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F8R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F8R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F8R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F8R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F8R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F8R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F8R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F8R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F8R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F8R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F8R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F8R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F8R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F8R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F8R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F8R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F8R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F8R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F8R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F8R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F8R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F8R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F8R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F8R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F8R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F8R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F8R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F8R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F8R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F9R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F9R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F9R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F9R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F9R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F9R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F9R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F9R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F9R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F9R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F9R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F9R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F9R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F9R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F9R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F9R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F9R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F9R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F9R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F9R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F9R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F9R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F9R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F9R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F9R1_FB24 ((uint32_t)0x01000000)

- #define CAN_F9R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F9R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F9R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F9R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F9R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F9R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F9R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F10R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F10R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F10R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F10R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F10R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F10R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F10R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F10R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F10R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F10R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F10R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F10R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F10R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F10R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F10R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F10R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F10R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F10R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F10R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F10R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F10R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F10R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F10R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F10R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F10R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F10R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F10R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F10R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F10R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F10R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F10R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F10R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F11R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F11R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F11R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F11R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F11R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F11R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F11R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F11R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F11R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F11R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F11R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F11R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F11R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F11R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F11R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F11R1_FB15 ((uint32_t)0x00008000)

- #define CAN_F11R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F11R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F11R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F11R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F11R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F11R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F11R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F11R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F11R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F11R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F11R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F11R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F11R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F11R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F11R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F11R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F12R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F12R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F12R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F12R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F12R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F12R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F12R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F12R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F12R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F12R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F12R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F12R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F12R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F12R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F12R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F12R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F12R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F12R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F12R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F12R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F12R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F12R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F12R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F12R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F12R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F12R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F12R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F12R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F12R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F12R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F12R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F12R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F13R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F13R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F13R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F13R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F13R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F13R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F13R1_FB6 ((uint32_t)0x00000040)

- #define CAN_F13R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F13R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F13R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F13R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F13R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F13R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F13R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F13R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F13R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F13R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F13R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F13R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F13R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F13R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F13R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F13R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F13R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F13R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F13R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F13R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F13R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F13R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F13R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F13R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F13R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F0R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F0R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F0R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F0R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F0R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F0R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F0R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F0R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F0R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F0R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F0R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F0R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F0R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F0R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F0R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F0R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F0R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F0R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F0R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F0R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F0R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F0R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F0R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F0R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F0R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F0R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F0R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F0R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F0R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F0R2_FB29 ((uint32_t)0x20000000)

- #define CAN_F0R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F0R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F1R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F1R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F1R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F1R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F1R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F1R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F1R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F1R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F1R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F1R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F1R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F1R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F1R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F1R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F1R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F1R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F1R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F1R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F1R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F1R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F1R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F1R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F1R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F1R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F1R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F1R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F1R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F1R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F1R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F1R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F1R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F1R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F2R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F2R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F2R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F2R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F2R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F2R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F2R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F2R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F2R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F2R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F2R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F2R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F2R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F2R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F2R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F2R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F2R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F2R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F2R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F2R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F2R2_FB20 ((uint32_t)0x00100000)

- #define CAN_F2R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F2R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F2R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F2R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F2R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F2R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F2R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F2R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F2R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F2R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F2R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F3R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F3R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F3R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F3R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F3R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F3R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F3R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F3R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F3R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F3R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F3R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F3R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F3R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F3R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F3R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F3R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F3R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F3R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F3R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F3R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F3R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F3R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F3R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F3R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F3R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F3R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F3R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F3R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F3R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F3R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F3R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F3R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F4R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F4R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F4R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F4R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F4R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F4R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F4R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F4R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F4R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F4R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F4R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F4R2_FB11 ((uint32_t)0x00000800)

- #define CAN_F4R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F4R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F4R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F4R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F4R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F4R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F4R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F4R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F4R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F4R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F4R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F4R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F4R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F4R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F4R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F4R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F4R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F4R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F4R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F4R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F5R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F5R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F5R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F5R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F5R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F5R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F5R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F5R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F5R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F5R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F5R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F5R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F5R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F5R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F5R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F5R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F5R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F5R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F5R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F5R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F5R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F5R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F5R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F5R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F5R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F5R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F5R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F5R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F5R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F5R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F5R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F5R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F6R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F6R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F6R2_FB2 ((uint32_t)0x00000004)

- #define CAN_F6R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F6R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F6R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F6R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F6R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F6R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F6R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F6R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F6R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F6R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F6R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F6R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F6R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F6R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F6R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F6R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F6R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F6R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F6R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F6R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F6R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F6R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F6R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F6R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F6R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F6R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F6R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F6R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F6R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F7R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F7R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F7R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F7R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F7R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F7R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F7R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F7R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F7R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F7R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F7R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F7R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F7R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F7R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F7R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F7R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F7R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F7R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F7R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F7R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F7R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F7R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F7R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F7R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F7R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F7R2_FB25 ((uint32_t)0x02000000)

- #define CAN_F7R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F7R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F7R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F7R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F7R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F7R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F8R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F8R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F8R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F8R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F8R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F8R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F8R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F8R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F8R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F8R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F8R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F8R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F8R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F8R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F8R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F8R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F8R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F8R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F8R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F8R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F8R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F8R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F8R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F8R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F8R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F8R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F8R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F8R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F8R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F8R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F8R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F8R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F9R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F9R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F9R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F9R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F9R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F9R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F9R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F9R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F9R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F9R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F9R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F9R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F9R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F9R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F9R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F9R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F9R2_FB16 ((uint32_t)0x00010000)

- #define CAN_F9R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F9R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F9R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F9R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F9R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F9R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F9R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F9R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F9R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F9R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F9R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F9R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F9R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F9R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F9R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F10R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F10R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F10R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F10R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F10R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F10R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F10R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F10R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F10R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F10R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F10R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F10R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F10R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F10R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F10R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F10R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F10R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F10R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F10R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F10R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F10R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F10R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F10R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F10R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F10R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F10R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F10R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F10R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F10R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F10R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F10R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F10R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F11R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F11R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F11R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F11R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F11R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F11R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F11R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F11R2_FB7 ((uint32_t)0x00000080)

- #define CAN_F11R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F11R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F11R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F11R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F11R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F11R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F11R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F11R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F11R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F11R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F11R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F11R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F11R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F11R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F11R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F11R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F11R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F11R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F11R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F11R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F11R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F11R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F11R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F11R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F12R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F12R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F12R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F12R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F12R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F12R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F12R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F12R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F12R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F12R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F12R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F12R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F12R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F12R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F12R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F12R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F12R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F12R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F12R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F12R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F12R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F12R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F12R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F12R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F12R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F12R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F12R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F12R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F12R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F12R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F12R2_FB30 ((uint32_t)0x40000000)

- #define CAN_F12R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F13R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F13R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F13R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F13R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F13R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F13R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F13R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F13R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F13R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F13R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F13R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F13R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F13R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F13R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F13R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F13R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F13R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F13R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F13R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F13R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F13R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F13R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F13R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F13R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F13R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F13R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F13R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F13R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F13R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F13R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F13R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F13R2_FB31 ((uint32_t)0x80000000)
- #define CRC_DR_DR ((uint32_t)0xFFFFFFFF)
- #define CRC_IDR_IDR ((uint8_t)0xFF)
- #define CRC_CR_RESET ((uint8_t)0x01)
- #define **CRYP_CR_ALGODIR** ((uint32_t)0x00000004)
- #define **CRYP_CR_ALGOMODE** ((uint32_t)0x00000038)
- #define **CRYP_CR_ALGOMODE_0** ((uint32_t)0x00000008)
- #define **CRYP_CR_ALGOMODE_1** ((uint32_t)0x00000010)
- #define **CRYP_CR_ALGOMODE_2** ((uint32_t)0x00000020)
- #define **CRYP_CR_ALGOMODE_TDES_ECB** ((uint32_t)0x00000000)
- #define **CRYP_CR_ALGOMODE_TDES_CBC** ((uint32_t)0x00000008)
- #define **CRYP_CR_ALGOMODE_DES_ECB** ((uint32_t)0x00000010)
- #define **CRYP_CR_ALGOMODE_DES_CBC** ((uint32_t)0x00000018)
- #define **CRYP_CR_ALGOMODE_AES_ECB** ((uint32_t)0x00000020)
- #define **CRYP_CR_ALGOMODE_AES_CBC** ((uint32_t)0x00000028)
- #define **CRYP_CR_ALGOMODE_AES_CTR** ((uint32_t)0x00000030)
- #define **CRYP_CR_ALGOMODE_AES_KEY** ((uint32_t)0x00000038)
- #define **CRYP_CR_DATATYPE** ((uint32_t)0x000000C0)
- #define **CRYP_CR_DATATYPE_0** ((uint32_t)0x00000040)
- #define **CRYP_CR_DATATYPE_1** ((uint32_t)0x00000080)
- #define **CRYP_CR_KEYSIZE** ((uint32_t)0x00000300)
- #define **CRYP_CR_KEYSIZE_0** ((uint32_t)0x00000100)
- #define **CRYP_CR_KEYSIZE_1** ((uint32_t)0x00000200)

- #define **CRYP_CR_FFLUSH** ((uint32_t)0x00004000)
- #define **CRYP_CR_CRYPEN** ((uint32_t)0x00008000)
- #define **CRYP_SR_IFEM** ((uint32_t)0x00000001)
- #define **CRYP_SR_IFNF** ((uint32_t)0x00000002)
- #define **CRYP_SR_OFNE** ((uint32_t)0x00000004)
- #define **CRYP_SR_OFFU** ((uint32_t)0x00000008)
- #define **CRYP_SR_BUSY** ((uint32_t)0x00000010)
- #define **CRYP_DMACR_DIEN** ((uint32_t)0x00000001)
- #define **CRYP_DMACR_DOEN** ((uint32_t)0x00000002)
- #define **CRYP_IMSCR_INIM** ((uint32_t)0x00000001)
- #define **CRYP_IMSCR_OUTIM** ((uint32_t)0x00000002)
- #define **CRYP_RISR_OUTRIS** ((uint32_t)0x00000001)
- #define **CRYP_RISR_INRIS** ((uint32_t)0x00000002)
- #define **CRYP_MISR_INMIS** ((uint32_t)0x00000001)
- #define **CRYP_MISR_OUTMIS** ((uint32_t)0x00000002)
- #define DAC_CR_EN1 ((uint32_t)0x00000001)
- #define DAC_CR_BOFF1 ((uint32_t)0x00000002)
- #define DAC_CR_TEN1 ((uint32_t)0x00000004)
- #define DAC_CR_TSEL1 ((uint32_t)0x00000038)
- #define DAC_CR_TSEL1_0 ((uint32_t)0x00000008)
- #define DAC_CR_TSEL1_1 ((uint32_t)0x00000010)
- #define DAC_CR_TSEL1_2 ((uint32_t)0x00000020)
- #define DAC_CR_WAVE1 ((uint32_t)0x000000C0)
- #define DAC_CR_WAVE1_0 ((uint32_t)0x00000040)
- #define DAC_CR_WAVE1_1 ((uint32_t)0x00000080)
- #define DAC_CR_MAMP1 ((uint32_t)0x00000F00)
- #define DAC_CR_MAMP1_0 ((uint32_t)0x00000100)
- #define DAC_CR_MAMP1_1 ((uint32_t)0x00000200)
- #define DAC_CR_MAMP1_2 ((uint32_t)0x00000400)
- #define DAC_CR_MAMP1_3 ((uint32_t)0x00000800)
- #define DAC_CR_DMAEN1 ((uint32_t)0x00001000)
- #define DAC_CR_EN2 ((uint32_t)0x00010000)
- #define DAC_CR_BOFF2 ((uint32_t)0x00020000)
- #define DAC_CR_TEN2 ((uint32_t)0x00040000)
- #define DAC_CR_TSEL2 ((uint32_t)0x00380000)
- #define DAC_CR_TSEL2_0 ((uint32_t)0x00080000)
- #define DAC_CR_TSEL2_1 ((uint32_t)0x00100000)
- #define DAC_CR_TSEL2_2 ((uint32_t)0x00200000)
- #define DAC_CR_WAVE2 ((uint32_t)0x00C00000)
- #define DAC_CR_WAVE2_0 ((uint32_t)0x00400000)
- #define DAC_CR_WAVE2_1 ((uint32_t)0x00800000)
- #define DAC_CR_MAMP2 ((uint32_t)0x0F000000)
- #define DAC_CR_MAMP2_0 ((uint32_t)0x01000000)
- #define DAC_CR_MAMP2_1 ((uint32_t)0x02000000)
- #define DAC_CR_MAMP2_2 ((uint32_t)0x04000000)
- #define DAC_CR_MAMP2_3 ((uint32_t)0x08000000)
- #define DAC_CR_DMAEN2 ((uint32_t)0x10000000)
- #define DAC_SWTRIGR_SWTRIG1 ((uint8_t)0x01)
- #define DAC_SWTRIGR_SWTRIG2 ((uint8_t)0x02)
- #define DAC_DHR12R1_DACC1DHR ((uint16_t)0x0FFF)
- #define DAC_DHR12L1_DACC1DHR ((uint16_t)0xFFF0)
- #define DAC_DHR8R1_DACC1DHR ((uint8_t)0xFF)
- #define DAC_DHR12R2_DACC2DHR ((uint16_t)0x0FFF)
- #define DAC_DHR12L2_DACC2DHR ((uint16_t)0xFFF0)
- #define DAC_DHR8R2_DACC2DHR ((uint8_t)0xFF)

- #define DAC_DHR12RD_DACC1DHR ((uint32_t)0x00000FFF)
- #define DAC_DHR12RD_DACC2DHR ((uint32_t)0x0FFF0000)
- #define DAC_DHR12LD_DACC1DHR ((uint32_t)0x0000FFF0)
- #define DAC_DHR12LD_DACC2DHR ((uint32_t)0xFFF00000)
- #define DAC_DHR8RD_DACC1DHR ((uint16_t)0x00FF)
- #define DAC_DHR8RD_DACC2DHR ((uint16_t)0xFF00)
- #define DAC_DOR1_DACC1DOR ((uint16_t)0x0FFF)
- #define DAC_DOR2_DACC2DOR ((uint16_t)0x0FFF)
- #define DAC_SR_DMAUDR1 ((uint32_t)0x00002000)
- #define DAC_SR_DMAUDR2 ((uint32_t)0x20000000)
- #define **DCMI_CR_CAPTURE** ((uint32_t)0x00000001)
- #define **DCMI_CR_CM** ((uint32_t)0x00000002)
- #define **DCMI_CR_CROP** ((uint32_t)0x00000004)
- #define **DCMI_CR_JPEG** ((uint32_t)0x00000008)
- #define **DCMI_CR_ESS** ((uint32_t)0x00000010)
- #define **DCMI_CR_PCKPOL** ((uint32_t)0x00000020)
- #define **DCMI_CR_HSPOL** ((uint32_t)0x00000040)
- #define **DCMI_CR_VSPOL** ((uint32_t)0x00000080)
- #define **DCMI_CR_FCRC_0** ((uint32_t)0x00000100)
- #define **DCMI_CR_FCRC_1** ((uint32_t)0x00000200)
- #define **DCMI_CR_EDM_0** ((uint32_t)0x00000400)
- #define **DCMI_CR_EDM_1** ((uint32_t)0x00000800)
- #define **DCMI_CR_CRE** ((uint32_t)0x00001000)
- #define **DCMI_CR_ENABLE** ((uint32_t)0x00004000)
- #define **DCMI_SR_HSYNC** ((uint32_t)0x00000001)
- #define **DCMI_SR_VSYNC** ((uint32_t)0x00000002)
- #define **DCMI_SR_FNE** ((uint32_t)0x00000004)
- #define **DCMI_RISR_FRAME_RIS** ((uint32_t)0x00000001)
- #define **DCMI_RISR_OVF_RIS** ((uint32_t)0x00000002)
- #define **DCMI_RISR_ERR_RIS** ((uint32_t)0x00000004)
- #define **DCMI_RISR_VSYNC_RIS** ((uint32_t)0x00000008)
- #define **DCMI_RISR_LINE_RIS** ((uint32_t)0x00000010)
- #define **DCMI_IER_FRAME_IE** ((uint32_t)0x00000001)
- #define **DCMI_IER_OVF_IE** ((uint32_t)0x00000002)
- #define **DCMI_IER_ERR_IE** ((uint32_t)0x00000004)
- #define **DCMI_IER_VSYNC_IE** ((uint32_t)0x00000008)
- #define **DCMI_IER_LINE_IE** ((uint32_t)0x00000010)
- #define **DCMI_MISR_FRAME_MIS** ((uint32_t)0x00000001)
- #define **DCMI_MISR_OVF_MIS** ((uint32_t)0x00000002)
- #define **DCMI_MISR_ERR_MIS** ((uint32_t)0x00000004)
- #define **DCMI_MISR_VSYNC_MIS** ((uint32_t)0x00000008)
- #define **DCMI_MISR_LINE_MIS** ((uint32_t)0x00000010)
- #define **DCMI_ICR_FRAME_ISC** ((uint32_t)0x00000001)
- #define **DCMI_ICR_OVF_ISC** ((uint32_t)0x00000002)
- #define **DCMI_ICR_ERR_ISC** ((uint32_t)0x00000004)
- #define **DCMI_ICR_VSYNC_ISC** ((uint32_t)0x00000008)
- #define **DCMI_ICR_LINE_ISC** ((uint32_t)0x00000010)
- #define **DMA_SxCR_CHSEL** ((uint32_t)0x0E000000)
- #define **DMA_SxCR_CHSEL_0** ((uint32_t)0x02000000)
- #define **DMA_SxCR_CHSEL_1** ((uint32_t)0x04000000)
- #define **DMA_SxCR_CHSEL_2** ((uint32_t)0x08000000)
- #define **DMA_SxCR_MBURST** ((uint32_t)0x01800000)
- #define **DMA_SxCR_MBURST_0** ((uint32_t)0x00800000)
- #define **DMA_SxCR_MBURST_1** ((uint32_t)0x01000000)
- #define **DMA_SxCR_PBURST** ((uint32_t)0x00600000)

- #define **DMA_SxCR_PBURST_0** ((uint32_t)0x00200000)
- #define **DMA_SxCR_PBURST_1** ((uint32_t)0x00400000)
- #define **DMA_SxCR_ACK** ((uint32_t)0x00100000)
- #define **DMA_SxCR_CT** ((uint32_t)0x00080000)
- #define **DMA_SxCR_DBM** ((uint32_t)0x00040000)
- #define **DMA_SxCR_PL** ((uint32_t)0x00030000)
- #define **DMA_SxCR_PL_0** ((uint32_t)0x00010000)
- #define **DMA_SxCR_PL_1** ((uint32_t)0x00020000)
- #define **DMA_SxCR_PINCOS** ((uint32_t)0x00008000)
- #define **DMA_SxCR_MSIZE** ((uint32_t)0x00006000)
- #define **DMA_SxCR_MSIZE_0** ((uint32_t)0x00002000)
- #define **DMA_SxCR_MSIZE_1** ((uint32_t)0x00004000)
- #define **DMA_SxCR_PSIZE** ((uint32_t)0x00001800)
- #define **DMA_SxCR_PSIZE_0** ((uint32_t)0x00000800)
- #define **DMA_SxCR_PSIZE_1** ((uint32_t)0x00001000)
- #define **DMA_SxCR_MINC** ((uint32_t)0x00000400)
- #define **DMA_SxCR_PINC** ((uint32_t)0x00000200)
- #define **DMA_SxCR_CIRC** ((uint32_t)0x00000100)
- #define **DMA_SxCR_DIR** ((uint32_t)0x000000C0)
- #define **DMA_SxCR_DIR_0** ((uint32_t)0x00000040)
- #define **DMA_SxCR_DIR_1** ((uint32_t)0x00000080)
- #define **DMA_SxCR_PFCTRL** ((uint32_t)0x00000020)
- #define **DMA_SxCR_TCIE** ((uint32_t)0x00000010)
- #define **DMA_SxCR_HTIE** ((uint32_t)0x00000008)
- #define **DMA_SxCR_TEIE** ((uint32_t)0x00000004)
- #define **DMA_SxCR_DMEIE** ((uint32_t)0x00000002)
- #define **DMA_SxCR_EN** ((uint32_t)0x00000001)
- #define **DMA_SxNDT** ((uint32_t)0x0000FFFF)
- #define **DMA_SxNDT_0** ((uint32_t)0x00000001)
- #define **DMA_SxNDT_1** ((uint32_t)0x00000002)
- #define **DMA_SxNDT_2** ((uint32_t)0x00000004)
- #define **DMA_SxNDT_3** ((uint32_t)0x00000008)
- #define **DMA_SxNDT_4** ((uint32_t)0x00000010)
- #define **DMA_SxNDT_5** ((uint32_t)0x00000020)
- #define **DMA_SxNDT_6** ((uint32_t)0x00000040)
- #define **DMA_SxNDT_7** ((uint32_t)0x00000080)
- #define **DMA_SxNDT_8** ((uint32_t)0x00000100)
- #define **DMA_SxNDT_9** ((uint32_t)0x00000200)
- #define **DMA_SxNDT_10** ((uint32_t)0x00000400)
- #define **DMA_SxNDT_11** ((uint32_t)0x00000800)
- #define **DMA_SxNDT_12** ((uint32_t)0x00001000)
- #define **DMA_SxNDT_13** ((uint32_t)0x00002000)
- #define **DMA_SxNDT_14** ((uint32_t)0x00004000)
- #define **DMA_SxNDT_15** ((uint32_t)0x00008000)
- #define **DMA_SxFCR_FEIE** ((uint32_t)0x00000080)
- #define **DMA_SxFCR_FS** ((uint32_t)0x00000038)
- #define **DMA_SxFCR_FS_0** ((uint32_t)0x00000008)
- #define **DMA_SxFCR_FS_1** ((uint32_t)0x00000010)
- #define **DMA_SxFCR_FS_2** ((uint32_t)0x00000020)
- #define **DMA_SxFCR_DMDIS** ((uint32_t)0x00000004)
- #define **DMA_SxFCR_FTH** ((uint32_t)0x00000003)
- #define **DMA_SxFCR_FTH_0** ((uint32_t)0x00000001)
- #define **DMA_SxFCR_FTH_1** ((uint32_t)0x00000002)
- #define **DMA_LISR_TCIF3** ((uint32_t)0x08000000)
- #define **DMA_LISR_HTIF3** ((uint32_t)0x04000000)

- #define **DMA_LISR_TEIF3** ((uint32_t)0x02000000)
- #define **DMA_LISR_DMEIF3** ((uint32_t)0x01000000)
- #define **DMA_LISR_FEIF3** ((uint32_t)0x00400000)
- #define **DMA_LISR_TCIF2** ((uint32_t)0x00200000)
- #define **DMA_LISR_HTIF2** ((uint32_t)0x00100000)
- #define **DMA_LISR_TEIF2** ((uint32_t)0x00080000)
- #define **DMA_LISR_DMEIF2** ((uint32_t)0x00040000)
- #define **DMA_LISR_FEIF2** ((uint32_t)0x00010000)
- #define **DMA_LISR_TCIF1** ((uint32_t)0x00000800)
- #define **DMA_LISR_HTIF1** ((uint32_t)0x00000400)
- #define **DMA_LISR_TEIF1** ((uint32_t)0x00000200)
- #define **DMA_LISR_DMEIF1** ((uint32_t)0x00000100)
- #define **DMA_LISR_FEIF1** ((uint32_t)0x00000040)
- #define **DMA_LISR_TCIF0** ((uint32_t)0x00000020)
- #define **DMA_LISR_HTIF0** ((uint32_t)0x00000010)
- #define **DMA_LISR_TEIF0** ((uint32_t)0x00000008)
- #define **DMA_LISR_DMEIF0** ((uint32_t)0x00000004)
- #define **DMA_LISR_FEIF0** ((uint32_t)0x00000001)
- #define **DMA_HISR_TCIF7** ((uint32_t)0x08000000)
- #define **DMA_HISR_HTIF7** ((uint32_t)0x04000000)
- #define **DMA_HISR_TEIF7** ((uint32_t)0x02000000)
- #define **DMA_HISR_DMEIF7** ((uint32_t)0x01000000)
- #define **DMA_HISR_FEIF7** ((uint32_t)0x00400000)
- #define **DMA_HISR_TCIF6** ((uint32_t)0x00200000)
- #define **DMA_HISR_HTIF6** ((uint32_t)0x00100000)
- #define **DMA_HISR_TEIF6** ((uint32_t)0x00080000)
- #define **DMA_HISR_DMEIF6** ((uint32_t)0x00040000)
- #define **DMA_HISR_FEIF6** ((uint32_t)0x00010000)
- #define **DMA_HISR_TCIF5** ((uint32_t)0x00000800)
- #define **DMA_HISR_HTIF5** ((uint32_t)0x00000400)
- #define **DMA_HISR_TEIF5** ((uint32_t)0x00000200)
- #define **DMA_HISR_DMEIF5** ((uint32_t)0x00000100)
- #define **DMA_HISR_FEIF5** ((uint32_t)0x00000040)
- #define **DMA_HISR_TCIF4** ((uint32_t)0x00000020)
- #define **DMA_HISR_HTIF4** ((uint32_t)0x00000010)
- #define **DMA_HISR_TEIF4** ((uint32_t)0x00000008)
- #define **DMA_HISR_DMEIF4** ((uint32_t)0x00000004)
- #define **DMA_HISR_FEIF4** ((uint32_t)0x00000001)
- #define **DMA_LIFCR_CTCIF3** ((uint32_t)0x08000000)
- #define **DMA_LIFCR_CHTIF3** ((uint32_t)0x04000000)
- #define **DMA_LIFCR_CTEIF3** ((uint32_t)0x02000000)
- #define **DMA_LIFCR_CDMEIF3** ((uint32_t)0x01000000)
- #define **DMA_LIFCR_CFEIF3** ((uint32_t)0x00400000)
- #define **DMA_LIFCR_CTCIF2** ((uint32_t)0x00200000)
- #define **DMA_LIFCR_CHTIF2** ((uint32_t)0x00100000)
- #define **DMA_LIFCR_CTEIF2** ((uint32_t)0x00080000)
- #define **DMA_LIFCR_CDMEIF2** ((uint32_t)0x00040000)
- #define **DMA_LIFCR_CFEIF2** ((uint32_t)0x00010000)
- #define **DMA_LIFCR_CTCIF1** ((uint32_t)0x00000800)
- #define **DMA_LIFCR_CHTIF1** ((uint32_t)0x00000400)
- #define **DMA_LIFCR_CTEIF1** ((uint32_t)0x00000200)
- #define **DMA_LIFCR_CDMEIF1** ((uint32_t)0x00000100)
- #define **DMA_LIFCR_CFEIF1** ((uint32_t)0x00000040)
- #define **DMA_LIFCR_CTCIF0** ((uint32_t)0x00000020)
- #define **DMA_LIFCR_CHTIF0** ((uint32_t)0x00000010)

- #define **DMA_LIFCR_CTEIF0** ((uint32_t)0x00000008)
- #define **DMA_LIFCR_CDMEIF0** ((uint32_t)0x00000004)
- #define **DMA_LIFCR_CFEIF0** ((uint32_t)0x00000001)
- #define **DMA_HIFCR_CTCIF7** ((uint32_t)0x08000000)
- #define **DMA_HIFCR_CHTIF7** ((uint32_t)0x04000000)
- #define **DMA_HIFCR_CTEIF7** ((uint32_t)0x02000000)
- #define **DMA_HIFCR_CDMEIF7** ((uint32_t)0x01000000)
- #define **DMA_HIFCR_CFEIF7** ((uint32_t)0x00400000)
- #define **DMA_HIFCR_CTCIF6** ((uint32_t)0x00200000)
- #define **DMA_HIFCR_CHTIF6** ((uint32_t)0x00100000)
- #define **DMA_HIFCR_CTEIF6** ((uint32_t)0x00080000)
- #define **DMA_HIFCR_CDMEIF6** ((uint32_t)0x00040000)
- #define **DMA_HIFCR_CFEIF6** ((uint32_t)0x00010000)
- #define **DMA_HIFCR_CTCIF5** ((uint32_t)0x00000800)
- #define **DMA_HIFCR_CHTIF5** ((uint32_t)0x00000400)
- #define **DMA_HIFCR_CTEIF5** ((uint32_t)0x00000200)
- #define **DMA_HIFCR_CDMEIF5** ((uint32_t)0x00000100)
- #define **DMA_HIFCR_CFEIF5** ((uint32_t)0x00000040)
- #define **DMA_HIFCR_CTCIF4** ((uint32_t)0x00000020)
- #define **DMA_HIFCR_CHTIF4** ((uint32_t)0x00000010)
- #define **DMA_HIFCR_CTEIF4** ((uint32_t)0x00000008)
- #define **DMA_HIFCR_CDMEIF4** ((uint32_t)0x00000004)
- #define **DMA_HIFCR_CFEIF4** ((uint32_t)0x00000001)
- #define EXTI_IMR_MR0 ((uint32_t)0x00000001)
- #define EXTI_IMR_MR1 ((uint32_t)0x00000002)
- #define EXTI_IMR_MR2 ((uint32_t)0x00000004)
- #define EXTI_IMR_MR3 ((uint32_t)0x00000008)
- #define EXTI_IMR_MR4 ((uint32_t)0x00000010)
- #define EXTI_IMR_MR5 ((uint32_t)0x00000020)
- #define EXTI_IMR_MR6 ((uint32_t)0x00000040)
- #define EXTI_IMR_MR7 ((uint32_t)0x00000080)
- #define EXTI_IMR_MR8 ((uint32_t)0x00000100)
- #define EXTI_IMR_MR9 ((uint32_t)0x00000200)
- #define EXTI_IMR_MR10 ((uint32_t)0x00000400)
- #define EXTI_IMR_MR11 ((uint32_t)0x00000800)
- #define EXTI_IMR_MR12 ((uint32_t)0x00001000)
- #define EXTI_IMR_MR13 ((uint32_t)0x00002000)
- #define EXTI_IMR_MR14 ((uint32_t)0x00004000)
- #define EXTI_IMR_MR15 ((uint32_t)0x00008000)
- #define EXTI_IMR_MR16 ((uint32_t)0x00010000)
- #define EXTI_IMR_MR17 ((uint32_t)0x00020000)
- #define EXTI_IMR_MR18 ((uint32_t)0x00040000)
- #define EXTI_IMR_MR19 ((uint32_t)0x00080000)
- #define EXTI_EMR_MR0 ((uint32_t)0x00000001)
- #define EXTI_EMR_MR1 ((uint32_t)0x00000002)
- #define EXTI_EMR_MR2 ((uint32_t)0x00000004)
- #define EXTI_EMR_MR3 ((uint32_t)0x00000008)
- #define EXTI_EMR_MR4 ((uint32_t)0x00000010)
- #define EXTI_EMR_MR5 ((uint32_t)0x00000020)
- #define EXTI_EMR_MR6 ((uint32_t)0x00000040)
- #define EXTI_EMR_MR7 ((uint32_t)0x00000080)
- #define EXTI_EMR_MR8 ((uint32_t)0x00000100)
- #define EXTI_EMR_MR9 ((uint32_t)0x00000200)
- #define EXTI_EMR_MR10 ((uint32_t)0x00000400)
- #define EXTI_EMR_MR11 ((uint32_t)0x00000800)

- #define EXTI_EMR_MR12 ((uint32_t)0x00001000)
- #define EXTI_EMR_MR13 ((uint32_t)0x00002000)
- #define EXTI_EMR_MR14 ((uint32_t)0x00004000)
- #define EXTI_EMR_MR15 ((uint32_t)0x00008000)
- #define EXTI_EMR_MR16 ((uint32_t)0x00010000)
- #define EXTI_EMR_MR17 ((uint32_t)0x00020000)
- #define EXTI_EMR_MR18 ((uint32_t)0x00040000)
- #define EXTI_EMR_MR19 ((uint32_t)0x00080000)
- #define EXTI_RTSR_TR0 ((uint32_t)0x00000001)
- #define EXTI_RTSR_TR1 ((uint32_t)0x00000002)
- #define EXTI_RTSR_TR2 ((uint32_t)0x00000004)
- #define EXTI_RTSR_TR3 ((uint32_t)0x00000008)
- #define EXTI_RTSR_TR4 ((uint32_t)0x00000010)
- #define EXTI_RTSR_TR5 ((uint32_t)0x00000020)
- #define EXTI_RTSR_TR6 ((uint32_t)0x00000040)
- #define EXTI_RTSR_TR7 ((uint32_t)0x00000080)
- #define EXTI_RTSR_TR8 ((uint32_t)0x00000100)
- #define EXTI_RTSR_TR9 ((uint32_t)0x00000200)
- #define EXTI_RTSR_TR10 ((uint32_t)0x00000400)
- #define EXTI_RTSR_TR11 ((uint32_t)0x00000800)
- #define EXTI_RTSR_TR12 ((uint32_t)0x00001000)
- #define EXTI_RTSR_TR13 ((uint32_t)0x00002000)
- #define EXTI_RTSR_TR14 ((uint32_t)0x00004000)
- #define EXTI_RTSR_TR15 ((uint32_t)0x00008000)
- #define EXTI_RTSR_TR16 ((uint32_t)0x00010000)
- #define EXTI_RTSR_TR17 ((uint32_t)0x00020000)
- #define EXTI_RTSR_TR18 ((uint32_t)0x00040000)
- #define EXTI_RTSR_TR19 ((uint32_t)0x00080000)
- #define EXTI_FTSR_TR0 ((uint32_t)0x00000001)
- #define EXTI_FTSR_TR1 ((uint32_t)0x00000002)
- #define EXTI_FTSR_TR2 ((uint32_t)0x00000004)
- #define EXTI_FTSR_TR3 ((uint32_t)0x00000008)
- #define EXTI_FTSR_TR4 ((uint32_t)0x00000010)
- #define EXTI_FTSR_TR5 ((uint32_t)0x00000020)
- #define EXTI_FTSR_TR6 ((uint32_t)0x00000040)
- #define EXTI_FTSR_TR7 ((uint32_t)0x00000080)
- #define EXTI_FTSR_TR8 ((uint32_t)0x00000100)
- #define EXTI_FTSR_TR9 ((uint32_t)0x00000200)
- #define EXTI_FTSR_TR10 ((uint32_t)0x00000400)
- #define EXTI_FTSR_TR11 ((uint32_t)0x00000800)
- #define EXTI_FTSR_TR12 ((uint32_t)0x00001000)
- #define EXTI_FTSR_TR13 ((uint32_t)0x00002000)
- #define EXTI_FTSR_TR14 ((uint32_t)0x00004000)
- #define EXTI_FTSR_TR15 ((uint32_t)0x00008000)
- #define EXTI_FTSR_TR16 ((uint32_t)0x00010000)
- #define EXTI_FTSR_TR17 ((uint32_t)0x00020000)
- #define EXTI_FTSR_TR18 ((uint32_t)0x00040000)
- #define EXTI_FTSR_TR19 ((uint32_t)0x00080000)
- #define EXTI_SWIER_SWIER0 ((uint32_t)0x00000001)
- #define EXTI_SWIER_SWIER1 ((uint32_t)0x00000002)
- #define EXTI_SWIER_SWIER2 ((uint32_t)0x00000004)
- #define EXTI_SWIER_SWIER3 ((uint32_t)0x00000008)
- #define EXTI_SWIER_SWIER4 ((uint32_t)0x00000010)
- #define EXTI_SWIER_SWIER5 ((uint32_t)0x00000020)
- #define EXTI_SWIER_SWIER6 ((uint32_t)0x00000040)

- #define EXTI_SWIER_SWIER7 ((uint32_t)0x00000080)
- #define EXTI_SWIER_SWIER8 ((uint32_t)0x00000100)
- #define EXTI_SWIER_SWIER9 ((uint32_t)0x00000200)
- #define EXTI_SWIER_SWIER10 ((uint32_t)0x00000400)
- #define EXTI_SWIER_SWIER11 ((uint32_t)0x00000800)
- #define EXTI_SWIER_SWIER12 ((uint32_t)0x00001000)
- #define EXTI_SWIER_SWIER13 ((uint32_t)0x00002000)
- #define EXTI_SWIER_SWIER14 ((uint32_t)0x00004000)
- #define EXTI_SWIER_SWIER15 ((uint32_t)0x00008000)
- #define EXTI_SWIER_SWIER16 ((uint32_t)0x00010000)
- #define EXTI_SWIER_SWIER17 ((uint32_t)0x00020000)
- #define EXTI_SWIER_SWIER18 ((uint32_t)0x00040000)
- #define EXTI_SWIER_SWIER19 ((uint32_t)0x00080000)
- #define EXTI_PR_PR0 ((uint32_t)0x00000001)
- #define EXTI_PR_PR1 ((uint32_t)0x00000002)
- #define EXTI_PR_PR2 ((uint32_t)0x00000004)
- #define EXTI_PR_PR3 ((uint32_t)0x00000008)
- #define EXTI_PR_PR4 ((uint32_t)0x00000010)
- #define EXTI_PR_PR5 ((uint32_t)0x00000020)
- #define EXTI_PR_PR6 ((uint32_t)0x00000040)
- #define EXTI_PR_PR7 ((uint32_t)0x00000080)
- #define EXTI_PR_PR8 ((uint32_t)0x00000100)
- #define EXTI_PR_PR9 ((uint32_t)0x00000200)
- #define EXTI_PR_PR10 ((uint32_t)0x00000400)
- #define EXTI_PR_PR11 ((uint32_t)0x00000800)
- #define EXTI_PR_PR12 ((uint32_t)0x00001000)
- #define EXTI_PR_PR13 ((uint32_t)0x00002000)
- #define EXTI_PR_PR14 ((uint32_t)0x00004000)
- #define EXTI_PR_PR15 ((uint32_t)0x00008000)
- #define EXTI_PR_PR16 ((uint32_t)0x00010000)
- #define EXTI_PR_PR17 ((uint32_t)0x00020000)
- #define EXTI_PR_PR18 ((uint32_t)0x00040000)
- #define EXTI_PR_PR19 ((uint32_t)0x00080000)
- #define **FLASH_ACR_LATENCY** ((uint32_t)0x00000007)
- #define **FLASH_ACR_LATENCY_0WS** ((uint32_t)0x00000000)
- #define **FLASH_ACR_LATENCY_1WS** ((uint32_t)0x00000001)
- #define **FLASH_ACR_LATENCY_2WS** ((uint32_t)0x00000002)
- #define **FLASH_ACR_LATENCY_3WS** ((uint32_t)0x00000003)
- #define **FLASH_ACR_LATENCY_4WS** ((uint32_t)0x00000004)
- #define **FLASH_ACR_LATENCY_5WS** ((uint32_t)0x00000005)
- #define **FLASH_ACR_LATENCY_6WS** ((uint32_t)0x00000006)
- #define **FLASH_ACR_LATENCY_7WS** ((uint32_t)0x00000007)
- #define **FLASH_ACR_PRFTEN** ((uint32_t)0x00000100)
- #define **FLASH_ACR_ICEN** ((uint32_t)0x00000200)
- #define **FLASH_ACR_DCEN** ((uint32_t)0x00000400)
- #define **FLASH_ACR_ICRST** ((uint32_t)0x00000800)
- #define **FLASH_ACR_DCRST** ((uint32_t)0x00001000)
- #define **FLASH_ACR_BYTE0_ADDRESS** ((uint32_t)0x40023C00)
- #define **FLASH_ACR_BYTE2_ADDRESS** ((uint32_t)0x40023C03)
- #define **FLASH_SR_EOP** ((uint32_t)0x00000001)
- #define **FLASH_SR_SOP** ((uint32_t)0x00000002)
- #define **FLASH_SR_WRPERR** ((uint32_t)0x00000010)
- #define **FLASH_SR_PGAERR** ((uint32_t)0x00000020)
- #define **FLASH_SR_PGPERR** ((uint32_t)0x00000040)
- #define **FLASH_SR_PGSERR** ((uint32_t)0x00000080)

- #define **FLASH_SR_BSY** ((uint32_t)0x00010000)
- #define **FLASH_CR_PG** ((uint32_t)0x00000001)
- #define **FLASH_CR_SER** ((uint32_t)0x00000002)
- #define **FLASH_CR_MER** ((uint32_t)0x00000004)
- #define **FLASH_CR_SNB_0** ((uint32_t)0x00000008)
- #define **FLASH_CR_SNB_1** ((uint32_t)0x00000010)
- #define **FLASH_CR_SNB_2** ((uint32_t)0x00000020)
- #define **FLASH_CR_SNB_3** ((uint32_t)0x00000040)
- #define **FLASH_CR_PSIZE_0** ((uint32_t)0x00000100)
- #define **FLASH_CR_PSIZE_1** ((uint32_t)0x00000200)
- #define **FLASH_CR_STRT** ((uint32_t)0x00010000)
- #define **FLASH_CR_EOPIE** ((uint32_t)0x01000000)
- #define **FLASH_CR_LOCK** ((uint32_t)0x80000000)
- #define **FLASH_OPTCR_OPTLOCK** ((uint32_t)0x00000001)
- #define **FLASH_OPTCR_OPTSTRT** ((uint32_t)0x00000002)
- #define **FLASH_OPTCR_BOR_LEV_0** ((uint32_t)0x00000004)
- #define **FLASH_OPTCR_BOR_LEV_1** ((uint32_t)0x00000008)
- #define **FLASH_OPTCR_BOR_LEV** ((uint32_t)0x0000000C)
- #define **FLASH_OPTCR_WDG_SW** ((uint32_t)0x00000020)
- #define **FLASH_OPTCR_nRST_STOP** ((uint32_t)0x00000040)
- #define **FLASH_OPTCR_nRST_STDBY** ((uint32_t)0x00000080)
- #define **FLASH_OPTCR_RDP_0** ((uint32_t)0x00000100)
- #define **FLASH_OPTCR_RDP_1** ((uint32_t)0x00000200)
- #define **FLASH_OPTCR_RDP_2** ((uint32_t)0x00000400)
- #define **FLASH_OPTCR_RDP_3** ((uint32_t)0x00000800)
- #define **FLASH_OPTCR_RDP_4** ((uint32_t)0x00001000)
- #define **FLASH_OPTCR_RDP_5** ((uint32_t)0x00002000)
- #define **FLASH_OPTCR_RDP_6** ((uint32_t)0x00004000)
- #define **FLASH_OPTCR_RDP_7** ((uint32_t)0x00008000)
- #define **FLASH_OPTCR_nWRP_0** ((uint32_t)0x00010000)
- #define **FLASH_OPTCR_nWRP_1** ((uint32_t)0x00020000)
- #define **FLASH_OPTCR_nWRP_2** ((uint32_t)0x00040000)
- #define **FLASH_OPTCR_nWRP_3** ((uint32_t)0x00080000)
- #define **FLASH_OPTCR_nWRP_4** ((uint32_t)0x00100000)
- #define **FLASH_OPTCR_nWRP_5** ((uint32_t)0x00200000)
- #define **FLASH_OPTCR_nWRP_6** ((uint32_t)0x00400000)
- #define **FLASH_OPTCR_nWRP_7** ((uint32_t)0x00800000)
- #define **FLASH_OPTCR_nWRP_8** ((uint32_t)0x01000000)
- #define **FLASH_OPTCR_nWRP_9** ((uint32_t)0x02000000)
- #define **FLASH_OPTCR_nWRP_10** ((uint32_t)0x04000000)
- #define **FLASH_OPTCR_nWRP_11** ((uint32_t)0x08000000)
- #define FSMC_BCR1_MBKEN ((uint32_t)0x00000001)
- #define FSMC_BCR1_MUXEN ((uint32_t)0x00000002)
- #define FSMC_BCR1_MTYP ((uint32_t)0x0000000C)
- #define FSMC_BCR1_MTYP_0 ((uint32_t)0x00000004)
- #define FSMC_BCR1_MTYP_1 ((uint32_t)0x00000008)
- #define FSMC_BCR1_MWID ((uint32_t)0x00000030)
- #define FSMC_BCR1_MWID_0 ((uint32_t)0x00000010)
- #define FSMC_BCR1_MWID_1 ((uint32_t)0x00000020)
- #define FSMC_BCR1_FACCEN ((uint32_t)0x00000040)
- #define FSMC_BCR1_BURSTEN ((uint32_t)0x00000100)
- #define FSMC_BCR1_WAITPOL ((uint32_t)0x00000200)
- #define FSMC_BCR1_WRAPMOD ((uint32_t)0x00000400)
- #define FSMC_BCR1_WAITCFG ((uint32_t)0x00000800)
- #define FSMC_BCR1_WREN ((uint32_t)0x00001000)

- #define FSMC_BCR1_WAITEN ((uint32_t)0x00002000)
- #define FSMC_BCR1_EXTMOD ((uint32_t)0x00004000)
- #define FSMC_BCR1_ASYNCWAIT ((uint32_t)0x00008000)
- #define FSMC_BCR1_CBURSTRW ((uint32_t)0x00080000)
- #define FSMC_BCR2_MBKEN ((uint32_t)0x00000001)
- #define FSMC_BCR2_MUXEN ((uint32_t)0x00000002)
- #define FSMC_BCR2_MTYP ((uint32_t)0x0000000C)
- #define FSMC_BCR2_MTYP_0 ((uint32_t)0x00000004)
- #define FSMC_BCR2_MTYP_1 ((uint32_t)0x00000008)
- #define FSMC_BCR2_MWID ((uint32_t)0x00000030)
- #define FSMC_BCR2_MWID_0 ((uint32_t)0x00000010)
- #define FSMC_BCR2_MWID_1 ((uint32_t)0x00000020)
- #define FSMC_BCR2_FACCEN ((uint32_t)0x00000040)
- #define FSMC_BCR2_BURSTEN ((uint32_t)0x00000100)
- #define FSMC_BCR2_WAITPOL ((uint32_t)0x00000200)
- #define FSMC_BCR2_WRAPMOD ((uint32_t)0x00000400)
- #define FSMC_BCR2_WAITCFG ((uint32_t)0x00000800)
- #define FSMC_BCR2_WREN ((uint32_t)0x00001000)
- #define FSMC_BCR2_WAITEN ((uint32_t)0x00002000)
- #define FSMC_BCR2_EXTMOD ((uint32_t)0x00004000)
- #define FSMC_BCR2_ASYNCWAIT ((uint32_t)0x00008000)
- #define FSMC_BCR2_CBURSTRW ((uint32_t)0x00080000)
- #define FSMC_BCR3_MBKEN ((uint32_t)0x00000001)
- #define FSMC_BCR3_MUXEN ((uint32_t)0x00000002)
- #define FSMC_BCR3_MTYP ((uint32_t)0x0000000C)
- #define FSMC_BCR3_MTYP_0 ((uint32_t)0x00000004)
- #define FSMC_BCR3_MTYP_1 ((uint32_t)0x00000008)
- #define FSMC_BCR3_MWID ((uint32_t)0x00000030)
- #define FSMC_BCR3_MWID_0 ((uint32_t)0x00000010)
- #define FSMC_BCR3_MWID_1 ((uint32_t)0x00000020)
- #define FSMC_BCR3_FACCEN ((uint32_t)0x00000040)
- #define FSMC_BCR3_BURSTEN ((uint32_t)0x00000100)
- #define FSMC_BCR3_WAITPOL ((uint32_t)0x00000200)
- #define FSMC_BCR3_WRAPMOD ((uint32_t)0x00000400)
- #define FSMC_BCR3_WAITCFG ((uint32_t)0x00000800)
- #define FSMC_BCR3_WREN ((uint32_t)0x00001000)
- #define FSMC_BCR3_WAITEN ((uint32_t)0x00002000)
- #define FSMC_BCR3_EXTMOD ((uint32_t)0x00004000)
- #define FSMC_BCR3_ASYNCWAIT ((uint32_t)0x00008000)
- #define FSMC_BCR3_CBURSTRW ((uint32_t)0x00080000)
- #define FSMC_BCR4_MBKEN ((uint32_t)0x00000001)
- #define FSMC_BCR4_MUXEN ((uint32_t)0x00000002)
- #define FSMC_BCR4_MTYP ((uint32_t)0x0000000C)
- #define FSMC_BCR4_MTYP_0 ((uint32_t)0x00000004)
- #define FSMC_BCR4_MTYP_1 ((uint32_t)0x00000008)
- #define FSMC_BCR4_MWID ((uint32_t)0x00000030)
- #define FSMC_BCR4_MWID_0 ((uint32_t)0x00000010)
- #define FSMC_BCR4_MWID_1 ((uint32_t)0x00000020)
- #define FSMC_BCR4_FACCEN ((uint32_t)0x00000040)
- #define FSMC_BCR4_BURSTEN ((uint32_t)0x00000100)
- #define FSMC_BCR4_WAITPOL ((uint32_t)0x00000200)
- #define FSMC_BCR4_WRAPMOD ((uint32_t)0x00000400)
- #define FSMC_BCR4_WAITCFG ((uint32_t)0x00000800)
- #define FSMC_BCR4_WREN ((uint32_t)0x00001000)
- #define FSMC_BCR4_WAITEN ((uint32_t)0x00002000)

- #define FSMC_BCR4_EXTMOD ((uint32_t)0x00004000)
- #define FSMC_BCR4_ASYNCWAIT ((uint32_t)0x00008000)
- #define FSMC_BCR4_CBURSTRW ((uint32_t)0x00080000)
- #define FSMC_BTR1_ADDSET ((uint32_t)0x0000000F)
- #define FSMC_BTR1_ADDSET_0 ((uint32_t)0x00000001)
- #define FSMC_BTR1_ADDSET_1 ((uint32_t)0x00000002)
- #define FSMC_BTR1_ADDSET_2 ((uint32_t)0x00000004)
- #define FSMC_BTR1_ADDSET_3 ((uint32_t)0x00000008)
- #define FSMC_BTR1_ADDHLD ((uint32_t)0x000000F0)
- #define FSMC_BTR1_ADDHLD_0 ((uint32_t)0x00000010)
- #define FSMC_BTR1_ADDHLD_1 ((uint32_t)0x00000020)
- #define FSMC_BTR1_ADDHLD_2 ((uint32_t)0x00000040)
- #define FSMC_BTR1_ADDHLD_3 ((uint32_t)0x00000080)
- #define FSMC_BTR1_DATAST ((uint32_t)0x0000FF00)
- #define FSMC_BTR1_DATAST_0 ((uint32_t)0x00000100)
- #define FSMC_BTR1_DATAST_1 ((uint32_t)0x00000200)
- #define FSMC_BTR1_DATAST_2 ((uint32_t)0x00000400)
- #define FSMC_BTR1_DATAST_3 ((uint32_t)0x00000800)
- #define FSMC_BTR1_BUSTURN ((uint32_t)0x000F0000)
- #define FSMC_BTR1_BUSTURN_0 ((uint32_t)0x00010000)
- #define FSMC_BTR1_BUSTURN_1 ((uint32_t)0x00020000)
- #define FSMC_BTR1_BUSTURN_2 ((uint32_t)0x00040000)
- #define FSMC_BTR1_BUSTURN_3 ((uint32_t)0x00080000)
- #define FSMC_BTR1_CLKDIV ((uint32_t)0x00F00000)
- #define FSMC_BTR1_CLKDIV_0 ((uint32_t)0x00100000)
- #define FSMC_BTR1_CLKDIV_1 ((uint32_t)0x00200000)
- #define FSMC_BTR1_CLKDIV_2 ((uint32_t)0x00400000)
- #define FSMC_BTR1_CLKDIV_3 ((uint32_t)0x00800000)
- #define FSMC_BTR1_DATLAT ((uint32_t)0x0F000000)
- #define FSMC_BTR1_DATLAT_0 ((uint32_t)0x01000000)
- #define FSMC_BTR1_DATLAT_1 ((uint32_t)0x02000000)
- #define FSMC_BTR1_DATLAT_2 ((uint32_t)0x04000000)
- #define FSMC_BTR1_DATLAT_3 ((uint32_t)0x08000000)
- #define FSMC_BTR1_ACCMOD ((uint32_t)0x30000000)
- #define FSMC_BTR1_ACCMOD_0 ((uint32_t)0x10000000)
- #define FSMC_BTR1_ACCMOD_1 ((uint32_t)0x20000000)
- #define FSMC_BTR2_ADDSET ((uint32_t)0x0000000F)
- #define FSMC_BTR2_ADDSET_0 ((uint32_t)0x00000001)
- #define FSMC_BTR2_ADDSET_1 ((uint32_t)0x00000002)
- #define FSMC_BTR2_ADDSET_2 ((uint32_t)0x00000004)
- #define FSMC_BTR2_ADDSET_3 ((uint32_t)0x00000008)
- #define FSMC_BTR2_ADDHLD ((uint32_t)0x000000F0)
- #define FSMC_BTR2_ADDHLD_0 ((uint32_t)0x00000010)
- #define FSMC_BTR2_ADDHLD_1 ((uint32_t)0x00000020)
- #define FSMC_BTR2_ADDHLD_2 ((uint32_t)0x00000040)
- #define FSMC_BTR2_ADDHLD_3 ((uint32_t)0x00000080)
- #define FSMC_BTR2_DATAST ((uint32_t)0x0000FF00)
- #define FSMC_BTR2_DATAST_0 ((uint32_t)0x00000100)
- #define FSMC_BTR2_DATAST_1 ((uint32_t)0x00000200)
- #define FSMC_BTR2_DATAST_2 ((uint32_t)0x00000400)
- #define FSMC_BTR2_DATAST_3 ((uint32_t)0x00000800)
- #define FSMC_BTR2_BUSTURN ((uint32_t)0x000F0000)
- #define FSMC_BTR2_BUSTURN_0 ((uint32_t)0x00010000)
- #define FSMC_BTR2_BUSTURN_1 ((uint32_t)0x00020000)
- #define FSMC_BTR2_BUSTURN_2 ((uint32_t)0x00040000)

- #define FSMC_BTR2_BUSTURN_3 ((uint32_t)0x00080000)
- #define FSMC_BTR2_CLKDIV ((uint32_t)0x00F00000)
- #define FSMC_BTR2_CLKDIV_0 ((uint32_t)0x00100000)
- #define FSMC_BTR2_CLKDIV_1 ((uint32_t)0x00200000)
- #define FSMC_BTR2_CLKDIV_2 ((uint32_t)0x00400000)
- #define FSMC_BTR2_CLKDIV_3 ((uint32_t)0x00800000)
- #define FSMC_BTR2_DATLAT ((uint32_t)0x0F000000)
- #define FSMC_BTR2_DATLAT_0 ((uint32_t)0x01000000)
- #define FSMC_BTR2_DATLAT_1 ((uint32_t)0x02000000)
- #define FSMC_BTR2_DATLAT_2 ((uint32_t)0x04000000)
- #define FSMC_BTR2_DATLAT_3 ((uint32_t)0x08000000)
- #define FSMC_BTR2_ACCMOD ((uint32_t)0x30000000)
- #define FSMC_BTR2_ACCMOD_0 ((uint32_t)0x10000000)
- #define FSMC_BTR2_ACCMOD_1 ((uint32_t)0x20000000)
- #define FSMC_BTR3_ADDSET ((uint32_t)0x0000000F)
- #define FSMC_BTR3_ADDSET_0 ((uint32_t)0x00000001)
- #define FSMC_BTR3_ADDSET_1 ((uint32_t)0x00000002)
- #define FSMC_BTR3_ADDSET_2 ((uint32_t)0x00000004)
- #define FSMC_BTR3_ADDSET_3 ((uint32_t)0x00000008)
- #define FSMC_BTR3_ADDHLD ((uint32_t)0x000000F0)
- #define FSMC_BTR3_ADDHLD_0 ((uint32_t)0x00000010)
- #define FSMC_BTR3_ADDHLD_1 ((uint32_t)0x00000020)
- #define FSMC_BTR3_ADDHLD_2 ((uint32_t)0x00000040)
- #define FSMC_BTR3_ADDHLD_3 ((uint32_t)0x00000080)
- #define FSMC_BTR3_DATAST ((uint32_t)0x0000FF00)
- #define FSMC_BTR3_DATAST_0 ((uint32_t)0x00000100)
- #define FSMC_BTR3_DATAST_1 ((uint32_t)0x00000200)
- #define FSMC_BTR3_DATAST_2 ((uint32_t)0x00000400)
- #define FSMC_BTR3_DATAST_3 ((uint32_t)0x00000800)
- #define FSMC_BTR3_BUSTURN ((uint32_t)0x000F0000)
- #define FSMC_BTR3_BUSTURN_0 ((uint32_t)0x00010000)
- #define FSMC_BTR3_BUSTURN_1 ((uint32_t)0x00020000)
- #define FSMC_BTR3_BUSTURN_2 ((uint32_t)0x00040000)
- #define FSMC_BTR3_BUSTURN_3 ((uint32_t)0x00080000)
- #define FSMC_BTR3_CLKDIV ((uint32_t)0x00F00000)
- #define FSMC_BTR3_CLKDIV_0 ((uint32_t)0x00100000)
- #define FSMC_BTR3_CLKDIV_1 ((uint32_t)0x00200000)
- #define FSMC_BTR3_CLKDIV_2 ((uint32_t)0x00400000)
- #define FSMC_BTR3_CLKDIV_3 ((uint32_t)0x00800000)
- #define FSMC_BTR3_DATLAT ((uint32_t)0x0F000000)
- #define FSMC_BTR3_DATLAT_0 ((uint32_t)0x01000000)
- #define FSMC_BTR3_DATLAT_1 ((uint32_t)0x02000000)
- #define FSMC_BTR3_DATLAT_2 ((uint32_t)0x04000000)
- #define FSMC_BTR3_DATLAT_3 ((uint32_t)0x08000000)
- #define FSMC_BTR3_ACCMOD ((uint32_t)0x30000000)
- #define FSMC_BTR3_ACCMOD_0 ((uint32_t)0x10000000)
- #define FSMC_BTR3_ACCMOD_1 ((uint32_t)0x20000000)
- #define FSMC_BTR4_ADDSET ((uint32_t)0x0000000F)
- #define FSMC_BTR4_ADDSET_0 ((uint32_t)0x00000001)
- #define FSMC_BTR4_ADDSET_1 ((uint32_t)0x00000002)
- #define FSMC_BTR4_ADDSET_2 ((uint32_t)0x00000004)
- #define FSMC_BTR4_ADDSET_3 ((uint32_t)0x00000008)
- #define FSMC_BTR4_ADDHLD ((uint32_t)0x000000F0)
- #define FSMC_BTR4_ADDHLD_0 ((uint32_t)0x00000010)
- #define FSMC_BTR4_ADDHLD_1 ((uint32_t)0x00000020)

- #define FSMC_BTR4_ADDHLD_2 ((uint32_t)0x00000040)
- #define FSMC_BTR4_ADDHLD_3 ((uint32_t)0x00000080)
- #define FSMC_BTR4_DATAST ((uint32_t)0x0000FF00)
- #define FSMC_BTR4_DATAST_0 ((uint32_t)0x00000100)
- #define FSMC_BTR4_DATAST_1 ((uint32_t)0x00000200)
- #define FSMC_BTR4_DATAST_2 ((uint32_t)0x00000400)
- #define FSMC_BTR4_DATAST_3 ((uint32_t)0x00000800)
- #define FSMC_BTR4_BUSTURN ((uint32_t)0x000F0000)
- #define FSMC_BTR4_BUSTURN_0 ((uint32_t)0x00010000)
- #define FSMC_BTR4_BUSTURN_1 ((uint32_t)0x00020000)
- #define FSMC_BTR4_BUSTURN_2 ((uint32_t)0x00040000)
- #define FSMC_BTR4_BUSTURN_3 ((uint32_t)0x00080000)
- #define FSMC_BTR4_CLKDIV ((uint32_t)0x00F00000)
- #define FSMC_BTR4_CLKDIV_0 ((uint32_t)0x00100000)
- #define FSMC_BTR4_CLKDIV_1 ((uint32_t)0x00200000)
- #define FSMC_BTR4_CLKDIV_2 ((uint32_t)0x00400000)
- #define FSMC_BTR4_CLKDIV_3 ((uint32_t)0x00800000)
- #define FSMC_BTR4_DATLAT ((uint32_t)0x0F000000)
- #define FSMC_BTR4_DATLAT_0 ((uint32_t)0x01000000)
- #define FSMC_BTR4_DATLAT_1 ((uint32_t)0x02000000)
- #define FSMC_BTR4_DATLAT_2 ((uint32_t)0x04000000)
- #define FSMC_BTR4_DATLAT_3 ((uint32_t)0x08000000)
- #define FSMC_BTR4_ACCMOD ((uint32_t)0x30000000)
- #define FSMC_BTR4_ACCMOD_0 ((uint32_t)0x10000000)
- #define FSMC_BTR4_ACCMOD_1 ((uint32_t)0x20000000)
- #define FSMC_BWTR1_ADDSET ((uint32_t)0x0000000F)
- #define FSMC_BWTR1_ADDSET_0 ((uint32_t)0x00000001)
- #define FSMC_BWTR1_ADDSET_1 ((uint32_t)0x00000002)
- #define FSMC_BWTR1_ADDSET_2 ((uint32_t)0x00000004)
- #define FSMC_BWTR1_ADDSET_3 ((uint32_t)0x00000008)
- #define FSMC_BWTR1_ADDHLD ((uint32_t)0x000000F0)
- #define FSMC_BWTR1_ADDHLD_0 ((uint32_t)0x00000010)
- #define FSMC_BWTR1_ADDHLD_1 ((uint32_t)0x00000020)
- #define FSMC_BWTR1_ADDHLD_2 ((uint32_t)0x00000040)
- #define FSMC_BWTR1_ADDHLD_3 ((uint32_t)0x00000080)
- #define FSMC_BWTR1_DATAST ((uint32_t)0x0000FF00)
- #define FSMC_BWTR1_DATAST_0 ((uint32_t)0x00000100)
- #define FSMC_BWTR1_DATAST_1 ((uint32_t)0x00000200)
- #define FSMC_BWTR1_DATAST_2 ((uint32_t)0x00000400)
- #define FSMC_BWTR1_DATAST_3 ((uint32_t)0x00000800)
- #define FSMC_BWTR1_CLKDIV ((uint32_t)0x00F00000)
- #define FSMC_BWTR1_CLKDIV_0 ((uint32_t)0x00100000)
- #define FSMC_BWTR1_CLKDIV_1 ((uint32_t)0x00200000)
- #define FSMC_BWTR1_CLKDIV_2 ((uint32_t)0x00400000)
- #define FSMC_BWTR1_CLKDIV_3 ((uint32_t)0x00800000)
- #define FSMC_BWTR1_DATLAT ((uint32_t)0x0F000000)
- #define FSMC_BWTR1_DATLAT_0 ((uint32_t)0x01000000)
- #define FSMC_BWTR1_DATLAT_1 ((uint32_t)0x02000000)
- #define FSMC_BWTR1_DATLAT_2 ((uint32_t)0x04000000)
- #define FSMC_BWTR1_DATLAT_3 ((uint32_t)0x08000000)
- #define FSMC_BWTR1_ACCMOD ((uint32_t)0x30000000)
- #define FSMC_BWTR1_ACCMOD_0 ((uint32_t)0x10000000)
- #define FSMC_BWTR1_ACCMOD_1 ((uint32_t)0x20000000)
- #define FSMC_BWTR2_ADDSET ((uint32_t)0x0000000F)
- #define FSMC_BWTR2_ADDSET_0 ((uint32_t)0x00000001)

- #define FSMC_BWTR2_ADDSET_1 ((uint32_t)0x00000002)
- #define FSMC_BWTR2_ADDSET_2 ((uint32_t)0x00000004)
- #define FSMC_BWTR2_ADDSET_3 ((uint32_t)0x00000008)
- #define FSMC_BWTR2_ADDHLD ((uint32_t)0x000000F0)
- #define FSMC_BWTR2_ADDHLD_0 ((uint32_t)0x00000010)
- #define FSMC_BWTR2_ADDHLD_1 ((uint32_t)0x00000020)
- #define FSMC_BWTR2_ADDHLD_2 ((uint32_t)0x00000040)
- #define FSMC_BWTR2_ADDHLD_3 ((uint32_t)0x00000080)
- #define FSMC_BWTR2_DATAST ((uint32_t)0x0000FF00)
- #define FSMC_BWTR2_DATAST_0 ((uint32_t)0x00000100)
- #define FSMC_BWTR2_DATAST_1 ((uint32_t)0x00000200)
- #define FSMC_BWTR2_DATAST_2 ((uint32_t)0x00000400)
- #define FSMC_BWTR2_DATAST_3 ((uint32_t)0x00000800)
- #define FSMC_BWTR2_CLKDIV ((uint32_t)0x00F00000)
- #define FSMC_BWTR2_CLKDIV_0 ((uint32_t)0x00100000)
- #define FSMC_BWTR2_CLKDIV_1 ((uint32_t)0x00200000)
- #define FSMC_BWTR2_CLKDIV_2 ((uint32_t)0x00400000)
- #define FSMC_BWTR2_CLKDIV_3 ((uint32_t)0x00800000)
- #define FSMC_BWTR2_DATLAT ((uint32_t)0x0F000000)
- #define FSMC_BWTR2_DATLAT_0 ((uint32_t)0x01000000)
- #define FSMC_BWTR2_DATLAT_1 ((uint32_t)0x02000000)
- #define FSMC_BWTR2_DATLAT_2 ((uint32_t)0x04000000)
- #define FSMC_BWTR2_DATLAT_3 ((uint32_t)0x08000000)
- #define FSMC_BWTR2_ACCMOD ((uint32_t)0x30000000)
- #define FSMC_BWTR2_ACCMOD_0 ((uint32_t)0x10000000)
- #define FSMC_BWTR2_ACCMOD_1 ((uint32_t)0x20000000)
- #define FSMC_BWTR3_ADDSET ((uint32_t)0x0000000F)
- #define FSMC_BWTR3_ADDSET_0 ((uint32_t)0x00000001)
- #define FSMC_BWTR3_ADDSET_1 ((uint32_t)0x00000002)
- #define FSMC_BWTR3_ADDSET_2 ((uint32_t)0x00000004)
- #define FSMC_BWTR3_ADDSET_3 ((uint32_t)0x00000008)
- #define FSMC_BWTR3_ADDHLD ((uint32_t)0x000000F0)
- #define FSMC_BWTR3_ADDHLD_0 ((uint32_t)0x00000010)
- #define FSMC_BWTR3_ADDHLD_1 ((uint32_t)0x00000020)
- #define FSMC_BWTR3_ADDHLD_2 ((uint32_t)0x00000040)
- #define FSMC_BWTR3_ADDHLD_3 ((uint32_t)0x00000080)
- #define FSMC_BWTR3_DATAST ((uint32_t)0x0000FF00)
- #define FSMC_BWTR3_DATAST_0 ((uint32_t)0x00000100)
- #define FSMC_BWTR3_DATAST_1 ((uint32_t)0x00000200)
- #define FSMC_BWTR3_DATAST_2 ((uint32_t)0x00000400)
- #define FSMC_BWTR3_DATAST_3 ((uint32_t)0x00000800)
- #define FSMC_BWTR3_CLKDIV ((uint32_t)0x00F00000)
- #define FSMC_BWTR3_CLKDIV_0 ((uint32_t)0x00100000)
- #define FSMC_BWTR3_CLKDIV_1 ((uint32_t)0x00200000)
- #define FSMC_BWTR3_CLKDIV_2 ((uint32_t)0x00400000)
- #define FSMC_BWTR3_CLKDIV_3 ((uint32_t)0x00800000)
- #define FSMC_BWTR3_DATLAT ((uint32_t)0x0F000000)
- #define FSMC_BWTR3_DATLAT_0 ((uint32_t)0x01000000)
- #define FSMC_BWTR3_DATLAT_1 ((uint32_t)0x02000000)
- #define FSMC_BWTR3_DATLAT_2 ((uint32_t)0x04000000)
- #define FSMC_BWTR3_DATLAT_3 ((uint32_t)0x08000000)
- #define FSMC_BWTR3_ACCMOD ((uint32_t)0x30000000)
- #define FSMC_BWTR3_ACCMOD_0 ((uint32_t)0x10000000)
- #define FSMC_BWTR3_ACCMOD_1 ((uint32_t)0x20000000)
- #define FSMC_BWTR4_ADDSET ((uint32_t)0x0000000F)

- #define FSMC_BWTR4_ADDSET_0 ((uint32_t)0x00000001)
- #define FSMC_BWTR4_ADDSET_1 ((uint32_t)0x00000002)
- #define FSMC_BWTR4_ADDSET_2 ((uint32_t)0x00000004)
- #define FSMC_BWTR4_ADDSET_3 ((uint32_t)0x00000008)
- #define FSMC_BWTR4_ADDHLD ((uint32_t)0x000000F0)
- #define FSMC_BWTR4_ADDHLD_0 ((uint32_t)0x00000010)
- #define FSMC_BWTR4_ADDHLD_1 ((uint32_t)0x00000020)
- #define FSMC_BWTR4_ADDHLD_2 ((uint32_t)0x00000040)
- #define FSMC_BWTR4_ADDHLD_3 ((uint32_t)0x00000080)
- #define FSMC_BWTR4_DATAST ((uint32_t)0x0000FF00)
- #define FSMC_BWTR4_DATAST_0 ((uint32_t)0x00000100)
- #define FSMC_BWTR4_DATAST_1 ((uint32_t)0x00000200)
- #define FSMC_BWTR4_DATAST_2 ((uint32_t)0x00000400)
- #define FSMC_BWTR4_DATAST_3 ((uint32_t)0x00000800)
- #define FSMC_BWTR4_CLKDIV ((uint32_t)0x00F00000)
- #define FSMC_BWTR4_CLKDIV_0 ((uint32_t)0x00100000)
- #define FSMC_BWTR4_CLKDIV_1 ((uint32_t)0x00200000)
- #define FSMC_BWTR4_CLKDIV_2 ((uint32_t)0x00400000)
- #define FSMC_BWTR4_CLKDIV_3 ((uint32_t)0x00800000)
- #define FSMC_BWTR4_DATLAT ((uint32_t)0x0F000000)
- #define FSMC_BWTR4_DATLAT_0 ((uint32_t)0x01000000)
- #define FSMC_BWTR4_DATLAT_1 ((uint32_t)0x02000000)
- #define FSMC_BWTR4_DATLAT_2 ((uint32_t)0x04000000)
- #define FSMC_BWTR4_DATLAT_3 ((uint32_t)0x08000000)
- #define FSMC_BWTR4_ACCMOD ((uint32_t)0x30000000)
- #define FSMC_BWTR4_ACCMOD_0 ((uint32_t)0x10000000)
- #define FSMC_BWTR4_ACCMOD_1 ((uint32_t)0x20000000)
- #define FSMC_PCR2_PWAITEN ((uint32_t)0x00000002)
- #define FSMC_PCR2_PBKEN ((uint32_t)0x00000004)
- #define FSMC_PCR2_PTYP ((uint32_t)0x00000008)
- #define FSMC_PCR2_PWID ((uint32_t)0x00000030)
- #define FSMC_PCR2_PWID_0 ((uint32_t)0x00000010)
- #define FSMC_PCR2_PWID_1 ((uint32_t)0x00000020)
- #define FSMC_PCR2_ECCEN ((uint32_t)0x00000040)
- #define FSMC_PCR2_TCLR ((uint32_t)0x00001E00)
- #define FSMC_PCR2_TCLR_0 ((uint32_t)0x00000200)
- #define FSMC_PCR2_TCLR_1 ((uint32_t)0x00000400)
- #define FSMC_PCR2_TCLR_2 ((uint32_t)0x00000800)
- #define FSMC_PCR2_TCLR_3 ((uint32_t)0x00001000)
- #define FSMC_PCR2_TAR ((uint32_t)0x0001E000)
- #define FSMC_PCR2_TAR_0 ((uint32_t)0x00002000)
- #define FSMC_PCR2_TAR_1 ((uint32_t)0x00004000)
- #define FSMC_PCR2_TAR_2 ((uint32_t)0x00008000)
- #define FSMC_PCR2_TAR_3 ((uint32_t)0x00010000)
- #define FSMC_PCR2_ECCPS ((uint32_t)0x000E0000)
- #define FSMC_PCR2_ECCPS_0 ((uint32_t)0x00020000)
- #define FSMC_PCR2_ECCPS_1 ((uint32_t)0x00040000)
- #define FSMC_PCR2_ECCPS_2 ((uint32_t)0x00080000)
- #define FSMC_PCR3_PWAITEN ((uint32_t)0x00000002)
- #define FSMC_PCR3_PBKEN ((uint32_t)0x00000004)
- #define FSMC_PCR3_PTYP ((uint32_t)0x00000008)
- #define FSMC_PCR3_PWID ((uint32_t)0x00000030)
- #define FSMC_PCR3_PWID_0 ((uint32_t)0x00000010)
- #define FSMC_PCR3_PWID_1 ((uint32_t)0x00000020)
- #define FSMC_PCR3_ECCEN ((uint32_t)0x00000040)

- #define FSMC_PCR3_TCLR ((uint32_t)0x00001E00)
- #define FSMC_PCR3_TCLR_0 ((uint32_t)0x00000200)
- #define FSMC_PCR3_TCLR_1 ((uint32_t)0x00000400)
- #define FSMC_PCR3_TCLR_2 ((uint32_t)0x00000800)
- #define FSMC_PCR3_TCLR_3 ((uint32_t)0x00001000)
- #define FSMC_PCR3_TAR ((uint32_t)0x0001E000)
- #define FSMC_PCR3_TAR_0 ((uint32_t)0x00002000)
- #define FSMC_PCR3_TAR_1 ((uint32_t)0x00004000)
- #define FSMC_PCR3_TAR_2 ((uint32_t)0x00008000)
- #define FSMC_PCR3_TAR_3 ((uint32_t)0x00010000)
- #define FSMC_PCR3_ECCPS ((uint32_t)0x000E0000)
- #define FSMC_PCR3_ECCPS_0 ((uint32_t)0x00020000)
- #define FSMC_PCR3_ECCPS_1 ((uint32_t)0x00040000)
- #define FSMC_PCR3_ECCPS_2 ((uint32_t)0x00080000)
- #define FSMC_PCR4_PWAITEN ((uint32_t)0x00000002)
- #define FSMC_PCR4_PBKEN ((uint32_t)0x00000004)
- #define FSMC_PCR4_PTYP ((uint32_t)0x00000008)
- #define FSMC_PCR4_PWID ((uint32_t)0x00000030)
- #define FSMC_PCR4_PWID_0 ((uint32_t)0x00000010)
- #define FSMC_PCR4_PWID_1 ((uint32_t)0x00000020)
- #define FSMC_PCR4_ECCEN ((uint32_t)0x00000040)
- #define FSMC_PCR4_TCLR ((uint32_t)0x00001E00)
- #define FSMC_PCR4_TCLR_0 ((uint32_t)0x00000200)
- #define FSMC_PCR4_TCLR_1 ((uint32_t)0x00000400)
- #define FSMC_PCR4_TCLR_2 ((uint32_t)0x00000800)
- #define FSMC_PCR4_TCLR_3 ((uint32_t)0x00001000)
- #define FSMC_PCR4_TAR ((uint32_t)0x0001E000)
- #define FSMC_PCR4_TAR_0 ((uint32_t)0x00002000)
- #define FSMC_PCR4_TAR_1 ((uint32_t)0x00004000)
- #define FSMC_PCR4_TAR_2 ((uint32_t)0x00008000)
- #define FSMC_PCR4_TAR_3 ((uint32_t)0x00010000)
- #define FSMC_PCR4_ECCPS ((uint32_t)0x000E0000)
- #define FSMC_PCR4_ECCPS_0 ((uint32_t)0x00020000)
- #define FSMC_PCR4_ECCPS_1 ((uint32_t)0x00040000)
- #define FSMC_PCR4_ECCPS_2 ((uint32_t)0x00080000)
- #define FSMC_SR2_IRS ((uint8_t)0x01)
- #define FSMC_SR2_ILS ((uint8_t)0x02)
- #define FSMC_SR2_IFS ((uint8_t)0x04)
- #define FSMC_SR2_IREN ((uint8_t)0x08)
- #define FSMC_SR2_ILEN ((uint8_t)0x10)
- #define FSMC_SR2_IFEN ((uint8_t)0x20)
- #define FSMC_SR2_FEMPT ((uint8_t)0x40)
- #define FSMC_SR3_IRS ((uint8_t)0x01)
- #define FSMC_SR3_ILS ((uint8_t)0x02)
- #define FSMC_SR3_IFS ((uint8_t)0x04)
- #define FSMC_SR3_IREN ((uint8_t)0x08)
- #define FSMC_SR3_ILEN ((uint8_t)0x10)
- #define FSMC_SR3_IFEN ((uint8_t)0x20)
- #define FSMC_SR3_FEMPT ((uint8_t)0x40)
- #define FSMC_SR4_IRS ((uint8_t)0x01)
- #define FSMC_SR4_ILS ((uint8_t)0x02)
- #define FSMC_SR4_IFS ((uint8_t)0x04)
- #define FSMC_SR4_IREN ((uint8_t)0x08)
- #define FSMC_SR4_ILEN ((uint8_t)0x10)
- #define FSMC_SR4_IFEN ((uint8_t)0x20)

- #define FSMC_SR4_FEMPT ((uint8_t)0x40)
- #define FSMC_PMEM2_MEMSET2 ((uint32_t)0x000000FF)
- #define FSMC_PMEM2_MEMSET2_0 ((uint32_t)0x00000001)
- #define FSMC_PMEM2_MEMSET2_1 ((uint32_t)0x00000002)
- #define FSMC_PMEM2_MEMSET2_2 ((uint32_t)0x00000004)
- #define FSMC_PMEM2_MEMSET2_3 ((uint32_t)0x00000008)
- #define FSMC_PMEM2_MEMSET2_4 ((uint32_t)0x00000010)
- #define FSMC_PMEM2_MEMSET2_5 ((uint32_t)0x00000020)
- #define FSMC_PMEM2_MEMSET2_6 ((uint32_t)0x00000040)
- #define FSMC_PMEM2_MEMSET2_7 ((uint32_t)0x00000080)
- #define FSMC_PMEM2_MEMWAIT2 ((uint32_t)0x0000FF00)
- #define FSMC_PMEM2_MEMWAIT2_0 ((uint32_t)0x00000100)
- #define FSMC_PMEM2_MEMWAIT2_1 ((uint32_t)0x00000200)
- #define FSMC_PMEM2_MEMWAIT2_2 ((uint32_t)0x00000400)
- #define FSMC_PMEM2_MEMWAIT2_3 ((uint32_t)0x00000800)
- #define FSMC_PMEM2_MEMWAIT2_4 ((uint32_t)0x00001000)
- #define FSMC_PMEM2_MEMWAIT2_5 ((uint32_t)0x00002000)
- #define FSMC_PMEM2_MEMWAIT2_6 ((uint32_t)0x00004000)
- #define FSMC_PMEM2_MEMWAIT2_7 ((uint32_t)0x00008000)
- #define FSMC_PMEM2_MEMHOLD2 ((uint32_t)0x00FF0000)
- #define FSMC_PMEM2_MEMHOLD2_0 ((uint32_t)0x00010000)
- #define FSMC_PMEM2_MEMHOLD2_1 ((uint32_t)0x00020000)
- #define FSMC_PMEM2_MEMHOLD2_2 ((uint32_t)0x00040000)
- #define FSMC_PMEM2_MEMHOLD2_3 ((uint32_t)0x00080000)
- #define FSMC_PMEM2_MEMHOLD2_4 ((uint32_t)0x00100000)
- #define FSMC_PMEM2_MEMHOLD2_5 ((uint32_t)0x00200000)
- #define FSMC_PMEM2_MEMHOLD2_6 ((uint32_t)0x00400000)
- #define FSMC_PMEM2_MEMHOLD2_7 ((uint32_t)0x00800000)
- #define FSMC_PMEM2_MEMHIZ2 ((uint32_t)0xFF000000)
- #define FSMC_PMEM2_MEMHIZ2_0 ((uint32_t)0x01000000)
- #define FSMC_PMEM2_MEMHIZ2_1 ((uint32_t)0x02000000)
- #define FSMC_PMEM2_MEMHIZ2_2 ((uint32_t)0x04000000)
- #define FSMC_PMEM2_MEMHIZ2_3 ((uint32_t)0x08000000)
- #define FSMC_PMEM2_MEMHIZ2_4 ((uint32_t)0x10000000)
- #define FSMC_PMEM2_MEMHIZ2_5 ((uint32_t)0x20000000)
- #define FSMC_PMEM2_MEMHIZ2_6 ((uint32_t)0x40000000)
- #define FSMC_PMEM2_MEMHIZ2_7 ((uint32_t)0x80000000)
- #define FSMC_PMEM3_MEMSET3 ((uint32_t)0x000000FF)
- #define FSMC_PMEM3_MEMSET3_0 ((uint32_t)0x00000001)
- #define FSMC_PMEM3_MEMSET3_1 ((uint32_t)0x00000002)
- #define FSMC_PMEM3_MEMSET3_2 ((uint32_t)0x00000004)
- #define FSMC_PMEM3_MEMSET3_3 ((uint32_t)0x00000008)
- #define FSMC_PMEM3_MEMSET3_4 ((uint32_t)0x00000010)
- #define FSMC_PMEM3_MEMSET3_5 ((uint32_t)0x00000020)
- #define FSMC_PMEM3_MEMSET3_6 ((uint32_t)0x00000040)
- #define FSMC_PMEM3_MEMSET3_7 ((uint32_t)0x00000080)
- #define FSMC_PMEM3_MEMWAIT3 ((uint32_t)0x0000FF00)
- #define FSMC_PMEM3_MEMWAIT3_0 ((uint32_t)0x00000100)
- #define FSMC_PMEM3_MEMWAIT3_1 ((uint32_t)0x00000200)
- #define FSMC_PMEM3_MEMWAIT3_2 ((uint32_t)0x00000400)
- #define FSMC_PMEM3_MEMWAIT3_3 ((uint32_t)0x00000800)
- #define FSMC_PMEM3_MEMWAIT3_4 ((uint32_t)0x00001000)
- #define FSMC_PMEM3_MEMWAIT3_5 ((uint32_t)0x00002000)
- #define FSMC_PMEM3_MEMWAIT3_6 ((uint32_t)0x00004000)
- #define FSMC_PMEM3_MEMWAIT3_7 ((uint32_t)0x00008000)

- #define FSMC_PMEM3_MEMHOLD3 ((uint32_t)0x00FF0000)
- #define FSMC_PMEM3_MEMHOLD3_0 ((uint32_t)0x00010000)
- #define FSMC_PMEM3_MEMHOLD3_1 ((uint32_t)0x00020000)
- #define FSMC_PMEM3_MEMHOLD3_2 ((uint32_t)0x00040000)
- #define FSMC_PMEM3_MEMHOLD3_3 ((uint32_t)0x00080000)
- #define FSMC_PMEM3_MEMHOLD3_4 ((uint32_t)0x00100000)
- #define FSMC_PMEM3_MEMHOLD3_5 ((uint32_t)0x00200000)
- #define FSMC_PMEM3_MEMHOLD3_6 ((uint32_t)0x00400000)
- #define FSMC_PMEM3_MEMHOLD3_7 ((uint32_t)0x00800000)
- #define FSMC_PMEM3_MEMHIZ3 ((uint32_t)0xFF000000)
- #define FSMC_PMEM3_MEMHIZ3_0 ((uint32_t)0x01000000)
- #define FSMC_PMEM3_MEMHIZ3_1 ((uint32_t)0x02000000)
- #define FSMC_PMEM3_MEMHIZ3_2 ((uint32_t)0x04000000)
- #define FSMC_PMEM3_MEMHIZ3_3 ((uint32_t)0x08000000)
- #define FSMC_PMEM3_MEMHIZ3_4 ((uint32_t)0x10000000)
- #define FSMC_PMEM3_MEMHIZ3_5 ((uint32_t)0x20000000)
- #define FSMC_PMEM3_MEMHIZ3_6 ((uint32_t)0x40000000)
- #define FSMC_PMEM3_MEMHIZ3_7 ((uint32_t)0x80000000)
- #define FSMC_PMEM4_MEMSET4 ((uint32_t)0x000000FF)
- #define FSMC_PMEM4_MEMSET4_0 ((uint32_t)0x00000001)
- #define FSMC_PMEM4_MEMSET4_1 ((uint32_t)0x00000002)
- #define FSMC_PMEM4_MEMSET4_2 ((uint32_t)0x00000004)
- #define FSMC_PMEM4_MEMSET4_3 ((uint32_t)0x00000008)
- #define FSMC_PMEM4_MEMSET4_4 ((uint32_t)0x00000010)
- #define FSMC_PMEM4_MEMSET4_5 ((uint32_t)0x00000020)
- #define FSMC_PMEM4_MEMSET4_6 ((uint32_t)0x00000040)
- #define FSMC_PMEM4_MEMSET4_7 ((uint32_t)0x00000080)
- #define FSMC_PMEM4_MEMWAIT4 ((uint32_t)0x0000FF00)
- #define FSMC_PMEM4_MEMWAIT4_0 ((uint32_t)0x00000100)
- #define FSMC_PMEM4_MEMWAIT4_1 ((uint32_t)0x00000200)
- #define FSMC_PMEM4_MEMWAIT4_2 ((uint32_t)0x00000400)
- #define FSMC_PMEM4_MEMWAIT4_3 ((uint32_t)0x00000800)
- #define FSMC_PMEM4_MEMWAIT4_4 ((uint32_t)0x00001000)
- #define FSMC_PMEM4_MEMWAIT4_5 ((uint32_t)0x00002000)
- #define FSMC_PMEM4_MEMWAIT4_6 ((uint32_t)0x00004000)
- #define FSMC_PMEM4_MEMWAIT4_7 ((uint32_t)0x00008000)
- #define FSMC_PMEM4_MEMHOLD4 ((uint32_t)0x00FF0000)
- #define FSMC_PMEM4_MEMHOLD4_0 ((uint32_t)0x00010000)
- #define FSMC_PMEM4_MEMHOLD4_1 ((uint32_t)0x00020000)
- #define FSMC_PMEM4_MEMHOLD4_2 ((uint32_t)0x00040000)
- #define FSMC_PMEM4_MEMHOLD4_3 ((uint32_t)0x00080000)
- #define FSMC_PMEM4_MEMHOLD4_4 ((uint32_t)0x00100000)
- #define FSMC_PMEM4_MEMHOLD4_5 ((uint32_t)0x00200000)
- #define FSMC_PMEM4_MEMHOLD4_6 ((uint32_t)0x00400000)
- #define FSMC_PMEM4_MEMHOLD4_7 ((uint32_t)0x00800000)
- #define FSMC_PMEM4_MEMHIZ4 ((uint32_t)0xFF000000)
- #define FSMC_PMEM4_MEMHIZ4_0 ((uint32_t)0x01000000)
- #define FSMC_PMEM4_MEMHIZ4_1 ((uint32_t)0x02000000)
- #define FSMC_PMEM4_MEMHIZ4_2 ((uint32_t)0x04000000)
- #define FSMC_PMEM4_MEMHIZ4_3 ((uint32_t)0x08000000)
- #define FSMC_PMEM4_MEMHIZ4_4 ((uint32_t)0x10000000)
- #define FSMC_PMEM4_MEMHIZ4_5 ((uint32_t)0x20000000)
- #define FSMC_PMEM4_MEMHIZ4_6 ((uint32_t)0x40000000)
- #define FSMC_PMEM4_MEMHIZ4_7 ((uint32_t)0x80000000)
- #define FSMC_PATT2_ATTSET2 ((uint32_t)0x000000FF)

- #define FSMC_PATT2_ATTSET2_0 ((uint32_t)0x00000001)
- #define FSMC_PATT2_ATTSET2_1 ((uint32_t)0x00000002)
- #define FSMC_PATT2_ATTSET2_2 ((uint32_t)0x00000004)
- #define FSMC_PATT2_ATTSET2_3 ((uint32_t)0x00000008)
- #define FSMC_PATT2_ATTSET2_4 ((uint32_t)0x00000010)
- #define FSMC_PATT2_ATTSET2_5 ((uint32_t)0x00000020)
- #define FSMC_PATT2_ATTSET2_6 ((uint32_t)0x00000040)
- #define FSMC_PATT2_ATTSET2_7 ((uint32_t)0x00000080)
- #define FSMC_PATT2_ATTWAIT2 ((uint32_t)0x0000FF00)
- #define FSMC_PATT2_ATTWAIT2_0 ((uint32_t)0x00000100)
- #define FSMC_PATT2_ATTWAIT2_1 ((uint32_t)0x00000200)
- #define FSMC_PATT2_ATTWAIT2_2 ((uint32_t)0x00000400)
- #define FSMC_PATT2_ATTWAIT2_3 ((uint32_t)0x00000800)
- #define FSMC_PATT2_ATTWAIT2_4 ((uint32_t)0x00001000)
- #define FSMC_PATT2_ATTWAIT2_5 ((uint32_t)0x00002000)
- #define FSMC_PATT2_ATTWAIT2_6 ((uint32_t)0x00004000)
- #define FSMC_PATT2_ATTWAIT2_7 ((uint32_t)0x00008000)
- #define FSMC_PATT2_ATTHOLD2 ((uint32_t)0x00FF0000)
- #define FSMC_PATT2_ATTHOLD2_0 ((uint32_t)0x00010000)
- #define FSMC_PATT2_ATTHOLD2_1 ((uint32_t)0x00020000)
- #define FSMC_PATT2_ATTHOLD2_2 ((uint32_t)0x00040000)
- #define FSMC_PATT2_ATTHOLD2_3 ((uint32_t)0x00080000)
- #define FSMC_PATT2_ATTHOLD2_4 ((uint32_t)0x00100000)
- #define FSMC_PATT2_ATTHOLD2_5 ((uint32_t)0x00200000)
- #define FSMC_PATT2_ATTHOLD2_6 ((uint32_t)0x00400000)
- #define FSMC_PATT2_ATTHOLD2_7 ((uint32_t)0x00800000)
- #define FSMC_PATT2_ATTHIZ2 ((uint32_t)0xFF000000)
- #define FSMC_PATT2_ATTHIZ2_0 ((uint32_t)0x01000000)
- #define FSMC_PATT2_ATTHIZ2_1 ((uint32_t)0x02000000)
- #define FSMC_PATT2_ATTHIZ2_2 ((uint32_t)0x04000000)
- #define FSMC_PATT2_ATTHIZ2_3 ((uint32_t)0x08000000)
- #define FSMC_PATT2_ATTHIZ2_4 ((uint32_t)0x10000000)
- #define FSMC_PATT2_ATTHIZ2_5 ((uint32_t)0x20000000)
- #define FSMC_PATT2_ATTHIZ2_6 ((uint32_t)0x40000000)
- #define FSMC_PATT2_ATTHIZ2_7 ((uint32_t)0x80000000)
- #define FSMC_PATT3_ATTSET3 ((uint32_t)0x000000FF)
- #define FSMC_PATT3_ATTSET3_0 ((uint32_t)0x00000001)
- #define FSMC_PATT3_ATTSET3_1 ((uint32_t)0x00000002)
- #define FSMC_PATT3_ATTSET3_2 ((uint32_t)0x00000004)
- #define FSMC_PATT3_ATTSET3_3 ((uint32_t)0x00000008)
- #define FSMC_PATT3_ATTSET3_4 ((uint32_t)0x00000010)
- #define FSMC_PATT3_ATTSET3_5 ((uint32_t)0x00000020)
- #define FSMC_PATT3_ATTSET3_6 ((uint32_t)0x00000040)
- #define FSMC_PATT3_ATTSET3_7 ((uint32_t)0x00000080)
- #define FSMC_PATT3_ATTWAIT3 ((uint32_t)0x0000FF00)
- #define FSMC_PATT3_ATTWAIT3_0 ((uint32_t)0x00000100)
- #define FSMC_PATT3_ATTWAIT3_1 ((uint32_t)0x00000200)
- #define FSMC_PATT3_ATTWAIT3_2 ((uint32_t)0x00000400)
- #define FSMC_PATT3_ATTWAIT3_3 ((uint32_t)0x00000800)
- #define FSMC_PATT3_ATTWAIT3_4 ((uint32_t)0x00001000)
- #define FSMC_PATT3_ATTWAIT3_5 ((uint32_t)0x00002000)
- #define FSMC_PATT3_ATTWAIT3_6 ((uint32_t)0x00004000)
- #define FSMC_PATT3_ATTWAIT3_7 ((uint32_t)0x00008000)
- #define FSMC_PATT3_ATTHOLD3 ((uint32_t)0x00FF0000)
- #define FSMC_PATT3_ATTHOLD3_0 ((uint32_t)0x00010000)

- #define FSMC_PATT3_ATTHOLD3_1 ((uint32_t)0x00020000)
- #define FSMC_PATT3_ATTHOLD3_2 ((uint32_t)0x00040000)
- #define FSMC_PATT3_ATTHOLD3_3 ((uint32_t)0x00080000)
- #define FSMC_PATT3_ATTHOLD3_4 ((uint32_t)0x00100000)
- #define FSMC_PATT3_ATTHOLD3_5 ((uint32_t)0x00200000)
- #define FSMC_PATT3_ATTHOLD3_6 ((uint32_t)0x00400000)
- #define FSMC_PATT3_ATTHOLD3_7 ((uint32_t)0x00800000)
- #define FSMC_PATT3_ATTHIZ3 ((uint32_t)0xFF000000)
- #define FSMC_PATT3_ATTHIZ3_0 ((uint32_t)0x01000000)
- #define FSMC_PATT3_ATTHIZ3_1 ((uint32_t)0x02000000)
- #define FSMC_PATT3_ATTHIZ3_2 ((uint32_t)0x04000000)
- #define FSMC_PATT3_ATTHIZ3_3 ((uint32_t)0x08000000)
- #define FSMC_PATT3_ATTHIZ3_4 ((uint32_t)0x10000000)
- #define FSMC_PATT3_ATTHIZ3_5 ((uint32_t)0x20000000)
- #define FSMC_PATT3_ATTHIZ3_6 ((uint32_t)0x40000000)
- #define FSMC_PATT3_ATTHIZ3_7 ((uint32_t)0x80000000)
- #define FSMC_PATT4_ATTSET4 ((uint32_t)0x000000FF)
- #define FSMC_PATT4_ATTSET4_0 ((uint32_t)0x00000001)
- #define FSMC_PATT4_ATTSET4_1 ((uint32_t)0x00000002)
- #define FSMC_PATT4_ATTSET4_2 ((uint32_t)0x00000004)
- #define FSMC_PATT4_ATTSET4_3 ((uint32_t)0x00000008)
- #define FSMC_PATT4_ATTSET4_4 ((uint32_t)0x00000010)
- #define FSMC_PATT4_ATTSET4_5 ((uint32_t)0x00000020)
- #define FSMC_PATT4_ATTSET4_6 ((uint32_t)0x00000040)
- #define FSMC_PATT4_ATTSET4_7 ((uint32_t)0x00000080)
- #define FSMC_PATT4_ATTWAIT4 ((uint32_t)0x0000FF00)
- #define FSMC_PATT4_ATTWAIT4_0 ((uint32_t)0x00000100)
- #define FSMC_PATT4_ATTWAIT4_1 ((uint32_t)0x00000200)
- #define FSMC_PATT4_ATTWAIT4_2 ((uint32_t)0x00000400)
- #define FSMC_PATT4_ATTWAIT4_3 ((uint32_t)0x00000800)
- #define FSMC_PATT4_ATTWAIT4_4 ((uint32_t)0x00001000)
- #define FSMC_PATT4_ATTWAIT4_5 ((uint32_t)0x00002000)
- #define FSMC_PATT4_ATTWAIT4_6 ((uint32_t)0x00004000)
- #define FSMC_PATT4_ATTWAIT4_7 ((uint32_t)0x00008000)
- #define FSMC_PATT4_ATTHOLD4 ((uint32_t)0x00FF0000)
- #define FSMC_PATT4_ATTHOLD4_0 ((uint32_t)0x00010000)
- #define FSMC_PATT4_ATTHOLD4_1 ((uint32_t)0x00020000)
- #define FSMC_PATT4_ATTHOLD4_2 ((uint32_t)0x00040000)
- #define FSMC_PATT4_ATTHOLD4_3 ((uint32_t)0x00080000)
- #define FSMC_PATT4_ATTHOLD4_4 ((uint32_t)0x00100000)
- #define FSMC_PATT4_ATTHOLD4_5 ((uint32_t)0x00200000)
- #define FSMC_PATT4_ATTHOLD4_6 ((uint32_t)0x00400000)
- #define FSMC_PATT4_ATTHOLD4_7 ((uint32_t)0x00800000)
- #define FSMC_PATT4_ATTHIZ4 ((uint32_t)0xFF000000)
- #define FSMC_PATT4_ATTHIZ4_0 ((uint32_t)0x01000000)
- #define FSMC_PATT4_ATTHIZ4_1 ((uint32_t)0x02000000)
- #define FSMC_PATT4_ATTHIZ4_2 ((uint32_t)0x04000000)
- #define FSMC_PATT4_ATTHIZ4_3 ((uint32_t)0x08000000)
- #define FSMC_PATT4_ATTHIZ4_4 ((uint32_t)0x10000000)
- #define FSMC_PATT4_ATTHIZ4_5 ((uint32_t)0x20000000)
- #define FSMC_PATT4_ATTHIZ4_6 ((uint32_t)0x40000000)
- #define FSMC_PATT4_ATTHIZ4_7 ((uint32_t)0x80000000)
- #define FSMC_PIO4_IOSET4 ((uint32_t)0x000000FF)
- #define FSMC_PIO4_IOSET4_0 ((uint32_t)0x00000001)
- #define FSMC_PIO4_IOSET4_1 ((uint32_t)0x00000002)

- #define FSMC_PIO4_IOSET4_2 ((uint32_t)0x00000004)
- #define FSMC_PIO4_IOSET4_3 ((uint32_t)0x00000008)
- #define FSMC_PIO4_IOSET4_4 ((uint32_t)0x00000010)
- #define FSMC_PIO4_IOSET4_5 ((uint32_t)0x00000020)
- #define FSMC_PIO4_IOSET4_6 ((uint32_t)0x00000040)
- #define FSMC_PIO4_IOSET4_7 ((uint32_t)0x00000080)
- #define FSMC_PIO4_IOWAIT4 ((uint32_t)0x0000FF00)
- #define FSMC_PIO4_IOWAIT4_0 ((uint32_t)0x00000100)
- #define FSMC_PIO4_IOWAIT4_1 ((uint32_t)0x00000200)
- #define FSMC_PIO4_IOWAIT4_2 ((uint32_t)0x00000400)
- #define FSMC_PIO4_IOWAIT4_3 ((uint32_t)0x00000800)
- #define FSMC_PIO4_IOWAIT4_4 ((uint32_t)0x00001000)
- #define FSMC_PIO4_IOWAIT4_5 ((uint32_t)0x00002000)
- #define FSMC_PIO4_IOWAIT4_6 ((uint32_t)0x00004000)
- #define FSMC_PIO4_IOWAIT4_7 ((uint32_t)0x00008000)
- #define FSMC_PIO4_IOHOLD4 ((uint32_t)0x00FF0000)
- #define FSMC_PIO4_IOHOLD4_0 ((uint32_t)0x00010000)
- #define FSMC_PIO4_IOHOLD4_1 ((uint32_t)0x00020000)
- #define FSMC_PIO4_IOHOLD4_2 ((uint32_t)0x00040000)
- #define FSMC_PIO4_IOHOLD4_3 ((uint32_t)0x00080000)
- #define FSMC_PIO4_IOHOLD4_4 ((uint32_t)0x00100000)
- #define FSMC_PIO4_IOHOLD4_5 ((uint32_t)0x00200000)
- #define FSMC_PIO4_IOHOLD4_6 ((uint32_t)0x00400000)
- #define FSMC_PIO4_IOHOLD4_7 ((uint32_t)0x00800000)
- #define FSMC_PIO4_IOHIZ4 ((uint32_t)0xFF000000)
- #define FSMC_PIO4_IOHIZ4_0 ((uint32_t)0x01000000)
- #define FSMC_PIO4_IOHIZ4_1 ((uint32_t)0x02000000)
- #define FSMC_PIO4_IOHIZ4_2 ((uint32_t)0x04000000)
- #define FSMC_PIO4_IOHIZ4_3 ((uint32_t)0x08000000)
- #define FSMC_PIO4_IOHIZ4_4 ((uint32_t)0x10000000)
- #define FSMC_PIO4_IOHIZ4_5 ((uint32_t)0x20000000)
- #define FSMC_PIO4_IOHIZ4_6 ((uint32_t)0x40000000)
- #define FSMC_PIO4_IOHIZ4_7 ((uint32_t)0x80000000)
- #define FSMC_ECCR2_ECC2 ((uint32_t)0xFFFFFFFF)
- #define FSMC_ECCR3_ECC3 ((uint32_t)0xFFFFFFFF)
- #define **GPIO_MODER_MODER0** ((uint32_t)0x00000003)
- #define **GPIO_MODER_MODER0_0** ((uint32_t)0x00000001)
- #define **GPIO_MODER_MODER0_1** ((uint32_t)0x00000002)
- #define **GPIO_MODER_MODER1** ((uint32_t)0x0000000C)
- #define **GPIO_MODER_MODER1_0** ((uint32_t)0x00000004)
- #define **GPIO_MODER_MODER1_1** ((uint32_t)0x00000008)
- #define **GPIO_MODER_MODER2** ((uint32_t)0x00000030)
- #define **GPIO_MODER_MODER2_0** ((uint32_t)0x00000010)
- #define **GPIO_MODER_MODER2_1** ((uint32_t)0x00000020)
- #define **GPIO_MODER_MODER3** ((uint32_t)0x000000C0)
- #define **GPIO_MODER_MODER3_0** ((uint32_t)0x00000040)
- #define **GPIO_MODER_MODER3_1** ((uint32_t)0x00000080)
- #define **GPIO_MODER_MODER4** ((uint32_t)0x00000300)
- #define **GPIO_MODER_MODER4_0** ((uint32_t)0x00000100)
- #define **GPIO_MODER_MODER4_1** ((uint32_t)0x00000200)
- #define **GPIO_MODER_MODER5** ((uint32_t)0x00000C00)
- #define **GPIO_MODER_MODER5_0** ((uint32_t)0x00000400)
- #define **GPIO_MODER_MODER5_1** ((uint32_t)0x00000800)
- #define **GPIO_MODER_MODER6** ((uint32_t)0x00003000)
- #define **GPIO_MODER_MODER6_0** ((uint32_t)0x00001000)

- #define **GPIO_MODER_MODER6_1** ((uint32_t)0x00002000)
- #define **GPIO_MODER_MODER7** ((uint32_t)0x0000C000)
- #define **GPIO_MODER_MODER7_0** ((uint32_t)0x00004000)
- #define **GPIO_MODER_MODER7_1** ((uint32_t)0x00008000)
- #define **GPIO_MODER_MODER8** ((uint32_t)0x00030000)
- #define **GPIO_MODER_MODER8_0** ((uint32_t)0x00010000)
- #define **GPIO_MODER_MODER8_1** ((uint32_t)0x00020000)
- #define **GPIO_MODER_MODER9** ((uint32_t)0x000C0000)
- #define **GPIO_MODER_MODER9_0** ((uint32_t)0x00040000)
- #define **GPIO_MODER_MODER9_1** ((uint32_t)0x00080000)
- #define **GPIO_MODER_MODER10** ((uint32_t)0x00300000)
- #define **GPIO_MODER_MODER10_0** ((uint32_t)0x00100000)
- #define **GPIO_MODER_MODER10_1** ((uint32_t)0x00200000)
- #define **GPIO_MODER_MODER11** ((uint32_t)0x00C00000)
- #define **GPIO_MODER_MODER11_0** ((uint32_t)0x00400000)
- #define **GPIO_MODER_MODER11_1** ((uint32_t)0x00800000)
- #define **GPIO_MODER_MODER12** ((uint32_t)0x03000000)
- #define **GPIO_MODER_MODER12_0** ((uint32_t)0x01000000)
- #define **GPIO_MODER_MODER12_1** ((uint32_t)0x02000000)
- #define **GPIO_MODER_MODER13** ((uint32_t)0x0C000000)
- #define **GPIO_MODER_MODER13_0** ((uint32_t)0x04000000)
- #define **GPIO_MODER_MODER13_1** ((uint32_t)0x08000000)
- #define **GPIO_MODER_MODER14** ((uint32_t)0x30000000)
- #define **GPIO_MODER_MODER14_0** ((uint32_t)0x10000000)
- #define **GPIO_MODER_MODER14_1** ((uint32_t)0x20000000)
- #define **GPIO_MODER_MODER15** ((uint32_t)0xC0000000)
- #define **GPIO_MODER_MODER15_0** ((uint32_t)0x40000000)
- #define **GPIO_MODER_MODER15_1** ((uint32_t)0x80000000)
- #define **GPIO_OTYPER_OT_0** ((uint32_t)0x00000001)
- #define **GPIO_OTYPER_OT_1** ((uint32_t)0x00000002)
- #define **GPIO_OTYPER_OT_2** ((uint32_t)0x00000004)
- #define **GPIO_OTYPER_OT_3** ((uint32_t)0x00000008)
- #define **GPIO_OTYPER_OT_4** ((uint32_t)0x00000010)
- #define **GPIO_OTYPER_OT_5** ((uint32_t)0x00000020)
- #define **GPIO_OTYPER_OT_6** ((uint32_t)0x00000040)
- #define **GPIO_OTYPER_OT_7** ((uint32_t)0x00000080)
- #define **GPIO_OTYPER_OT_8** ((uint32_t)0x00000100)
- #define **GPIO_OTYPER_OT_9** ((uint32_t)0x00000200)
- #define **GPIO_OTYPER_OT_10** ((uint32_t)0x00000400)
- #define **GPIO_OTYPER_OT_11** ((uint32_t)0x00000800)
- #define **GPIO_OTYPER_OT_12** ((uint32_t)0x00001000)
- #define **GPIO_OTYPER_OT_13** ((uint32_t)0x00002000)
- #define **GPIO_OTYPER_OT_14** ((uint32_t)0x00004000)
- #define **GPIO_OTYPER_OT_15** ((uint32_t)0x00008000)
- #define **GPIO_OSPEEDER_OSPEEDR0** ((uint32_t)0x00000003)
- #define **GPIO_OSPEEDER_OSPEEDR0_0** ((uint32_t)0x00000001)
- #define **GPIO_OSPEEDER_OSPEEDR0_1** ((uint32_t)0x00000002)
- #define **GPIO_OSPEEDER_OSPEEDR1** ((uint32_t)0x0000000C)
- #define **GPIO_OSPEEDER_OSPEEDR1_0** ((uint32_t)0x00000004)
- #define **GPIO_OSPEEDER_OSPEEDR1_1** ((uint32_t)0x00000008)
- #define **GPIO_OSPEEDER_OSPEEDR2** ((uint32_t)0x00000030)
- #define **GPIO_OSPEEDER_OSPEEDR2_0** ((uint32_t)0x00000010)
- #define **GPIO_OSPEEDER_OSPEEDR2_1** ((uint32_t)0x00000020)
- #define **GPIO_OSPEEDER_OSPEEDR3** ((uint32_t)0x000000C0)
- #define **GPIO_OSPEEDER_OSPEEDR3_0** ((uint32_t)0x00000040)

- #define **GPIO_OSPEEDER_OSPEEDR3_1** ((uint32_t)0x00000080)
- #define **GPIO_OSPEEDER_OSPEEDR4** ((uint32_t)0x00000300)
- #define **GPIO_OSPEEDER_OSPEEDR4_0** ((uint32_t)0x00000100)
- #define **GPIO_OSPEEDER_OSPEEDR4_1** ((uint32_t)0x00000200)
- #define **GPIO_OSPEEDER_OSPEEDR5** ((uint32_t)0x00000C00)
- #define **GPIO_OSPEEDER_OSPEEDR5_0** ((uint32_t)0x00000400)
- #define **GPIO_OSPEEDER_OSPEEDR5_1** ((uint32_t)0x00000800)
- #define **GPIO_OSPEEDER_OSPEEDR6** ((uint32_t)0x00003000)
- #define **GPIO_OSPEEDER_OSPEEDR6_0** ((uint32_t)0x00001000)
- #define **GPIO_OSPEEDER_OSPEEDR6_1** ((uint32_t)0x00002000)
- #define **GPIO_OSPEEDER_OSPEEDR7** ((uint32_t)0x0000C000)
- #define **GPIO_OSPEEDER_OSPEEDR7_0** ((uint32_t)0x00004000)
- #define **GPIO_OSPEEDER_OSPEEDR7_1** ((uint32_t)0x00008000)
- #define **GPIO_OSPEEDER_OSPEEDR8** ((uint32_t)0x00030000)
- #define **GPIO_OSPEEDER_OSPEEDR8_0** ((uint32_t)0x00010000)
- #define **GPIO_OSPEEDER_OSPEEDR8_1** ((uint32_t)0x00020000)
- #define **GPIO_OSPEEDER_OSPEEDR9** ((uint32_t)0x000C0000)
- #define **GPIO_OSPEEDER_OSPEEDR9_0** ((uint32_t)0x00040000)
- #define **GPIO_OSPEEDER_OSPEEDR9_1** ((uint32_t)0x00080000)
- #define **GPIO_OSPEEDER_OSPEEDR10** ((uint32_t)0x00300000)
- #define **GPIO_OSPEEDER_OSPEEDR10_0** ((uint32_t)0x00100000)
- #define **GPIO_OSPEEDER_OSPEEDR10_1** ((uint32_t)0x00200000)
- #define **GPIO_OSPEEDER_OSPEEDR11** ((uint32_t)0x00C00000)
- #define **GPIO_OSPEEDER_OSPEEDR11_0** ((uint32_t)0x00400000)
- #define **GPIO_OSPEEDER_OSPEEDR11_1** ((uint32_t)0x00800000)
- #define **GPIO_OSPEEDER_OSPEEDR12** ((uint32_t)0x03000000)
- #define **GPIO_OSPEEDER_OSPEEDR12_0** ((uint32_t)0x01000000)
- #define **GPIO_OSPEEDER_OSPEEDR12_1** ((uint32_t)0x02000000)
- #define **GPIO_OSPEEDER_OSPEEDR13** ((uint32_t)0x0C000000)
- #define **GPIO_OSPEEDER_OSPEEDR13_0** ((uint32_t)0x04000000)
- #define **GPIO_OSPEEDER_OSPEEDR13_1** ((uint32_t)0x08000000)
- #define **GPIO_OSPEEDER_OSPEEDR14** ((uint32_t)0x30000000)
- #define **GPIO_OSPEEDER_OSPEEDR14_0** ((uint32_t)0x10000000)
- #define **GPIO_OSPEEDER_OSPEEDR14_1** ((uint32_t)0x20000000)
- #define **GPIO_OSPEEDER_OSPEEDR15** ((uint32_t)0xC0000000)
- #define **GPIO_OSPEEDER_OSPEEDR15_0** ((uint32_t)0x40000000)
- #define **GPIO_OSPEEDER_OSPEEDR15_1** ((uint32_t)0x80000000)
- #define **GPIO_PUPDR_PUPDR0** ((uint32_t)0x00000003)
- #define **GPIO_PUPDR_PUPDR0_0** ((uint32_t)0x00000001)
- #define **GPIO_PUPDR_PUPDR0_1** ((uint32_t)0x00000002)
- #define **GPIO_PUPDR_PUPDR1** ((uint32_t)0x0000000C)
- #define **GPIO_PUPDR_PUPDR1_0** ((uint32_t)0x00000004)
- #define **GPIO_PUPDR_PUPDR1_1** ((uint32_t)0x00000008)
- #define **GPIO_PUPDR_PUPDR2** ((uint32_t)0x00000030)
- #define **GPIO_PUPDR_PUPDR2_0** ((uint32_t)0x00000010)
- #define **GPIO_PUPDR_PUPDR2_1** ((uint32_t)0x00000020)
- #define **GPIO_PUPDR_PUPDR3** ((uint32_t)0x000000C0)
- #define **GPIO_PUPDR_PUPDR3_0** ((uint32_t)0x00000040)
- #define **GPIO_PUPDR_PUPDR3_1** ((uint32_t)0x00000080)
- #define **GPIO_PUPDR_PUPDR4** ((uint32_t)0x00000300)
- #define **GPIO_PUPDR_PUPDR4_0** ((uint32_t)0x00000100)
- #define **GPIO_PUPDR_PUPDR4_1** ((uint32_t)0x00000200)
- #define **GPIO_PUPDR_PUPDR5** ((uint32_t)0x00000C00)
- #define **GPIO_PUPDR_PUPDR5_0** ((uint32_t)0x00000400)
- #define **GPIO_PUPDR_PUPDR5_1** ((uint32_t)0x00000800)

- #define **GPIO_PUPDR_PUPDR6** ((uint32_t)0x00003000)
- #define **GPIO_PUPDR_PUPDR6_0** ((uint32_t)0x00001000)
- #define **GPIO_PUPDR_PUPDR6_1** ((uint32_t)0x00002000)
- #define **GPIO_PUPDR_PUPDR7** ((uint32_t)0x0000C000)
- #define **GPIO_PUPDR_PUPDR7_0** ((uint32_t)0x00004000)
- #define **GPIO_PUPDR_PUPDR7_1** ((uint32_t)0x00008000)
- #define **GPIO_PUPDR_PUPDR8** ((uint32_t)0x00030000)
- #define **GPIO_PUPDR_PUPDR8_0** ((uint32_t)0x00010000)
- #define **GPIO_PUPDR_PUPDR8_1** ((uint32_t)0x00020000)
- #define **GPIO_PUPDR_PUPDR9** ((uint32_t)0x000C0000)
- #define **GPIO_PUPDR_PUPDR9_0** ((uint32_t)0x00040000)
- #define **GPIO_PUPDR_PUPDR9_1** ((uint32_t)0x00080000)
- #define **GPIO_PUPDR_PUPDR10** ((uint32_t)0x00300000)
- #define **GPIO_PUPDR_PUPDR10_0** ((uint32_t)0x00100000)
- #define **GPIO_PUPDR_PUPDR10_1** ((uint32_t)0x00200000)
- #define **GPIO_PUPDR_PUPDR11** ((uint32_t)0x00C00000)
- #define **GPIO_PUPDR_PUPDR11_0** ((uint32_t)0x00400000)
- #define **GPIO_PUPDR_PUPDR11_1** ((uint32_t)0x00800000)
- #define **GPIO_PUPDR_PUPDR12** ((uint32_t)0x03000000)
- #define **GPIO_PUPDR_PUPDR12_0** ((uint32_t)0x01000000)
- #define **GPIO_PUPDR_PUPDR12_1** ((uint32_t)0x02000000)
- #define **GPIO_PUPDR_PUPDR13** ((uint32_t)0x0C000000)
- #define **GPIO_PUPDR_PUPDR13_0** ((uint32_t)0x04000000)
- #define **GPIO_PUPDR_PUPDR13_1** ((uint32_t)0x08000000)
- #define **GPIO_PUPDR_PUPDR14** ((uint32_t)0x30000000)
- #define **GPIO_PUPDR_PUPDR14_0** ((uint32_t)0x10000000)
- #define **GPIO_PUPDR_PUPDR14_1** ((uint32_t)0x20000000)
- #define **GPIO_PUPDR_PUPDR15** ((uint32_t)0xC0000000)
- #define **GPIO_PUPDR_PUPDR15_0** ((uint32_t)0x40000000)
- #define **GPIO_PUPDR_PUPDR15_1** ((uint32_t)0x80000000)
- #define **GPIO_IDR_IDR_0** ((uint32_t)0x00000001)
- #define **GPIO_IDR_IDR_1** ((uint32_t)0x00000002)
- #define **GPIO_IDR_IDR_2** ((uint32_t)0x00000004)
- #define **GPIO_IDR_IDR_3** ((uint32_t)0x00000008)
- #define **GPIO_IDR_IDR_4** ((uint32_t)0x00000010)
- #define **GPIO_IDR_IDR_5** ((uint32_t)0x00000020)
- #define **GPIO_IDR_IDR_6** ((uint32_t)0x00000040)
- #define **GPIO_IDR_IDR_7** ((uint32_t)0x00000080)
- #define **GPIO_IDR_IDR_8** ((uint32_t)0x00000100)
- #define **GPIO_IDR_IDR_9** ((uint32_t)0x00000200)
- #define **GPIO_IDR_IDR_10** ((uint32_t)0x00000400)
- #define **GPIO_IDR_IDR_11** ((uint32_t)0x00000800)
- #define **GPIO_IDR_IDR_12** ((uint32_t)0x00001000)
- #define **GPIO_IDR_IDR_13** ((uint32_t)0x00002000)
- #define **GPIO_IDR_IDR_14** ((uint32_t)0x00004000)
- #define **GPIO_IDR_IDR_15** ((uint32_t)0x00008000)
- #define **GPIO_OTYPER_IDR_0** GPIO_IDR_IDR_0
- #define **GPIO_OTYPER_IDR_1** GPIO_IDR_IDR_1
- #define **GPIO_OTYPER_IDR_2** GPIO_IDR_IDR_2
- #define **GPIO_OTYPER_IDR_3** GPIO_IDR_IDR_3
- #define **GPIO_OTYPER_IDR_4** GPIO_IDR_IDR_4
- #define **GPIO_OTYPER_IDR_5** GPIO_IDR_IDR_5
- #define **GPIO_OTYPER_IDR_6** GPIO_IDR_IDR_6
- #define **GPIO_OTYPER_IDR_7** GPIO_IDR_IDR_7
- #define **GPIO_OTYPER_IDR_8** GPIO_IDR_IDR_8

- #define **GPIO_OTYPER_IDR_9** GPIO_IDR_IDR_9
- #define **GPIO_OTYPER_IDR_10** GPIO_IDR_IDR_10
- #define **GPIO_OTYPER_IDR_11** GPIO_IDR_IDR_11
- #define **GPIO_OTYPER_IDR_12** GPIO_IDR_IDR_12
- #define **GPIO_OTYPER_IDR_13** GPIO_IDR_IDR_13
- #define **GPIO_OTYPER_IDR_14** GPIO_IDR_IDR_14
- #define **GPIO_OTYPER_IDR_15** GPIO_IDR_IDR_15
- #define **GPIO_ODR_ODR_0** ((uint32_t)0x00000001)
- #define **GPIO_ODR_ODR_1** ((uint32_t)0x00000002)
- #define **GPIO_ODR_ODR_2** ((uint32_t)0x00000004)
- #define **GPIO_ODR_ODR_3** ((uint32_t)0x00000008)
- #define **GPIO_ODR_ODR_4** ((uint32_t)0x00000010)
- #define **GPIO_ODR_ODR_5** ((uint32_t)0x00000020)
- #define **GPIO_ODR_ODR_6** ((uint32_t)0x00000040)
- #define **GPIO_ODR_ODR_7** ((uint32_t)0x00000080)
- #define **GPIO_ODR_ODR_8** ((uint32_t)0x00000100)
- #define **GPIO_ODR_ODR_9** ((uint32_t)0x00000200)
- #define **GPIO_ODR_ODR_10** ((uint32_t)0x00000400)
- #define **GPIO_ODR_ODR_11** ((uint32_t)0x00000800)
- #define **GPIO_ODR_ODR_12** ((uint32_t)0x00001000)
- #define **GPIO_ODR_ODR_13** ((uint32_t)0x00002000)
- #define **GPIO_ODR_ODR_14** ((uint32_t)0x00004000)
- #define **GPIO_ODR_ODR_15** ((uint32_t)0x00008000)
- #define **GPIO_OTYPER_ODR_0** GPIO_ODR_ODR_0
- #define **GPIO_OTYPER_ODR_1** GPIO_ODR_ODR_1
- #define **GPIO_OTYPER_ODR_2** GPIO_ODR_ODR_2
- #define **GPIO_OTYPER_ODR_3** GPIO_ODR_ODR_3
- #define **GPIO_OTYPER_ODR_4** GPIO_ODR_ODR_4
- #define **GPIO_OTYPER_ODR_5** GPIO_ODR_ODR_5
- #define **GPIO_OTYPER_ODR_6** GPIO_ODR_ODR_6
- #define **GPIO_OTYPER_ODR_7** GPIO_ODR_ODR_7
- #define **GPIO_OTYPER_ODR_8** GPIO_ODR_ODR_8
- #define **GPIO_OTYPER_ODR_9** GPIO_ODR_ODR_9
- #define **GPIO_OTYPER_ODR_10** GPIO_ODR_ODR_10
- #define **GPIO_OTYPER_ODR_11** GPIO_ODR_ODR_11
- #define **GPIO_OTYPER_ODR_12** GPIO_ODR_ODR_12
- #define **GPIO_OTYPER_ODR_13** GPIO_ODR_ODR_13
- #define **GPIO_OTYPER_ODR_14** GPIO_ODR_ODR_14
- #define **GPIO_OTYPER_ODR_15** GPIO_ODR_ODR_15
- #define **GPIO_BSRR_BS_0** ((uint32_t)0x00000001)
- #define **GPIO_BSRR_BS_1** ((uint32_t)0x00000002)
- #define **GPIO_BSRR_BS_2** ((uint32_t)0x00000004)
- #define **GPIO_BSRR_BS_3** ((uint32_t)0x00000008)
- #define **GPIO_BSRR_BS_4** ((uint32_t)0x00000010)
- #define **GPIO_BSRR_BS_5** ((uint32_t)0x00000020)
- #define **GPIO_BSRR_BS_6** ((uint32_t)0x00000040)
- #define **GPIO_BSRR_BS_7** ((uint32_t)0x00000080)
- #define **GPIO_BSRR_BS_8** ((uint32_t)0x00000100)
- #define **GPIO_BSRR_BS_9** ((uint32_t)0x00000200)
- #define **GPIO_BSRR_BS_10** ((uint32_t)0x00000400)
- #define **GPIO_BSRR_BS_11** ((uint32_t)0x00000800)
- #define **GPIO_BSRR_BS_12** ((uint32_t)0x00001000)
- #define **GPIO_BSRR_BS_13** ((uint32_t)0x00002000)
- #define **GPIO_BSRR_BS_14** ((uint32_t)0x00004000)
- #define **GPIO_BSRR_BS_15** ((uint32_t)0x00008000)

- #define **GPIO_BSRR_BR_0** ((uint32_t)0x00010000)
- #define **GPIO_BSRR_BR_1** ((uint32_t)0x00020000)
- #define **GPIO_BSRR_BR_2** ((uint32_t)0x00040000)
- #define **GPIO_BSRR_BR_3** ((uint32_t)0x00080000)
- #define **GPIO_BSRR_BR_4** ((uint32_t)0x00100000)
- #define **GPIO_BSRR_BR_5** ((uint32_t)0x00200000)
- #define **GPIO_BSRR_BR_6** ((uint32_t)0x00400000)
- #define **GPIO_BSRR_BR_7** ((uint32_t)0x00800000)
- #define **GPIO_BSRR_BR_8** ((uint32_t)0x01000000)
- #define **GPIO_BSRR_BR_9** ((uint32_t)0x02000000)
- #define **GPIO_BSRR_BR_10** ((uint32_t)0x04000000)
- #define **GPIO_BSRR_BR_11** ((uint32_t)0x08000000)
- #define **GPIO_BSRR_BR_12** ((uint32_t)0x10000000)
- #define **GPIO_BSRR_BR_13** ((uint32_t)0x20000000)
- #define **GPIO_BSRR_BR_14** ((uint32_t)0x40000000)
- #define **GPIO_BSRR_BR_15** ((uint32_t)0x80000000)
- #define **HASH_CR_INIT** ((uint32_t)0x00000004)
- #define **HASH_CR_DMAE** ((uint32_t)0x00000008)
- #define **HASH_CR_DATATYPE** ((uint32_t)0x00000030)
- #define **HASH_CR_DATATYPE_0** ((uint32_t)0x00000010)
- #define **HASH_CR_DATATYPE_1** ((uint32_t)0x00000020)
- #define **HASH_CR_MODE** ((uint32_t)0x00000040)
- #define **HASH_CR_ALGO** ((uint32_t)0x00000080)
- #define **HASH_CR_NBW** ((uint32_t)0x00000F00)
- #define **HASH_CR_NBW_0** ((uint32_t)0x00000100)
- #define **HASH_CR_NBW_1** ((uint32_t)0x00000200)
- #define **HASH_CR_NBW_2** ((uint32_t)0x00000400)
- #define **HASH_CR_NBW_3** ((uint32_t)0x00000800)
- #define **HASH_CR_DINNE** ((uint32_t)0x00001000)
- #define **HASH_CR_LKEY** ((uint32_t)0x00010000)
- #define **HASH_STR_NBW** ((uint32_t)0x0000001F)
- #define **HASH_STR_NBW_0** ((uint32_t)0x00000001)
- #define **HASH_STR_NBW_1** ((uint32_t)0x00000002)
- #define **HASH_STR_NBW_2** ((uint32_t)0x00000004)
- #define **HASH_STR_NBW_3** ((uint32_t)0x00000008)
- #define **HASH_STR_NBW_4** ((uint32_t)0x00000010)
- #define **HASH_STR_DCAL** ((uint32_t)0x00000100)
- #define **HASH_IMR_DINIM** ((uint32_t)0x00000001)
- #define **HASH_IMR_DCIM** ((uint32_t)0x00000002)
- #define **HASH_SR_DINIS** ((uint32_t)0x00000001)
- #define **HASH_SR_DCIS** ((uint32_t)0x00000002)
- #define **HASH_SR_DMAS** ((uint32_t)0x00000004)
- #define **HASH_SR_BUSY** ((uint32_t)0x00000008)
- #define I2C_CR1_PE ((uint16_t)0x0001)
- #define I2C_CR1_SMBUS ((uint16_t)0x0002)
- #define I2C_CR1_SMBTYPE ((uint16_t)0x0008)
- #define I2C_CR1_ENARP ((uint16_t)0x0010)
- #define I2C_CR1_ENPEC ((uint16_t)0x0020)
- #define I2C_CR1_ENGC ((uint16_t)0x0040)
- #define I2C_CR1_NOSTRETCH ((uint16_t)0x0080)
- #define I2C_CR1_START ((uint16_t)0x0100)
- #define I2C_CR1_STOP ((uint16_t)0x0200)
- #define I2C_CR1_ACK ((uint16_t)0x0400)
- #define I2C_CR1_POS ((uint16_t)0x0800)
- #define I2C_CR1_PEC ((uint16_t)0x1000)

- #define I2C_CR1_ALERT ((uint16_t)0x2000)
- #define I2C_CR1_SWRST ((uint16_t)0x8000)
- #define I2C_CR2_FREQ ((uint16_t)0x003F)
- #define I2C_CR2_FREQ_0 ((uint16_t)0x0001)
- #define I2C_CR2_FREQ_1 ((uint16_t)0x0002)
- #define I2C_CR2_FREQ_2 ((uint16_t)0x0004)
- #define I2C_CR2_FREQ_3 ((uint16_t)0x0008)
- #define I2C_CR2_FREQ_4 ((uint16_t)0x0010)
- #define I2C_CR2_FREQ_5 ((uint16_t)0x0020)
- #define I2C_CR2_ITERREN ((uint16_t)0x0100)
- #define I2C_CR2_ITEVTEN ((uint16_t)0x0200)
- #define I2C_CR2_ITBUFEN ((uint16_t)0x0400)
- #define I2C_CR2_DMAEN ((uint16_t)0x0800)
- #define I2C_CR2_LAST ((uint16_t)0x1000)
- #define I2C_OAR1_ADD1_7 ((uint16_t)0x00FE)
- #define I2C_OAR1_ADD8_9 ((uint16_t)0x0300)
- #define I2C_OAR1_ADD0 ((uint16_t)0x0001)
- #define I2C_OAR1_ADD1 ((uint16_t)0x0002)
- #define I2C_OAR1_ADD2 ((uint16_t)0x0004)
- #define I2C_OAR1_ADD3 ((uint16_t)0x0008)
- #define I2C_OAR1_ADD4 ((uint16_t)0x0010)
- #define I2C_OAR1_ADD5 ((uint16_t)0x0020)
- #define I2C_OAR1_ADD6 ((uint16_t)0x0040)
- #define I2C_OAR1_ADD7 ((uint16_t)0x0080)
- #define I2C_OAR1_ADD8 ((uint16_t)0x0100)
- #define I2C_OAR1_ADD9 ((uint16_t)0x0200)
- #define I2C_OAR1_ADDMODE ((uint16_t)0x8000)
- #define I2C_OAR2_ENDUAL ((uint8_t)0x01)
- #define I2C_OAR2_ADD2 ((uint8_t)0xFE)
- #define I2C_DR_DR ((uint8_t)0xFF)
- #define I2C_SR1_SB ((uint16_t)0x0001)
- #define I2C_SR1_ADDR ((uint16_t)0x0002)
- #define I2C_SR1_BTF ((uint16_t)0x0004)
- #define I2C_SR1_ADD10 ((uint16_t)0x0008)
- #define I2C_SR1_STOPF ((uint16_t)0x0010)
- #define I2C_SR1_RXNE ((uint16_t)0x0040)
- #define I2C_SR1_TXE ((uint16_t)0x0080)
- #define I2C_SR1_BERR ((uint16_t)0x0100)
- #define I2C_SR1_ARLO ((uint16_t)0x0200)
- #define I2C_SR1_AF ((uint16_t)0x0400)
- #define I2C_SR1_OVR ((uint16_t)0x0800)
- #define I2C_SR1_PECERR ((uint16_t)0x1000)
- #define I2C_SR1_TIMEOUT ((uint16_t)0x4000)
- #define I2C_SR1_SMBALERT ((uint16_t)0x8000)
- #define I2C_SR2_MSL ((uint16_t)0x0001)
- #define I2C_SR2_BUSY ((uint16_t)0x0002)
- #define I2C_SR2_TRA ((uint16_t)0x0004)
- #define I2C_SR2_GENCALL ((uint16_t)0x0010)
- #define I2C_SR2_SMBDEFAULT ((uint16_t)0x0020)
- #define I2C_SR2_SMBHOST ((uint16_t)0x0040)
- #define I2C_SR2_DUALF ((uint16_t)0x0080)
- #define I2C_SR2_PEC ((uint16_t)0xFF00)
- #define I2C_CCR_CCR ((uint16_t)0x0FFF)
- #define I2C_CCR_DUTY ((uint16_t)0x4000)
- #define I2C_CCR_FS ((uint16_t)0x8000)

- #define I2C_TRISE_TRISE ((uint8_t)0x3F)
- #define IWDG_KR_KEY ((uint16_t)0xFFFF)
- #define IWDG_PR_PR ((uint8_t)0x07)
- #define IWDG_PR_PR_0 ((uint8_t)0x01)
- #define IWDG_PR_PR_1 ((uint8_t)0x02)
- #define IWDG_PR_PR_2 ((uint8_t)0x04)
- #define IWDG_RLR_RL ((uint16_t)0x0FFF)
- #define IWDG_SR_PVU ((uint8_t)0x01)
- #define IWDG_SR_RVU ((uint8_t)0x02)
- #define PWR_CR_LPDS ((uint16_t)0x0001)
- #define PWR_CR_PDDS ((uint16_t)0x0002)
- #define PWR_CR_CWUF ((uint16_t)0x0004)
- #define PWR_CR_CSBF ((uint16_t)0x0008)
- #define PWR_CR_PVDE ((uint16_t)0x0010)
- #define PWR_CR_PLS ((uint16_t)0x00E0)
- #define PWR_CR_PLS_0 ((uint16_t)0x0020)
- #define PWR_CR_PLS_1 ((uint16_t)0x0040)
- #define PWR_CR_PLS_2 ((uint16_t)0x0080)
- #define PWR_CR_PLS_LEV0 ((uint16_t)0x0000)
- #define PWR_CR_PLS_LEV1 ((uint16_t)0x0020)
- #define PWR_CR_PLS_LEV2 ((uint16_t)0x0040)
- #define PWR_CR_PLS_LEV3 ((uint16_t)0x0060)
- #define PWR_CR_PLS_LEV4 ((uint16_t)0x0080)
- #define PWR_CR_PLS_LEV5 ((uint16_t)0x00A0)
- #define PWR_CR_PLS_LEV6 ((uint16_t)0x00C0)
- #define PWR_CR_PLS_LEV7 ((uint16_t)0x00E0)
- #define PWR_CR_DBP ((uint16_t)0x0100)
- #define PWR_CR_FPDS ((uint16_t)0x0200)
- #define PWR_CR_VOS ((uint16_t)0x4000)
- #define **PWR_CR_PMODE** PWR_CR_VOS
- #define PWR_CSR_WUF ((uint16_t)0x0001)
- #define PWR_CSR_SBF ((uint16_t)0x0002)
- #define PWR_CSR_PVDO ((uint16_t)0x0004)
- #define PWR_CSR_BRR ((uint16_t)0x0008)
- #define PWR_CSR_EWUP ((uint16_t)0x0100)
- #define PWR_CSR_BRE ((uint16_t)0x0200)
- #define PWR_CSR_VOSRDY ((uint16_t)0x4000)
- #define **PWR_CSR_REGRDY** PWR_CSR_VOSRDY
- #define **RCC_CR_HSION** ((uint32_t)0x00000001)
- #define **RCC_CR_HSIRDY** ((uint32_t)0x00000002)
- #define **RCC_CR_HSITRIM** ((uint32_t)0x000000F8)
- #define RCC_CR_HSITRIM_0 ((uint32_t)0x00000008)
- #define RCC_CR_HSITRIM_1 ((uint32_t)0x00000010)
- #define RCC_CR_HSITRIM_2 ((uint32_t)0x00000020)
- #define RCC_CR_HSITRIM_3 ((uint32_t)0x00000040)
- #define RCC_CR_HSITRIM_4 ((uint32_t)0x00000080)
- #define **RCC_CR_HSICAL** ((uint32_t)0x0000FF00)
- #define RCC_CR_HSICAL_0 ((uint32_t)0x00000100)
- #define RCC_CR_HSICAL_1 ((uint32_t)0x00000200)
- #define RCC_CR_HSICAL_2 ((uint32_t)0x00000400)
- #define RCC_CR_HSICAL_3 ((uint32_t)0x00000800)
- #define RCC_CR_HSICAL_4 ((uint32_t)0x00001000)
- #define RCC_CR_HSICAL_5 ((uint32_t)0x00002000)
- #define RCC_CR_HSICAL_6 ((uint32_t)0x00004000)
- #define RCC_CR_HSICAL_7 ((uint32_t)0x00008000)

- #define **RCC_CR_HSEON** ((uint32_t)0x00010000)
- #define **RCC_CR_HSERDY** ((uint32_t)0x00020000)
- #define **RCC_CR_HSEBYP** ((uint32_t)0x00040000)
- #define **RCC_CR_CSSON** ((uint32_t)0x00080000)
- #define **RCC_CR_PLLON** ((uint32_t)0x01000000)
- #define **RCC_CR_PLLRDY** ((uint32_t)0x02000000)
- #define **RCC_CR_PLLI2SON** ((uint32_t)0x04000000)
- #define **RCC_CR_PLLI2SRDY** ((uint32_t)0x08000000)
- #define **RCC_PLLCFGR_PLLM** ((uint32_t)0x0000003F)
- #define **RCC_PLLCFGR_PLLM_0** ((uint32_t)0x00000001)
- #define **RCC_PLLCFGR_PLLM_1** ((uint32_t)0x00000002)
- #define **RCC_PLLCFGR_PLLM_2** ((uint32_t)0x00000004)
- #define **RCC_PLLCFGR_PLLM_3** ((uint32_t)0x00000008)
- #define **RCC_PLLCFGR_PLLM_4** ((uint32_t)0x00000010)
- #define **RCC_PLLCFGR_PLLM_5** ((uint32_t)0x00000020)
- #define **RCC_PLLCFGR_PLLN** ((uint32_t)0x00007FC0)
- #define **RCC_PLLCFGR_PLLN_0** ((uint32_t)0x00000040)
- #define **RCC_PLLCFGR_PLLN_1** ((uint32_t)0x00000080)
- #define **RCC_PLLCFGR_PLLN_2** ((uint32_t)0x00000100)
- #define **RCC_PLLCFGR_PLLN_3** ((uint32_t)0x00000200)
- #define **RCC_PLLCFGR_PLLN_4** ((uint32_t)0x00000400)
- #define **RCC_PLLCFGR_PLLN_5** ((uint32_t)0x00000800)
- #define **RCC_PLLCFGR_PLLN_6** ((uint32_t)0x00001000)
- #define **RCC_PLLCFGR_PLLN_7** ((uint32_t)0x00002000)
- #define **RCC_PLLCFGR_PLLN_8** ((uint32_t)0x00004000)
- #define **RCC_PLLCFGR_PLLP** ((uint32_t)0x00030000)
- #define **RCC_PLLCFGR_PLLP_0** ((uint32_t)0x00010000)
- #define **RCC_PLLCFGR_PLLP_1** ((uint32_t)0x00020000)
- #define **RCC_PLLCFGR_PLLSRC** ((uint32_t)0x00400000)
- #define **RCC_PLLCFGR_PLLSRC_HSE** ((uint32_t)0x00400000)
- #define **RCC_PLLCFGR_PLLSRC_HSI** ((uint32_t)0x00000000)
- #define **RCC_PLLCFGR_PLLQ** ((uint32_t)0x0F000000)
- #define **RCC_PLLCFGR_PLLQ_0** ((uint32_t)0x01000000)
- #define **RCC_PLLCFGR_PLLQ_1** ((uint32_t)0x02000000)
- #define **RCC_PLLCFGR_PLLQ_2** ((uint32_t)0x04000000)
- #define **RCC_PLLCFGR_PLLQ_3** ((uint32_t)0x08000000)
- #define RCC_CFGR_SW ((uint32_t)0x00000003)
- #define RCC_CFGR_SW_0 ((uint32_t)0x00000001)
- #define RCC_CFGR_SW_1 ((uint32_t)0x00000002)
- #define RCC_CFGR_SW_HSI ((uint32_t)0x00000000)
- #define RCC_CFGR_SW_HSE ((uint32_t)0x00000001)
- #define RCC_CFGR_SW_PLL ((uint32_t)0x00000002)
- #define RCC_CFGR_SWS ((uint32_t)0x0000000C)
- #define RCC_CFGR_SWS_0 ((uint32_t)0x00000004)
- #define RCC_CFGR_SWS_1 ((uint32_t)0x00000008)
- #define RCC_CFGR_SWS_HSI ((uint32_t)0x00000000)
- #define RCC_CFGR_SWS_HSE ((uint32_t)0x00000004)
- #define RCC_CFGR_SWS_PLL ((uint32_t)0x00000008)
- #define RCC_CFGR_HPRE ((uint32_t)0x000000F0)
- #define RCC_CFGR_HPRE_0 ((uint32_t)0x00000010)
- #define RCC_CFGR_HPRE_1 ((uint32_t)0x00000020)
- #define RCC_CFGR_HPRE_2 ((uint32_t)0x00000040)
- #define RCC_CFGR_HPRE_3 ((uint32_t)0x00000080)
- #define RCC_CFGR_HPRE_DIV1 ((uint32_t)0x00000000)
- #define RCC_CFGR_HPRE_DIV2 ((uint32_t)0x00000080)

- #define RCC_CFGR_HPRE_DIV4 ((uint32_t)0x00000090)
- #define RCC_CFGR_HPRE_DIV8 ((uint32_t)0x000000A0)
- #define RCC_CFGR_HPRE_DIV16 ((uint32_t)0x000000B0)
- #define RCC_CFGR_HPRE_DIV64 ((uint32_t)0x000000C0)
- #define RCC_CFGR_HPRE_DIV128 ((uint32_t)0x000000D0)
- #define RCC_CFGR_HPRE_DIV256 ((uint32_t)0x000000E0)
- #define RCC_CFGR_HPRE_DIV512 ((uint32_t)0x000000F0)
- #define RCC_CFGR_PPRE1 ((uint32_t)0x00001C00)
- #define RCC_CFGR_PPRE1_0 ((uint32_t)0x00000400)
- #define RCC_CFGR_PPRE1_1 ((uint32_t)0x00000800)
- #define RCC_CFGR_PPRE1_2 ((uint32_t)0x00001000)
- #define RCC_CFGR_PPRE1_DIV1 ((uint32_t)0x00000000)
- #define RCC_CFGR_PPRE1_DIV2 ((uint32_t)0x00001000)
- #define RCC_CFGR_PPRE1_DIV4 ((uint32_t)0x00001400)
- #define RCC_CFGR_PPRE1_DIV8 ((uint32_t)0x00001800)
- #define RCC_CFGR_PPRE1_DIV16 ((uint32_t)0x00001C00)
- #define RCC_CFGR_PPRE2 ((uint32_t)0x0000E000)
- #define RCC_CFGR_PPRE2_0 ((uint32_t)0x00002000)
- #define RCC_CFGR_PPRE2_1 ((uint32_t)0x00004000)
- #define RCC_CFGR_PPRE2_2 ((uint32_t)0x00008000)
- #define RCC_CFGR_PPRE2_DIV1 ((uint32_t)0x00000000)
- #define RCC_CFGR_PPRE2_DIV2 ((uint32_t)0x00008000)
- #define RCC_CFGR_PPRE2_DIV4 ((uint32_t)0x0000A000)
- #define RCC_CFGR_PPRE2_DIV8 ((uint32_t)0x0000C000)
- #define RCC_CFGR_PPRE2_DIV16 ((uint32_t)0x0000E000)
- #define **RCC_CFGR_RTCPRE** ((uint32_t)0x001F0000)
- #define **RCC_CFGR_RTCPRE_0** ((uint32_t)0x00010000)
- #define **RCC_CFGR_RTCPRE_1** ((uint32_t)0x00020000)
- #define **RCC_CFGR_RTCPRE_2** ((uint32_t)0x00040000)
- #define **RCC_CFGR_RTCPRE_3** ((uint32_t)0x00080000)
- #define RCC_CFGR_RTCPRE_4 ((uint32_t)0x00100000)
- #define **RCC_CFGR_MCO1** ((uint32_t)0x00600000)
- #define **RCC_CFGR_MCO1_0** ((uint32_t)0x00200000)
- #define **RCC_CFGR_MCO1_1** ((uint32_t)0x00400000)
- #define **RCC_CFGR_I2SSRC** ((uint32_t)0x00800000)
- #define **RCC_CFGR_MCO1PRE** ((uint32_t)0x07000000)
- #define **RCC_CFGR_MCO1PRE_0** ((uint32_t)0x01000000)
- #define **RCC_CFGR_MCO1PRE_1** ((uint32_t)0x02000000)
- #define **RCC_CFGR_MCO1PRE_2** ((uint32_t)0x04000000)
- #define **RCC_CFGR_MCO2PRE** ((uint32_t)0x38000000)
- #define **RCC_CFGR_MCO2PRE_0** ((uint32_t)0x08000000)
- #define **RCC_CFGR_MCO2PRE_1** ((uint32_t)0x10000000)
- #define **RCC_CFGR_MCO2PRE_2** ((uint32_t)0x20000000)
- #define **RCC_CFGR_MCO2** ((uint32_t)0xC0000000)
- #define **RCC_CFGR_MCO2_0** ((uint32_t)0x40000000)
- #define **RCC_CFGR_MCO2_1** ((uint32_t)0x80000000)
- #define **RCC_CIR_LSIRDYF** ((uint32_t)0x00000001)
- #define **RCC_CIR_LSERDYF** ((uint32_t)0x00000002)
- #define **RCC_CIR_HSIRDYF** ((uint32_t)0x00000004)
- #define **RCC_CIR_HSERDYF** ((uint32_t)0x00000008)
- #define **RCC_CIR_PLLRDYF** ((uint32_t)0x00000010)
- #define **RCC_CIR_PLLI2SRDYF** ((uint32_t)0x00000020)
- #define **RCC_CIR_CSSF** ((uint32_t)0x00000080)
- #define **RCC_CIR_LSIRDYIE** ((uint32_t)0x00000100)
- #define **RCC_CIR_LSERDYIE** ((uint32_t)0x00000200)

- #define **RCC_CIR_HSIRDYIE** ((uint32_t)0x00000400)
- #define **RCC_CIR_HSERDYIE** ((uint32_t)0x00000800)
- #define **RCC_CIR_PLLRDYIE** ((uint32_t)0x00001000)
- #define **RCC_CIR_PLLI2SRDYIE** ((uint32_t)0x00002000)
- #define **RCC_CIR_LSIRDYC** ((uint32_t)0x00010000)
- #define **RCC_CIR_LSERDYC** ((uint32_t)0x00020000)
- #define **RCC_CIR_HSIRDYC** ((uint32_t)0x00040000)
- #define **RCC_CIR_HSERDYC** ((uint32_t)0x00080000)
- #define **RCC_CIR_PLLRDYC** ((uint32_t)0x00100000)
- #define **RCC_CIR_PLLI2SRDYC** ((uint32_t)0x00200000)
- #define **RCC_CIR_CSSC** ((uint32_t)0x00800000)
- #define **RCC_AHB1RSTR_GPIOARST** ((uint32_t)0x00000001)
- #define **RCC_AHB1RSTR_GPIOBRST** ((uint32_t)0x00000002)
- #define **RCC_AHB1RSTR_GPIOCRST** ((uint32_t)0x00000004)
- #define **RCC_AHB1RSTR_GPIODRST** ((uint32_t)0x00000008)
- #define **RCC_AHB1RSTR_GPIOERST** ((uint32_t)0x00000010)
- #define **RCC_AHB1RSTR_GPIOFRST** ((uint32_t)0x00000020)
- #define **RCC_AHB1RSTR_GPIOGRST** ((uint32_t)0x00000040)
- #define **RCC_AHB1RSTR_GPIOHRST** ((uint32_t)0x00000080)
- #define **RCC_AHB1RSTR_GPIOIRST** ((uint32_t)0x00000100)
- #define **RCC_AHB1RSTR_CRCRST** ((uint32_t)0x00001000)
- #define **RCC_AHB1RSTR_DMA1RST** ((uint32_t)0x00200000)
- #define **RCC_AHB1RSTR_DMA2RST** ((uint32_t)0x00400000)
- #define **RCC_AHB1RSTR_ETHMACRST** ((uint32_t)0x02000000)
- #define **RCC_AHB1RSTR_OTGHRST** ((uint32_t)0x10000000)
- #define **RCC_AHB2RSTR_DCMIRST** ((uint32_t)0x00000001)
- #define **RCC_AHB2RSTR_CRYPRST** ((uint32_t)0x00000010)
- #define **RCC_AHB2RSTR_HSAHRST** ((uint32_t)0x00000020)
- #define **RCC_AHB2RSTR_RNGRST** ((uint32_t)0x00000040)
- #define **RCC_AHB2RSTR_OTGFSRST** ((uint32_t)0x00000080)
- #define **RCC_AHB3RSTR_FSMCRST** ((uint32_t)0x00000001)
- #define **RCC_APB1RSTR_TIM2RST** ((uint32_t)0x00000001)
- #define **RCC_APB1RSTR_TIM3RST** ((uint32_t)0x00000002)
- #define **RCC_APB1RSTR_TIM4RST** ((uint32_t)0x00000004)
- #define **RCC_APB1RSTR_TIM5RST** ((uint32_t)0x00000008)
- #define **RCC_APB1RSTR_TIM6RST** ((uint32_t)0x00000010)
- #define **RCC_APB1RSTR_TIM7RST** ((uint32_t)0x00000020)
- #define **RCC_APB1RSTR_TIM12RST** ((uint32_t)0x00000040)
- #define **RCC_APB1RSTR_TIM13RST** ((uint32_t)0x00000080)
- #define **RCC_APB1RSTR_TIM14RST** ((uint32_t)0x00000100)
- #define **RCC_APB1RSTR_WWDGEN** ((uint32_t)0x00000800)
- #define **RCC_APB1RSTR_SPI2RST** ((uint32_t)0x00008000)
- #define **RCC_APB1RSTR_SPI3RST** ((uint32_t)0x00010000)
- #define **RCC_APB1RSTR_USART2RST** ((uint32_t)0x00020000)
- #define **RCC_APB1RSTR_USART3RST** ((uint32_t)0x00040000)
- #define **RCC_APB1RSTR_UART4RST** ((uint32_t)0x00080000)
- #define **RCC_APB1RSTR_UART5RST** ((uint32_t)0x00100000)
- #define **RCC_APB1RSTR_I2C1RST** ((uint32_t)0x00200000)
- #define **RCC_APB1RSTR_I2C2RST** ((uint32_t)0x00400000)
- #define **RCC_APB1RSTR_I2C3RST** ((uint32_t)0x00800000)
- #define **RCC_APB1RSTR_CAN1RST** ((uint32_t)0x02000000)
- #define **RCC_APB1RSTR_CAN2RST** ((uint32_t)0x04000000)
- #define **RCC_APB1RSTR_PWRRST** ((uint32_t)0x10000000)
- #define **RCC_APB1RSTR_DACRST** ((uint32_t)0x20000000)
- #define **RCC_APB2RSTR_TIM1RST** ((uint32_t)0x00000001)

- #define **RCC_APB2RSTR_TIM8RST** ((uint32_t)0x00000002)
- #define **RCC_APB2RSTR_USART1RST** ((uint32_t)0x00000010)
- #define **RCC_APB2RSTR_USART6RST** ((uint32_t)0x00000020)
- #define **RCC_APB2RSTR_ADCRST** ((uint32_t)0x00000100)
- #define **RCC_APB2RSTR_SDIORST** ((uint32_t)0x00000800)
- #define **RCC_APB2RSTR_SPI1RST** ((uint32_t)0x00001000)
- #define **RCC_APB2RSTR_SYSCFGRST** ((uint32_t)0x00004000)
- #define **RCC_APB2RSTR_TIM9RST** ((uint32_t)0x00010000)
- #define **RCC_APB2RSTR_TIM10RST** ((uint32_t)0x00020000)
- #define **RCC_APB2RSTR_TIM11RST** ((uint32_t)0x00040000)
- #define **RCC_APB2RSTR_SPI1** RCC_APB2RSTR_SPI1RST
- #define **RCC_AHB1ENR_GPIOAEN** ((uint32_t)0x00000001)
- #define **RCC_AHB1ENR_GPIOBEN** ((uint32_t)0x00000002)
- #define **RCC_AHB1ENR_GPIOCEN** ((uint32_t)0x00000004)
- #define **RCC_AHB1ENR_GPIODEN** ((uint32_t)0x00000008)
- #define **RCC_AHB1ENR_GPIOEEN** ((uint32_t)0x00000010)
- #define **RCC_AHB1ENR_GPIOFEN** ((uint32_t)0x00000020)
- #define **RCC_AHB1ENR_GPIOGEN** ((uint32_t)0x00000040)
- #define **RCC_AHB1ENR_GPIOHEN** ((uint32_t)0x00000080)
- #define **RCC_AHB1ENR_GPIOIEN** ((uint32_t)0x00000100)
- #define **RCC_AHB1ENR_CRCEN** ((uint32_t)0x00001000)
- #define **RCC_AHB1ENR_BKPSRAMEN** ((uint32_t)0x00040000)
- #define **RCC_AHB1ENR_CCMDATARAMEN** ((uint32_t)0x00100000)
- #define **RCC_AHB1ENR_DMA1EN** ((uint32_t)0x00200000)
- #define **RCC_AHB1ENR_DMA2EN** ((uint32_t)0x00400000)
- #define **RCC_AHB1ENR_ETHMACEN** ((uint32_t)0x02000000)
- #define **RCC_AHB1ENR_ETHMACTXEN** ((uint32_t)0x04000000)
- #define **RCC_AHB1ENR_ETHMACRXEN** ((uint32_t)0x08000000)
- #define **RCC_AHB1ENR_ETHMACPTPEN** ((uint32_t)0x10000000)
- #define **RCC_AHB1ENR_OTGHSEN** ((uint32_t)0x20000000)
- #define **RCC_AHB1ENR_OTGHSULPIEN** ((uint32_t)0x40000000)
- #define **RCC_AHB2ENR_DCMIEN** ((uint32_t)0x00000001)
- #define **RCC_AHB2ENR_CRYPEN** ((uint32_t)0x00000010)
- #define **RCC_AHB2ENR_HASHEN** ((uint32_t)0x00000020)
- #define **RCC_AHB2ENR_RNGEN** ((uint32_t)0x00000040)
- #define **RCC_AHB2ENR_OTGFSEN** ((uint32_t)0x00000080)
- #define **RCC_AHB3ENR_FSMCEN** ((uint32_t)0x00000001)
- #define **RCC_APB1ENR_TIM2EN** ((uint32_t)0x00000001)
- #define **RCC_APB1ENR_TIM3EN** ((uint32_t)0x00000002)
- #define **RCC_APB1ENR_TIM4EN** ((uint32_t)0x00000004)
- #define **RCC_APB1ENR_TIM5EN** ((uint32_t)0x00000008)
- #define **RCC_APB1ENR_TIM6EN** ((uint32_t)0x00000010)
- #define **RCC_APB1ENR_TIM7EN** ((uint32_t)0x00000020)
- #define **RCC_APB1ENR_TIM12EN** ((uint32_t)0x00000040)
- #define **RCC_APB1ENR_TIM13EN** ((uint32_t)0x00000080)
- #define **RCC_APB1ENR_TIM14EN** ((uint32_t)0x00000100)
- #define **RCC_APB1ENR_WWDGEN** ((uint32_t)0x00000800)
- #define **RCC_APB1ENR_SPI2EN** ((uint32_t)0x00004000)
- #define **RCC_APB1ENR_SPI3EN** ((uint32_t)0x00008000)
- #define **RCC_APB1ENR_USART2EN** ((uint32_t)0x00020000)
- #define **RCC_APB1ENR_USART3EN** ((uint32_t)0x00040000)
- #define **RCC_APB1ENR_UART4EN** ((uint32_t)0x00080000)
- #define **RCC_APB1ENR_UART5EN** ((uint32_t)0x00100000)
- #define **RCC_APB1ENR_I2C1EN** ((uint32_t)0x00200000)
- #define **RCC_APB1ENR_I2C2EN** ((uint32_t)0x00400000)

- #define **RCC_APB1ENR_I2C3EN** ((uint32_t)0x00800000)
- #define **RCC_APB1ENR_CAN1EN** ((uint32_t)0x02000000)
- #define **RCC_APB1ENR_CAN2EN** ((uint32_t)0x04000000)
- #define **RCC_APB1ENR_PWREN** ((uint32_t)0x10000000)
- #define **RCC_APB1ENR_DACEN** ((uint32_t)0x20000000)
- #define **RCC_APB2ENR_TIM1EN** ((uint32_t)0x00000001)
- #define **RCC_APB2ENR_TIM8EN** ((uint32_t)0x00000002)
- #define **RCC_APB2ENR_USART1EN** ((uint32_t)0x00000010)
- #define **RCC_APB2ENR_USART6EN** ((uint32_t)0x00000020)
- #define **RCC_APB2ENR_ADC1EN** ((uint32_t)0x00000100)
- #define **RCC_APB2ENR_ADC2EN** ((uint32_t)0x00000200)
- #define **RCC_APB2ENR_ADC3EN** ((uint32_t)0x00000400)
- #define **RCC_APB2ENR_SDIOEN** ((uint32_t)0x00000800)
- #define **RCC_APB2ENR_SPI1EN** ((uint32_t)0x00001000)
- #define **RCC_APB2ENR_SYSCFGEN** ((uint32_t)0x00004000)
- #define **RCC_APB2ENR_TIM11EN** ((uint32_t)0x00040000)
- #define **RCC_APB2ENR_TIM10EN** ((uint32_t)0x00020000)
- #define **RCC_APB2ENR_TIM9EN** ((uint32_t)0x00010000)
- #define **RCC_AHB1LPENR_GPIOALPEN** ((uint32_t)0x00000001)
- #define **RCC_AHB1LPENR_GPIOBLPEN** ((uint32_t)0x00000002)
- #define **RCC_AHB1LPENR_GPIOCLPEN** ((uint32_t)0x00000004)
- #define **RCC_AHB1LPENR_GPIODLPEN** ((uint32_t)0x00000008)
- #define **RCC_AHB1LPENR_GPIOELPEN** ((uint32_t)0x00000010)
- #define **RCC_AHB1LPENR_GPIOFLPEN** ((uint32_t)0x00000020)
- #define **RCC_AHB1LPENR_GPIOGLPEN** ((uint32_t)0x00000040)
- #define **RCC_AHB1LPENR_GPIOHLPEN** ((uint32_t)0x00000080)
- #define **RCC_AHB1LPENR_GPIOILPEN** ((uint32_t)0x00000100)
- #define **RCC_AHB1LPENR_CRCLPEN** ((uint32_t)0x00001000)
- #define **RCC_AHB1LPENR_FLITFLPEN** ((uint32_t)0x00008000)
- #define **RCC_AHB1LPENR_SRAM1LPEN** ((uint32_t)0x00010000)
- #define **RCC_AHB1LPENR_SRAM2LPEN** ((uint32_t)0x00020000)
- #define **RCC_AHB1LPENR_BKPSRAMLPEN** ((uint32_t)0x00040000)
- #define **RCC_AHB1LPENR_DMA1LPEN** ((uint32_t)0x00200000)
- #define **RCC_AHB1LPENR_DMA2LPEN** ((uint32_t)0x00400000)
- #define **RCC_AHB1LPENR_ETHMACLPEN** ((uint32_t)0x02000000)
- #define **RCC_AHB1LPENR_ETHMACTXLPEN** ((uint32_t)0x04000000)
- #define **RCC_AHB1LPENR_ETHMACRXLPEN** ((uint32_t)0x08000000)
- #define **RCC_AHB1LPENR_ETHMACPTPLPEN** ((uint32_t)0x10000000)
- #define **RCC_AHB1LPENR_OTGHSLPEN** ((uint32_t)0x20000000)
- #define **RCC_AHB1LPENR_OTGHSULPILPEN** ((uint32_t)0x40000000)
- #define **RCC_AHB2LPENR_DCMILPEN** ((uint32_t)0x00000001)
- #define **RCC_AHB2LPENR_CRYPLPEN** ((uint32_t)0x00000010)
- #define **RCC_AHB2LPENR_HASHLPEN** ((uint32_t)0x00000020)
- #define **RCC_AHB2LPENR_RNGLPEN** ((uint32_t)0x00000040)
- #define **RCC_AHB2LPENR_OTGFSLPEN** ((uint32_t)0x00000080)
- #define **RCC_AHB3LPENR_FSMCLPEN** ((uint32_t)0x00000001)
- #define **RCC_APB1LPENR_TIM2LPEN** ((uint32_t)0x00000001)
- #define **RCC_APB1LPENR_TIM3LPEN** ((uint32_t)0x00000002)
- #define **RCC_APB1LPENR_TIM4LPEN** ((uint32_t)0x00000004)
- #define **RCC_APB1LPENR_TIM5LPEN** ((uint32_t)0x00000008)
- #define **RCC_APB1LPENR_TIM6LPEN** ((uint32_t)0x00000010)
- #define **RCC_APB1LPENR_TIM7LPEN** ((uint32_t)0x00000020)
- #define **RCC_APB1LPENR_TIM12LPEN** ((uint32_t)0x00000040)
- #define **RCC_APB1LPENR_TIM13LPEN** ((uint32_t)0x00000080)
- #define **RCC_APB1LPENR_TIM14LPEN** ((uint32_t)0x00000100)

- #define **RCC_APB1LPENR_WWDGLPEN** ((uint32_t)0x00000800)
- #define **RCC_APB1LPENR_SPI2LPEN** ((uint32_t)0x00004000)
- #define **RCC_APB1LPENR_SPI3LPEN** ((uint32_t)0x00008000)
- #define **RCC_APB1LPENR_USART2LPEN** ((uint32_t)0x00020000)
- #define **RCC_APB1LPENR_USART3LPEN** ((uint32_t)0x00040000)
- #define **RCC_APB1LPENR_UART4LPEN** ((uint32_t)0x00080000)
- #define **RCC_APB1LPENR_UART5LPEN** ((uint32_t)0x00100000)
- #define **RCC_APB1LPENR_I2C1LPEN** ((uint32_t)0x00200000)
- #define **RCC_APB1LPENR_I2C2LPEN** ((uint32_t)0x00400000)
- #define **RCC_APB1LPENR_I2C3LPEN** ((uint32_t)0x00800000)
- #define **RCC_APB1LPENR_CAN1LPEN** ((uint32_t)0x02000000)
- #define **RCC_APB1LPENR_CAN2LPEN** ((uint32_t)0x04000000)
- #define **RCC_APB1LPENR_PWRLPEN** ((uint32_t)0x10000000)
- #define **RCC_APB1LPENR_DACLPEN** ((uint32_t)0x20000000)
- #define **RCC_APB2LPENR_TIM1LPEN** ((uint32_t)0x00000001)
- #define **RCC_APB2LPENR_TIM8LPEN** ((uint32_t)0x00000002)
- #define **RCC_APB2LPENR_USART1LPEN** ((uint32_t)0x00000010)
- #define **RCC_APB2LPENR_USART6LPEN** ((uint32_t)0x00000020)
- #define **RCC_APB2LPENR_ADC1LPEN** ((uint32_t)0x00000100)
- #define **RCC_APB2LPENR_ADC2PEN** ((uint32_t)0x00000200)
- #define **RCC_APB2LPENR_ADC3LPEN** ((uint32_t)0x00000400)
- #define **RCC_APB2LPENR_SDIOLPEN** ((uint32_t)0x00000800)
- #define **RCC_APB2LPENR_SPI1LPEN** ((uint32_t)0x00001000)
- #define **RCC_APB2LPENR_SYSCFGLPEN** ((uint32_t)0x00004000)
- #define **RCC_APB2LPENR_TIM9LPEN** ((uint32_t)0x00010000)
- #define **RCC_APB2LPENR_TIM10LPEN** ((uint32_t)0x00020000)
- #define **RCC_APB2LPENR_TIM11LPEN** ((uint32_t)0x00040000)
- #define **RCC_BDCR_LSEON** ((uint32_t)0x00000001)
- #define **RCC_BDCR_LSERDY** ((uint32_t)0x00000002)
- #define **RCC_BDCR_LSEBYP** ((uint32_t)0x00000004)
- #define **RCC_BDCR_RTCSEL** ((uint32_t)0x00000300)
- #define **RCC_BDCR_RTCSEL_0** ((uint32_t)0x00000100)
- #define **RCC_BDCR_RTCSEL_1** ((uint32_t)0x00000200)
- #define **RCC_BDCR_RTCEN** ((uint32_t)0x00008000)
- #define **RCC_BDCR_BDRST** ((uint32_t)0x00010000)
- #define **RCC_CSR_LSION** ((uint32_t)0x00000001)
- #define **RCC_CSR_LSIRDY** ((uint32_t)0x00000002)
- #define **RCC_CSR_RMVF** ((uint32_t)0x01000000)
- #define **RCC_CSR_BORRSTF** ((uint32_t)0x02000000)
- #define **RCC_CSR_PADRSTF** ((uint32_t)0x04000000)
- #define **RCC_CSR_PORRSTF** ((uint32_t)0x08000000)
- #define **RCC_CSR_SFTRSTF** ((uint32_t)0x10000000)
- #define **RCC_CSR_WDGRSTF** ((uint32_t)0x20000000)
- #define **RCC_CSR_WWDGRSTF** ((uint32_t)0x40000000)
- #define **RCC_CSR_LPWRRSTF** ((uint32_t)0x80000000)
- #define **RCC_SSCGR_MODPER** ((uint32_t)0x00001FFF)
- #define **RCC_SSCGR_INCSTEP** ((uint32_t)0x0FFFE000)
- #define **RCC_SSCGR_SPREADSEL** ((uint32_t)0x40000000)
- #define **RCC_SSCGR_SSCGEN** ((uint32_t)0x80000000)
- #define **RCC_PLLI2SCFGR_PLLI2SN** ((uint32_t)0x00007FC0)
- #define **RCC_PLLI2SCFGR_PLLI2SR** ((uint32_t)0x70000000)
- #define **RNG_CR_RNGEN** ((uint32_t)0x00000004)
- #define **RNG_CR_IE** ((uint32_t)0x00000008)
- #define **RNG_SR_DRDY** ((uint32_t)0x00000001)
- #define **RNG_SR_CECS** ((uint32_t)0x00000002)

- #define **RNG_SR_SECS** ((uint32_t)0x00000004)
- #define **RNG_SR_CEIS** ((uint32_t)0x00000020)
- #define **RNG_SR_SEIS** ((uint32_t)0x00000040)
- #define **RTC_TR_PM** ((uint32_t)0x00400000)
- #define **RTC_TR_HT** ((uint32_t)0x00300000)
- #define **RTC_TR_HT_0** ((uint32_t)0x00100000)
- #define **RTC_TR_HT_1** ((uint32_t)0x00200000)
- #define **RTC_TR_HU** ((uint32_t)0x000F0000)
- #define **RTC_TR_HU_0** ((uint32_t)0x00010000)
- #define **RTC_TR_HU_1** ((uint32_t)0x00020000)
- #define **RTC_TR_HU_2** ((uint32_t)0x00040000)
- #define **RTC_TR_HU_3** ((uint32_t)0x00080000)
- #define **RTC_TR_MNT** ((uint32_t)0x00007000)
- #define **RTC_TR_MNT_0** ((uint32_t)0x00001000)
- #define **RTC_TR_MNT_1** ((uint32_t)0x00002000)
- #define **RTC_TR_MNT_2** ((uint32_t)0x00004000)
- #define **RTC_TR_MNU** ((uint32_t)0x00000F00)
- #define **RTC_TR_MNU_0** ((uint32_t)0x00000100)
- #define **RTC_TR_MNU_1** ((uint32_t)0x00000200)
- #define **RTC_TR_MNU_2** ((uint32_t)0x00000400)
- #define **RTC_TR_MNU_3** ((uint32_t)0x00000800)
- #define **RTC_TR_ST** ((uint32_t)0x00000070)
- #define **RTC_TR_ST_0** ((uint32_t)0x00000010)
- #define **RTC_TR_ST_1** ((uint32_t)0x00000020)
- #define **RTC_TR_ST_2** ((uint32_t)0x00000040)
- #define **RTC_TR_SU** ((uint32_t)0x0000000F)
- #define **RTC_TR_SU_0** ((uint32_t)0x00000001)
- #define **RTC_TR_SU_1** ((uint32_t)0x00000002)
- #define **RTC_TR_SU_2** ((uint32_t)0x00000004)
- #define **RTC_TR_SU_3** ((uint32_t)0x00000008)
- #define **RTC_DR_YT** ((uint32_t)0x00F00000)
- #define **RTC_DR_YT_0** ((uint32_t)0x00100000)
- #define **RTC_DR_YT_1** ((uint32_t)0x00200000)
- #define **RTC_DR_YT_2** ((uint32_t)0x00400000)
- #define **RTC_DR_YT_3** ((uint32_t)0x00800000)
- #define **RTC_DR_YU** ((uint32_t)0x000F0000)
- #define **RTC_DR_YU_0** ((uint32_t)0x00010000)
- #define **RTC_DR_YU_1** ((uint32_t)0x00020000)
- #define **RTC_DR_YU_2** ((uint32_t)0x00040000)
- #define **RTC_DR_YU_3** ((uint32_t)0x00080000)
- #define **RTC_DR_WDU** ((uint32_t)0x0000E000)
- #define **RTC_DR_WDU_0** ((uint32_t)0x00002000)
- #define **RTC_DR_WDU_1** ((uint32_t)0x00004000)
- #define **RTC_DR_WDU_2** ((uint32_t)0x00008000)
- #define **RTC_DR_MT** ((uint32_t)0x00001000)
- #define **RTC_DR_MU** ((uint32_t)0x00000F00)
- #define **RTC_DR_MU_0** ((uint32_t)0x00000100)
- #define **RTC_DR_MU_1** ((uint32_t)0x00000200)
- #define **RTC_DR_MU_2** ((uint32_t)0x00000400)
- #define **RTC_DR_MU_3** ((uint32_t)0x00000800)
- #define **RTC_DR_DT** ((uint32_t)0x00000030)
- #define **RTC_DR_DT_0** ((uint32_t)0x00000010)
- #define **RTC_DR_DT_1** ((uint32_t)0x00000020)
- #define **RTC_DR_DU** ((uint32_t)0x0000000F)
- #define **RTC_DR_DU_0** ((uint32_t)0x00000001)

- #define **RTC_DR_DU_1** ((uint32_t)0x00000002)
- #define **RTC_DR_DU_2** ((uint32_t)0x00000004)
- #define **RTC_DR_DU_3** ((uint32_t)0x00000008)
- #define **RTC_CR_COE** ((uint32_t)0x00800000)
- #define **RTC_CR_OSEL** ((uint32_t)0x00600000)
- #define **RTC_CR_OSEL_0** ((uint32_t)0x00200000)
- #define **RTC_CR_OSEL_1** ((uint32_t)0x00400000)
- #define **RTC_CR_POL** ((uint32_t)0x00100000)
- #define **RTC_CR_COSEL** ((uint32_t)0x00080000)
- #define **RTC_CR_BCK** ((uint32_t)0x00040000)
- #define **RTC_CR_SUB1H** ((uint32_t)0x00020000)
- #define **RTC_CR_ADD1H** ((uint32_t)0x00010000)
- #define **RTC_CR_TSIE** ((uint32_t)0x00008000)
- #define **RTC_CR_WUTIE** ((uint32_t)0x00004000)
- #define **RTC_CR_ALRBIE** ((uint32_t)0x00002000)
- #define **RTC_CR_ALRAIE** ((uint32_t)0x00001000)
- #define **RTC_CR_TSE** ((uint32_t)0x00000800)
- #define **RTC_CR_WUTE** ((uint32_t)0x00000400)
- #define **RTC_CR_ALRBE** ((uint32_t)0x00000200)
- #define **RTC_CR_ALRAE** ((uint32_t)0x00000100)
- #define **RTC_CR_DCE** ((uint32_t)0x00000080)
- #define **RTC_CR_FMT** ((uint32_t)0x00000040)
- #define **RTC_CR_BYPSHAD** ((uint32_t)0x00000020)
- #define **RTC_CR_REFCKON** ((uint32_t)0x00000010)
- #define **RTC_CR_TSEDGE** ((uint32_t)0x00000008)
- #define **RTC_CR_WUCKSEL** ((uint32_t)0x00000007)
- #define **RTC_CR_WUCKSEL_0** ((uint32_t)0x00000001)
- #define **RTC_CR_WUCKSEL_1** ((uint32_t)0x00000002)
- #define **RTC_CR_WUCKSEL_2** ((uint32_t)0x00000004)
- #define **RTC_ISR_RECALPF** ((uint32_t)0x00010000)
- #define **RTC_ISR_TAMP1F** ((uint32_t)0x00002000)
- #define **RTC_ISR_TSOVF** ((uint32_t)0x00001000)
- #define **RTC_ISR_TSF** ((uint32_t)0x00000800)
- #define **RTC_ISR_WUTF** ((uint32_t)0x00000400)
- #define **RTC_ISR_ALRBF** ((uint32_t)0x00000200)
- #define **RTC_ISR_ALRAF** ((uint32_t)0x00000100)
- #define **RTC_ISR_INIT** ((uint32_t)0x00000080)
- #define **RTC_ISR_INITF** ((uint32_t)0x00000040)
- #define **RTC_ISR_RSF** ((uint32_t)0x00000020)
- #define **RTC_ISR_INITS** ((uint32_t)0x00000010)
- #define **RTC_ISR_SHPF** ((uint32_t)0x00000008)
- #define **RTC_ISR_WUTWF** ((uint32_t)0x00000004)
- #define **RTC_ISR_ALRBWF** ((uint32_t)0x00000002)
- #define **RTC_ISR_ALRAWF** ((uint32_t)0x00000001)
- #define **RTC_PRER_PREDIV_A** ((uint32_t)0x007F0000)
- #define **RTC_PRER_PREDIV_S** ((uint32_t)0x00001FFF)
- #define **RTC_WUTR_WUT** ((uint32_t)0x0000FFFF)
- #define **RTC_CALIBR_DCS** ((uint32_t)0x00000080)
- #define **RTC_CALIBR_DC** ((uint32_t)0x0000001F)
- #define **RTC_ALRMAR_MSK4** ((uint32_t)0x80000000)
- #define **RTC_ALRMAR_WDSEL** ((uint32_t)0x40000000)
- #define **RTC_ALRMAR_DT** ((uint32_t)0x30000000)
- #define **RTC_ALRMAR_DT_0** ((uint32_t)0x10000000)
- #define **RTC_ALRMAR_DT_1** ((uint32_t)0x20000000)
- #define **RTC_ALRMAR_DU** ((uint32_t)0x0F000000)

- #define **RTC_ALRMAR_DU_0** ((uint32_t)0x01000000)
- #define **RTC_ALRMAR_DU_1** ((uint32_t)0x02000000)
- #define **RTC_ALRMAR_DU_2** ((uint32_t)0x04000000)
- #define **RTC_ALRMAR_DU_3** ((uint32_t)0x08000000)
- #define **RTC_ALRMAR_MSK3** ((uint32_t)0x00800000)
- #define **RTC_ALRMAR_PM** ((uint32_t)0x00400000)
- #define **RTC_ALRMAR_HT** ((uint32_t)0x00300000)
- #define **RTC_ALRMAR_HT_0** ((uint32_t)0x00100000)
- #define **RTC_ALRMAR_HT_1** ((uint32_t)0x00200000)
- #define **RTC_ALRMAR_HU** ((uint32_t)0x000F0000)
- #define **RTC_ALRMAR_HU_0** ((uint32_t)0x00010000)
- #define **RTC_ALRMAR_HU_1** ((uint32_t)0x00020000)
- #define **RTC_ALRMAR_HU_2** ((uint32_t)0x00040000)
- #define **RTC_ALRMAR_HU_3** ((uint32_t)0x00080000)
- #define **RTC_ALRMAR_MSK2** ((uint32_t)0x00008000)
- #define **RTC_ALRMAR_MNT** ((uint32_t)0x00007000)
- #define **RTC_ALRMAR_MNT_0** ((uint32_t)0x00001000)
- #define **RTC_ALRMAR_MNT_1** ((uint32_t)0x00002000)
- #define **RTC_ALRMAR_MNT_2** ((uint32_t)0x00004000)
- #define **RTC_ALRMAR_MNU** ((uint32_t)0x00000F00)
- #define **RTC_ALRMAR_MNU_0** ((uint32_t)0x00000100)
- #define **RTC_ALRMAR_MNU_1** ((uint32_t)0x00000200)
- #define **RTC_ALRMAR_MNU_2** ((uint32_t)0x00000400)
- #define **RTC_ALRMAR_MNU_3** ((uint32_t)0x00000800)
- #define **RTC_ALRMAR_MSK1** ((uint32_t)0x00000080)
- #define **RTC_ALRMAR_ST** ((uint32_t)0x00000070)
- #define **RTC_ALRMAR_ST_0** ((uint32_t)0x00000010)
- #define **RTC_ALRMAR_ST_1** ((uint32_t)0x00000020)
- #define **RTC_ALRMAR_ST_2** ((uint32_t)0x00000040)
- #define **RTC_ALRMAR_SU** ((uint32_t)0x0000000F)
- #define **RTC_ALRMAR_SU_0** ((uint32_t)0x00000001)
- #define **RTC_ALRMAR_SU_1** ((uint32_t)0x00000002)
- #define **RTC_ALRMAR_SU_2** ((uint32_t)0x00000004)
- #define **RTC_ALRMAR_SU_3** ((uint32_t)0x00000008)
- #define **RTC_ALRMBR_MSK4** ((uint32_t)0x80000000)
- #define **RTC_ALRMBR_WDSEL** ((uint32_t)0x40000000)
- #define **RTC_ALRMBR_DT** ((uint32_t)0x30000000)
- #define **RTC_ALRMBR_DT_0** ((uint32_t)0x10000000)
- #define **RTC_ALRMBR_DT_1** ((uint32_t)0x20000000)
- #define **RTC_ALRMBR_DU** ((uint32_t)0x0F000000)
- #define **RTC_ALRMBR_DU_0** ((uint32_t)0x01000000)
- #define **RTC_ALRMBR_DU_1** ((uint32_t)0x02000000)
- #define **RTC_ALRMBR_DU_2** ((uint32_t)0x04000000)
- #define **RTC_ALRMBR_DU_3** ((uint32_t)0x08000000)
- #define **RTC_ALRMBR_MSK3** ((uint32_t)0x00800000)
- #define **RTC_ALRMBR_PM** ((uint32_t)0x00400000)
- #define **RTC_ALRMBR_HT** ((uint32_t)0x00300000)
- #define **RTC_ALRMBR_HT_0** ((uint32_t)0x00100000)
- #define **RTC_ALRMBR_HT_1** ((uint32_t)0x00200000)
- #define **RTC_ALRMBR_HU** ((uint32_t)0x000F0000)
- #define **RTC_ALRMBR_HU_0** ((uint32_t)0x00010000)
- #define **RTC_ALRMBR_HU_1** ((uint32_t)0x00020000)
- #define **RTC_ALRMBR_HU_2** ((uint32_t)0x00040000)
- #define **RTC_ALRMBR_HU_3** ((uint32_t)0x00080000)
- #define **RTC_ALRMBR_MSK2** ((uint32_t)0x00008000)

- #define **RTC_ALRMBR_MNT** ((uint32_t)0x00007000)
- #define **RTC_ALRMBR_MNT_0** ((uint32_t)0x00001000)
- #define **RTC_ALRMBR_MNT_1** ((uint32_t)0x00002000)
- #define **RTC_ALRMBR_MNT_2** ((uint32_t)0x00004000)
- #define **RTC_ALRMBR_MNU** ((uint32_t)0x00000F00)
- #define **RTC_ALRMBR_MNU_0** ((uint32_t)0x00000100)
- #define **RTC_ALRMBR_MNU_1** ((uint32_t)0x00000200)
- #define **RTC_ALRMBR_MNU_2** ((uint32_t)0x00000400)
- #define **RTC_ALRMBR_MNU_3** ((uint32_t)0x00000800)
- #define **RTC_ALRMBR_MSK1** ((uint32_t)0x00000080)
- #define **RTC_ALRMBR_ST** ((uint32_t)0x00000070)
- #define **RTC_ALRMBR_ST_0** ((uint32_t)0x00000010)
- #define **RTC_ALRMBR_ST_1** ((uint32_t)0x00000020)
- #define **RTC_ALRMBR_ST_2** ((uint32_t)0x00000040)
- #define **RTC_ALRMBR_SU** ((uint32_t)0x0000000F)
- #define **RTC_ALRMBR_SU_0** ((uint32_t)0x00000001)
- #define **RTC_ALRMBR_SU_1** ((uint32_t)0x00000002)
- #define **RTC_ALRMBR_SU_2** ((uint32_t)0x00000004)
- #define **RTC_ALRMBR_SU_3** ((uint32_t)0x00000008)
- #define **RTC_WPR_KEY** ((uint32_t)0x000000FF)
- #define **RTC_SSR_SS** ((uint32_t)0x0000FFFF)
- #define **RTC_SHIFTR_SUBFS** ((uint32_t)0x00007FFF)
- #define **RTC_SHIFTR_ADD1S** ((uint32_t)0x80000000)
- #define **RTC_TSTR_PM** ((uint32_t)0x00400000)
- #define **RTC_TSTR_HT** ((uint32_t)0x00300000)
- #define **RTC_TSTR_HT_0** ((uint32_t)0x00100000)
- #define **RTC_TSTR_HT_1** ((uint32_t)0x00200000)
- #define **RTC_TSTR_HU** ((uint32_t)0x000F0000)
- #define **RTC_TSTR_HU_0** ((uint32_t)0x00010000)
- #define **RTC_TSTR_HU_1** ((uint32_t)0x00020000)
- #define **RTC_TSTR_HU_2** ((uint32_t)0x00040000)
- #define **RTC_TSTR_HU_3** ((uint32_t)0x00080000)
- #define **RTC_TSTR_MNT** ((uint32_t)0x00007000)
- #define **RTC_TSTR_MNT_0** ((uint32_t)0x00001000)
- #define **RTC_TSTR_MNT_1** ((uint32_t)0x00002000)
- #define **RTC_TSTR_MNT_2** ((uint32_t)0x00004000)
- #define **RTC_TSTR_MNU** ((uint32_t)0x00000F00)
- #define **RTC_TSTR_MNU_0** ((uint32_t)0x00000100)
- #define **RTC_TSTR_MNU_1** ((uint32_t)0x00000200)
- #define **RTC_TSTR_MNU_2** ((uint32_t)0x00000400)
- #define **RTC_TSTR_MNU_3** ((uint32_t)0x00000800)
- #define **RTC_TSTR_ST** ((uint32_t)0x00000070)
- #define **RTC_TSTR_ST_0** ((uint32_t)0x00000010)
- #define **RTC_TSTR_ST_1** ((uint32_t)0x00000020)
- #define **RTC_TSTR_ST_2** ((uint32_t)0x00000040)
- #define **RTC_TSTR_SU** ((uint32_t)0x0000000F)
- #define **RTC_TSTR_SU_0** ((uint32_t)0x00000001)
- #define **RTC_TSTR_SU_1** ((uint32_t)0x00000002)
- #define **RTC_TSTR_SU_2** ((uint32_t)0x00000004)
- #define **RTC_TSTR_SU_3** ((uint32_t)0x00000008)
- #define **RTC_TSDR_WDU** ((uint32_t)0x0000E000)
- #define **RTC_TSDR_WDU_0** ((uint32_t)0x00002000)
- #define **RTC_TSDR_WDU_1** ((uint32_t)0x00004000)
- #define **RTC_TSDR_WDU_2** ((uint32_t)0x00008000)
- #define **RTC_TSDR_MT** ((uint32_t)0x00001000)

- #define **RTC_TSDR_MU** ((uint32_t)0x00000F00)
- #define **RTC_TSDR_MU_0** ((uint32_t)0x00000100)
- #define **RTC_TSDR_MU_1** ((uint32_t)0x00000200)
- #define **RTC_TSDR_MU_2** ((uint32_t)0x00000400)
- #define **RTC_TSDR_MU_3** ((uint32_t)0x00000800)
- #define **RTC_TSDR_DT** ((uint32_t)0x00000030)
- #define **RTC_TSDR_DT_0** ((uint32_t)0x00000010)
- #define **RTC_TSDR_DT_1** ((uint32_t)0x00000020)
- #define **RTC_TSDR_DU** ((uint32_t)0x0000000F)
- #define **RTC_TSDR_DU_0** ((uint32_t)0x00000001)
- #define **RTC_TSDR_DU_1** ((uint32_t)0x00000002)
- #define **RTC_TSDR_DU_2** ((uint32_t)0x00000004)
- #define **RTC_TSDR_DU_3** ((uint32_t)0x00000008)
- #define **RTC_TSSSR_SS** ((uint32_t)0x0000FFFF)
- #define **RTC_CALR_CALP** ((uint32_t)0x00008000)
- #define **RTC_CALR_CALW8** ((uint32_t)0x00004000)
- #define **RTC_CALR_CALW16** ((uint32_t)0x00002000)
- #define **RTC_CALR_CALM** ((uint32_t)0x000001FF)
- #define **RTC_CALR_CALM_0** ((uint32_t)0x00000001)
- #define **RTC_CALR_CALM_1** ((uint32_t)0x00000002)
- #define **RTC_CALR_CALM_2** ((uint32_t)0x00000004)
- #define **RTC_CALR_CALM_3** ((uint32_t)0x00000008)
- #define **RTC_CALR_CALM_4** ((uint32_t)0x00000010)
- #define **RTC_CALR_CALM_5** ((uint32_t)0x00000020)
- #define **RTC_CALR_CALM_6** ((uint32_t)0x00000040)
- #define **RTC_CALR_CALM_7** ((uint32_t)0x00000080)
- #define **RTC_CALR_CALM_8** ((uint32_t)0x00000100)
- #define **RTC_TAFCR_ALARMOUTTYPE** ((uint32_t)0x00040000)
- #define **RTC_TAFCR_TSINSEL** ((uint32_t)0x00020000)
- #define **RTC_TAFCR_TAMPINSEL** ((uint32_t)0x00010000)
- #define **RTC_TAFCR_TAMPPUDIS** ((uint32_t)0x00008000)
- #define **RTC_TAFCR_TAMPPRCH** ((uint32_t)0x00006000)
- #define **RTC_TAFCR_TAMPPRCH_0** ((uint32_t)0x00002000)
- #define **RTC_TAFCR_TAMPPRCH_1** ((uint32_t)0x00004000)
- #define **RTC_TAFCR_TAMPFLT** ((uint32_t)0x00001800)
- #define **RTC_TAFCR_TAMPFLT_0** ((uint32_t)0x00000800)
- #define **RTC_TAFCR_TAMPFLT_1** ((uint32_t)0x00001000)
- #define **RTC_TAFCR_TAMPFREQ** ((uint32_t)0x00000700)
- #define **RTC_TAFCR_TAMPFREQ_0** ((uint32_t)0x00000100)
- #define **RTC_TAFCR_TAMPFREQ_1** ((uint32_t)0x00000200)
- #define **RTC_TAFCR_TAMPFREQ_2** ((uint32_t)0x00000400)
- #define **RTC_TAFCR_TAMPTS** ((uint32_t)0x00000080)
- #define **RTC_TAFCR_TAMPIE** ((uint32_t)0x00000004)
- #define **RTC_TAFCR_TAMP1TRG** ((uint32_t)0x00000002)
- #define **RTC_TAFCR_TAMP1E** ((uint32_t)0x00000001)
- #define **RTC_ALRMASSR_MASKSS** ((uint32_t)0x0F000000)
- #define **RTC_ALRMASSR_MASKSS_0** ((uint32_t)0x01000000)
- #define **RTC_ALRMASSR_MASKSS_1** ((uint32_t)0x02000000)
- #define **RTC_ALRMASSR_MASKSS_2** ((uint32_t)0x04000000)
- #define **RTC_ALRMASSR_MASKSS_3** ((uint32_t)0x08000000)
- #define **RTC_ALRMASSR_SS** ((uint32_t)0x00007FFF)
- #define **RTC_ALRMBSSR_MASKSS** ((uint32_t)0x0F000000)
- #define **RTC_ALRMBSSR_MASKSS_0** ((uint32_t)0x01000000)
- #define **RTC_ALRMBSSR_MASKSS_1** ((uint32_t)0x02000000)
- #define **RTC_ALRMBSSR_MASKSS_2** ((uint32_t)0x04000000)

- #define **RTC_ALRMBSSR_MASKSS_3** ((uint32_t)0x08000000)
- #define **RTC_ALRMBSSR_SS** ((uint32_t)0x00007FFF)
- #define **RTC_BKP0R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP1R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP2R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP3R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP4R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP5R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP6R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP7R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP8R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP9R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP10R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP11R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP12R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP13R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP14R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP15R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP16R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP17R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP18R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP19R** ((uint32_t)0xFFFFFFFF)
- #define SDIO_POWER_PWRCTRL ((uint8_t)0x03)
- #define SDIO_POWER_PWRCTRL_0 ((uint8_t)0x01)
- #define SDIO_POWER_PWRCTRL_1 ((uint8_t)0x02)
- #define SDIO_CLKCR_CLKDIV ((uint16_t)0x00FF)
- #define SDIO_CLKCR_CLKEN ((uint16_t)0x0100)
- #define SDIO_CLKCR_PWRSAV ((uint16_t)0x0200)
- #define SDIO_CLKCR_BYPASS ((uint16_t)0x0400)
- #define SDIO_CLKCR_WIDBUS ((uint16_t)0x1800)
- #define SDIO_CLKCR_WIDBUS_0 ((uint16_t)0x0800)
- #define SDIO_CLKCR_WIDBUS_1 ((uint16_t)0x1000)
- #define SDIO_CLKCR_NEGEDGE ((uint16_t)0x2000)
- #define SDIO_CLKCR_HWFC_EN ((uint16_t)0x4000)
- #define SDIO_ARG_CMDARG ((uint32_t)0xFFFFFFFF)
- #define SDIO_CMD_CMDINDEX ((uint16_t)0x003F)
- #define SDIO_CMD_WAITRESP ((uint16_t)0x00C0)
- #define SDIO_CMD_WAITRESP_0 ((uint16_t)0x0040)
- #define SDIO_CMD_WAITRESP_1 ((uint16_t)0x0080)
- #define SDIO_CMD_WAITINT ((uint16_t)0x0100)
- #define SDIO_CMD_WAITPEND ((uint16_t)0x0200)
- #define SDIO_CMD_CPSMEN ((uint16_t)0x0400)
- #define SDIO_CMD_SDIOSUSPEND ((uint16_t)0x0800)
- #define SDIO_CMD_ENCMDCOMPL ((uint16_t)0x1000)
- #define SDIO_CMD_NIEN ((uint16_t)0x2000)
- #define SDIO_CMD_CEATACMD ((uint16_t)0x4000)
- #define SDIO_RESPCMD_RESPCMD ((uint8_t)0x3F)
- #define SDIO_RESP0_CARDSTATUS0 ((uint32_t)0xFFFFFFFF)
- #define SDIO_RESP1_CARDSTATUS1 ((uint32_t)0xFFFFFFFF)
- #define SDIO_RESP2_CARDSTATUS2 ((uint32_t)0xFFFFFFFF)
- #define SDIO_RESP3_CARDSTATUS3 ((uint32_t)0xFFFFFFFF)
- #define SDIO_RESP4_CARDSTATUS4 ((uint32_t)0xFFFFFFFF)
- #define SDIO_DTIMER_DATATIME ((uint32_t)0xFFFFFFFF)
- #define SDIO_DLEN_DATALENGTH ((uint32_t)0x01FFFFFF)
- #define SDIO_DCTRL_DTEN ((uint16_t)0x0001)

- #define SDIO_DCTRL_DTDIR ((uint16_t)0x0002)
- #define SDIO_DCTRL_DTMODE ((uint16_t)0x0004)
- #define SDIO_DCTRL_DMAEN ((uint16_t)0x0008)
- #define SDIO_DCTRL_DBLOCKSIZE ((uint16_t)0x00F0)
- #define SDIO_DCTRL_DBLOCKSIZE_0 ((uint16_t)0x0010)
- #define SDIO_DCTRL_DBLOCKSIZE_1 ((uint16_t)0x0020)
- #define SDIO_DCTRL_DBLOCKSIZE_2 ((uint16_t)0x0040)
- #define SDIO_DCTRL_DBLOCKSIZE_3 ((uint16_t)0x0080)
- #define SDIO_DCTRL_RWSTART ((uint16_t)0x0100)
- #define SDIO_DCTRL_RWSTOP ((uint16_t)0x0200)
- #define SDIO_DCTRL_RWMOD ((uint16_t)0x0400)
- #define SDIO_DCTRL_SDIOEN ((uint16_t)0x0800)
- #define SDIO_DCOUNT_DATACOUNT ((uint32_t)0x01FFFFFF)
- #define SDIO_STA_CCRCFAIL ((uint32_t)0x00000001)
- #define SDIO_STA_DCRCFAIL ((uint32_t)0x00000002)
- #define SDIO_STA_CTIMEOUT ((uint32_t)0x00000004)
- #define SDIO_STA_DTIMEOUT ((uint32_t)0x00000008)
- #define SDIO_STA_TXUNDERR ((uint32_t)0x00000010)
- #define SDIO_STA_RXOVERR ((uint32_t)0x00000020)
- #define SDIO_STA_CMDREND ((uint32_t)0x00000040)
- #define SDIO_STA_CMDSENT ((uint32_t)0x00000080)
- #define SDIO_STA_DATAEND ((uint32_t)0x00000100)
- #define SDIO_STA_STBITERR ((uint32_t)0x00000200)
- #define SDIO_STA_DBCKEND ((uint32_t)0x00000400)
- #define SDIO_STA_CMDACT ((uint32_t)0x00000800)
- #define SDIO_STA_TXACT ((uint32_t)0x00001000)
- #define SDIO_STA_RXACT ((uint32_t)0x00002000)
- #define SDIO_STA_TXFIFOHE ((uint32_t)0x00004000)
- #define SDIO_STA_RXFIFOHF ((uint32_t)0x00008000)
- #define SDIO_STA_TXFIFOF ((uint32_t)0x00010000)
- #define SDIO_STA_RXFIFOF ((uint32_t)0x00020000)
- #define SDIO_STA_TXFIFOE ((uint32_t)0x00040000)
- #define SDIO_STA_RXFIFOE ((uint32_t)0x00080000)
- #define SDIO_STA_TXDAVL ((uint32_t)0x00100000)
- #define SDIO_STA_RXDAVL ((uint32_t)0x00200000)
- #define SDIO_STA_SDIOIT ((uint32_t)0x00400000)
- #define SDIO_STA_CEATAEND ((uint32_t)0x00800000)
- #define SDIO_ICR_CCRCFAILC ((uint32_t)0x00000001)
- #define SDIO_ICR_DCRCFAILC ((uint32_t)0x00000002)
- #define SDIO_ICR_CTIMEOUTC ((uint32_t)0x00000004)
- #define SDIO_ICR_DTIMEOUTC ((uint32_t)0x00000008)
- #define SDIO_ICR_TXUNDERRC ((uint32_t)0x00000010)
- #define SDIO_ICR_RXOVERRC ((uint32_t)0x00000020)
- #define SDIO_ICR_CMDRENDC ((uint32_t)0x00000040)
- #define SDIO_ICR_CMDSENTC ((uint32_t)0x00000080)
- #define SDIO_ICR_DATAENDC ((uint32_t)0x00000100)
- #define SDIO_ICR_STBITERRC ((uint32_t)0x00000200)
- #define SDIO_ICR_DBCKENDC ((uint32_t)0x00000400)
- #define SDIO_ICR_SDIOITC ((uint32_t)0x00400000)
- #define SDIO_ICR_CEATAENDC ((uint32_t)0x00800000)
- #define SDIO_MASK_CCRCFAILIE ((uint32_t)0x00000001)
- #define SDIO_MASK_DCRCFAILIE ((uint32_t)0x00000002)
- #define SDIO_MASK_CTIMEOUTIE ((uint32_t)0x00000004)
- #define SDIO_MASK_DTIMEOUTIE ((uint32_t)0x00000008)
- #define SDIO_MASK_TXUNDERRIE ((uint32_t)0x00000010)

- #define SDIO_MASK_RXOVERRIE ((uint32_t)0x00000020)
- #define SDIO_MASK_CMDRENDIE ((uint32_t)0x00000040)
- #define SDIO_MASK_CMDSENTIE ((uint32_t)0x00000080)
- #define SDIO_MASK_DATAENDIE ((uint32_t)0x00000100)
- #define SDIO_MASK_STBITERRIE ((uint32_t)0x00000200)
- #define SDIO_MASK_DBCKENDIE ((uint32_t)0x00000400)
- #define SDIO_MASK_CMDACTIE ((uint32_t)0x00000800)
- #define SDIO_MASK_TXACTIE ((uint32_t)0x00001000)
- #define SDIO_MASK_RXACTIE ((uint32_t)0x00002000)
- #define SDIO_MASK_TXFIFOHEIE ((uint32_t)0x00004000)
- #define SDIO_MASK_RXFIFOHFIE ((uint32_t)0x00008000)
- #define SDIO_MASK_TXFIFOFIE ((uint32_t)0x00010000)
- #define SDIO_MASK_RXFIFOFIE ((uint32_t)0x00020000)
- #define SDIO_MASK_TXFIFOEIE ((uint32_t)0x00040000)
- #define SDIO_MASK_RXFIFOEIE ((uint32_t)0x00080000)
- #define SDIO_MASK_TXDAVLIE ((uint32_t)0x00100000)
- #define SDIO_MASK_RXDAVLIE ((uint32_t)0x00200000)
- #define SDIO_MASK_SDIOITIE ((uint32_t)0x00400000)
- #define SDIO_MASK_CEATAENDIE ((uint32_t)0x00800000)
- #define SDIO_FIFOCNT_FIFOCOUNT ((uint32_t)0x00FFFFFF)
- #define SDIO_FIFO_FIFODATA ((uint32_t)0xFFFFFFFF)
- #define SPI_CR1_CPHA ((uint16_t)0x0001)
- #define SPI_CR1_CPOL ((uint16_t)0x0002)
- #define SPI_CR1_MSTR ((uint16_t)0x0004)
- #define SPI_CR1_BR ((uint16_t)0x0038)
- #define SPI_CR1_BR_0 ((uint16_t)0x0008)
- #define SPI_CR1_BR_1 ((uint16_t)0x0010)
- #define SPI_CR1_BR_2 ((uint16_t)0x0020)
- #define SPI_CR1_SPE ((uint16_t)0x0040)
- #define SPI_CR1_LSBFIRST ((uint16_t)0x0080)
- #define SPI_CR1_SSI ((uint16_t)0x0100)
- #define SPI_CR1_SSM ((uint16_t)0x0200)
- #define SPI_CR1_RXONLY ((uint16_t)0x0400)
- #define SPI_CR1_DFF ((uint16_t)0x0800)
- #define SPI_CR1_CRCNEXT ((uint16_t)0x1000)
- #define SPI_CR1_CRCEN ((uint16_t)0x2000)
- #define SPI_CR1_BIDIOE ((uint16_t)0x4000)
- #define SPI_CR1_BIDIMODE ((uint16_t)0x8000)
- #define SPI_CR2_RXDMAEN ((uint8_t)0x01)
- #define SPI_CR2_TXDMAEN ((uint8_t)0x02)
- #define SPI_CR2_SSOE ((uint8_t)0x04)
- #define SPI_CR2_ERRIE ((uint8_t)0x20)
- #define SPI_CR2_RXNEIE ((uint8_t)0x40)
- #define SPI_CR2_TXEIE ((uint8_t)0x80)
- #define SPI_SR_RXNE ((uint8_t)0x01)
- #define SPI_SR_TXE ((uint8_t)0x02)
- #define SPI_SR_CHSIDE ((uint8_t)0x04)
- #define SPI_SR_UDR ((uint8_t)0x08)
- #define SPI_SR_CRCERR ((uint8_t)0x10)
- #define SPI_SR_MODF ((uint8_t)0x20)
- #define SPI_SR_OVR ((uint8_t)0x40)
- #define SPI_SR_BSY ((uint8_t)0x80)
- #define SPI_DR_DR ((uint16_t)0xFFFF)
- #define SPI_CRCPR_CRCPOLY ((uint16_t)0xFFFF)
- #define SPI_RXCRCR_RXCRC ((uint16_t)0xFFFF)

- #define SPI_TXCRCR_TXCRC ((uint16_t)0xFFFF)
- #define SPI_I2SCFGR_CHLEN ((uint16_t)0x0001)
- #define SPI_I2SCFGR_DATLEN ((uint16_t)0x0006)
- #define SPI_I2SCFGR_DATLEN_0 ((uint16_t)0x0002)
- #define SPI_I2SCFGR_DATLEN_1 ((uint16_t)0x0004)
- #define SPI_I2SCFGR_CKPOL ((uint16_t)0x0008)
- #define SPI_I2SCFGR_I2SSTD ((uint16_t)0x0030)
- #define SPI_I2SCFGR_I2SSTD_0 ((uint16_t)0x0010)
- #define SPI_I2SCFGR_I2SSTD_1 ((uint16_t)0x0020)
- #define SPI_I2SCFGR_PCMSYNC ((uint16_t)0x0080)
- #define SPI_I2SCFGR_I2SCFG ((uint16_t)0x0300)
- #define SPI_I2SCFGR_I2SCFG_0 ((uint16_t)0x0100)
- #define SPI_I2SCFGR_I2SCFG_1 ((uint16_t)0x0200)
- #define SPI_I2SCFGR_I2SE ((uint16_t)0x0400)
- #define SPI_I2SCFGR_I2SMOD ((uint16_t)0x0800)
- #define SPI_I2SPR_I2SDIV ((uint16_t)0x00FF)
- #define SPI_I2SPR_ODD ((uint16_t)0x0100)
- #define SPI_I2SPR_MCKOE ((uint16_t)0x0200)
- #define SYSCFG_MEMRMP_MEM_MODE ((uint32_t)0x00000003)
- #define **SYSCFG_MEMRMP_MEM_MODE_0** ((uint32_t)0x00000001)
- #define **SYSCFG_MEMRMP_MEM_MODE_1** ((uint32_t)0x00000002)
- #define SYSCFG_PMC_MII_RMII_SEL ((uint32_t)0x00800000)
- #define **SYSCFG_PMC_MII_RMII** SYSCFG_PMC_MII_RMII_SEL
- #define SYSCFG_EXTICR1_EXTI0 ((uint16_t)0x000F)
- #define SYSCFG_EXTICR1_EXTI1 ((uint16_t)0x00F0)
- #define SYSCFG_EXTICR1_EXTI2 ((uint16_t)0x0F00)
- #define SYSCFG_EXTICR1_EXTI3 ((uint16_t)0xF000)
- #define SYSCFG_EXTICR1_EXTI0_PA ((uint16_t)0x0000)

  EXTI0 configuration

- #define SYSCFG_EXTICR1_EXTI0_PB ((uint16_t)0x0001)
- #define SYSCFG_EXTICR1_EXTI0_PC ((uint16_t)0x0002)
- #define SYSCFG_EXTICR1_EXTI0_PD ((uint16_t)0x0003)
- #define SYSCFG_EXTICR1_EXTI0_PE ((uint16_t)0x0004)
- #define SYSCFG_EXTICR1_EXTI0_PF ((uint16_t)0x0005)
- #define SYSCFG_EXTICR1_EXTI0_PG ((uint16_t)0x0006)
- #define SYSCFG_EXTICR1_EXTI0_PH ((uint16_t)0x0007)
- #define SYSCFG_EXTICR1_EXTI0_PI ((uint16_t)0x0008)
- #define SYSCFG_EXTICR1_EXTI1_PA ((uint16_t)0x0000)

  EXTI1 configuration

- #define SYSCFG_EXTICR1_EXTI1_PB ((uint16_t)0x0010)
- #define SYSCFG_EXTICR1_EXTI1_PC ((uint16_t)0x0020)
- #define SYSCFG_EXTICR1_EXTI1_PD ((uint16_t)0x0030)
- #define SYSCFG_EXTICR1_EXTI1_PE ((uint16_t)0x0040)
- #define SYSCFG_EXTICR1_EXTI1_PF ((uint16_t)0x0050)
- #define SYSCFG_EXTICR1_EXTI1_PG ((uint16_t)0x0060)
- #define SYSCFG_EXTICR1_EXTI1_PH ((uint16_t)0x0070)
- #define SYSCFG_EXTICR1_EXTI1_PI ((uint16_t)0x0080)
- #define SYSCFG_EXTICR1_EXTI2_PA ((uint16_t)0x0000)

  EXTI2 configuration

- #define SYSCFG_EXTICR1_EXTI2_PB ((uint16_t)0x0100)
- #define SYSCFG_EXTICR1_EXTI2_PC ((uint16_t)0x0200)

- #define SYSCFG_EXTICR1_EXTI2_PD ((uint16_t)0x0300)
- #define SYSCFG_EXTICR1_EXTI2_PE ((uint16_t)0x0400)
- #define SYSCFG_EXTICR1_EXTI2_PF ((uint16_t)0x0500)
- #define SYSCFG_EXTICR1_EXTI2_PG ((uint16_t)0x0600)
- #define SYSCFG_EXTICR1_EXTI2_PH ((uint16_t)0x0700)
- #define SYSCFG_EXTICR1_EXTI2_PI ((uint16_t)0x0800)
- #define SYSCFG_EXTICR1_EXTI3_PA ((uint16_t)0x0000)

  EXTI3 configuration

- #define SYSCFG_EXTICR1_EXTI3_PB ((uint16_t)0x1000)
- #define SYSCFG_EXTICR1_EXTI3_PC ((uint16_t)0x2000)
- #define SYSCFG_EXTICR1_EXTI3_PD ((uint16_t)0x3000)
- #define SYSCFG_EXTICR1_EXTI3_PE ((uint16_t)0x4000)
- #define SYSCFG_EXTICR1_EXTI3_PF ((uint16_t)0x5000)
- #define SYSCFG_EXTICR1_EXTI3_PG ((uint16_t)0x6000)
- #define SYSCFG_EXTICR1_EXTI3_PH ((uint16_t)0x7000)
- #define SYSCFG_EXTICR1_EXTI3_PI ((uint16_t)0x8000)
- #define SYSCFG_EXTICR2_EXTI4 ((uint16_t)0x000F)
- #define SYSCFG_EXTICR2_EXTI5 ((uint16_t)0x00F0)
- #define SYSCFG_EXTICR2_EXTI6 ((uint16_t)0x0F00)
- #define SYSCFG_EXTICR2_EXTI7 ((uint16_t)0xF000)
- #define SYSCFG_EXTICR2_EXTI4_PA ((uint16_t)0x0000)

  EXTI4 configuration

- #define SYSCFG_EXTICR2_EXTI4_PB ((uint16_t)0x0001)
- #define SYSCFG_EXTICR2_EXTI4_PC ((uint16_t)0x0002)
- #define SYSCFG_EXTICR2_EXTI4_PD ((uint16_t)0x0003)
- #define SYSCFG_EXTICR2_EXTI4_PE ((uint16_t)0x0004)
- #define SYSCFG_EXTICR2_EXTI4_PF ((uint16_t)0x0005)
- #define SYSCFG_EXTICR2_EXTI4_PG ((uint16_t)0x0006)
- #define SYSCFG_EXTICR2_EXTI4_PH ((uint16_t)0x0007)
- #define SYSCFG_EXTICR2_EXTI4_PI ((uint16_t)0x0008)
- #define SYSCFG_EXTICR2_EXTI5_PA ((uint16_t)0x0000)

  EXTI5 configuration

- #define SYSCFG_EXTICR2_EXTI5_PB ((uint16_t)0x0010)
- #define SYSCFG_EXTICR2_EXTI5_PC ((uint16_t)0x0020)
- #define SYSCFG_EXTICR2_EXTI5_PD ((uint16_t)0x0030)
- #define SYSCFG_EXTICR2_EXTI5_PE ((uint16_t)0x0040)
- #define SYSCFG_EXTICR2_EXTI5_PF ((uint16_t)0x0050)
- #define SYSCFG_EXTICR2_EXTI5_PG ((uint16_t)0x0060)
- #define SYSCFG_EXTICR2_EXTI5_PH ((uint16_t)0x0070)
- #define SYSCFG_EXTICR2_EXTI5_PI ((uint16_t)0x0080)
- #define SYSCFG_EXTICR2_EXTI6_PA ((uint16_t)0x0000)

  EXTI6 configuration

- #define SYSCFG_EXTICR2_EXTI6_PB ((uint16_t)0x0100)
- #define SYSCFG_EXTICR2_EXTI6_PC ((uint16_t)0x0200)
- #define SYSCFG_EXTICR2_EXTI6_PD ((uint16_t)0x0300)
- #define SYSCFG_EXTICR2_EXTI6_PE ((uint16_t)0x0400)
- #define SYSCFG_EXTICR2_EXTI6_PF ((uint16_t)0x0500)
- #define SYSCFG_EXTICR2_EXTI6_PG ((uint16_t)0x0600)
- #define SYSCFG_EXTICR2_EXTI6_PH ((uint16_t)0x0700)
- #define SYSCFG_EXTICR2_EXTI6_PI ((uint16_t)0x0800)

- #define SYSCFG_EXTICR2_EXTI7_PA ((uint16_t)0x0000)

  EXTI7 configuration

- #define SYSCFG_EXTICR2_EXTI7_PB ((uint16_t)0x1000)
- #define SYSCFG_EXTICR2_EXTI7_PC ((uint16_t)0x2000)
- #define SYSCFG_EXTICR2_EXTI7_PD ((uint16_t)0x3000)
- #define SYSCFG_EXTICR2_EXTI7_PE ((uint16_t)0x4000)
- #define SYSCFG_EXTICR2_EXTI7_PF ((uint16_t)0x5000)
- #define SYSCFG_EXTICR2_EXTI7_PG ((uint16_t)0x6000)
- #define SYSCFG_EXTICR2_EXTI7_PH ((uint16_t)0x7000)
- #define SYSCFG_EXTICR2_EXTI7_PI ((uint16_t)0x8000)
- #define SYSCFG_EXTICR3_EXTI8 ((uint16_t)0x000F)
- #define SYSCFG_EXTICR3_EXTI9 ((uint16_t)0x00F0)
- #define SYSCFG_EXTICR3_EXTI10 ((uint16_t)0x0F00)
- #define SYSCFG_EXTICR3_EXTI11 ((uint16_t)0xF000)
- #define SYSCFG_EXTICR3_EXTI8_PA ((uint16_t)0x0000)

  EXTI8 configuration

- #define SYSCFG_EXTICR3_EXTI8_PB ((uint16_t)0x0001)
- #define SYSCFG_EXTICR3_EXTI8_PC ((uint16_t)0x0002)
- #define SYSCFG_EXTICR3_EXTI8_PD ((uint16_t)0x0003)
- #define SYSCFG_EXTICR3_EXTI8_PE ((uint16_t)0x0004)
- #define SYSCFG_EXTICR3_EXTI8_PF ((uint16_t)0x0005)
- #define SYSCFG_EXTICR3_EXTI8_PG ((uint16_t)0x0006)
- #define SYSCFG_EXTICR3_EXTI8_PH ((uint16_t)0x0007)
- #define SYSCFG_EXTICR3_EXTI8_PI ((uint16_t)0x0008)
- #define SYSCFG_EXTICR3_EXTI9_PA ((uint16_t)0x0000)

  EXTI9 configuration

- #define SYSCFG_EXTICR3_EXTI9_PB ((uint16_t)0x0010)
- #define SYSCFG_EXTICR3_EXTI9_PC ((uint16_t)0x0020)
- #define SYSCFG_EXTICR3_EXTI9_PD ((uint16_t)0x0030)
- #define SYSCFG_EXTICR3_EXTI9_PE ((uint16_t)0x0040)
- #define SYSCFG_EXTICR3_EXTI9_PF ((uint16_t)0x0050)
- #define SYSCFG_EXTICR3_EXTI9_PG ((uint16_t)0x0060)
- #define SYSCFG_EXTICR3_EXTI9_PH ((uint16_t)0x0070)
- #define SYSCFG_EXTICR3_EXTI9_PI ((uint16_t)0x0080)
- #define SYSCFG_EXTICR3_EXTI10_PA ((uint16_t)0x0000)

  EXTI10 configuration

- #define SYSCFG_EXTICR3_EXTI10_PB ((uint16_t)0x0100)
- #define SYSCFG_EXTICR3_EXTI10_PC ((uint16_t)0x0200)
- #define SYSCFG_EXTICR3_EXTI10_PD ((uint16_t)0x0300)
- #define SYSCFG_EXTICR3_EXTI10_PE ((uint16_t)0x0400)
- #define SYSCFG_EXTICR3_EXTI10_PF ((uint16_t)0x0500)
- #define SYSCFG_EXTICR3_EXTI10_PG ((uint16_t)0x0600)
- #define SYSCFG_EXTICR3_EXTI10_PH ((uint16_t)0x0700)
- #define SYSCFG_EXTICR3_EXTI10_PI ((uint16_t)0x0800)
- #define SYSCFG_EXTICR3_EXTI11_PA ((uint16_t)0x0000)

  EXTI11 configuration

- #define SYSCFG_EXTICR3_EXTI11_PB ((uint16_t)0x1000)
- #define SYSCFG_EXTICR3_EXTI11_PC ((uint16_t)0x2000)
- #define SYSCFG_EXTICR3_EXTI11_PD ((uint16_t)0x3000)

- #define SYSCFG_EXTICR3_EXTI11_PE ((uint16_t)0x4000)
- #define SYSCFG_EXTICR3_EXTI11_PF ((uint16_t)0x5000)
- #define SYSCFG_EXTICR3_EXTI11_PG ((uint16_t)0x6000)
- #define SYSCFG_EXTICR3_EXTI11_PH ((uint16_t)0x7000)
- #define SYSCFG_EXTICR3_EXTI11_PI ((uint16_t)0x8000)
- #define SYSCFG_EXTICR4_EXTI12 ((uint16_t)0x000F)
- #define SYSCFG_EXTICR4_EXTI13 ((uint16_t)0x00F0)
- #define SYSCFG_EXTICR4_EXTI14 ((uint16_t)0x0F00)
- #define SYSCFG_EXTICR4_EXTI15 ((uint16_t)0xF000)
- #define SYSCFG_EXTICR4_EXTI12_PA ((uint16_t)0x0000)

  EXTI12 configuration

- #define SYSCFG_EXTICR4_EXTI12_PB ((uint16_t)0x0001)
- #define SYSCFG_EXTICR4_EXTI12_PC ((uint16_t)0x0002)
- #define SYSCFG_EXTICR4_EXTI12_PD ((uint16_t)0x0003)
- #define SYSCFG_EXTICR4_EXTI12_PE ((uint16_t)0x0004)
- #define SYSCFG_EXTICR4_EXTI12_PF ((uint16_t)0x0005)
- #define SYSCFG_EXTICR4_EXTI12_PG ((uint16_t)0x0006)
- #define SYSCFG_EXTICR3_EXTI12_PH ((uint16_t)0x0007)
- #define SYSCFG_EXTICR4_EXTI13_PA ((uint16_t)0x0000)

  EXTI13 configuration

- #define SYSCFG_EXTICR4_EXTI13_PB ((uint16_t)0x0010)
- #define SYSCFG_EXTICR4_EXTI13_PC ((uint16_t)0x0020)
- #define SYSCFG_EXTICR4_EXTI13_PD ((uint16_t)0x0030)
- #define SYSCFG_EXTICR4_EXTI13_PE ((uint16_t)0x0040)
- #define SYSCFG_EXTICR4_EXTI13_PF ((uint16_t)0x0050)
- #define SYSCFG_EXTICR4_EXTI13_PG ((uint16_t)0x0060)
- #define SYSCFG_EXTICR3_EXTI13_PH ((uint16_t)0x0070)
- #define SYSCFG_EXTICR4_EXTI14_PA ((uint16_t)0x0000)

  EXTI14 configuration

- #define SYSCFG_EXTICR4_EXTI14_PB ((uint16_t)0x0100)
- #define SYSCFG_EXTICR4_EXTI14_PC ((uint16_t)0x0200)
- #define SYSCFG_EXTICR4_EXTI14_PD ((uint16_t)0x0300)
- #define SYSCFG_EXTICR4_EXTI14_PE ((uint16_t)0x0400)
- #define SYSCFG_EXTICR4_EXTI14_PF ((uint16_t)0x0500)
- #define SYSCFG_EXTICR4_EXTI14_PG ((uint16_t)0x0600)
- #define SYSCFG_EXTICR3_EXTI14_PH ((uint16_t)0x0700)
- #define SYSCFG_EXTICR4_EXTI15_PA ((uint16_t)0x0000)

  EXTI15 configuration

- #define SYSCFG_EXTICR4_EXTI15_PB ((uint16_t)0x1000)
- #define SYSCFG_EXTICR4_EXTI15_PC ((uint16_t)0x2000)
- #define SYSCFG_EXTICR4_EXTI15_PD ((uint16_t)0x3000)
- #define SYSCFG_EXTICR4_EXTI15_PE ((uint16_t)0x4000)
- #define SYSCFG_EXTICR4_EXTI15_PF ((uint16_t)0x5000)
- #define SYSCFG_EXTICR4_EXTI15_PG ((uint16_t)0x6000)
- #define SYSCFG_EXTICR3_EXTI15_PH ((uint16_t)0x7000)
- #define SYSCFG_CMPCR_CMP_PD ((uint32_t)0x00000001)
- #define SYSCFG_CMPCR_READY ((uint32_t)0x00000100)
- #define TIM_CR1_CEN ((uint16_t)0x0001)
- #define TIM_CR1_UDIS ((uint16_t)0x0002)
- #define TIM_CR1_URS ((uint16_t)0x0004)

- #define TIM_CR1_OPM ((uint16_t)0x0008)
- #define TIM_CR1_DIR ((uint16_t)0x0010)
- #define TIM_CR1_CMS ((uint16_t)0x0060)
- #define TIM_CR1_CMS_0 ((uint16_t)0x0020)
- #define TIM_CR1_CMS_1 ((uint16_t)0x0040)
- #define TIM_CR1_ARPE ((uint16_t)0x0080)
- #define TIM_CR1_CKD ((uint16_t)0x0300)
- #define TIM_CR1_CKD_0 ((uint16_t)0x0100)
- #define TIM_CR1_CKD_1 ((uint16_t)0x0200)
- #define TIM_CR2_CCPC ((uint16_t)0x0001)
- #define TIM_CR2_CCUS ((uint16_t)0x0004)
- #define TIM_CR2_CCDS ((uint16_t)0x0008)
- #define TIM_CR2_MMS ((uint16_t)0x0070)
- #define TIM_CR2_MMS_0 ((uint16_t)0x0010)
- #define TIM_CR2_MMS_1 ((uint16_t)0x0020)
- #define TIM_CR2_MMS_2 ((uint16_t)0x0040)
- #define TIM_CR2_TI1S ((uint16_t)0x0080)
- #define TIM_CR2_OIS1 ((uint16_t)0x0100)
- #define TIM_CR2_OIS1N ((uint16_t)0x0200)
- #define TIM_CR2_OIS2 ((uint16_t)0x0400)
- #define TIM_CR2_OIS2N ((uint16_t)0x0800)
- #define TIM_CR2_OIS3 ((uint16_t)0x1000)
- #define TIM_CR2_OIS3N ((uint16_t)0x2000)
- #define TIM_CR2_OIS4 ((uint16_t)0x4000)
- #define TIM_SMCR_SMS ((uint16_t)0x0007)
- #define TIM_SMCR_SMS_0 ((uint16_t)0x0001)
- #define TIM_SMCR_SMS_1 ((uint16_t)0x0002)
- #define TIM_SMCR_SMS_2 ((uint16_t)0x0004)
- #define TIM_SMCR_TS ((uint16_t)0x0070)
- #define TIM_SMCR_TS_0 ((uint16_t)0x0010)
- #define TIM_SMCR_TS_1 ((uint16_t)0x0020)
- #define TIM_SMCR_TS_2 ((uint16_t)0x0040)
- #define TIM_SMCR_MSM ((uint16_t)0x0080)
- #define TIM_SMCR_ETF ((uint16_t)0x0F00)
- #define TIM_SMCR_ETF_0 ((uint16_t)0x0100)
- #define TIM_SMCR_ETF_1 ((uint16_t)0x0200)
- #define TIM_SMCR_ETF_2 ((uint16_t)0x0400)
- #define TIM_SMCR_ETF_3 ((uint16_t)0x0800)
- #define TIM_SMCR_ETPS ((uint16_t)0x3000)
- #define TIM_SMCR_ETPS_0 ((uint16_t)0x1000)
- #define TIM_SMCR_ETPS_1 ((uint16_t)0x2000)
- #define TIM_SMCR_ECE ((uint16_t)0x4000)
- #define TIM_SMCR_ETP ((uint16_t)0x8000)
- #define TIM_DIER_UIE ((uint16_t)0x0001)
- #define TIM_DIER_CC1IE ((uint16_t)0x0002)
- #define TIM_DIER_CC2IE ((uint16_t)0x0004)
- #define TIM_DIER_CC3IE ((uint16_t)0x0008)
- #define TIM_DIER_CC4IE ((uint16_t)0x0010)
- #define TIM_DIER_COMIE ((uint16_t)0x0020)
- #define TIM_DIER_TIE ((uint16_t)0x0040)
- #define TIM_DIER_BIE ((uint16_t)0x0080)
- #define TIM_DIER_UDE ((uint16_t)0x0100)
- #define TIM_DIER_CC1DE ((uint16_t)0x0200)
- #define TIM_DIER_CC2DE ((uint16_t)0x0400)
- #define TIM_DIER_CC3DE ((uint16_t)0x0800)

- #define TIM_DIER_CC4DE ((uint16_t)0x1000)
- #define TIM_DIER_COMDE ((uint16_t)0x2000)
- #define TIM_DIER_TDE ((uint16_t)0x4000)
- #define TIM_SR_UIF ((uint16_t)0x0001)
- #define TIM_SR_CC1IF ((uint16_t)0x0002)
- #define TIM_SR_CC2IF ((uint16_t)0x0004)
- #define TIM_SR_CC3IF ((uint16_t)0x0008)
- #define TIM_SR_CC4IF ((uint16_t)0x0010)
- #define TIM_SR_COMIF ((uint16_t)0x0020)
- #define TIM_SR_TIF ((uint16_t)0x0040)
- #define TIM_SR_BIF ((uint16_t)0x0080)
- #define TIM_SR_CC1OF ((uint16_t)0x0200)
- #define TIM_SR_CC2OF ((uint16_t)0x0400)
- #define TIM_SR_CC3OF ((uint16_t)0x0800)
- #define TIM_SR_CC4OF ((uint16_t)0x1000)
- #define TIM_EGR_UG ((uint8_t)0x01)
- #define TIM_EGR_CC1G ((uint8_t)0x02)
- #define TIM_EGR_CC2G ((uint8_t)0x04)
- #define TIM_EGR_CC3G ((uint8_t)0x08)
- #define TIM_EGR_CC4G ((uint8_t)0x10)
- #define TIM_EGR_COMG ((uint8_t)0x20)
- #define TIM_EGR_TG ((uint8_t)0x40)
- #define TIM_EGR_BG ((uint8_t)0x80)
- #define TIM_CCMR1_CC1S ((uint16_t)0x0003)
- #define TIM_CCMR1_CC1S_0 ((uint16_t)0x0001)
- #define TIM_CCMR1_CC1S_1 ((uint16_t)0x0002)
- #define TIM_CCMR1_OC1FE ((uint16_t)0x0004)
- #define TIM_CCMR1_OC1PE ((uint16_t)0x0008)
- #define TIM_CCMR1_OC1M ((uint16_t)0x0070)
- #define TIM_CCMR1_OC1M_0 ((uint16_t)0x0010)
- #define TIM_CCMR1_OC1M_1 ((uint16_t)0x0020)
- #define TIM_CCMR1_OC1M_2 ((uint16_t)0x0040)
- #define TIM_CCMR1_OC1CE ((uint16_t)0x0080)
- #define TIM_CCMR1_CC2S ((uint16_t)0x0300)
- #define TIM_CCMR1_CC2S_0 ((uint16_t)0x0100)
- #define TIM_CCMR1_CC2S_1 ((uint16_t)0x0200)
- #define TIM_CCMR1_OC2FE ((uint16_t)0x0400)
- #define TIM_CCMR1_OC2PE ((uint16_t)0x0800)
- #define TIM_CCMR1_OC2M ((uint16_t)0x7000)
- #define TIM_CCMR1_OC2M_0 ((uint16_t)0x1000)
- #define TIM_CCMR1_OC2M_1 ((uint16_t)0x2000)
- #define TIM_CCMR1_OC2M_2 ((uint16_t)0x4000)
- #define TIM_CCMR1_OC2CE ((uint16_t)0x8000)
- #define TIM_CCMR1_IC1PSC ((uint16_t)0x000C)
- #define TIM_CCMR1_IC1PSC_0 ((uint16_t)0x0004)
- #define TIM_CCMR1_IC1PSC_1 ((uint16_t)0x0008)
- #define TIM_CCMR1_IC1F ((uint16_t)0x00F0)
- #define TIM_CCMR1_IC1F_0 ((uint16_t)0x0010)
- #define TIM_CCMR1_IC1F_1 ((uint16_t)0x0020)
- #define TIM_CCMR1_IC1F_2 ((uint16_t)0x0040)
- #define TIM_CCMR1_IC1F_3 ((uint16_t)0x0080)
- #define TIM_CCMR1_IC2PSC ((uint16_t)0x0C00)
- #define TIM_CCMR1_IC2PSC_0 ((uint16_t)0x0400)
- #define TIM_CCMR1_IC2PSC_1 ((uint16_t)0x0800)
- #define TIM_CCMR1_IC2F ((uint16_t)0xF000)

- #define TIM_CCMR1_IC2F_0 ((uint16_t)0x1000)
- #define TIM_CCMR1_IC2F_1 ((uint16_t)0x2000)
- #define TIM_CCMR1_IC2F_2 ((uint16_t)0x4000)
- #define TIM_CCMR1_IC2F_3 ((uint16_t)0x8000)
- #define TIM_CCMR2_CC3S ((uint16_t)0x0003)
- #define TIM_CCMR2_CC3S_0 ((uint16_t)0x0001)
- #define TIM_CCMR2_CC3S_1 ((uint16_t)0x0002)
- #define TIM_CCMR2_OC3FE ((uint16_t)0x0004)
- #define TIM_CCMR2_OC3PE ((uint16_t)0x0008)
- #define TIM_CCMR2_OC3M ((uint16_t)0x0070)
- #define TIM_CCMR2_OC3M_0 ((uint16_t)0x0010)
- #define TIM_CCMR2_OC3M_1 ((uint16_t)0x0020)
- #define TIM_CCMR2_OC3M_2 ((uint16_t)0x0040)
- #define TIM_CCMR2_OC3CE ((uint16_t)0x0080)
- #define TIM_CCMR2_CC4S ((uint16_t)0x0300)
- #define TIM_CCMR2_CC4S_0 ((uint16_t)0x0100)
- #define TIM_CCMR2_CC4S_1 ((uint16_t)0x0200)
- #define TIM_CCMR2_OC4FE ((uint16_t)0x0400)
- #define TIM_CCMR2_OC4PE ((uint16_t)0x0800)
- #define TIM_CCMR2_OC4M ((uint16_t)0x7000)
- #define TIM_CCMR2_OC4M_0 ((uint16_t)0x1000)
- #define TIM_CCMR2_OC4M_1 ((uint16_t)0x2000)
- #define TIM_CCMR2_OC4M_2 ((uint16_t)0x4000)
- #define TIM_CCMR2_OC4CE ((uint16_t)0x8000)
- #define TIM_CCMR2_IC3PSC ((uint16_t)0x000C)
- #define TIM_CCMR2_IC3PSC_0 ((uint16_t)0x0004)
- #define TIM_CCMR2_IC3PSC_1 ((uint16_t)0x0008)
- #define TIM_CCMR2_IC3F ((uint16_t)0x00F0)
- #define TIM_CCMR2_IC3F_0 ((uint16_t)0x0010)
- #define TIM_CCMR2_IC3F_1 ((uint16_t)0x0020)
- #define TIM_CCMR2_IC3F_2 ((uint16_t)0x0040)
- #define TIM_CCMR2_IC3F_3 ((uint16_t)0x0080)
- #define TIM_CCMR2_IC4PSC ((uint16_t)0x0C00)
- #define TIM_CCMR2_IC4PSC_0 ((uint16_t)0x0400)
- #define TIM_CCMR2_IC4PSC_1 ((uint16_t)0x0800)
- #define TIM_CCMR2_IC4F ((uint16_t)0xF000)
- #define TIM_CCMR2_IC4F_0 ((uint16_t)0x1000)
- #define TIM_CCMR2_IC4F_1 ((uint16_t)0x2000)
- #define TIM_CCMR2_IC4F_2 ((uint16_t)0x4000)
- #define TIM_CCMR2_IC4F_3 ((uint16_t)0x8000)
- #define TIM_CCER_CC1E ((uint16_t)0x0001)
- #define TIM_CCER_CC1P ((uint16_t)0x0002)
- #define TIM_CCER_CC1NE ((uint16_t)0x0004)
- #define TIM_CCER_CC1NP ((uint16_t)0x0008)
- #define TIM_CCER_CC2E ((uint16_t)0x0010)
- #define TIM_CCER_CC2P ((uint16_t)0x0020)
- #define TIM_CCER_CC2NE ((uint16_t)0x0040)
- #define TIM_CCER_CC2NP ((uint16_t)0x0080)
- #define TIM_CCER_CC3E ((uint16_t)0x0100)
- #define TIM_CCER_CC3P ((uint16_t)0x0200)
- #define TIM_CCER_CC3NE ((uint16_t)0x0400)
- #define TIM_CCER_CC3NP ((uint16_t)0x0800)
- #define TIM_CCER_CC4E ((uint16_t)0x1000)
- #define TIM_CCER_CC4P ((uint16_t)0x2000)
- #define TIM_CCER_CC4NP ((uint16_t)0x8000)

- #define TIM_CNT_CNT ((uint16_t)0xFFFF)
- #define TIM_PSC_PSC ((uint16_t)0xFFFF)
- #define TIM_ARR_ARR ((uint16_t)0xFFFF)
- #define TIM_RCR_REP ((uint8_t)0xFF)
- #define TIM_CCR1_CCR1 ((uint16_t)0xFFFF)
- #define TIM_CCR2_CCR2 ((uint16_t)0xFFFF)
- #define TIM_CCR3_CCR3 ((uint16_t)0xFFFF)
- #define TIM_CCR4_CCR4 ((uint16_t)0xFFFF)
- #define TIM_BDTR_DTG ((uint16_t)0x00FF)
- #define TIM_BDTR_DTG_0 ((uint16_t)0x0001)
- #define TIM_BDTR_DTG_1 ((uint16_t)0x0002)
- #define TIM_BDTR_DTG_2 ((uint16_t)0x0004)
- #define TIM_BDTR_DTG_3 ((uint16_t)0x0008)
- #define TIM_BDTR_DTG_4 ((uint16_t)0x0010)
- #define TIM_BDTR_DTG_5 ((uint16_t)0x0020)
- #define TIM_BDTR_DTG_6 ((uint16_t)0x0040)
- #define TIM_BDTR_DTG_7 ((uint16_t)0x0080)
- #define TIM_BDTR_LOCK ((uint16_t)0x0300)
- #define TIM_BDTR_LOCK_0 ((uint16_t)0x0100)
- #define TIM_BDTR_LOCK_1 ((uint16_t)0x0200)
- #define TIM_BDTR_OSSI ((uint16_t)0x0400)
- #define TIM_BDTR_OSSR ((uint16_t)0x0800)
- #define TIM_BDTR_BKE ((uint16_t)0x1000)
- #define TIM_BDTR_BKP ((uint16_t)0x2000)
- #define TIM_BDTR_AOE ((uint16_t)0x4000)
- #define TIM_BDTR_MOE ((uint16_t)0x8000)
- #define TIM_DCR_DBA ((uint16_t)0x001F)
- #define TIM_DCR_DBA_0 ((uint16_t)0x0001)
- #define TIM_DCR_DBA_1 ((uint16_t)0x0002)
- #define TIM_DCR_DBA_2 ((uint16_t)0x0004)
- #define TIM_DCR_DBA_3 ((uint16_t)0x0008)
- #define TIM_DCR_DBA_4 ((uint16_t)0x0010)
- #define TIM_DCR_DBL ((uint16_t)0x1F00)
- #define TIM_DCR_DBL_0 ((uint16_t)0x0100)
- #define TIM_DCR_DBL_1 ((uint16_t)0x0200)
- #define TIM_DCR_DBL_2 ((uint16_t)0x0400)
- #define TIM_DCR_DBL_3 ((uint16_t)0x0800)
- #define TIM_DCR_DBL_4 ((uint16_t)0x1000)
- #define TIM_DMAR_DMAB ((uint16_t)0xFFFF)
- #define TIM_OR_TI4_RMP ((uint16_t)0x00C0)
- #define TIM_OR_TI4_RMP_0 ((uint16_t)0x0040)
- #define TIM_OR_TI4_RMP_1 ((uint16_t)0x0080)
- #define TIM_OR_ITR1_RMP ((uint16_t)0x0C00)
- #define TIM_OR_ITR1_RMP_0 ((uint16_t)0x0400)
- #define TIM_OR_ITR1_RMP_1 ((uint16_t)0x0800)
- #define USART_SR_PE ((uint16_t)0x0001)
- #define USART_SR_FE ((uint16_t)0x0002)
- #define USART_SR_NE ((uint16_t)0x0004)
- #define USART_SR_ORE ((uint16_t)0x0008)
- #define USART_SR_IDLE ((uint16_t)0x0010)
- #define USART_SR_RXNE ((uint16_t)0x0020)
- #define USART_SR_TC ((uint16_t)0x0040)
- #define USART_SR_TXE ((uint16_t)0x0080)
- #define USART_SR_LBD ((uint16_t)0x0100)
- #define USART_SR_CTS ((uint16_t)0x0200)

- #define USART_DR_DR ((uint16_t)0x01FF)
- #define USART_BRR_DIV_Fraction ((uint16_t)0x000F)
- #define USART_BRR_DIV_Mantissa ((uint16_t)0xFFF0)
- #define USART_CR1_SBK ((uint16_t)0x0001)
- #define USART_CR1_RWU ((uint16_t)0x0002)
- #define USART_CR1_RE ((uint16_t)0x0004)
- #define USART_CR1_TE ((uint16_t)0x0008)
- #define USART_CR1_IDLEIE ((uint16_t)0x0010)
- #define USART_CR1_RXNEIE ((uint16_t)0x0020)
- #define USART_CR1_TCIE ((uint16_t)0x0040)
- #define USART_CR1_TXEIE ((uint16_t)0x0080)
- #define USART_CR1_PEIE ((uint16_t)0x0100)
- #define USART_CR1_PS ((uint16_t)0x0200)
- #define USART_CR1_PCE ((uint16_t)0x0400)
- #define USART_CR1_WAKE ((uint16_t)0x0800)
- #define USART_CR1_M ((uint16_t)0x1000)
- #define USART_CR1_UE ((uint16_t)0x2000)
- #define USART_CR1_OVER8 ((uint16_t)0x8000)
- #define USART_CR2_ADD ((uint16_t)0x000F)
- #define USART_CR2_LBDL ((uint16_t)0x0020)
- #define USART_CR2_LBDIE ((uint16_t)0x0040)
- #define USART_CR2_LBCL ((uint16_t)0x0100)
- #define USART_CR2_CPHA ((uint16_t)0x0200)
- #define USART_CR2_CPOL ((uint16_t)0x0400)
- #define USART_CR2_CLKEN ((uint16_t)0x0800)
- #define USART_CR2_STOP ((uint16_t)0x3000)
- #define USART_CR2_STOP_0 ((uint16_t)0x1000)
- #define USART_CR2_STOP_1 ((uint16_t)0x2000)
- #define USART_CR2_LINEN ((uint16_t)0x4000)
- #define USART_CR3_EIE ((uint16_t)0x0001)
- #define USART_CR3_IREN ((uint16_t)0x0002)
- #define USART_CR3_IRLP ((uint16_t)0x0004)
- #define USART_CR3_HDSEL ((uint16_t)0x0008)
- #define USART_CR3_NACK ((uint16_t)0x0010)
- #define USART_CR3_SCEN ((uint16_t)0x0020)
- #define USART_CR3_DMAR ((uint16_t)0x0040)
- #define USART_CR3_DMAT ((uint16_t)0x0080)
- #define USART_CR3_RTSE ((uint16_t)0x0100)
- #define USART_CR3_CTSE ((uint16_t)0x0200)
- #define USART_CR3_CTSIE ((uint16_t)0x0400)
- #define USART_CR3_ONEBIT ((uint16_t)0x0800)
- #define USART_GTPR_PSC ((uint16_t)0x00FF)
- #define USART_GTPR_PSC_0 ((uint16_t)0x0001)
- #define USART_GTPR_PSC_1 ((uint16_t)0x0002)
- #define USART_GTPR_PSC_2 ((uint16_t)0x0004)
- #define USART_GTPR_PSC_3 ((uint16_t)0x0008)
- #define USART_GTPR_PSC_4 ((uint16_t)0x0010)
- #define USART_GTPR_PSC_5 ((uint16_t)0x0020)
- #define USART_GTPR_PSC_6 ((uint16_t)0x0040)
- #define USART_GTPR_PSC_7 ((uint16_t)0x0080)
- #define USART_GTPR_GT ((uint16_t)0xFF00)
- #define WWDG_CR_T ((uint8_t)0x7F)
- #define WWDG_CR_T0 ((uint8_t)0x01)
- #define WWDG_CR_T1 ((uint8_t)0x02)
- #define WWDG_CR_T2 ((uint8_t)0x04)

- #define WWDG_CR_T3 ((uint8_t)0x08)
- #define WWDG_CR_T4 ((uint8_t)0x10)
- #define WWDG_CR_T5 ((uint8_t)0x20)
- #define WWDG_CR_T6 ((uint8_t)0x40)
- #define WWDG_CR_WDGA ((uint8_t)0x80)
- #define WWDG_CFR_W ((uint16_t)0x007F)
- #define WWDG_CFR_W0 ((uint16_t)0x0001)
- #define WWDG_CFR_W1 ((uint16_t)0x0002)
- #define WWDG_CFR_W2 ((uint16_t)0x0004)
- #define WWDG_CFR_W3 ((uint16_t)0x0008)
- #define WWDG_CFR_W4 ((uint16_t)0x0010)
- #define WWDG_CFR_W5 ((uint16_t)0x0020)
- #define WWDG_CFR_W6 ((uint16_t)0x0040)
- #define WWDG_CFR_WDGTB ((uint16_t)0x0180)
- #define WWDG_CFR_WDGTB0 ((uint16_t)0x0080)
- #define WWDG_CFR_WDGTB1 ((uint16_t)0x0100)
- #define WWDG_CFR_EWI ((uint16_t)0x0200)
- #define WWDG_SR_EWIF ((uint8_t)0x01)
- #define **DBGMCU_IDCODE_DEV_ID** ((uint32_t)0x00000FFF)
- #define **DBGMCU_IDCODE_REV_ID** ((uint32_t)0xFFFF0000)
- #define **DBGMCU_CR_DBG_SLEEP** ((uint32_t)0x00000001)
- #define **DBGMCU_CR_DBG_STOP** ((uint32_t)0x00000002)
- #define **DBGMCU_CR_DBG_STANDBY** ((uint32_t)0x00000004)
- #define **DBGMCU_CR_TRACE_IOEN** ((uint32_t)0x00000020)
- #define **DBGMCU_CR_TRACE_MODE** ((uint32_t)0x000000C0)
- #define DBGMCU_CR_TRACE_MODE_0 ((uint32_t)0x00000040)
- #define DBGMCU_CR_TRACE_MODE_1 ((uint32_t)0x00000080)
- #define **DBGMCU_APB1_FZ_DBG_TIM2_STOP** ((uint32_t)0x00000001)
- #define **DBGMCU_APB1_FZ_DBG_TIM3_STOP** ((uint32_t)0x00000002)
- #define **DBGMCU_APB1_FZ_DBG_TIM4_STOP** ((uint32_t)0x00000004)
- #define **DBGMCU_APB1_FZ_DBG_TIM5_STOP** ((uint32_t)0x00000008)
- #define **DBGMCU_APB1_FZ_DBG_TIM6_STOP** ((uint32_t)0x00000010)
- #define **DBGMCU_APB1_FZ_DBG_TIM7_STOP** ((uint32_t)0x00000020)
- #define **DBGMCU_APB1_FZ_DBG_TIM12_STOP** ((uint32_t)0x00000040)
- #define **DBGMCU_APB1_FZ_DBG_TIM13_STOP** ((uint32_t)0x00000080)
- #define **DBGMCU_APB1_FZ_DBG_TIM14_STOP** ((uint32_t)0x00000100)
- #define **DBGMCU_APB1_FZ_DBG_RTC_STOP** ((uint32_t)0x00000400)
- #define **DBGMCU_APB1_FZ_DBG_WWDG_STOP** ((uint32_t)0x00000800)
- #define **DBGMCU_APB1_FZ_DBG_IWDG_STOP** ((uint32_t)0x00001000)
- #define **DBGMCU_APB1_FZ_DBG_I2C1_SMBUS_TIMEOUT** ((uint32_t)0x00200000)
- #define **DBGMCU_APB1_FZ_DBG_I2C2_SMBUS_TIMEOUT** ((uint32_t)0x00400000)
- #define **DBGMCU_APB1_FZ_DBG_I2C3_SMBUS_TIMEOUT** ((uint32_t)0x00800000)
- #define **DBGMCU_APB1_FZ_DBG_CAN1_STOP** ((uint32_t)0x02000000)
- #define **DBGMCU_APB1_FZ_DBG_CAN2_STOP** ((uint32_t)0x04000000)
- #define **DBGMCU_APB1_FZ_DBG_IWDEG_STOP** DBGMCU_APB1_FZ_DBG_IWDG_STOP
- #define **DBGMCU_APB1_FZ_DBG_TIM1_STOP** ((uint32_t)0x00000001)
- #define **DBGMCU_APB1_FZ_DBG_TIM8_STOP** ((uint32_t)0x00000002)
- #define **DBGMCU_APB1_FZ_DBG_TIM9_STOP** ((uint32_t)0x00010000)
- #define **DBGMCU_APB1_FZ_DBG_TIM10_STOP** ((uint32_t)0x00020000)
- #define **DBGMCU_APB1_FZ_DBG_TIM11_STOP** ((uint32_t)0x00040000)
- #define **ETH_MACCR_WD** ((uint32_t)0x00800000) /* Watchdog disable */
- #define **ETH_MACCR_JD** ((uint32_t)0x00400000) /* Jabber disable */
- #define **ETH_MACCR_IFG** ((uint32_t)0x000E0000) /* Inter-frame gap */
- #define **ETH_MACCR_IFG_96Bit** ((uint32_t)0x00000000) /* Minimum IFG between frames during transmission is 96Bit */

- #define **ETH_MACCR_IFG_88Bit** ((uint32_t)0x00020000) /* Minimum IFG between frames during transmission is 88Bit */
- #define **ETH_MACCR_IFG_80Bit** ((uint32_t)0x00040000) /* Minimum IFG between frames during transmission is 80Bit */
- #define **ETH_MACCR_IFG_72Bit** ((uint32_t)0x00060000) /* Minimum IFG between frames during transmission is 72Bit */
- #define **ETH_MACCR_IFG_64Bit** ((uint32_t)0x00080000) /* Minimum IFG between frames during transmission is 64Bit */
- #define **ETH_MACCR_IFG_56Bit** ((uint32_t)0x000A0000) /* Minimum IFG between frames during transmission is 56Bit */
- #define **ETH_MACCR_IFG_48Bit** ((uint32_t)0x000C0000) /* Minimum IFG between frames during transmission is 48Bit */
- #define **ETH_MACCR_IFG_40Bit** ((uint32_t)0x000E0000) /* Minimum IFG between frames during transmission is 40Bit */
- #define **ETH_MACCR_CSD** ((uint32_t)0x00010000) /* Carrier sense disable (during transmission) */
- #define **ETH_MACCR_FES** ((uint32_t)0x00004000) /* Fast ethernet speed */
- #define **ETH_MACCR_ROD** ((uint32_t)0x00002000) /* Receive own disable */
- #define **ETH_MACCR_LM** ((uint32_t)0x00001000) /* loopback mode */
- #define **ETH_MACCR_DM** ((uint32_t)0x00000800) /* Duplex mode */
- #define **ETH_MACCR_IPCO** ((uint32_t)0x00000400) /* IP Checksum offload */
- #define **ETH_MACCR_RD** ((uint32_t)0x00000200) /* Retry disable */
- #define **ETH_MACCR_APCS** ((uint32_t)0x00000080) /* Automatic Pad/CRC stripping */
- #define ETH_MACCR_BL
- #define **ETH_MACCR_BL_10** ((uint32_t)0x00000000) /* k = min (n, 10) */
- #define **ETH_MACCR_BL_8** ((uint32_t)0x00000020) /* k = min (n, 8) */
- #define **ETH_MACCR_BL_4** ((uint32_t)0x00000040) /* k = min (n, 4) */
- #define **ETH_MACCR_BL_1** ((uint32_t)0x00000060) /* k = min (n, 1) */
- #define **ETH_MACCR_DC** ((uint32_t)0x00000010) /* Defferal check */
- #define **ETH_MACCR_TE** ((uint32_t)0x00000008) /* Transmitter enable */
- #define **ETH_MACCR_RE** ((uint32_t)0x00000004) /* Receiver enable */
- #define **ETH_MACFFR_RA** ((uint32_t)0x80000000) /* Receive all */
- #define **ETH_MACFFR_HPF** ((uint32_t)0x00000400) /* Hash or perfect filter */
- #define **ETH_MACFFR_SAF** ((uint32_t)0x00000200) /* Source address filter enable */
- #define **ETH_MACFFR_SAIF** ((uint32_t)0x00000100) /* SA inverse filtering */
- #define **ETH_MACFFR_PCF** ((uint32_t)0x000000C0) /* Pass control frames: 3 cases */
- #define **ETH_MACFFR_PCF_BlockAll** ((uint32_t)0x00000040) /* MAC filters all control frames from reaching the application */
- #define **ETH_MACFFR_PCF_ForwardAll** ((uint32_t)0x00000080) /* MAC forwards all control frames to application even if they fail the Address Filter */
- #define **ETH_MACFFR_PCF_ForwardPassedAddrFilter** ((uint32_t)0x000000C0) /* MAC forwards control frames that pass the Address Filter. */
- #define **ETH_MACFFR_BFD** ((uint32_t)0x00000020) /* Broadcast frame disable */
- #define **ETH_MACFFR_PAM** ((uint32_t)0x00000010) /* Pass all mutlicast */
- #define **ETH_MACFFR_DAIF** ((uint32_t)0x00000008) /* DA Inverse filtering */
- #define **ETH_MACFFR_HM** ((uint32_t)0x00000004) /* Hash multicast */
- #define **ETH_MACFFR_HU** ((uint32_t)0x00000002) /* Hash unicast */
- #define **ETH_MACFFR_PM** ((uint32_t)0x00000001) /* Promiscuous mode */
- #define **ETH_MACHTHR_HTH** ((uint32_t)0xFFFFFFFF) /* Hash table high */
- #define **ETH_MACHTLR_HTL** ((uint32_t)0xFFFFFFFF) /* Hash table low */
- #define **ETH_MACMIIAR_PA** ((uint32_t)0x0000F800) /* Physical layer address */
- #define **ETH_MACMIIAR_MR** ((uint32_t)0x000007C0) /* MII register in the selected PHY */
- #define **ETH_MACMIIAR_CR** ((uint32_t)0x0000001C) /* CR clock range: 6 cases */
- #define **ETH_MACMIIAR_CR_Div42** ((uint32_t)0x00000000) /* HCLK:60-100 MHz; MDC clock= HCLK/42 */
- #define **ETH_MACMIIAR_CR_Div62** ((uint32_t)0x00000004) /* HCLK:100-150 MHz; MDC clock= HCLK/62 */

- #define **ETH_MACMIIAR_CR_Div16** ((uint32_t)0x00000008) /* HCLK:20-35 MHz; MDC clock= HCLK/16 */
- #define **ETH_MACMIIAR_CR_Div26** ((uint32_t)0x0000000C) /* HCLK:35-60 MHz; MDC clock= HCLK/26 */
- #define **ETH_MACMIIAR_CR_Div102** ((uint32_t)0x00000010) /* HCLK:150-168 MHz; MDC clock= HCLK/102 */
- #define **ETH_MACMIIAR_MW** ((uint32_t)0x00000002) /* MII write */
- #define **ETH_MACMIIAR_MB** ((uint32_t)0x00000001) /* MII busy */
- #define **ETH_MACMIIDR_MD** ((uint32_t)0x0000FFFF) /* MII data: read/write data from/to PHY */
- #define **ETH_MACFCR_PT** ((uint32_t)0xFFFF0000) /* Pause time */
- #define **ETH_MACFCR_ZQPD** ((uint32_t)0x00000080) /* Zero-quanta pause disable */
- #define **ETH_MACFCR_PLT** ((uint32_t)0x00000030) /* Pause low threshold: 4 cases */
- #define **ETH_MACFCR_PLT_Minus4** ((uint32_t)0x00000000) /* Pause time minus 4 slot times */
- #define **ETH_MACFCR_PLT_Minus28** ((uint32_t)0x00000010) /* Pause time minus 28 slot times */
- #define **ETH_MACFCR_PLT_Minus144** ((uint32_t)0x00000020) /* Pause time minus 144 slot times */
- #define **ETH_MACFCR_PLT_Minus256** ((uint32_t)0x00000030) /* Pause time minus 256 slot times */
- #define **ETH_MACFCR_UPFD** ((uint32_t)0x00000008) /* Unicast pause frame detect */
- #define **ETH_MACFCR_RFCE** ((uint32_t)0x00000004) /* Receive flow control enable */
- #define **ETH_MACFCR_TFCE** ((uint32_t)0x00000002) /* Transmit flow control enable */
- #define **ETH_MACFCR_FCBBPA** ((uint32_t)0x00000001) /* Flow control busy/backpressure activate */
- #define **ETH_MACVLANTR_VLANTC** ((uint32_t)0x00010000) /* 12-bit VLAN tag comparison */
- #define **ETH_MACVLANTR_VLANTI** ((uint32_t)0x0000FFFF) /* VLAN tag identifier (for receive frames) */
- #define **ETH_MACRWUFFR_D** ((uint32_t)0xFFFFFFFF) /* Wake-up frame filter register data */
- #define **ETH_MACPMTCSR_WFFRPR** ((uint32_t)0x80000000) /* Wake-Up Frame Filter Register Pointer Reset */
- #define **ETH_MACPMTCSR_GU** ((uint32_t)0x00000200) /* Global Unicast */
- #define **ETH_MACPMTCSR_WFR** ((uint32_t)0x00000040) /* Wake-Up Frame Received */
- #define **ETH_MACPMTCSR_MPR** ((uint32_t)0x00000020) /* Magic Packet Received */
- #define **ETH_MACPMTCSR_WFE** ((uint32_t)0x00000004) /* Wake-Up Frame Enable */
- #define **ETH_MACPMTCSR_MPE** ((uint32_t)0x00000002) /* Magic Packet Enable */
- #define **ETH_MACPMTCSR_PD** ((uint32_t)0x00000001) /* Power Down */
- #define **ETH_MACSR_TSTS** ((uint32_t)0x00000200) /* Time stamp trigger status */
- #define **ETH_MACSR_MMCTS** ((uint32_t)0x00000040) /* MMC transmit status */
- #define **ETH_MACSR_MMMCRS** ((uint32_t)0x00000020) /* MMC receive status */
- #define **ETH_MACSR_MMCS** ((uint32_t)0x00000010) /* MMC status */
- #define **ETH_MACSR_PMTS** ((uint32_t)0x00000008) /* PMT status */
- #define **ETH_MACIMR_TSTIM** ((uint32_t)0x00000200) /* Time stamp trigger interrupt mask */
- #define **ETH_MACIMR_PMTIM** ((uint32_t)0x00000008) /* PMT interrupt mask */
- #define **ETH_MACA0HR_MACA0H** ((uint32_t)0x0000FFFF) /* MAC address0 high */
- #define **ETH_MACA0LR_MACA0L** ((uint32_t)0xFFFFFFFF) /* MAC address0 low */
- #define **ETH_MACA1HR_AE** ((uint32_t)0x80000000) /* Address enable */
- #define **ETH_MACA1HR_SA** ((uint32_t)0x40000000) /* Source address */
- #define **ETH_MACA1HR_MBC** ((uint32_t)0x3F000000) /* Mask byte control: bits to mask for comparison of the MAC Address bytes */
- #define **ETH_MACA1HR_MBC_HBits15_8** ((uint32_t)0x20000000) /* Mask MAC Address high reg bits [15:8] */
- #define **ETH_MACA1HR_MBC_HBits7_0** ((uint32_t)0x10000000) /* Mask MAC Address high reg bits [7:0] */
- #define **ETH_MACA1HR_MBC_LBits31_24** ((uint32_t)0x08000000) /* Mask MAC Address low reg bits [31:24] */
- #define **ETH_MACA1HR_MBC_LBits23_16** ((uint32_t)0x04000000) /* Mask MAC Address low reg bits [23:16] */
- #define **ETH_MACA1HR_MBC_LBits15_8** ((uint32_t)0x02000000) /* Mask MAC Address low reg bits [15:8] */
- #define **ETH_MACA1HR_MBC_LBits7_0** ((uint32_t)0x01000000) /* Mask MAC Address low reg bits [7:0] */

- #define **ETH_MACA1HR_MACA1H** ((uint32_t)0x0000FFFF) /* MAC address1 high */
- #define **ETH_MACA1LR_MACA1L** ((uint32_t)0xFFFFFFFF) /* MAC address1 low */
- #define **ETH_MACA2HR_AE** ((uint32_t)0x80000000) /* Address enable */
- #define **ETH_MACA2HR_SA** ((uint32_t)0x40000000) /* Source address */
- #define **ETH_MACA2HR_MBC** ((uint32_t)0x3F000000) /* Mask byte control */
- #define **ETH_MACA2HR_MBC_HBits15_8** ((uint32_t)0x20000000) /* Mask MAC Address high reg bits [15:8] */
- #define **ETH_MACA2HR_MBC_HBits7_0** ((uint32_t)0x10000000) /* Mask MAC Address high reg bits [7:0] */
- #define **ETH_MACA2HR_MBC_LBits31_24** ((uint32_t)0x08000000) /* Mask MAC Address low reg bits [31:24] */
- #define **ETH_MACA2HR_MBC_LBits23_16** ((uint32_t)0x04000000) /* Mask MAC Address low reg bits [23:16] */
- #define **ETH_MACA2HR_MBC_LBits15_8** ((uint32_t)0x02000000) /* Mask MAC Address low reg bits [15:8] */
- #define **ETH_MACA2HR_MBC_LBits7_0** ((uint32_t)0x01000000) /* Mask MAC Address low reg bits [70] */
- #define **ETH_MACA2HR_MACA2H** ((uint32_t)0x0000FFFF) /* MAC address1 high */
- #define **ETH_MACA2LR_MACA2L** ((uint32_t)0xFFFFFFFF) /* MAC address2 low */
- #define **ETH_MACA3HR_AE** ((uint32_t)0x80000000) /* Address enable */
- #define **ETH_MACA3HR_SA** ((uint32_t)0x40000000) /* Source address */
- #define **ETH_MACA3HR_MBC** ((uint32_t)0x3F000000) /* Mask byte control */
- #define **ETH_MACA3HR_MBC_HBits15_8** ((uint32_t)0x20000000) /* Mask MAC Address high reg bits [15:8] */
- #define **ETH_MACA3HR_MBC_HBits7_0** ((uint32_t)0x10000000) /* Mask MAC Address high reg bits [7:0] */
- #define **ETH_MACA3HR_MBC_LBits31_24** ((uint32_t)0x08000000) /* Mask MAC Address low reg bits [31:24] */
- #define **ETH_MACA3HR_MBC_LBits23_16** ((uint32_t)0x04000000) /* Mask MAC Address low reg bits [23:16] */
- #define **ETH_MACA3HR_MBC_LBits15_8** ((uint32_t)0x02000000) /* Mask MAC Address low reg bits [15:8] */
- #define **ETH_MACA3HR_MBC_LBits7_0** ((uint32_t)0x01000000) /* Mask MAC Address low reg bits [70] */
- #define **ETH_MACA3HR_MACA3H** ((uint32_t)0x0000FFFF) /* MAC address3 high */
- #define **ETH_MACA3LR_MACA3L** ((uint32_t)0xFFFFFFFF) /* MAC address3 low */
- #define **ETH_MMCCR_MCFHP** ((uint32_t)0x00000020) /* MMC counter Full-Half preset */
- #define **ETH_MMCCR_MCP** ((uint32_t)0x00000010) /* MMC counter preset */
- #define **ETH_MMCCR_MCF** ((uint32_t)0x00000008) /* MMC Counter Freeze */
- #define **ETH_MMCCR_ROR** ((uint32_t)0x00000004) /* Reset on Read */
- #define **ETH_MMCCR_CSR** ((uint32_t)0x00000002) /* Counter Stop Rollover */
- #define **ETH_MMCCR_CR** ((uint32_t)0x00000001) /* Counters Reset */
- #define **ETH_MMCRIR_RGUFS** ((uint32_t)0x00020000) /* Set when Rx good unicast frames counter reaches half the maximum value */
- #define **ETH_MMCRIR_RFAES** ((uint32_t)0x00000040) /* Set when Rx alignment error counter reaches half the maximum value */
- #define **ETH_MMCRIR_RFCES** ((uint32_t)0x00000020) /* Set when Rx crc error counter reaches half the maximum value */
- #define **ETH_MMCTIR_TGFS** ((uint32_t)0x00200000) /* Set when Tx good frame count counter reaches half the maximum value */
- #define **ETH_MMCTIR_TGFMSCS** ((uint32_t)0x00008000) /* Set when Tx good multi col counter reaches half the maximum value */
- #define **ETH_MMCTIR_TGFSCS** ((uint32_t)0x00004000) /* Set when Tx good single col counter reaches half the maximum value */
- #define **ETH_MMCRIMR_RGUFM** ((uint32_t)0x00020000) /* Mask the interrupt when Rx good unicast frames counter reaches half the maximum value */
- #define **ETH_MMCRIMR_RFAEM** ((uint32_t)0x00000040) /* Mask the interrupt when when Rx alignment error counter reaches half the maximum value */

- #define **ETH_MMCRIMR_RFCEM** ((uint32_t)0x00000020) /* Mask the interrupt when Rx crc error counter reaches half the maximum value */
- #define **ETH_MMCTIMR_TGFM** ((uint32_t)0x00200000) /* Mask the interrupt when Tx good frame count counter reaches half the maximum value */
- #define **ETH_MMCTIMR_TGFMSCM** ((uint32_t)0x00008000) /* Mask the interrupt when Tx good multi col counter reaches half the maximum value */
- #define **ETH_MMCTIMR_TGFSCM** ((uint32_t)0x00004000) /* Mask the interrupt when Tx good single col counter reaches half the maximum value */
- #define **ETH_MMCTGFSCCR_TGFSCC** ((uint32_t)0xFFFFFFFF) /* Number of successfully transmitted frames after a single collision in Half-duplex mode. */
- #define **ETH_MMCTGFMSCCR_TGFMSCC** ((uint32_t)0xFFFFFFFF) /* Number of successfully transmitted frames after more than a single collision in Half-duplex mode. */
- #define **ETH_MMCTGFCR_TGFC** ((uint32_t)0xFFFFFFFF) /* Number of good frames transmitted. */
- #define **ETH_MMCRFCECR_RFCEC** ((uint32_t)0xFFFFFFFF) /* Number of frames received with CRC error. */
- #define **ETH_MMCRFAECR_RFAEC** ((uint32_t)0xFFFFFFFF) /* Number of frames received with alignment (dribble) error */
- #define **ETH_MMCRGUFCR_RGUFC** ((uint32_t)0xFFFFFFFF) /* Number of good unicast frames received. */
- #define **ETH_PTPTSCR_TSCNT** ((uint32_t)0x00030000) /* Time stamp clock node type */
- #define **ETH_PTPTSSR_TSSMRME** ((uint32_t)0x00008000) /* Time stamp snapshot for message relevant to master enable */
- #define **ETH_PTPTSSR_TSSEME** ((uint32_t)0x00004000) /* Time stamp snapshot for event message enable */
- #define **ETH_PTPTSSR_TSSIPV4FE** ((uint32_t)0x00002000) /* Time stamp snapshot for IPv4 frames enable */
- #define **ETH_PTPTSSR_TSSIPV6FE** ((uint32_t)0x00001000) /* Time stamp snapshot for IPv6 frames enable */
- #define **ETH_PTPTSSR_TSSPTPOEFE** ((uint32_t)0x00000800) /* Time stamp snapshot for PTP over ethernet frames enable */
- #define **ETH_PTPTSSR_TSPTPPSV2E** ((uint32_t)0x00000400) /* Time stamp PTP packet snooping for version2 format enable */
- #define **ETH_PTPTSSR_TSSSR** ((uint32_t)0x00000200) /* Time stamp Sub-seconds rollover */
- #define **ETH_PTPTSSR_TSSARFE** ((uint32_t)0x00000100) /* Time stamp snapshot for all received frames enable */
- #define **ETH_PTPTSCR_TSARU** ((uint32_t)0x00000020) /* Addend register update */
- #define **ETH_PTPTSCR_TSITE** ((uint32_t)0x00000010) /* Time stamp interrupt trigger enable */
- #define **ETH_PTPTSCR_TSSTU** ((uint32_t)0x00000008) /* Time stamp update */
- #define **ETH_PTPTSCR_TSSTI** ((uint32_t)0x00000004) /* Time stamp initialize */
- #define **ETH_PTPTSCR_TSFCU** ((uint32_t)0x00000002) /* Time stamp fine or coarse update */
- #define **ETH_PTPTSCR_TSE** ((uint32_t)0x00000001) /* Time stamp enable */
- #define **ETH_PTPSSIR_STSSI** ((uint32_t)0x000000FF) /* System time Sub-second increment value */
- #define **ETH_PTPTSHR_STS** ((uint32_t)0xFFFFFFFF) /* System Time second */
- #define **ETH_PTPTSLR_STPNS** ((uint32_t)0x80000000) /* System Time Positive or negative time */
- #define **ETH_PTPTSLR_STSS** ((uint32_t)0x7FFFFFFF) /* System Time sub-seconds */
- #define **ETH_PTPTSHUR_TSUS** ((uint32_t)0xFFFFFFFF) /* Time stamp update seconds */
- #define **ETH_PTPTSLUR_TSUPNS** ((uint32_t)0x80000000) /* Time stamp update Positive or negative time */
- #define **ETH_PTPTSLUR_TSUSS** ((uint32_t)0x7FFFFFFF) /* Time stamp update sub-seconds */
- #define **ETH_PTPTSAR_TSA** ((uint32_t)0xFFFFFFFF) /* Time stamp addend */
- #define **ETH_PTPTTHR_TTSH** ((uint32_t)0xFFFFFFFF) /* Target time stamp high */
- #define **ETH_PTPTTLR_TTSL** ((uint32_t)0xFFFFFFFF) /* Target time stamp low */
- #define **ETH_PTPTSSR_TSTTR** ((uint32_t)0x00000020) /* Time stamp target time reached */
- #define **ETH_PTPTSSR_TSSO** ((uint32_t)0x00000010) /* Time stamp seconds overflow */
- #define **ETH_DMABMR_AAB** ((uint32_t)0x02000000) /* Address-Aligned beats */
- #define **ETH_DMABMR_FPM** ((uint32_t)0x01000000) /* 4xPBL mode */

- #define **ETH_DMABMR_USP** ((uint32_t)0x00800000) /* Use separate PBL */
- #define **ETH_DMABMR_RDP** ((uint32_t)0x007E0000) /* RxDMA PBL */
- #define **ETH_DMABMR_RDP_1Beat** ((uint32_t)0x00020000) /* maximum number of beats to be transferred in one RxDMA transaction is 1 */
- #define **ETH_DMABMR_RDP_2Beat** ((uint32_t)0x00040000) /* maximum number of beats to be transferred in one RxDMA transaction is 2 */
- #define **ETH_DMABMR_RDP_4Beat** ((uint32_t)0x00080000) /* maximum number of beats to be transferred in one RxDMA transaction is 4 */
- #define **ETH_DMABMR_RDP_8Beat** ((uint32_t)0x00100000) /* maximum number of beats to be transferred in one RxDMA transaction is 8 */
- #define **ETH_DMABMR_RDP_16Beat** ((uint32_t)0x00200000) /* maximum number of beats to be transferred in one RxDMA transaction is 16 */
- #define **ETH_DMABMR_RDP_32Beat** ((uint32_t)0x00400000) /* maximum number of beats to be transferred in one RxDMA transaction is 32 */
- #define **ETH_DMABMR_RDP_4xPBL_4Beat** ((uint32_t)0x01020000) /* maximum number of beats to be transferred in one RxDMA transaction is 4 */
- #define **ETH_DMABMR_RDP_4xPBL_8Beat** ((uint32_t)0x01040000) /* maximum number of beats to be transferred in one RxDMA transaction is 8 */
- #define **ETH_DMABMR_RDP_4xPBL_16Beat** ((uint32_t)0x01080000) /* maximum number of beats to be transferred in one RxDMA transaction is 16 */
- #define **ETH_DMABMR_RDP_4xPBL_32Beat** ((uint32_t)0x01100000) /* maximum number of beats to be transferred in one RxDMA transaction is 32 */
- #define **ETH_DMABMR_RDP_4xPBL_64Beat** ((uint32_t)0x01200000) /* maximum number of beats to be transferred in one RxDMA transaction is 64 */
- #define **ETH_DMABMR_RDP_4xPBL_128Beat** ((uint32_t)0x01400000) /* maximum number of beats to be transferred in one RxDMA transaction is 128 */
- #define **ETH_DMABMR_FB** ((uint32_t)0x00010000) /* Fixed Burst */
- #define **ETH_DMABMR_RTPR** ((uint32_t)0x0000C000) /* Rx Tx priority ratio */
- #define **ETH_DMABMR_RTPR_1_1** ((uint32_t)0x00000000) /* Rx Tx priority ratio */
- #define **ETH_DMABMR_RTPR_2_1** ((uint32_t)0x00004000) /* Rx Tx priority ratio */
- #define **ETH_DMABMR_RTPR_3_1** ((uint32_t)0x00008000) /* Rx Tx priority ratio */
- #define **ETH_DMABMR_RTPR_4_1** ((uint32_t)0x0000C000) /* Rx Tx priority ratio */
- #define **ETH_DMABMR_PBL** ((uint32_t)0x00003F00) /* Programmable burst length */
- #define **ETH_DMABMR_PBL_1Beat** ((uint32_t)0x00000100) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 1 */
- #define **ETH_DMABMR_PBL_2Beat** ((uint32_t)0x00000200) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 2 */
- #define **ETH_DMABMR_PBL_4Beat** ((uint32_t)0x00000400) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 4 */
- #define **ETH_DMABMR_PBL_8Beat** ((uint32_t)0x00000800) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 8 */
- #define **ETH_DMABMR_PBL_16Beat** ((uint32_t)0x00001000) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 16 */
- #define **ETH_DMABMR_PBL_32Beat** ((uint32_t)0x00002000) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 32 */
- #define **ETH_DMABMR_PBL_4xPBL_4Beat** ((uint32_t)0x01000100) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 4 */
- #define **ETH_DMABMR_PBL_4xPBL_8Beat** ((uint32_t)0x01000200) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 8 */
- #define **ETH_DMABMR_PBL_4xPBL_16Beat** ((uint32_t)0x01000400) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 16 */
- #define **ETH_DMABMR_PBL_4xPBL_32Beat** ((uint32_t)0x01000800) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 32 */
- #define **ETH_DMABMR_PBL_4xPBL_64Beat** ((uint32_t)0x01001000) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 64 */

- #define **ETH_DMABMR_PBL_4xPBL_128Beat** ((uint32_t)0x01002000) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 128 */
- #define **ETH_DMABMR_EDE** ((uint32_t)0x00000080) /* Enhanced Descriptor Enable */
- #define **ETH_DMABMR_DSL** ((uint32_t)0x0000007C) /* Descriptor Skip Length */
- #define **ETH_DMABMR_DA** ((uint32_t)0x00000002) /* DMA arbitration scheme */
- #define **ETH_DMABMR_SR** ((uint32_t)0x00000001) /* Software reset */
- #define **ETH_DMATPDR_TPD** ((uint32_t)0xFFFFFFFF) /* Transmit poll demand */
- #define **ETH_DMARPDR_RPD** ((uint32_t)0xFFFFFFFF) /* Receive poll demand */
- #define **ETH_DMARDLAR_SRL** ((uint32_t)0xFFFFFFFF) /* Start of receive list */
- #define **ETH_DMATDLAR_STL** ((uint32_t)0xFFFFFFFF) /* Start of transmit list */
- #define **ETH_DMASR_TSTS** ((uint32_t)0x20000000) /* Time-stamp trigger status */
- #define **ETH_DMASR_PMTS** ((uint32_t)0x10000000) /* PMT status */
- #define **ETH_DMASR_MMCS** ((uint32_t)0x08000000) /* MMC status */
- #define **ETH_DMASR_EBS** ((uint32_t)0x03800000) /* Error bits status */
- #define **ETH_DMASR_EBS_DescAccess** ((uint32_t)0x02000000) /* Error bits 0-data buffer, 1-desc. access */
- #define **ETH_DMASR_EBS_ReadTransf** ((uint32_t)0x01000000) /* Error bits 0-write trnsf, 1-read transfr */
- #define **ETH_DMASR_EBS_DataTransfTx** ((uint32_t)0x00800000) /* Error bits 0-Rx DMA, 1-Tx DMA */
- #define **ETH_DMASR_TPS** ((uint32_t)0x00700000) /* Transmit process state */
- #define **ETH_DMASR_TPS_Stopped** ((uint32_t)0x00000000) /* Stopped - Reset or Stop Tx Command issued */
- #define **ETH_DMASR_TPS_Fetching** ((uint32_t)0x00100000) /* Running - fetching the Tx descriptor */
- #define **ETH_DMASR_TPS_Waiting** ((uint32_t)0x00200000) /* Running - waiting for status */
- #define **ETH_DMASR_TPS_Reading** ((uint32_t)0x00300000) /* Running - reading the data from host memory */
- #define **ETH_DMASR_TPS_Suspended** ((uint32_t)0x00600000) /* Suspended - Tx Descriptor unavailabe */
- #define **ETH_DMASR_TPS_Closing** ((uint32_t)0x00700000) /* Running - closing Rx descriptor */
- #define **ETH_DMASR_RPS** ((uint32_t)0x000E0000) /* Receive process state */
- #define **ETH_DMASR_RPS_Stopped** ((uint32_t)0x00000000) /* Stopped - Reset or Stop Rx Command issued */
- #define **ETH_DMASR_RPS_Fetching** ((uint32_t)0x00020000) /* Running - fetching the Rx descriptor */
- #define **ETH_DMASR_RPS_Waiting** ((uint32_t)0x00060000) /* Running - waiting for packet */
- #define **ETH_DMASR_RPS_Suspended** ((uint32_t)0x00080000) /* Suspended - Rx Descriptor unavailable */
- #define **ETH_DMASR_RPS_Closing** ((uint32_t)0x000A0000) /* Running - closing descriptor */
- #define **ETH_DMASR_RPS_Queuing** ((uint32_t)0x000E0000) /* Running - queuing the recieve frame into host memory */
- #define **ETH_DMASR_NIS** ((uint32_t)0x00010000) /* Normal interrupt summary */
- #define **ETH_DMASR_AIS** ((uint32_t)0x00008000) /* Abnormal interrupt summary */
- #define **ETH_DMASR_ERS** ((uint32_t)0x00004000) /* Early receive status */
- #define **ETH_DMASR_FBES** ((uint32_t)0x00002000) /* Fatal bus error status */
- #define **ETH_DMASR_ETS** ((uint32_t)0x00000400) /* Early transmit status */
- #define **ETH_DMASR_RWTS** ((uint32_t)0x00000200) /* Receive watchdog timeout status */
- #define **ETH_DMASR_RPSS** ((uint32_t)0x00000100) /* Receive process stopped status */
- #define **ETH_DMASR_RBUS** ((uint32_t)0x00000080) /* Receive buffer unavailable status */
- #define **ETH_DMASR_RS** ((uint32_t)0x00000040) /* Receive status */
- #define **ETH_DMASR_TUS** ((uint32_t)0x00000020) /* Transmit underflow status */
- #define **ETH_DMASR_ROS** ((uint32_t)0x00000010) /* Receive overflow status */
- #define **ETH_DMASR_TJTS** ((uint32_t)0x00000008) /* Transmit jabber timeout status */
- #define **ETH_DMASR_TBUS** ((uint32_t)0x00000004) /* Transmit buffer unavailable status */
- #define **ETH_DMASR_TPSS** ((uint32_t)0x00000002) /* Transmit process stopped status */
- #define **ETH_DMASR_TS** ((uint32_t)0x00000001) /* Transmit status */
- #define **ETH_DMAOMR_DTCEFD** ((uint32_t)0x04000000) /* Disable Dropping of TCP/IP checksum error frames */

- #define **ETH_DMAOMR_RSF** ((uint32_t)0x02000000) /* Receive store and forward */
- #define **ETH_DMAOMR_DFRF** ((uint32_t)0x01000000) /* Disable flushing of received frames */
- #define **ETH_DMAOMR_TSF** ((uint32_t)0x00200000) /* Transmit store and forward */
- #define **ETH_DMAOMR_FTF** ((uint32_t)0x00100000) /* Flush transmit FIFO */
- #define **ETH_DMAOMR_TTC** ((uint32_t)0x0001C000) /* Transmit threshold control */
- #define **ETH_DMAOMR_TTC_64Bytes** ((uint32_t)0x00000000) /* threshold level of the MTL Transmit FIFO is 64 Bytes */
- #define **ETH_DMAOMR_TTC_128Bytes** ((uint32_t)0x00004000) /* threshold level of the MTL Transmit FIFO is 128 Bytes */
- #define **ETH_DMAOMR_TTC_192Bytes** ((uint32_t)0x00008000) /* threshold level of the MTL Transmit FIFO is 192 Bytes */
- #define **ETH_DMAOMR_TTC_256Bytes** ((uint32_t)0x0000C000) /* threshold level of the MTL Transmit FIFO is 256 Bytes */
- #define **ETH_DMAOMR_TTC_40Bytes** ((uint32_t)0x00010000) /* threshold level of the MTL Transmit FIFO is 40 Bytes */
- #define **ETH_DMAOMR_TTC_32Bytes** ((uint32_t)0x00014000) /* threshold level of the MTL Transmit FIFO is 32 Bytes */
- #define **ETH_DMAOMR_TTC_24Bytes** ((uint32_t)0x00018000) /* threshold level of the MTL Transmit FIFO is 24 Bytes */
- #define **ETH_DMAOMR_TTC_16Bytes** ((uint32_t)0x0001C000) /* threshold level of the MTL Transmit FIFO is 16 Bytes */
- #define **ETH_DMAOMR_ST** ((uint32_t)0x00002000) /* Start/stop transmission command */
- #define **ETH_DMAOMR_FEF** ((uint32_t)0x00000080) /* Forward error frames */
- #define **ETH_DMAOMR_FUGF** ((uint32_t)0x00000040) /* Forward undersized good frames */
- #define **ETH_DMAOMR_RTC** ((uint32_t)0x00000018) /* receive threshold control */
- #define **ETH_DMAOMR_RTC_64Bytes** ((uint32_t)0x00000000) /* threshold level of the MTL Receive FIFO is 64 Bytes */
- #define **ETH_DMAOMR_RTC_32Bytes** ((uint32_t)0x00000008) /* threshold level of the MTL Receive FIFO is 32 Bytes */
- #define **ETH_DMAOMR_RTC_96Bytes** ((uint32_t)0x00000010) /* threshold level of the MTL Receive FIFO is 96 Bytes */
- #define **ETH_DMAOMR_RTC_128Bytes** ((uint32_t)0x00000018) /* threshold level of the MTL Receive FIFO is 128 Bytes */
- #define **ETH_DMAOMR_OSF** ((uint32_t)0x00000004) /* operate on second frame */
- #define **ETH_DMAOMR_SR** ((uint32_t)0x00000002) /* Start/stop receive */
- #define **ETH_DMAIER_NISE** ((uint32_t)0x00010000) /* Normal interrupt summary enable */
- #define **ETH_DMAIER_AISE** ((uint32_t)0x00008000) /* Abnormal interrupt summary enable */
- #define **ETH_DMAIER_ERIE** ((uint32_t)0x00004000) /* Early receive interrupt enable */
- #define **ETH_DMAIER_FBEIE** ((uint32_t)0x00002000) /* Fatal bus error interrupt enable */
- #define **ETH_DMAIER_ETIE** ((uint32_t)0x00000400) /* Early transmit interrupt enable */
- #define **ETH_DMAIER_RWTIE** ((uint32_t)0x00000200) /* Receive watchdog timeout interrupt enable */
- #define **ETH_DMAIER_RPSIE** ((uint32_t)0x00000100) /* Receive process stopped interrupt enable */
- #define **ETH_DMAIER_RBUIE** ((uint32_t)0x00000080) /* Receive buffer unavailable interrupt enable */
- #define **ETH_DMAIER_RIE** ((uint32_t)0x00000040) /* Receive interrupt enable */
- #define **ETH_DMAIER_TUIE** ((uint32_t)0x00000020) /* Transmit Underflow interrupt enable */
- #define **ETH_DMAIER_ROIE** ((uint32_t)0x00000010) /* Receive Overflow interrupt enable */
- #define **ETH_DMAIER_TJTIE** ((uint32_t)0x00000008) /* Transmit jabber timeout interrupt enable */
- #define **ETH_DMAIER_TBUIE** ((uint32_t)0x00000004) /* Transmit buffer unavailable interrupt enable */
- #define **ETH_DMAIER_TPSIE** ((uint32_t)0x00000002) /* Transmit process stopped interrupt enable */
- #define **ETH_DMAIER_TIE** ((uint32_t)0x00000001) /* Transmit interrupt enable */
- #define **ETH_DMAMFBOCR_OFOC** ((uint32_t)0x10000000) /* Overflow bit for FIFO overflow counter */
- #define **ETH_DMAMFBOCR_MFA** ((uint32_t)0x0FFE0000) /* Number of frames missed by the application */
- #define **ETH_DMAMFBOCR_OMFC** ((uint32_t)0x00010000) /* Overflow bit for missed frame counter */
- #define **ETH_DMAMFBOCR_MFC** ((uint32_t)0x0000FFFF) /* Number of frames missed by the controller */

- #define **ETH_DMACHTDR_HTDAP** ((uint32_t)0xFFFFFFFF) /* Host transmit descriptor address pointer */
- #define **ETH_DMACHRDR_HRDAP** ((uint32_t)0xFFFFFFFF) /* Host receive descriptor address pointer */
- #define **ETH_DMACHTBAR_HTBAP** ((uint32_t)0xFFFFFFFF) /* Host transmit buffer address pointer */
- #define **ETH_DMACHRBAR_HRBAP** ((uint32_t)0xFFFFFFFF) /* Host receive buffer address pointer */

5.154.1 Detailed Description

5.154.2 Macro Definition Documentation

5.154.2.1 ADC_CCR_ADCPRE

```
#define ADC_CCR_ADCPRE ((uint32_t)0x00030000)
```

ADCPRE[1:0] bits (ADC prescaler)

5.154.2.2 ADC_CCR_ADCPRE_0

```
#define ADC_CCR_ADCPRE_0 ((uint32_t)0x00010000)
```

Bit 0

5.154.2.3 ADC_CCR_ADCPRE_1

```
#define ADC_CCR_ADCPRE_1 ((uint32_t)0x00020000)
```

Bit 1

5.154.2.4 ADC_CCR_DDS

```
#define ADC_CCR_DDS ((uint32_t)0x00002000)
```

DMA disable selection (Multi-ADC mode)

5.154.2.5 ADC_CCR_DELAY

```
#define ADC_CCR_DELAY ((uint32_t)0x00000F00)
```

DELAY[3:0] bits (Delay between 2 sampling phases)

5.154.2.6 ADC_CCR_DELAY_0

```
#define ADC_CCR_DELAY_0 ((uint32_t)0x00000100)
```

Bit 0

5.154.2.7 ADC_CCR_DELAY_1

```
#define ADC_CCR_DELAY_1 ((uint32_t)0x00000200)
```

Bit 1

5.154.2.8 ADC_CCR_DELAY_2

```
#define ADC_CCR_DELAY_2 ((uint32_t)0x00000400)
```

Bit 2

5.154.2.9 ADC_CCR_DELAY_3

```
#define ADC_CCR_DELAY_3 ((uint32_t)0x00000800)
```

Bit 3

5.154.2.10 ADC_CCR_DMA

```
#define ADC_CCR_DMA ((uint32_t)0x0000C000)
```

DMA[1:0] bits (Direct Memory Access mode for multimode)

5.154.2.11 ADC_CCR_DMA_0

```
#define ADC_CCR_DMA_0 ((uint32_t)0x00004000)
```

Bit 0

5.154.2.12 ADC_CCR_DMA_1

```
#define ADC_CCR_DMA_1 ((uint32_t)0x00008000)
```

Bit 1

5.154.2.13 ADC_CCR_MULTI

```
#define ADC_CCR_MULTI ((uint32_t)0x0000001F)
```

MULTI[4:0] bits (Multi-ADC mode selection)

5.154.2.14 ADC_CCR_MULTI_0

```
#define ADC_CCR_MULTI_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.15 ADC_CCR_MULTI_1

```
#define ADC_CCR_MULTI_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.16 ADC_CCR_MULTI_2

```
#define ADC_CCR_MULTI_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.17 ADC_CCR_MULTI_3

```
#define ADC_CCR_MULTI_3 ((uint32_t)0x00000008)
```

Bit 3

5.154.2.18 ADC_CCR_MULTI_4

```
#define ADC_CCR_MULTI_4 ((uint32_t)0x00000010)
```

Bit 4

5.154.2.19 ADC_CCR_TSVREFE

```
#define ADC_CCR_TSVREFE ((uint32_t)0x00800000)
```

Temperature Sensor and VREFINT Enable

5.154.2.20 ADC_CCR_VBATE

```
#define ADC_CCR_VBATE ((uint32_t)0x00400000)
```

VBAT Enable

5.154.2.21 ADC_CDR_DATA1

```
#define ADC_CDR_DATA1 ((uint32_t)0x0000FFFF)
```

1st data of a pair of regular conversions

5.154.2.22 ADC_CDR_DATA2

```
#define ADC_CDR_DATA2 ((uint32_t)0xFFFF0000)
```

2nd data of a pair of regular conversions

5.154.2.23 ADC_CR1_AWDCH

```
#define ADC_CR1_AWDCH ((uint32_t)0x0000001F)
```

AWDCH[4:0] bits (Analog watchdog channel select bits)

5.154.2.24 ADC_CR1_AWDCH_0

```
#define ADC_CR1_AWDCH_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.25 ADC_CR1_AWDCH_1

```
#define ADC_CR1_AWDCH_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.26 ADC_CR1_AWDCH_2

```
#define ADC_CR1_AWDCH_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.27 ADC_CR1_AWDCH_3

```
#define ADC_CR1_AWDCH_3 ((uint32_t)0x00000008)
```

Bit 3

5.154.2.28 ADC_CR1_AWDCH_4

```
#define ADC_CR1_AWDCH_4 ((uint32_t)0x00000010)
```

Bit 4

5.154.2.29 ADC_CR1_AWDEN

```
#define ADC_CR1_AWDEN ((uint32_t)0x00800000)
```

Analog watchdog enable on regular channels

5.154.2.30 ADC_CR1_AWDIE

```
#define ADC_CR1_AWDIE ((uint32_t)0x00000040)
```

AAnalog Watchdog interrupt enable

5.154.2.31 ADC_CR1_AWDSGL

```
#define ADC_CR1_AWDSGL ((uint32_t)0x00000200)
```

Enable the watchdog on a single channel in scan mode

5.154.2.32 ADC_CR1_DISCEN

```
#define ADC_CR1_DISCEN ((uint32_t)0x00000800)
```

Discontinuous mode on regular channels

5.154.2.33 ADC_CR1_DISCNUM

```
#define ADC_CR1_DISCNUM ((uint32_t)0x0000E000)
```

DISCNUM[2:0] bits (Discontinuous mode channel count)

5.154.2.34 ADC_CR1_DISCNUM_0

```
#define ADC_CR1_DISCNUM_0 ((uint32_t)0x00002000)
```

Bit 0

5.154.2.35 ADC_CR1_DISCNUM_1

```
#define ADC_CR1_DISCNUM_1 ((uint32_t)0x00004000)
```

Bit 1

5.154.2.36 ADC_CR1_DISCNUM_2

```
#define ADC_CR1_DISCNUM_2 ((uint32_t)0x00008000)
```

Bit 2

5.154.2.37 ADC_CR1_EOCIE

```
#define ADC_CR1_EOCIE ((uint32_t)0x00000020)
```

Interrupt enable for EOC

5.154.2.38 ADC_CR1_JAUTO

```
#define ADC_CR1_JAUTO ((uint32_t)0x00000400)
```

Automatic injected group conversion

5.154.2.39 ADC_CR1_JAWDEN

```
#define ADC_CR1_JAWDEN ((uint32_t)0x00400000)
```

Analog watchdog enable on injected channels

5.154.2.40 ADC_CR1_JDISCEN

```
#define ADC_CR1_JDISCEN ((uint32_t)0x00001000)
```

Discontinuous mode on injected channels

5.154.2.41 ADC_CR1_JEOCIE

```
#define ADC_CR1_JEOCIE ((uint32_t)0x00000080)
```

Interrupt enable for injected channels

5.154.2.42 ADC_CR1_OVRIE

```
#define ADC_CR1_OVRIE ((uint32_t)0x04000000)
```

overrun interrupt enable

5.154.2.43 ADC_CR1_RES

```
#define ADC_CR1_RES ((uint32_t)0x03000000)
```

RES[2:0] bits (Resolution)

5.154.2.44 ADC_CR1_RES_0

```
#define ADC_CR1_RES_0 ((uint32_t)0x01000000)
```

Bit 0

5.154.2.45 ADC_CR1_RES_1

```
#define ADC_CR1_RES_1 ((uint32_t)0x02000000)
```

Bit 1

5.154.2.46 ADC_CR1_SCAN

```
#define ADC_CR1_SCAN ((uint32_t)0x00000100)
```

Scan mode

5.154.2.47 ADC_CR2_ADON

```
#define ADC_CR2_ADON ((uint32_t)0x00000001)
```

A/D Converter ON / OFF

5.154.2.48 ADC_CR2_ALIGN

```
#define ADC_CR2_ALIGN ((uint32_t)0x00000800)
```

Data Alignment

5.154.2.49 ADC_CR2_CONT

```
#define ADC_CR2_CONT ((uint32_t)0x00000002)
```

Continuous Conversion

5.154.2.50 ADC_CR2_DDS

```
#define ADC_CR2_DDS ((uint32_t)0x00000200)
```

DMA disable selection (Single ADC)

5.154.2.51 ADC_CR2_DMA

```
#define ADC_CR2_DMA ((uint32_t)0x00000100)
```

Direct Memory access mode

5.154.2.52 ADC_CR2_EOCS

```
#define ADC_CR2_EOCS ((uint32_t)0x00000400)
```

End of conversion selection

5.154.2.53 ADC_CR2_EXTEN

```
#define ADC_CR2_EXTEN ((uint32_t)0x30000000)
```

EXTEN[1:0] bits (External Trigger Conversion mode for regular channelsp)

5.154.2.54 ADC_CR2_EXTEN_0

```
#define ADC_CR2_EXTEN_0 ((uint32_t)0x10000000)
```

Bit 0

5.154.2.55 ADC_CR2_EXTEN_1

```
#define ADC_CR2_EXTEN_1 ((uint32_t)0x20000000)
```

Bit 1

5.154.2.56 ADC_CR2_EXTSEL

```
#define ADC_CR2_EXTSEL ((uint32_t)0x0F000000)
```

EXTSEL[3:0] bits (External Event Select for regular group)

5.154.2.57 ADC_CR2_EXTSEL_0

```
#define ADC_CR2_EXTSEL_0 ((uint32_t)0x01000000)
```

Bit 0

5.154.2.58 ADC_CR2_EXTSEL_1

```
#define ADC_CR2_EXTSEL_1 ((uint32_t)0x02000000)
```

Bit 1

5.154.2.59 ADC_CR2_EXTSEL_2

```
#define ADC_CR2_EXTSEL_2 ((uint32_t)0x04000000)
```

Bit 2

5.154.2.60 ADC_CR2_EXTSEL_3

```
#define ADC_CR2_EXTSEL_3 ((uint32_t)0x08000000)
```

Bit 3

5.154.2.61 ADC_CR2_JEXTEN

```
#define ADC_CR2_JEXTEN ((uint32_t)0x00300000)
```

JEXTEN[1:0] bits (External Trigger Conversion mode for injected channelsp)

5.154.2.62 ADC_CR2_JEXTEN_0

```
#define ADC_CR2_JEXTEN_0 ((uint32_t)0x00100000)
```

Bit 0

5.154.2.63 ADC_CR2_JEXTEN_1

```
#define ADC_CR2_JEXTEN_1 ((uint32_t)0x00200000)
```

Bit 1

5.154.2.64 ADC_CR2_JEXTSEL

```
#define ADC_CR2_JEXTSEL ((uint32_t)0x000F0000)
```

JEXTSEL[3:0] bits (External event select for injected group)

5.154.2.65 ADC_CR2_JEXTSEL_0

```
#define ADC_CR2_JEXTSEL_0 ((uint32_t)0x00010000)
```

Bit 0

5.154.2.66 ADC_CR2_JEXTSEL_1

```
#define ADC_CR2_JEXTSEL_1 ((uint32_t)0x00020000)
```

Bit 1

5.154.2.67 ADC_CR2_JEXTSEL_2

```
#define ADC_CR2_JEXTSEL_2 ((uint32_t)0x00040000)
```

Bit 2

5.154.2.68 ADC_CR2_JEXTSEL_3

```
#define ADC_CR2_JEXTSEL_3 ((uint32_t)0x00080000)
```

Bit 3

5.154.2.69 ADC_CR2_JSWSTART

```
#define ADC_CR2_JSWSTART ((uint32_t)0x00400000)
```

Start Conversion of injected channels

5.154.2.70 ADC_CR2_SWSTART

```
#define ADC_CR2_SWSTART ((uint32_t)0x40000000)
```

Start Conversion of regular channels

5.154.2.71 ADC_CSR_AWD1

```
#define ADC_CSR_AWD1 ((uint32_t)0x00000001)
```

ADC1 Analog watchdog flag

5.154.2.72 ADC_CSR_AWD2

```
#define ADC_CSR_AWD2 ((uint32_t)0x00000100)
```

ADC2 Analog watchdog flag

5.154.2.73 ADC_CSR_AWD3

```
#define ADC_CSR_AWD3 ((uint32_t)0x00010000)
```

ADC3 Analog watchdog flag

5.154.2.74 ADC_CSR_DOVR1

```
#define ADC_CSR_DOVR1 ((uint32_t)0x00000020)
```

ADC1 DMA overrun flag

5.154.2.75 ADC_CSR_DOVR2

```
#define ADC_CSR_DOVR2 ((uint32_t)0x00002000)
```

ADC2 DMA overrun flag

5.154.2.76 ADC_CSR_DOVR3

```
#define ADC_CSR_DOVR3 ((uint32_t)0x00200000)
```

ADC3 DMA overrun flag

5.154.2.77 ADC_CSR_EOC1

```
#define ADC_CSR_EOC1 ((uint32_t)0x00000002)
```

ADC1 End of conversion

5.154.2.78 ADC_CSR_EOC2

```
#define ADC_CSR_EOC2 ((uint32_t)0x00000200)
```

ADC2 End of conversion

5.154.2.79 ADC_CSR_EOC3

```
#define ADC_CSR_EOC3 ((uint32_t)0x00020000)
```

ADC3 End of conversion

5.154.2.80 ADC_CSR_JEOC1

```
#define ADC_CSR_JEOC1 ((uint32_t)0x00000004)
```

ADC1 Injected channel end of conversion

5.154.2.81 ADC_CSR_JEOC2

```
#define ADC_CSR_JEOC2 ((uint32_t)0x00000400)
```

ADC2 Injected channel end of conversion

5.154.2.82 ADC_CSR_JEOC3

```
#define ADC_CSR_JEOC3 ((uint32_t)0x00040000)
```

ADC3 Injected channel end of conversion

5.154.2.83 ADC_CSR_JSTRT1

```
#define ADC_CSR_JSTRT1 ((uint32_t)0x00000008)
```

ADC1 Injected channel Start flag

5.154.2.84 ADC_CSR_JSTRT2

```
#define ADC_CSR_JSTRT2 ((uint32_t)0x00000800)
```

ADC2 Injected channel Start flag

5.154.2.85 ADC_CSR_JSTRT3

```
#define ADC_CSR_JSTRT3 ((uint32_t)0x00080000)
```

ADC3 Injected channel Start flag

5.154.2.86 ADC_CSR_STRT1

```
#define ADC_CSR_STRT1 ((uint32_t)0x00000010)
```

ADC1 Regular channel Start flag

5.154.2.87 ADC_CSR_STRT2

```
#define ADC_CSR_STRT2 ((uint32_t)0x00001000)
```

ADC2 Regular channel Start flag

5.154.2.88 ADC_CSR_STRT3

```
#define ADC_CSR_STRT3 ((uint32_t)0x00100000)
```

ADC3 Regular channel Start flag

5.154.2.89 ADC_DR_ADC2DATA

```
#define ADC_DR_ADC2DATA ((uint32_t)0xFFFF0000)
```

ADC2 data

5.154.2.90 ADC_DR_DATA

```
#define ADC_DR_DATA ((uint32_t)0x0000FFFF)
```

Regular data

5.154.2.91 ADC_HTR_HT

```
#define ADC_HTR_HT ((uint16_t)0x0FFF)
```

Analog watchdog high threshold

5.154.2.92 ADC_JDR1_JDATA

```
#define ADC_JDR1_JDATA ((uint16_t)0xFFFF)
```

Injected data

5.154.2.93 ADC_JDR2_JDATA

```
#define ADC_JDR2_JDATA ((uint16_t)0xFFFF)
```

Injected data

5.154.2.94 ADC_JDR3_JDATA

```
#define ADC_JDR3_JDATA ((uint16_t)0xFFFF)
```

Injected data

5.154.2.95 ADC_JDR4_JDATA

```
#define ADC_JDR4_JDATA ((uint16_t)0xFFFF)
```

Injected data

5.154.2.96 ADC_JOFR1_JOFFSET1

```
#define ADC_JOFR1_JOFFSET1 ((uint16_t)0x0FFF)
```

Data offset for injected channel 1

5.154.2.97 ADC_JOFR2_JOFFSET2

```
#define ADC_JOFR2_JOFFSET2 ((uint16_t)0x0FFF)
```

Data offset for injected channel 2

5.154.2.98 ADC_JOFR3_JOFFSET3

```
#define ADC_JOFR3_JOFFSET3 ((uint16_t)0x0FFF)
```

Data offset for injected channel 3

5.154.2.99 ADC_JOFR4_JOFFSET4

```
#define ADC_JOFR4_JOFFSET4 ((uint16_t)0x0FFF)
```

Data offset for injected channel 4

5.154.2.100 ADC_JSQR_JL

```
#define ADC_JSQR_JL ((uint32_t)0x00300000)
```

JL[1:0] bits (Injected Sequence length)

5.154.2.101 ADC_JSQR_JL_0

```
#define ADC_JSQR_JL_0 ((uint32_t)0x00100000)
```

Bit 0

5.154.2.102 ADC_JSQR_JL_1

```
#define ADC_JSQR_JL_1 ((uint32_t)0x00200000)
```

Bit 1

5.154.2.103 ADC_JSQR_JSQ1

```
#define ADC_JSQR_JSQ1 ((uint32_t)0x0000001F)
```

JSQ1[4:0] bits (1st conversion in injected sequence)

5.154.2.104 ADC_JSQR_JSQ1_0

```
#define ADC_JSQR_JSQ1_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.105 ADC_JSQR_JSQ1_1

```
#define ADC_JSQR_JSQ1_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.106 ADC_JSQR_JSQ1_2

```
#define ADC_JSQR_JSQ1_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.107 ADC_JSQR_JSQ1_3

```
#define ADC_JSQR_JSQ1_3 ((uint32_t)0x00000008)
```

Bit 3

5.154.2.108 ADC_JSQR_JSQ1_4

```
#define ADC_JSQR_JSQ1_4 ((uint32_t)0x00000010)
```

Bit 4

5.154.2.109 ADC_JSQR_JSQ2

```
#define ADC_JSQR_JSQ2 ((uint32_t)0x000003E0)
```

JSQ2[4:0] bits (2nd conversion in injected sequence)

5.154.2.110 ADC_JSQR_JSQ2_0

```
#define ADC_JSQR_JSQ2_0 ((uint32_t)0x00000020)
```

Bit 0

5.154.2.111 ADC_JSQR_JSQ2_1

```
#define ADC_JSQR_JSQ2_1 ((uint32_t)0x00000040)
```

Bit 1

5.154.2.112 ADC_JSQR_JSQ2_2

```
#define ADC_JSQR_JSQ2_2 ((uint32_t)0x00000080)
```

Bit 2

5.154.2.113 ADC_JSQR_JSQ2_3

```
#define ADC_JSQR_JSQ2_3 ((uint32_t)0x00000100)
```

Bit 3

5.154.2.114 ADC_JSQR_JSQ2_4

```
#define ADC_JSQR_JSQ2_4 ((uint32_t)0x00000200)
```

Bit 4

5.154.2.115 ADC_JSQR_JSQ3

```
#define ADC_JSQR_JSQ3 ((uint32_t)0x00007C00)
```

JSQ3[4:0] bits (3rd conversion in injected sequence)

5.154.2.116 ADC_JSQR_JSQ3_0

```
#define ADC_JSQR_JSQ3_0 ((uint32_t)0x00000400)
```

Bit 0

5.154.2.117 ADC_JSQR_JSQ3_1

```
#define ADC_JSQR_JSQ3_1 ((uint32_t)0x00000800)
```

Bit 1

5.154.2.118 ADC_JSQR_JSQ3_2

```
#define ADC_JSQR_JSQ3_2 ((uint32_t)0x00001000)
```

Bit 2

5.154.2.119 ADC_JSQR_JSQ3_3

```
#define ADC_JSQR_JSQ3_3 ((uint32_t)0x00002000)
```

Bit 3

5.154.2.120 ADC_JSQR_JSQ3_4

```
#define ADC_JSQR_JSQ3_4 ((uint32_t)0x00004000)
```

Bit 4

5.154.2.121 ADC_JSQR_JSQ4

```
#define ADC_JSQR_JSQ4 ((uint32_t)0x000F8000)
```

JSQ4[4:0] bits (4th conversion in injected sequence)

5.154.2.122 ADC_JSQR_JSQ4_0

```
#define ADC_JSQR_JSQ4_0 ((uint32_t)0x00008000)
```

Bit 0

5.154.2.123 ADC_JSQR_JSQ4_1

```
#define ADC_JSQR_JSQ4_1 ((uint32_t)0x00010000)
```

Bit 1

5.154.2.124 ADC_JSQR_JSQ4_2

```
#define ADC_JSQR_JSQ4_2 ((uint32_t)0x00020000)
```

Bit 2

5.154.2.125 ADC_JSQR_JSQ4_3

```
#define ADC_JSQR_JSQ4_3 ((uint32_t)0x00040000)
```

Bit 3

5.154.2.126 ADC_JSQR_JSQ4_4

```
#define ADC_JSQR_JSQ4_4 ((uint32_t)0x00080000)
```

Bit 4

5.154.2.127 ADC_LTR_LT

```
#define ADC_LTR_LT ((uint16_t)0x0FFF)
```

Analog watchdog low threshold

5.154.2.128 ADC_SMPR1_SMP10

```
#define ADC_SMPR1_SMP10 ((uint32_t)0x00000007)
```

SMP10[2:0] bits (Channel 10 Sample time selection)

5.154.2.129 ADC_SMPR1_SMP10_0

```
#define ADC_SMPR1_SMP10_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.130 ADC_SMPR1_SMP10_1

```
#define ADC_SMPR1_SMP10_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.131 ADC_SMPR1_SMP10_2

```
#define ADC_SMPR1_SMP10_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.132 ADC_SMPR1_SMP11

```
#define ADC_SMPR1_SMP11 ((uint32_t)0x00000038)
```

SMP11[2:0] bits (Channel 11 Sample time selection)

5.154.2.133 ADC_SMPR1_SMP11_0

```
#define ADC_SMPR1_SMP11_0 ((uint32_t)0x00000008)
```

Bit 0

5.154.2.134 ADC_SMPR1_SMP11_1

```
#define ADC_SMPR1_SMP11_1 ((uint32_t)0x00000010)
```

Bit 1

5.154.2.135 ADC_SMPR1_SMP11_2

```
#define ADC_SMPR1_SMP11_2 ((uint32_t)0x00000020)
```

Bit 2

5.154.2.136 ADC_SMPR1_SMP12

```
#define ADC_SMPR1_SMP12 ((uint32_t)0x000001C0)
```

SMP12[2:0] bits (Channel 12 Sample time selection)

5.154.2.137 ADC_SMPR1_SMP12_0

```
#define ADC_SMPR1_SMP12_0 ((uint32_t)0x00000040)
```

Bit 0

5.154.2.138 ADC_SMPR1_SMP12_1

```
#define ADC_SMPR1_SMP12_1 ((uint32_t)0x00000080)
```

Bit 1

5.154.2.139 ADC_SMPR1_SMP12_2

```
#define ADC_SMPR1_SMP12_2 ((uint32_t)0x00000100)
```

Bit 2

5.154.2.140 ADC_SMPR1_SMP13

```
#define ADC_SMPR1_SMP13 ((uint32_t)0x00000E00)
```

SMP13[2:0] bits (Channel 13 Sample time selection)

5.154.2.141 ADC_SMPR1_SMP13_0

```
#define ADC_SMPR1_SMP13_0 ((uint32_t)0x00000200)
```

Bit 0

5.154.2.142 ADC_SMPR1_SMP13_1

```
#define ADC_SMPR1_SMP13_1 ((uint32_t)0x00000400)
```

Bit 1

5.154.2.143 ADC_SMPR1_SMP13_2

```
#define ADC_SMPR1_SMP13_2 ((uint32_t)0x00000800)
```

Bit 2

5.154.2.144 ADC_SMPR1_SMP14

```
#define ADC_SMPR1_SMP14 ((uint32_t)0x00007000)
```

SMP14[2:0] bits (Channel 14 Sample time selection)

5.154.2.145 ADC_SMPR1_SMP14_0

```
#define ADC_SMPR1_SMP14_0 ((uint32_t)0x00001000)
```

Bit 0

5.154.2.146 ADC_SMPR1_SMP14_1

```
#define ADC_SMPR1_SMP14_1 ((uint32_t)0x00002000)
```

Bit 1

5.154.2.147 ADC_SMPR1_SMP14_2

```
#define ADC_SMPR1_SMP14_2 ((uint32_t)0x00004000)
```

Bit 2

5.154.2.148 ADC_SMPR1_SMP15

```
#define ADC_SMPR1_SMP15 ((uint32_t)0x00038000)
```

SMP15[2:0] bits (Channel 15 Sample time selection)

5.154.2.149 ADC_SMPR1_SMP15_0

```
#define ADC_SMPR1_SMP15_0 ((uint32_t)0x00008000)
```

Bit 0

5.154.2.150 ADC_SMPR1_SMP15_1

```
#define ADC_SMPR1_SMP15_1 ((uint32_t)0x00010000)
```

Bit 1

5.154.2.151 ADC_SMPR1_SMP15_2

```
#define ADC_SMPR1_SMP15_2 ((uint32_t)0x00020000)
```

Bit 2

5.154.2.152 ADC_SMPR1_SMP16

```
#define ADC_SMPR1_SMP16 ((uint32_t)0x001C0000)
```

SMP16[2:0] bits (Channel 16 Sample time selection)

5.154.2.153 ADC_SMPR1_SMP16_0

```
#define ADC_SMPR1_SMP16_0 ((uint32_t)0x00040000)
```

Bit 0

5.154.2.154 ADC_SMPR1_SMP16_1

```
#define ADC_SMPR1_SMP16_1 ((uint32_t)0x00080000)
```

Bit 1

5.154.2.155 ADC_SMPR1_SMP16_2

```
#define ADC_SMPR1_SMP16_2 ((uint32_t)0x00100000)
```

Bit 2

5.154.2.156 ADC_SMPR1_SMP17

```
#define ADC_SMPR1_SMP17 ((uint32_t)0x00E00000)
```

SMP17[2:0] bits (Channel 17 Sample time selection)

5.154.2.157 ADC_SMPR1_SMP17_0

```
#define ADC_SMPR1_SMP17_0 ((uint32_t)0x00200000)
```

Bit 0

5.154.2.158 ADC_SMPR1_SMP17_1

```
#define ADC_SMPR1_SMP17_1 ((uint32_t)0x00400000)
```

Bit 1

5.154.2.159 ADC_SMPR1_SMP17_2

```
#define ADC_SMPR1_SMP17_2 ((uint32_t)0x00800000)
```

Bit 2

5.154.2.160 ADC_SMPR1_SMP18

```
#define ADC_SMPR1_SMP18 ((uint32_t)0x07000000)
```

SMP18[2:0] bits (Channel 18 Sample time selection)

5.154.2.161 ADC_SMPR1_SMP18_0

```
#define ADC_SMPR1_SMP18_0 ((uint32_t)0x01000000)
```

Bit 0

5.154.2.162 ADC_SMPR1_SMP18_1

```
#define ADC_SMPR1_SMP18_1 ((uint32_t)0x02000000)
```

Bit 1

5.154.2.163 ADC_SMPR1_SMP18_2

```
#define ADC_SMPR1_SMP18_2 ((uint32_t)0x04000000)
```

Bit 2

5.154.2.164 ADC_SMPR2_SMP0

```
#define ADC_SMPR2_SMP0 ((uint32_t)0x00000007)
```

SMP0[2:0] bits (Channel 0 Sample time selection)

5.154.2.165 ADC_SMPR2_SMP0_0

```
#define ADC_SMPR2_SMP0_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.166 ADC_SMPR2_SMP0_1

```
#define ADC_SMPR2_SMP0_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.167 ADC_SMPR2_SMP0_2

```
#define ADC_SMPR2_SMP0_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.168 ADC_SMPR2_SMP1

```
#define ADC_SMPR2_SMP1 ((uint32_t)0x00000038)
```

SMP1[2:0] bits (Channel 1 Sample time selection)

5.154.2.169 ADC_SMPR2_SMP1_0

```
#define ADC_SMPR2_SMP1_0 ((uint32_t)0x00000008)
```

Bit 0

5.154.2.170 ADC_SMPR2_SMP1_1

```
#define ADC_SMPR2_SMP1_1 ((uint32_t)0x00000010)
```

Bit 1

5.154.2.171 ADC_SMPR2_SMP1_2

```
#define ADC_SMPR2_SMP1_2 ((uint32_t)0x00000020)
```

Bit 2

5.154.2.172 ADC_SMPR2_SMP2

```
#define ADC_SMPR2_SMP2 ((uint32_t)0x000001C0)
```

SMP2[2:0] bits (Channel 2 Sample time selection)

5.154.2.173 ADC_SMPR2_SMP2_0

```
#define ADC_SMPR2_SMP2_0 ((uint32_t)0x00000040)
```

Bit 0

5.154.2.174 ADC_SMPR2_SMP2_1

```
#define ADC_SMPR2_SMP2_1 ((uint32_t)0x00000080)
```

Bit 1

5.154.2.175 ADC_SMPR2_SMP2_2

```
#define ADC_SMPR2_SMP2_2 ((uint32_t)0x00000100)
```

Bit 2

5.154.2.176 ADC_SMPR2_SMP3

```
#define ADC_SMPR2_SMP3 ((uint32_t)0x00000E00)
```

SMP3[2:0] bits (Channel 3 Sample time selection)

5.154.2.177 ADC_SMPR2_SMP3_0

```
#define ADC_SMPR2_SMP3_0 ((uint32_t)0x00000200)
```

Bit 0

5.154.2.178 ADC_SMPR2_SMP3_1

```
#define ADC_SMPR2_SMP3_1 ((uint32_t)0x00000400)
```

Bit 1

5.154.2.179 ADC_SMPR2_SMP3_2

```
#define ADC_SMPR2_SMP3_2 ((uint32_t)0x00000800)
```

Bit 2

5.154.2.180 ADC_SMPR2_SMP4

```
#define ADC_SMPR2_SMP4 ((uint32_t)0x00007000)
```

SMP4[2:0] bits (Channel 4 Sample time selection)

5.154.2.181 ADC_SMPR2_SMP4_0

```
#define ADC_SMPR2_SMP4_0 ((uint32_t)0x00001000)
```

Bit 0

5.154.2.182 ADC_SMPR2_SMP4_1

```
#define ADC_SMPR2_SMP4_1 ((uint32_t)0x00002000)
```

Bit 1

5.154.2.183 ADC_SMPR2_SMP4_2

```
#define ADC_SMPR2_SMP4_2 ((uint32_t)0x00004000)
```

Bit 2

5.154.2.184 ADC_SMPR2_SMP5

```
#define ADC_SMPR2_SMP5 ((uint32_t)0x00038000)
```

SMP5[2:0] bits (Channel 5 Sample time selection)

5.154.2.185 ADC_SMPR2_SMP5_0

```
#define ADC_SMPR2_SMP5_0 ((uint32_t)0x00008000)
```

Bit 0

5.154.2.186 ADC_SMPR2_SMP5_1

```
#define ADC_SMPR2_SMP5_1 ((uint32_t)0x00010000)
```

Bit 1

5.154.2.187 ADC_SMPR2_SMP5_2

```
#define ADC_SMPR2_SMP5_2 ((uint32_t)0x00020000)
```

Bit 2

5.154.2.188 ADC_SMPR2_SMP6

```
#define ADC_SMPR2_SMP6 ((uint32_t)0x001C0000)
```

SMP6[2:0] bits (Channel 6 Sample time selection)

5.154.2.189 ADC_SMPR2_SMP6_0

```
#define ADC_SMPR2_SMP6_0 ((uint32_t)0x00040000)
```

Bit 0

5.154.2.190 ADC_SMPR2_SMP6_1

```
#define ADC_SMPR2_SMP6_1 ((uint32_t)0x00080000)
```

Bit 1

5.154.2.191 ADC_SMPR2_SMP6_2

```
#define ADC_SMPR2_SMP6_2 ((uint32_t)0x00100000)
```

Bit 2

5.154.2.192 ADC_SMPR2_SMP7

```
#define ADC_SMPR2_SMP7 ((uint32_t)0x00E00000)
```

SMP7[2:0] bits (Channel 7 Sample time selection)

5.154.2.193 ADC_SMPR2_SMP7_0

```
#define ADC_SMPR2_SMP7_0 ((uint32_t)0x00200000)
```

Bit 0

5.154.2.194 ADC_SMPR2_SMP7_1

```
#define ADC_SMPR2_SMP7_1 ((uint32_t)0x00400000)
```

Bit 1

5.154.2.195 ADC_SMPR2_SMP7_2

```
#define ADC_SMPR2_SMP7_2 ((uint32_t)0x00800000)
```

Bit 2

5.154.2.196 ADC_SMPR2_SMP8

```
#define ADC_SMPR2_SMP8 ((uint32_t)0x07000000)
```

SMP8[2:0] bits (Channel 8 Sample time selection)

5.154.2.197 ADC_SMPR2_SMP8_0

```
#define ADC_SMPR2_SMP8_0 ((uint32_t)0x01000000)
```

Bit 0

5.154.2.198 ADC_SMPR2_SMP8_1

```
#define ADC_SMPR2_SMP8_1 ((uint32_t)0x02000000)
```

Bit 1

5.154.2.199 ADC_SMPR2_SMP8_2

```
#define ADC_SMPR2_SMP8_2 ((uint32_t)0x04000000)
```

Bit 2

5.154.2.200 ADC_SMPR2_SMP9

```
#define ADC_SMPR2_SMP9 ((uint32_t)0x38000000)
```

SMP9[2:0] bits (Channel 9 Sample time selection)

5.154.2.201 ADC_SMPR2_SMP9_0

```
#define ADC_SMPR2_SMP9_0 ((uint32_t)0x08000000)
```

Bit 0

5.154.2.202 ADC_SMPR2_SMP9_1

```
#define ADC_SMPR2_SMP9_1 ((uint32_t)0x10000000)
```

Bit 1

5.154.2.203 ADC_SMPR2_SMP9_2

```
#define ADC_SMPR2_SMP9_2 ((uint32_t)0x20000000)
```

Bit 2

5.154.2.204 ADC_SQR1_L

```
#define ADC_SQR1_L ((uint32_t)0x00F00000)
```

L[3:0] bits (Regular channel sequence length)

5.154.2.205 ADC_SQR1_L_0

```
#define ADC_SQR1_L_0 ((uint32_t)0x00100000)
```

Bit 0

5.154.2.206 ADC_SQR1_L_1

```
#define ADC_SQR1_L_1 ((uint32_t)0x00200000)
```

Bit 1

5.154.2.207 ADC_SQR1_L_2

```
#define ADC_SQR1_L_2 ((uint32_t)0x00400000)
```

Bit 2

5.154.2.208 ADC_SQR1_L_3

```
#define ADC_SQR1_L_3 ((uint32_t)0x00800000)
```

Bit 3

5.154.2.209 ADC_SQR1_SQ13

```
#define ADC_SQR1_SQ13 ((uint32_t)0x0000001F)
```

SQ13[4:0] bits (13th conversion in regular sequence)

5.154.2.210 ADC_SQR1_SQ13_0

```
#define ADC_SQR1_SQ13_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.211 ADC_SQR1_SQ13_1

```
#define ADC_SQR1_SQ13_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.212 ADC_SQR1_SQ13_2

```
#define ADC_SQR1_SQ13_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.213 ADC_SQR1_SQ13_3

```
#define ADC_SQR1_SQ13_3 ((uint32_t)0x00000008)
```

Bit 3

5.154.2.214 ADC_SQR1_SQ13_4

```
#define ADC_SQR1_SQ13_4 ((uint32_t)0x00000010)
```

Bit 4

5.154.2.215 ADC_SQR1_SQ14

```
#define ADC_SQR1_SQ14 ((uint32_t)0x000003E0)
```

SQ14[4:0] bits (14th conversion in regular sequence)

5.154.2.216 ADC_SQR1_SQ14_0

```
#define ADC_SQR1_SQ14_0 ((uint32_t)0x00000020)
```

Bit 0

5.154.2.217 ADC_SQR1_SQ14_1

```
#define ADC_SQR1_SQ14_1 ((uint32_t)0x00000040)
```

Bit 1

5.154.2.218 ADC_SQR1_SQ14_2

```
#define ADC_SQR1_SQ14_2 ((uint32_t)0x00000080)
```

Bit 2

5.154.2.219 ADC_SQR1_SQ14_3

```
#define ADC_SQR1_SQ14_3 ((uint32_t)0x00000100)
```

Bit 3

5.154.2.220 ADC_SQR1_SQ14_4

```
#define ADC_SQR1_SQ14_4 ((uint32_t)0x00000200)
```

Bit 4

5.154.2.221 ADC_SQR1_SQ15

```
#define ADC_SQR1_SQ15 ((uint32_t)0x00007C00)
```

SQ15[4:0] bits (15th conversion in regular sequence)

5.154.2.222 ADC_SQR1_SQ15_0

```
#define ADC_SQR1_SQ15_0 ((uint32_t)0x00000400)
```

Bit 0

5.154.2.223 ADC_SQR1_SQ15_1

```
#define ADC_SQR1_SQ15_1 ((uint32_t)0x00000800)
```

Bit 1

5.154.2.224 ADC_SQR1_SQ15_2

```
#define ADC_SQR1_SQ15_2 ((uint32_t)0x00001000)
```

Bit 2

5.154.2.225 ADC_SQR1_SQ15_3

```
#define ADC_SQR1_SQ15_3 ((uint32_t)0x00002000)
```

Bit 3

5.154.2.226 ADC_SQR1_SQ15_4

```
#define ADC_SQR1_SQ15_4 ((uint32_t)0x00004000)
```

Bit 4

5.154.2.227 ADC_SQR1_SQ16

```
#define ADC_SQR1_SQ16 ((uint32_t)0x000F8000)
```

SQ16[4:0] bits (16th conversion in regular sequence)

5.154.2.228 ADC_SQR1_SQ16_0

```
#define ADC_SQR1_SQ16_0 ((uint32_t)0x00008000)
```

Bit 0

5.154.2.229 ADC_SQR1_SQ16_1

```
#define ADC_SQR1_SQ16_1 ((uint32_t)0x00010000)
```

Bit 1

5.154.2.230 ADC_SQR1_SQ16_2

```
#define ADC_SQR1_SQ16_2 ((uint32_t)0x00020000)
```

Bit 2

5.154.2.231 ADC_SQR1_SQ16_3

```
#define ADC_SQR1_SQ16_3 ((uint32_t)0x00040000)
```

Bit 3

5.154.2.232 ADC_SQR1_SQ16_4

```
#define ADC_SQR1_SQ16_4 ((uint32_t)0x00080000)
```

Bit 4

5.154.2.233 ADC_SQR2_SQ10

```
#define ADC_SQR2_SQ10 ((uint32_t)0x000F8000)
```

SQ10[4:0] bits (10th conversion in regular sequence)

5.154.2.234 ADC_SQR2_SQ10_0

```
#define ADC_SQR2_SQ10_0 ((uint32_t)0x00008000)
```

Bit 0

5.154.2.235 ADC_SQR2_SQ10_1

```
#define ADC_SQR2_SQ10_1 ((uint32_t)0x00010000)
```

Bit 1

5.154.2.236 ADC_SQR2_SQ10_2

```
#define ADC_SQR2_SQ10_2 ((uint32_t)0x00020000)
```

Bit 2

5.154.2.237 ADC_SQR2_SQ10_3

```
#define ADC_SQR2_SQ10_3 ((uint32_t)0x00040000)
```

Bit 3

5.154.2.238 ADC_SQR2_SQ10_4

```
#define ADC_SQR2_SQ10_4 ((uint32_t)0x00080000)
```

Bit 4

5.154.2.239 ADC_SQR2_SQ11

```
#define ADC_SQR2_SQ11 ((uint32_t)0x01F00000)
```

SQ11[4:0] bits (11th conversion in regular sequence)

5.154.2.240 ADC_SQR2_SQ11_0

```
#define ADC_SQR2_SQ11_0 ((uint32_t)0x00100000)
```

Bit 0

5.154.2.241 ADC_SQR2_SQ11_1

```
#define ADC_SQR2_SQ11_1 ((uint32_t)0x00200000)
```

Bit 1

5.154.2.242 ADC_SQR2_SQ11_2

```
#define ADC_SQR2_SQ11_2 ((uint32_t)0x00400000)
```

Bit 2

5.154.2.243 ADC_SQR2_SQ11_3

```
#define ADC_SQR2_SQ11_3 ((uint32_t)0x00800000)
```

Bit 3

5.154.2.244 ADC_SQR2_SQ11_4

```
#define ADC_SQR2_SQ11_4 ((uint32_t)0x01000000)
```

Bit 4

5.154.2.245 ADC_SQR2_SQ12

```
#define ADC_SQR2_SQ12 ((uint32_t)0x3E000000)
```

SQ12[4:0] bits (12th conversion in regular sequence)

5.154.2.246 ADC_SQR2_SQ12_0

```
#define ADC_SQR2_SQ12_0 ((uint32_t)0x02000000)
```

Bit 0

5.154.2.247 ADC_SQR2_SQ12_1

```
#define ADC_SQR2_SQ12_1 ((uint32_t)0x04000000)
```

Bit 1

5.154.2.248 ADC_SQR2_SQ12_2

```
#define ADC_SQR2_SQ12_2 ((uint32_t)0x08000000)
```

Bit 2

5.154.2.249 ADC_SQR2_SQ12_3

```
#define ADC_SQR2_SQ12_3 ((uint32_t)0x10000000)
```

Bit 3

5.154.2.250 ADC_SQR2_SQ12_4

```
#define ADC_SQR2_SQ12_4 ((uint32_t)0x20000000)
```

Bit 4

5.154.2.251 ADC_SQR2_SQ7

```
#define ADC_SQR2_SQ7 ((uint32_t)0x0000001F)
```

SQ7[4:0] bits (7th conversion in regular sequence)

5.154.2.252 ADC_SQR2_SQ7_0

```
#define ADC_SQR2_SQ7_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.253 ADC_SQR2_SQ7_1

```
#define ADC_SQR2_SQ7_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.254 ADC_SQR2_SQ7_2

```
#define ADC_SQR2_SQ7_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.255 ADC_SQR2_SQ7_3

```
#define ADC_SQR2_SQ7_3 ((uint32_t)0x00000008)
```

Bit 3

5.154.2.256 ADC_SQR2_SQ7_4

```
#define ADC_SQR2_SQ7_4 ((uint32_t)0x00000010)
```

Bit 4

5.154.2.257 ADC_SQR2_SQ8

```
#define ADC_SQR2_SQ8 ((uint32_t)0x000003E0)
```

SQ8[4:0] bits (8th conversion in regular sequence)

5.154.2.258 ADC_SQR2_SQ8_0

```
#define ADC_SQR2_SQ8_0 ((uint32_t)0x00000020)
```

Bit 0

5.154.2.259 ADC_SQR2_SQ8_1

```
#define ADC_SQR2_SQ8_1 ((uint32_t)0x00000040)
```

Bit 1

5.154.2.260 ADC_SQR2_SQ8_2

```
#define ADC_SQR2_SQ8_2 ((uint32_t)0x00000080)
```

Bit 2

5.154.2.261 ADC_SQR2_SQ8_3

```
#define ADC_SQR2_SQ8_3 ((uint32_t)0x00000100)
```

Bit 3

5.154.2.262 ADC_SQR2_SQ8_4

```
#define ADC_SQR2_SQ8_4 ((uint32_t)0x00000200)
```

Bit 4

5.154.2.263 ADC_SQR2_SQ9

```
#define ADC_SQR2_SQ9 ((uint32_t)0x00007C00)
```

SQ9[4:0] bits (9th conversion in regular sequence)

5.154.2.264 ADC_SQR2_SQ9_0

```
#define ADC_SQR2_SQ9_0 ((uint32_t)0x00000400)
```

Bit 0

5.154.2.265 ADC_SQR2_SQ9_1

```
#define ADC_SQR2_SQ9_1 ((uint32_t)0x00000800)
```

Bit 1

5.154.2.266 ADC_SQR2_SQ9_2

```
#define ADC_SQR2_SQ9_2 ((uint32_t)0x00001000)
```

Bit 2

5.154.2.267 ADC_SQR2_SQ9_3

```
#define ADC_SQR2_SQ9_3 ((uint32_t)0x00002000)
```

Bit 3

5.154.2.268 ADC_SQR2_SQ9_4

```
#define ADC_SQR2_SQ9_4 ((uint32_t)0x00004000)
```

Bit 4

5.154.2.269 ADC_SQR3_SQ1

```
#define ADC_SQR3_SQ1 ((uint32_t)0x0000001F)
```

SQ1[4:0] bits (1st conversion in regular sequence)

5.154.2.270 ADC_SQR3_SQ1_0

```
#define ADC_SQR3_SQ1_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.271 ADC_SQR3_SQ1_1

```
#define ADC_SQR3_SQ1_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.272 ADC_SQR3_SQ1_2

```
#define ADC_SQR3_SQ1_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.273 ADC_SQR3_SQ1_3

```
#define ADC_SQR3_SQ1_3 ((uint32_t)0x00000008)
```

Bit 3

5.154.2.274 ADC_SQR3_SQ1_4

```
#define ADC_SQR3_SQ1_4 ((uint32_t)0x00000010)
```

Bit 4

5.154.2.275 ADC_SQR3_SQ2

```
#define ADC_SQR3_SQ2 ((uint32_t)0x000003E0)
```

SQ2[4:0] bits (2nd conversion in regular sequence)

5.154.2.276 ADC_SQR3_SQ2_0

```
#define ADC_SQR3_SQ2_0 ((uint32_t)0x00000020)
```

Bit 0

5.154.2.277 ADC_SQR3_SQ2_1

```
#define ADC_SQR3_SQ2_1 ((uint32_t)0x00000040)
```

Bit 1

5.154.2.278 ADC_SQR3_SQ2_2

```
#define ADC_SQR3_SQ2_2 ((uint32_t)0x00000080)
```

Bit 2

5.154.2.279 ADC_SQR3_SQ2_3

```
#define ADC_SQR3_SQ2_3 ((uint32_t)0x00000100)
```

Bit 3

5.154.2.280 ADC_SQR3_SQ2_4

```
#define ADC_SQR3_SQ2_4 ((uint32_t)0x00000200)
```

Bit 4

5.154.2.281 ADC_SQR3_SQ3

```
#define ADC_SQR3_SQ3 ((uint32_t)0x00007C00)
```

SQ3[4:0] bits (3rd conversion in regular sequence)

5.154.2.282 ADC_SQR3_SQ3_0

```
#define ADC_SQR3_SQ3_0 ((uint32_t)0x00000400)
```

Bit 0

5.154.2.283 ADC_SQR3_SQ3_1

```
#define ADC_SQR3_SQ3_1 ((uint32_t)0x00000800)
```

Bit 1

5.154.2.284 ADC_SQR3_SQ3_2

```
#define ADC_SQR3_SQ3_2 ((uint32_t)0x00001000)
```

Bit 2

5.154.2.285 ADC_SQR3_SQ3_3

```
#define ADC_SQR3_SQ3_3 ((uint32_t)0x00002000)
```

Bit 3

5.154.2.286 ADC_SQR3_SQ3_4

```
#define ADC_SQR3_SQ3_4 ((uint32_t)0x00004000)
```

Bit 4

5.154.2.287 ADC_SQR3_SQ4

```
#define ADC_SQR3_SQ4 ((uint32_t)0x000F8000)
```

SQ4[4:0] bits (4th conversion in regular sequence)

5.154.2.288 ADC_SQR3_SQ4_0

```
#define ADC_SQR3_SQ4_0 ((uint32_t)0x00008000)
```

Bit 0

5.154.2.289 ADC_SQR3_SQ4_1

```
#define ADC_SQR3_SQ4_1 ((uint32_t)0x00010000)
```

Bit 1

5.154.2.290 ADC_SQR3_SQ4_2

```
#define ADC_SQR3_SQ4_2 ((uint32_t)0x00020000)
```

Bit 2

5.154.2.291 ADC_SQR3_SQ4_3

```
#define ADC_SQR3_SQ4_3 ((uint32_t)0x00040000)
```

Bit 3

5.154.2.292 ADC_SQR3_SQ4_4

```
#define ADC_SQR3_SQ4_4 ((uint32_t)0x00080000)
```

Bit 4

5.154.2.293 ADC_SQR3_SQ5

```
#define ADC_SQR3_SQ5 ((uint32_t)0x01F00000)
```

SQ5[4:0] bits (5th conversion in regular sequence)

5.154.2.294 ADC_SQR3_SQ5_0

```
#define ADC_SQR3_SQ5_0 ((uint32_t)0x00100000)
```

Bit 0

5.154.2.295 ADC_SQR3_SQ5_1

```
#define ADC_SQR3_SQ5_1 ((uint32_t)0x00200000)
```

Bit 1

5.154.2.296 ADC_SQR3_SQ5_2

```
#define ADC_SQR3_SQ5_2 ((uint32_t)0x00400000)
```

Bit 2

5.154.2.297 ADC_SQR3_SQ5_3

```
#define ADC_SQR3_SQ5_3 ((uint32_t)0x00800000)
```

Bit 3

5.154.2.298 ADC_SQR3_SQ5_4

```
#define ADC_SQR3_SQ5_4 ((uint32_t)0x01000000)
```

Bit 4

5.154.2.299 ADC_SQR3_SQ6

```
#define ADC_SQR3_SQ6 ((uint32_t)0x3E000000)
```

SQ6[4:0] bits (6th conversion in regular sequence)

5.154.2.300 ADC_SQR3_SQ6_0

```
#define ADC_SQR3_SQ6_0 ((uint32_t)0x02000000)
```

Bit 0

5.154.2.301 ADC_SQR3_SQ6_1

```
#define ADC_SQR3_SQ6_1 ((uint32_t)0x04000000)
```

Bit 1

5.154.2.302 ADC_SQR3_SQ6_2

```
#define ADC_SQR3_SQ6_2 ((uint32_t)0x08000000)
```

Bit 2

5.154.2.303 ADC_SQR3_SQ6_3

```
#define ADC_SQR3_SQ6_3 ((uint32_t)0x10000000)
```

Bit 3

5.154.2.304 ADC_SQR3_SQ6_4

```
#define ADC_SQR3_SQ6_4 ((uint32_t)0x20000000)
```

Bit 4

5.154.2.305 ADC_SR_AWD

```
#define ADC_SR_AWD ((uint8_t)0x01)
```

Analog watchdog flag

5.154.2.306 ADC_SR_EOC

```
#define ADC_SR_EOC ((uint8_t)0x02)
```

End of conversion

5.154.2.307 ADC_SR_JEOC

```
#define ADC_SR_JEOC ((uint8_t)0x04)
```

Injected channel end of conversion

5.154.2.308 ADC_SR_JSTRT

```
#define ADC_SR_JSTRT ((uint8_t)0x08)
```

Injected channel Start flag

5.154.2.309 ADC_SR_OVR

```
#define ADC_SR_OVR ((uint8_t)0x20)
```

Overrun flag

5.154.2.310 ADC_SR_STRT

```
#define ADC_SR_STRT ((uint8_t)0x10)
```

Regular channel Start flag

5.154.2.311 CAN_BTR_BRP

```
#define CAN_BTR_BRP ((uint32_t)0x000003FF)
```

Baud Rate Prescaler

5.154.2.312 CAN_BTR_LBKM

```
#define CAN_BTR_LBKM ((uint32_t)0x40000000)
```

Loop Back Mode (Debug)

5.154.2.313 CAN_BTR_SILM

```
#define CAN_BTR_SILM ((uint32_t)0x80000000)
```

Silent Mode Mailbox registers

5.154.2.314 CAN_BTR_SJW

```
#define CAN_BTR_SJW ((uint32_t)0x03000000)
```

Resynchronization Jump Width

5.154.2.315 CAN_BTR_TS1

```
#define CAN_BTR_TS1 ((uint32_t)0x000F0000)
```

Time Segment 1

5.154.2.316 CAN_BTR_TS2

```
#define CAN_BTR_TS2 ((uint32_t)0x00700000)
```

Time Segment 2

5.154.2.317 CAN_ESR_BOFF

```
#define CAN_ESR_BOFF ((uint32_t)0x00000004)
```

Bus-Off Flag

5.154.2.318 CAN_ESR_EPVF

```
#define CAN_ESR_EPVF ((uint32_t)0x00000002)
```

Error Passive Flag

5.154.2.319 CAN_ESR_EWGF

```
#define CAN_ESR_EWGF ((uint32_t)0x00000001)
```

Error Warning Flag

5.154.2.320 CAN_ESR_LEC

```
#define CAN_ESR_LEC ((uint32_t)0x00000070)
```

LEC[2:0] bits (Last Error Code)

5.154.2.321 CAN_ESR_LEC_0

```
#define CAN_ESR_LEC_0 ((uint32_t)0x00000010)
```

Bit 0

5.154.2.322 CAN_ESR_LEC_1

```
#define CAN_ESR_LEC_1 ((uint32_t)0x00000020)
```

Bit 1

5.154.2.323 CAN_ESR_LEC_2

```
#define CAN_ESR_LEC_2 ((uint32_t)0x00000040)
```

Bit 2

5.154.2.324 CAN_ESR_REC

```
#define CAN_ESR_REC ((uint32_t)0xFF000000)
```

Receive Error Counter

5.154.2.325 CAN_ESR_TEC

```
#define CAN_ESR_TEC ((uint32_t)0x00FF0000)
```

Least significant byte of the 9-bit Transmit Error Counter

5.154.2.326 CAN_F0R1_FB0

```
#define CAN_F0R1_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.327 CAN_F0R1_FB1

```
#define CAN_F0R1_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.328 CAN_F0R1_FB10

```
#define CAN_F0R1_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.329 CAN_F0R1_FB11

```
#define CAN_F0R1_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.330 CAN_F0R1_FB12

```
#define CAN_F0R1_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.331 CAN_F0R1_FB13

```
#define CAN_F0R1_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.332 CAN_F0R1_FB14

```
#define CAN_F0R1_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.333 CAN_F0R1_FB15

```
#define CAN_F0R1_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.334 CAN_F0R1_FB16

```
#define CAN_F0R1_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.335 CAN_F0R1_FB17

```
#define CAN_F0R1_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.336 CAN_F0R1_FB18

```
#define CAN_F0R1_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.337 CAN_F0R1_FB19

```
#define CAN_F0R1_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.338 CAN_F0R1_FB2

```
#define CAN_F0R1_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.339 CAN_F0R1_FB20

```
#define CAN_F0R1_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.340 CAN_F0R1_FB21

```
#define CAN_F0R1_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.341 CAN_F0R1_FB22

```
#define CAN_F0R1_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.342 CAN_F0R1_FB23

```
#define CAN_F0R1_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.343 CAN_F0R1_FB24

```
#define CAN_F0R1_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.344 CAN_F0R1_FB25

```
#define CAN_F0R1_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.345 CAN_F0R1_FB26

```
#define CAN_F0R1_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.346 CAN_F0R1_FB27

```
#define CAN_F0R1_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.347 CAN_F0R1_FB28

```
#define CAN_F0R1_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.348 CAN_F0R1_FB29

```
#define CAN_F0R1_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.349 CAN_F0R1_FB3

```
#define CAN_F0R1_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.350 CAN_F0R1_FB30

```
#define CAN_F0R1_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.351 CAN_F0R1_FB31

```
#define CAN_F0R1_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.352 CAN_F0R1_FB4

```
#define CAN_F0R1_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.353 CAN_F0R1_FB5

```
#define CAN_F0R1_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.354 CAN_F0R1_FB6

```
#define CAN_F0R1_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.355 CAN_F0R1_FB7

```
#define CAN_F0R1_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.356 CAN_F0R1_FB8

```
#define CAN_F0R1_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.357 CAN_F0R1_FB9

```
#define CAN_F0R1_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.358 CAN_F0R2_FB0

```
#define CAN_F0R2_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.359 CAN_F0R2_FB1

```
#define CAN_F0R2_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.360 CAN_F0R2_FB10

```
#define CAN_F0R2_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.361 CAN_F0R2_FB11

```
#define CAN_F0R2_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.362 CAN_F0R2_FB12

```
#define CAN_F0R2_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.363 CAN_F0R2_FB13

```
#define CAN_F0R2_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.364 CAN_F0R2_FB14

```
#define CAN_F0R2_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.365 CAN_F0R2_FB15

```
#define CAN_F0R2_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.366 CAN_F0R2_FB16

```
#define CAN_F0R2_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.367 CAN_F0R2_FB17

```
#define CAN_F0R2_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.368 CAN_F0R2_FB18

```
#define CAN_F0R2_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.369 CAN_F0R2_FB19

```
#define CAN_F0R2_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.370 CAN_F0R2_FB2

```
#define CAN_F0R2_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.371 CAN_F0R2_FB20

```
#define CAN_F0R2_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.372 CAN_F0R2_FB21

```
#define CAN_F0R2_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.373 CAN_F0R2_FB22

```
#define CAN_F0R2_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.374 CAN_F0R2_FB23

```
#define CAN_F0R2_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.375 CAN_F0R2_FB24

```
#define CAN_F0R2_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.376 CAN_F0R2_FB25

```
#define CAN_F0R2_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.377 CAN_F0R2_FB26

```
#define CAN_F0R2_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.378 CAN_F0R2_FB27

```
#define CAN_F0R2_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.379 CAN_F0R2_FB28

```
#define CAN_F0R2_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.380 CAN_F0R2_FB29

```
#define CAN_F0R2_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.381 CAN_F0R2_FB3

```
#define CAN_F0R2_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.382 CAN_F0R2_FB30

```
#define CAN_F0R2_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.383 CAN_F0R2_FB31

```
#define CAN_F0R2_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.384 CAN_F0R2_FB4

```
#define CAN_F0R2_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.385 CAN_F0R2_FB5

```
#define CAN_F0R2_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.386 CAN_F0R2_FB6

```
#define CAN_F0R2_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.387 CAN_F0R2_FB7

```
#define CAN_F0R2_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.388 CAN_F0R2_FB8

```
#define CAN_F0R2_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.389 CAN_F0R2_FB9

```
#define CAN_F0R2_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.390 CAN_F10R1_FB0

```
#define CAN_F10R1_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.391 CAN_F10R1_FB1

```
#define CAN_F10R1_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.392 CAN_F10R1_FB10

```
#define CAN_F10R1_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.393 CAN_F10R1_FB11

```
#define CAN_F10R1_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.394 CAN_F10R1_FB12

```
#define CAN_F10R1_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.395 CAN_F10R1_FB13

```
#define CAN_F10R1_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.396 CAN_F10R1_FB14

```
#define CAN_F10R1_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.397 CAN_F10R1_FB15

```
#define CAN_F10R1_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.398 CAN_F10R1_FB16

```
#define CAN_F10R1_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.399 CAN_F10R1_FB17

```
#define CAN_F10R1_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.400 CAN_F10R1_FB18

```
#define CAN_F10R1_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.401 CAN_F10R1_FB19

```
#define CAN_F10R1_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.402 CAN_F10R1_FB2

```
#define CAN_F10R1_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.403 CAN_F10R1_FB20

```
#define CAN_F10R1_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.404 CAN_F10R1_FB21

```
#define CAN_F10R1_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.405 CAN_F10R1_FB22

```
#define CAN_F10R1_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.406 CAN_F10R1_FB23

```
#define CAN_F10R1_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.407 CAN_F10R1_FB24

```
#define CAN_F10R1_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.408 CAN_F10R1_FB25

```
#define CAN_F10R1_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.409 CAN_F10R1_FB26

```
#define CAN_F10R1_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.410 CAN_F10R1_FB27

```
#define CAN_F10R1_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.411 CAN_F10R1_FB28

```
#define CAN_F10R1_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.412 CAN_F10R1_FB29

```
#define CAN_F10R1_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.413 CAN_F10R1_FB3

```
#define CAN_F10R1_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.414 CAN_F10R1_FB30

```
#define CAN_F10R1_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.415 CAN_F10R1_FB31

```
#define CAN_F10R1_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.416 CAN_F10R1_FB4

```
#define CAN_F10R1_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.417 CAN_F10R1_FB5

```
#define CAN_F10R1_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.418 CAN_F10R1_FB6

```
#define CAN_F10R1_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.419 CAN_F10R1_FB7

```
#define CAN_F10R1_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.420 CAN_F10R1_FB8

```
#define CAN_F10R1_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.421 CAN_F10R1_FB9

```
#define CAN_F10R1_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.422 CAN_F10R2_FB0

```
#define CAN_F10R2_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.423 CAN_F10R2_FB1

```
#define CAN_F10R2_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.424 CAN_F10R2_FB10

```
#define CAN_F10R2_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.425 CAN_F10R2_FB11

```
#define CAN_F10R2_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.426 CAN_F10R2_FB12

```
#define CAN_F10R2_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.427 CAN_F10R2_FB13

```
#define CAN_F10R2_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.428 CAN_F10R2_FB14

```
#define CAN_F10R2_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.429 CAN_F10R2_FB15

```
#define CAN_F10R2_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.430 CAN_F10R2_FB16

```
#define CAN_F10R2_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.431 CAN_F10R2_FB17

```
#define CAN_F10R2_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.432 CAN_F10R2_FB18

```
#define CAN_F10R2_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.433 CAN_F10R2_FB19

```
#define CAN_F10R2_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.434 CAN_F10R2_FB2

```
#define CAN_F10R2_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.435 CAN_F10R2_FB20

```
#define CAN_F10R2_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.436 CAN_F10R2_FB21

```
#define CAN_F10R2_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.437 CAN_F10R2_FB22

```
#define CAN_F10R2_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.438 CAN_F10R2_FB23

```
#define CAN_F10R2_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.439 CAN_F10R2_FB24

```
#define CAN_F10R2_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.440 CAN_F10R2_FB25

```
#define CAN_F10R2_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.441 CAN_F10R2_FB26

```
#define CAN_F10R2_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.442 CAN_F10R2_FB27

```
#define CAN_F10R2_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.443 CAN_F10R2_FB28

```
#define CAN_F10R2_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.444 CAN_F10R2_FB29

```
#define CAN_F10R2_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.445 CAN_F10R2_FB3

```
#define CAN_F10R2_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.446 CAN_F10R2_FB30

```
#define CAN_F10R2_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.447 CAN_F10R2_FB31

```
#define CAN_F10R2_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.448 CAN_F10R2_FB4

```
#define CAN_F10R2_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.449 CAN_F10R2_FB5

```
#define CAN_F10R2_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.450 CAN_F10R2_FB6

```
#define CAN_F10R2_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.451 CAN_F10R2_FB7

```
#define CAN_F10R2_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.452 CAN_F10R2_FB8

```
#define CAN_F10R2_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.453 CAN_F10R2_FB9

```
#define CAN_F10R2_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.454 CAN_F11R1_FB0

```
#define CAN_F11R1_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.455 CAN_F11R1_FB1

```
#define CAN_F11R1_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.456 CAN_F11R1_FB10

```
#define CAN_F11R1_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.457 CAN_F11R1_FB11

```
#define CAN_F11R1_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.458 CAN_F11R1_FB12

```
#define CAN_F11R1_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.459 CAN_F11R1_FB13

```
#define CAN_F11R1_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.460 CAN_F11R1_FB14

```
#define CAN_F11R1_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.461 CAN_F11R1_FB15

```
#define CAN_F11R1_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.462 CAN_F11R1_FB16

```
#define CAN_F11R1_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.463 CAN_F11R1_FB17

```
#define CAN_F11R1_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.464 CAN_F11R1_FB18

```
#define CAN_F11R1_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.465 CAN_F11R1_FB19

```
#define CAN_F11R1_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.466 CAN_F11R1_FB2

```
#define CAN_F11R1_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.467 CAN_F11R1_FB20

```
#define CAN_F11R1_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.468 CAN_F11R1_FB21

```
#define CAN_F11R1_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.469 CAN_F11R1_FB22

```
#define CAN_F11R1_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.470 CAN_F11R1_FB23

```
#define CAN_F11R1_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.471 CAN_F11R1_FB24

```
#define CAN_F11R1_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.472 CAN_F11R1_FB25

```
#define CAN_F11R1_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.473 CAN_F11R1_FB26

```
#define CAN_F11R1_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.474 CAN_F11R1_FB27

```
#define CAN_F11R1_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.475 CAN_F11R1_FB28

```
#define CAN_F11R1_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.476 CAN_F11R1_FB29

```
#define CAN_F11R1_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.477 CAN_F11R1_FB3

```
#define CAN_F11R1_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.478 CAN_F11R1_FB30

```
#define CAN_F11R1_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.479 CAN_F11R1_FB31

```
#define CAN_F11R1_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.480 CAN_F11R1_FB4

```
#define CAN_F11R1_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.481 CAN_F11R1_FB5

```
#define CAN_F11R1_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.482 CAN_F11R1_FB6

```
#define CAN_F11R1_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.483 CAN_F11R1_FB7

```
#define CAN_F11R1_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.484 CAN_F11R1_FB8

```
#define CAN_F11R1_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.485 CAN_F11R1_FB9

```
#define CAN_F11R1_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.486 CAN_F11R2_FB0

```
#define CAN_F11R2_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.487 CAN_F11R2_FB1

```
#define CAN_F11R2_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.488 CAN_F11R2_FB10

```
#define CAN_F11R2_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.489 CAN_F11R2_FB11

```
#define CAN_F11R2_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.490 CAN_F11R2_FB12

```
#define CAN_F11R2_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.491 CAN_F11R2_FB13

```
#define CAN_F11R2_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.492 CAN_F11R2_FB14

```
#define CAN_F11R2_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.493 CAN_F11R2_FB15

```
#define CAN_F11R2_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.494 CAN_F11R2_FB16

```
#define CAN_F11R2_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.495 CAN_F11R2_FB17

```
#define CAN_F11R2_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.496 CAN_F11R2_FB18

```
#define CAN_F11R2_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.497 CAN_F11R2_FB19

```
#define CAN_F11R2_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.498 CAN_F11R2_FB2

```
#define CAN_F11R2_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.499 CAN_F11R2_FB20

```
#define CAN_F11R2_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.500 CAN_F11R2_FB21

```
#define CAN_F11R2_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.501 CAN_F11R2_FB22

```
#define CAN_F11R2_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.502 CAN_F11R2_FB23

```
#define CAN_F11R2_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.503 CAN_F11R2_FB24

```
#define CAN_F11R2_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.504 CAN_F11R2_FB25

```
#define CAN_F11R2_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.505 CAN_F11R2_FB26

```
#define CAN_F11R2_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.506 CAN_F11R2_FB27

```
#define CAN_F11R2_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.507 CAN_F11R2_FB28

```
#define CAN_F11R2_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.508 CAN_F11R2_FB29

```
#define CAN_F11R2_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.509 CAN_F11R2_FB3

```
#define CAN_F11R2_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.510 CAN_F11R2_FB30

```
#define CAN_F11R2_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.511 CAN_F11R2_FB31

```
#define CAN_F11R2_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.512 CAN_F11R2_FB4

```
#define CAN_F11R2_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.513 CAN_F11R2_FB5

```
#define CAN_F11R2_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.514 CAN_F11R2_FB6

```
#define CAN_F11R2_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.515 CAN_F11R2_FB7

```
#define CAN_F11R2_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.516 CAN_F11R2_FB8

```
#define CAN_F11R2_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.517 CAN_F11R2_FB9

```
#define CAN_F11R2_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.518 CAN_F12R1_FB0

```
#define CAN_F12R1_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.519 CAN_F12R1_FB1

```
#define CAN_F12R1_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.520 CAN_F12R1_FB10

```
#define CAN_F12R1_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.521 CAN_F12R1_FB11

```
#define CAN_F12R1_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.522 CAN_F12R1_FB12

```
#define CAN_F12R1_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.523 CAN_F12R1_FB13

```
#define CAN_F12R1_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.524 CAN_F12R1_FB14

```
#define CAN_F12R1_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.525 CAN_F12R1_FB15

```
#define CAN_F12R1_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.526 CAN_F12R1_FB16

```
#define CAN_F12R1_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.527 CAN_F12R1_FB17

```
#define CAN_F12R1_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.528 CAN_F12R1_FB18

```
#define CAN_F12R1_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.529 CAN_F12R1_FB19

```
#define CAN_F12R1_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.530 CAN_F12R1_FB2

```
#define CAN_F12R1_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.531 CAN_F12R1_FB20

```
#define CAN_F12R1_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.532 CAN_F12R1_FB21

```
#define CAN_F12R1_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.533 CAN_F12R1_FB22

```
#define CAN_F12R1_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.534 CAN_F12R1_FB23

```
#define CAN_F12R1_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.535 CAN_F12R1_FB24

```
#define CAN_F12R1_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.536 CAN_F12R1_FB25

```
#define CAN_F12R1_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.537 CAN_F12R1_FB26

```
#define CAN_F12R1_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.538 CAN_F12R1_FB27

```
#define CAN_F12R1_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.539 CAN_F12R1_FB28

```
#define CAN_F12R1_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.540 CAN_F12R1_FB29

```
#define CAN_F12R1_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.541 CAN_F12R1_FB3

```
#define CAN_F12R1_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.542 CAN_F12R1_FB30

```
#define CAN_F12R1_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.543 CAN_F12R1_FB31

```
#define CAN_F12R1_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.544 CAN_F12R1_FB4

```
#define CAN_F12R1_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.545 CAN_F12R1_FB5

```
#define CAN_F12R1_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.546 CAN_F12R1_FB6

```
#define CAN_F12R1_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.547 CAN_F12R1_FB7

```
#define CAN_F12R1_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.548 CAN_F12R1_FB8

```
#define CAN_F12R1_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.549 CAN_F12R1_FB9

```
#define CAN_F12R1_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.550 CAN_F12R2_FB0

```
#define CAN_F12R2_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.551 CAN_F12R2_FB1

```
#define CAN_F12R2_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.552 CAN_F12R2_FB10

```
#define CAN_F12R2_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.553 CAN_F12R2_FB11

```
#define CAN_F12R2_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.554 CAN_F12R2_FB12

```
#define CAN_F12R2_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.555 CAN_F12R2_FB13

```
#define CAN_F12R2_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.556 CAN_F12R2_FB14

```
#define CAN_F12R2_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.557 CAN_F12R2_FB15

```
#define CAN_F12R2_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.558 CAN_F12R2_FB16

```
#define CAN_F12R2_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.559 CAN_F12R2_FB17

```
#define CAN_F12R2_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.560 CAN_F12R2_FB18

```
#define CAN_F12R2_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.561 CAN_F12R2_FB19

```
#define CAN_F12R2_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.562 CAN_F12R2_FB2

```
#define CAN_F12R2_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.563 CAN_F12R2_FB20

```
#define CAN_F12R2_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.564 CAN_F12R2_FB21

```
#define CAN_F12R2_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.565 CAN_F12R2_FB22

```
#define CAN_F12R2_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.566 CAN_F12R2_FB23

```
#define CAN_F12R2_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.567 CAN_F12R2_FB24

```
#define CAN_F12R2_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.568 CAN_F12R2_FB25

```
#define CAN_F12R2_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.569 CAN_F12R2_FB26

```
#define CAN_F12R2_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.570 CAN_F12R2_FB27

```
#define CAN_F12R2_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.571 CAN_F12R2_FB28

```
#define CAN_F12R2_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.572 CAN_F12R2_FB29

```
#define CAN_F12R2_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.573 CAN_F12R2_FB3

```
#define CAN_F12R2_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.574 CAN_F12R2_FB30

```
#define CAN_F12R2_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.575 CAN_F12R2_FB31

```
#define CAN_F12R2_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.576 CAN_F12R2_FB4

```
#define CAN_F12R2_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.577 CAN_F12R2_FB5

```
#define CAN_F12R2_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.578 CAN_F12R2_FB6

```
#define CAN_F12R2_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.579 CAN_F12R2_FB7

```
#define CAN_F12R2_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.580 CAN_F12R2_FB8

```
#define CAN_F12R2_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.581 CAN_F12R2_FB9

```
#define CAN_F12R2_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.582 CAN_F13R1_FB0

```
#define CAN_F13R1_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.583 CAN_F13R1_FB1

```
#define CAN_F13R1_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.584 CAN_F13R1_FB10

```
#define CAN_F13R1_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.585 CAN_F13R1_FB11

```
#define CAN_F13R1_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.586 CAN_F13R1_FB12

```
#define CAN_F13R1_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.587 CAN_F13R1_FB13

```
#define CAN_F13R1_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.588 CAN_F13R1_FB14

```
#define CAN_F13R1_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.589 CAN_F13R1_FB15

```
#define CAN_F13R1_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.590 CAN_F13R1_FB16

```
#define CAN_F13R1_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.591 CAN_F13R1_FB17

```
#define CAN_F13R1_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.592 CAN_F13R1_FB18

```
#define CAN_F13R1_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.593 CAN_F13R1_FB19

```
#define CAN_F13R1_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.594 CAN_F13R1_FB2

```
#define CAN_F13R1_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.595 CAN_F13R1_FB20

```
#define CAN_F13R1_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.596 CAN_F13R1_FB21

```
#define CAN_F13R1_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.597 CAN_F13R1_FB22

```
#define CAN_F13R1_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.598 CAN_F13R1_FB23

```
#define CAN_F13R1_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.599 CAN_F13R1_FB24

```
#define CAN_F13R1_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.600 CAN_F13R1_FB25

```
#define CAN_F13R1_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.601 CAN_F13R1_FB26

```
#define CAN_F13R1_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.602 CAN_F13R1_FB27

```
#define CAN_F13R1_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.603 CAN_F13R1_FB28

```
#define CAN_F13R1_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.604 CAN_F13R1_FB29

```
#define CAN_F13R1_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.605 CAN_F13R1_FB3

```
#define CAN_F13R1_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.606 CAN_F13R1_FB30

```
#define CAN_F13R1_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.607 CAN_F13R1_FB31

```
#define CAN_F13R1_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.608 CAN_F13R1_FB4

```
#define CAN_F13R1_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.609 CAN_F13R1_FB5

```
#define CAN_F13R1_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.610 CAN_F13R1_FB6

```
#define CAN_F13R1_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.611 CAN_F13R1_FB7

```
#define CAN_F13R1_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.612 CAN_F13R1_FB8

```
#define CAN_F13R1_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.613 CAN_F13R1_FB9

```
#define CAN_F13R1_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.614 CAN_F13R2_FB0

```
#define CAN_F13R2_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.615 CAN_F13R2_FB1

```
#define CAN_F13R2_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.616 CAN_F13R2_FB10

```
#define CAN_F13R2_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.617 CAN_F13R2_FB11

```
#define CAN_F13R2_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.618 CAN_F13R2_FB12

```
#define CAN_F13R2_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.619 CAN_F13R2_FB13

```
#define CAN_F13R2_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.620 CAN_F13R2_FB14

```
#define CAN_F13R2_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.621 CAN_F13R2_FB15

```
#define CAN_F13R2_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.622 CAN_F13R2_FB16

```
#define CAN_F13R2_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.623 CAN_F13R2_FB17

```
#define CAN_F13R2_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.624 CAN_F13R2_FB18

```
#define CAN_F13R2_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.625 CAN_F13R2_FB19

```
#define CAN_F13R2_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.626 CAN_F13R2_FB2

```
#define CAN_F13R2_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.627 CAN_F13R2_FB20

```
#define CAN_F13R2_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.628 CAN_F13R2_FB21

```
#define CAN_F13R2_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.629 CAN_F13R2_FB22

```
#define CAN_F13R2_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.630 CAN_F13R2_FB23

```
#define CAN_F13R2_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.631 CAN_F13R2_FB24

```
#define CAN_F13R2_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.632 CAN_F13R2_FB25

```
#define CAN_F13R2_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.633 CAN_F13R2_FB26

```
#define CAN_F13R2_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.634 CAN_F13R2_FB27

```
#define CAN_F13R2_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.635 CAN_F13R2_FB28

```
#define CAN_F13R2_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.636 CAN_F13R2_FB29

```
#define CAN_F13R2_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.637 CAN_F13R2_FB3

```
#define CAN_F13R2_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.638 CAN_F13R2_FB30

```
#define CAN_F13R2_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.639 CAN_F13R2_FB31

```
#define CAN_F13R2_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.640 CAN_F13R2_FB4

```
#define CAN_F13R2_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.641 CAN_F13R2_FB5

```
#define CAN_F13R2_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.642 CAN_F13R2_FB6

```
#define CAN_F13R2_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.643 CAN_F13R2_FB7

```
#define CAN_F13R2_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.644 CAN_F13R2_FB8

```
#define CAN_F13R2_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.645 CAN_F13R2_FB9

```
#define CAN_F13R2_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.646 CAN_F1R1_FB0

```
#define CAN_F1R1_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.647 CAN_F1R1_FB1

```
#define CAN_F1R1_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.648 CAN_F1R1_FB10

```
#define CAN_F1R1_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.649 CAN_F1R1_FB11

```
#define CAN_F1R1_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.650 CAN_F1R1_FB12

```
#define CAN_F1R1_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.651 CAN_F1R1_FB13

```
#define CAN_F1R1_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.652 CAN_F1R1_FB14

```
#define CAN_F1R1_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.653 CAN_F1R1_FB15

```
#define CAN_F1R1_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.654 CAN_F1R1_FB16

```
#define CAN_F1R1_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.655 CAN_F1R1_FB17

```
#define CAN_F1R1_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.656 CAN_F1R1_FB18

```
#define CAN_F1R1_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.657 CAN_F1R1_FB19

```
#define CAN_F1R1_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.658 CAN_F1R1_FB2

```
#define CAN_F1R1_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.659 CAN_F1R1_FB20

```
#define CAN_F1R1_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.660 CAN_F1R1_FB21

```
#define CAN_F1R1_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.661 CAN_F1R1_FB22

```
#define CAN_F1R1_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.662 CAN_F1R1_FB23

```
#define CAN_F1R1_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.663 CAN_F1R1_FB24

```
#define CAN_F1R1_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.664 CAN_F1R1_FB25

```
#define CAN_F1R1_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.665 CAN_F1R1_FB26

```
#define CAN_F1R1_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.666 CAN_F1R1_FB27

```
#define CAN_F1R1_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.667 CAN_F1R1_FB28

```
#define CAN_F1R1_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.668 CAN_F1R1_FB29

```
#define CAN_F1R1_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.669 CAN_F1R1_FB3

```
#define CAN_F1R1_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.670 CAN_F1R1_FB30

```
#define CAN_F1R1_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.671 CAN_F1R1_FB31

```
#define CAN_F1R1_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.672 CAN_F1R1_FB4

```
#define CAN_F1R1_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.673 CAN_F1R1_FB5

```
#define CAN_F1R1_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.674 CAN_F1R1_FB6

```
#define CAN_F1R1_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.675 CAN_F1R1_FB7

```
#define CAN_F1R1_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.676 CAN_F1R1_FB8

```
#define CAN_F1R1_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.677 CAN_F1R1_FB9

```
#define CAN_F1R1_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.678 CAN_F1R2_FB0

```
#define CAN_F1R2_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.679 CAN_F1R2_FB1

```
#define CAN_F1R2_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.680 CAN_F1R2_FB10

```
#define CAN_F1R2_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.681 CAN_F1R2_FB11

```
#define CAN_F1R2_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.682 CAN_F1R2_FB12

```
#define CAN_F1R2_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.683 CAN_F1R2_FB13

```
#define CAN_F1R2_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.684 CAN_F1R2_FB14

```
#define CAN_F1R2_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.685 CAN_F1R2_FB15

```
#define CAN_F1R2_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.686 CAN_F1R2_FB16

```
#define CAN_F1R2_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.687 CAN_F1R2_FB17

```
#define CAN_F1R2_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.688 CAN_F1R2_FB18

```
#define CAN_F1R2_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.689 CAN_F1R2_FB19

```
#define CAN_F1R2_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.690 CAN_F1R2_FB2

```
#define CAN_F1R2_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.691 CAN_F1R2_FB20

```
#define CAN_F1R2_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.692 CAN_F1R2_FB21

```
#define CAN_F1R2_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.693 CAN_F1R2_FB22

```
#define CAN_F1R2_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.694 CAN_F1R2_FB23

```
#define CAN_F1R2_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.695 CAN_F1R2_FB24

```
#define CAN_F1R2_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.696 CAN_F1R2_FB25

```
#define CAN_F1R2_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.697 CAN_F1R2_FB26

```
#define CAN_F1R2_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.698 CAN_F1R2_FB27

```
#define CAN_F1R2_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.699 CAN_F1R2_FB28

```
#define CAN_F1R2_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.700 CAN_F1R2_FB29

```
#define CAN_F1R2_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.701 CAN_F1R2_FB3

```
#define CAN_F1R2_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.702 CAN_F1R2_FB30

```
#define CAN_F1R2_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.703 CAN_F1R2_FB31

```
#define CAN_F1R2_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.704 CAN_F1R2_FB4

```
#define CAN_F1R2_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.705 CAN_F1R2_FB5

```
#define CAN_F1R2_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.706 CAN_F1R2_FB6

```
#define CAN_F1R2_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.707 CAN_F1R2_FB7

```
#define CAN_F1R2_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.708 CAN_F1R2_FB8

```
#define CAN_F1R2_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.709 CAN_F1R2_FB9

```
#define CAN_F1R2_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.710 CAN_F2R1_FB0

```
#define CAN_F2R1_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.711 CAN_F2R1_FB1

```
#define CAN_F2R1_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.712 CAN_F2R1_FB10

```
#define CAN_F2R1_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.713 CAN_F2R1_FB11

```
#define CAN_F2R1_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.714 CAN_F2R1_FB12

```
#define CAN_F2R1_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.715 CAN_F2R1_FB13

```
#define CAN_F2R1_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.716 CAN_F2R1_FB14

```
#define CAN_F2R1_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.717 CAN_F2R1_FB15

```
#define CAN_F2R1_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.718 CAN_F2R1_FB16

```
#define CAN_F2R1_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.719 CAN_F2R1_FB17

```
#define CAN_F2R1_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.720 CAN_F2R1_FB18

```
#define CAN_F2R1_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.721 CAN_F2R1_FB19

```
#define CAN_F2R1_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.722 CAN_F2R1_FB2

```
#define CAN_F2R1_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.723 CAN_F2R1_FB20

```
#define CAN_F2R1_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.724 CAN_F2R1_FB21

```
#define CAN_F2R1_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.725 CAN_F2R1_FB22

```
#define CAN_F2R1_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.726 CAN_F2R1_FB23

```
#define CAN_F2R1_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.727 CAN_F2R1_FB24

```
#define CAN_F2R1_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.728 CAN_F2R1_FB25

```
#define CAN_F2R1_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.729 CAN_F2R1_FB26

```
#define CAN_F2R1_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.730 CAN_F2R1_FB27

```
#define CAN_F2R1_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.731 CAN_F2R1_FB28

```
#define CAN_F2R1_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.732 CAN_F2R1_FB29

```
#define CAN_F2R1_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.733 CAN_F2R1_FB3

```
#define CAN_F2R1_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.734 CAN_F2R1_FB30

```
#define CAN_F2R1_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.735 CAN_F2R1_FB31

```
#define CAN_F2R1_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.736 CAN_F2R1_FB4

```
#define CAN_F2R1_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.737 CAN_F2R1_FB5

```
#define CAN_F2R1_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.738 CAN_F2R1_FB6

```
#define CAN_F2R1_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.739 CAN_F2R1_FB7

```
#define CAN_F2R1_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.740 CAN_F2R1_FB8

```
#define CAN_F2R1_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.741 CAN_F2R1_FB9

```
#define CAN_F2R1_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.742 CAN_F2R2_FB0

```
#define CAN_F2R2_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.743 CAN_F2R2_FB1

```
#define CAN_F2R2_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.744 CAN_F2R2_FB10

```
#define CAN_F2R2_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.745 CAN_F2R2_FB11

```
#define CAN_F2R2_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.746 CAN_F2R2_FB12

```
#define CAN_F2R2_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.747 CAN_F2R2_FB13

```
#define CAN_F2R2_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.748 CAN_F2R2_FB14

```
#define CAN_F2R2_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.749 CAN_F2R2_FB15

```
#define CAN_F2R2_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.750 CAN_F2R2_FB16

```
#define CAN_F2R2_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.751 CAN_F2R2_FB17

```
#define CAN_F2R2_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.752 CAN_F2R2_FB18

```
#define CAN_F2R2_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.753 CAN_F2R2_FB19

```
#define CAN_F2R2_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.754 CAN_F2R2_FB2

```
#define CAN_F2R2_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.755 CAN_F2R2_FB20

```
#define CAN_F2R2_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.756 CAN_F2R2_FB21

```
#define CAN_F2R2_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.757 CAN_F2R2_FB22

```
#define CAN_F2R2_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.758 CAN_F2R2_FB23

```
#define CAN_F2R2_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.759 CAN_F2R2_FB24

```
#define CAN_F2R2_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.760 CAN_F2R2_FB25

```
#define CAN_F2R2_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.761 CAN_F2R2_FB26

```
#define CAN_F2R2_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.762 CAN_F2R2_FB27

```
#define CAN_F2R2_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.763 CAN_F2R2_FB28

```
#define CAN_F2R2_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.764 CAN_F2R2_FB29

```
#define CAN_F2R2_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.765 CAN_F2R2_FB3

```
#define CAN_F2R2_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.766 CAN_F2R2_FB30

```
#define CAN_F2R2_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.767 CAN_F2R2_FB31

```
#define CAN_F2R2_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.768 CAN_F2R2_FB4

```
#define CAN_F2R2_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.769 CAN_F2R2_FB5

```
#define CAN_F2R2_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.770 CAN_F2R2_FB6

```
#define CAN_F2R2_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.771 CAN_F2R2_FB7

```
#define CAN_F2R2_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.772 CAN_F2R2_FB8

```
#define CAN_F2R2_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.773 CAN_F2R2_FB9

```
#define CAN_F2R2_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.774 CAN_F3R1_FB0

```
#define CAN_F3R1_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.775 CAN_F3R1_FB1

```
#define CAN_F3R1_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.776 CAN_F3R1_FB10

```
#define CAN_F3R1_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.777 CAN_F3R1_FB11

```
#define CAN_F3R1_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.778 CAN_F3R1_FB12

```
#define CAN_F3R1_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.779 CAN_F3R1_FB13

```
#define CAN_F3R1_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.780 CAN_F3R1_FB14

```
#define CAN_F3R1_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.781 CAN_F3R1_FB15

```
#define CAN_F3R1_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.782 CAN_F3R1_FB16

```
#define CAN_F3R1_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.783 CAN_F3R1_FB17

```
#define CAN_F3R1_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.784 CAN_F3R1_FB18

```
#define CAN_F3R1_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.785 CAN_F3R1_FB19

```
#define CAN_F3R1_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.786 CAN_F3R1_FB2

```
#define CAN_F3R1_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.787 CAN_F3R1_FB20

```
#define CAN_F3R1_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.788 CAN_F3R1_FB21

```
#define CAN_F3R1_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.789 CAN_F3R1_FB22

```
#define CAN_F3R1_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.790 CAN_F3R1_FB23

```
#define CAN_F3R1_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.791 CAN_F3R1_FB24

```
#define CAN_F3R1_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.792 CAN_F3R1_FB25

```
#define CAN_F3R1_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.793 CAN_F3R1_FB26

```
#define CAN_F3R1_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.794 CAN_F3R1_FB27

```
#define CAN_F3R1_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.795 CAN_F3R1_FB28

```
#define CAN_F3R1_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.796 CAN_F3R1_FB29

```
#define CAN_F3R1_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.797 CAN_F3R1_FB3

```
#define CAN_F3R1_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.798 CAN_F3R1_FB30

```
#define CAN_F3R1_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.799 CAN_F3R1_FB31

```
#define CAN_F3R1_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.800 CAN_F3R1_FB4

```
#define CAN_F3R1_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.801 CAN_F3R1_FB5

```
#define CAN_F3R1_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.802 CAN_F3R1_FB6

```
#define CAN_F3R1_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.803 CAN_F3R1_FB7

```
#define CAN_F3R1_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.804 CAN_F3R1_FB8

```
#define CAN_F3R1_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.805 CAN_F3R1_FB9

```
#define CAN_F3R1_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.806 CAN_F3R2_FB0

```
#define CAN_F3R2_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.807 CAN_F3R2_FB1

```
#define CAN_F3R2_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.808 CAN_F3R2_FB10

```
#define CAN_F3R2_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.809 CAN_F3R2_FB11

```
#define CAN_F3R2_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.810 CAN_F3R2_FB12

```
#define CAN_F3R2_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.811 CAN_F3R2_FB13

```
#define CAN_F3R2_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.812 CAN_F3R2_FB14

```
#define CAN_F3R2_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.813 CAN_F3R2_FB15

```
#define CAN_F3R2_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.814 CAN_F3R2_FB16

```
#define CAN_F3R2_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.815 CAN_F3R2_FB17

```
#define CAN_F3R2_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.816 CAN_F3R2_FB18

```
#define CAN_F3R2_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.817 CAN_F3R2_FB19

```
#define CAN_F3R2_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.818 CAN_F3R2_FB2

```
#define CAN_F3R2_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.819 CAN_F3R2_FB20

```
#define CAN_F3R2_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.820 CAN_F3R2_FB21

```
#define CAN_F3R2_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.821 CAN_F3R2_FB22

```
#define CAN_F3R2_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.822 CAN_F3R2_FB23

```
#define CAN_F3R2_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.823 CAN_F3R2_FB24

```
#define CAN_F3R2_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.824 CAN_F3R2_FB25

```
#define CAN_F3R2_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.825 CAN_F3R2_FB26

```
#define CAN_F3R2_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.826 CAN_F3R2_FB27

```
#define CAN_F3R2_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.827 CAN_F3R2_FB28

```
#define CAN_F3R2_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.828 CAN_F3R2_FB29

```
#define CAN_F3R2_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.829 CAN_F3R2_FB3

```
#define CAN_F3R2_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.830 CAN_F3R2_FB30

```
#define CAN_F3R2_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.831 CAN_F3R2_FB31

```
#define CAN_F3R2_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.832 CAN_F3R2_FB4

```
#define CAN_F3R2_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.833 CAN_F3R2_FB5

```
#define CAN_F3R2_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.834 CAN_F3R2_FB6

```
#define CAN_F3R2_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.835 CAN_F3R2_FB7

```
#define CAN_F3R2_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.836 CAN_F3R2_FB8

```
#define CAN_F3R2_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.837 CAN_F3R2_FB9

```
#define CAN_F3R2_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.838 CAN_F4R1_FB0

```
#define CAN_F4R1_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.839 CAN_F4R1_FB1

```
#define CAN_F4R1_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.840 CAN_F4R1_FB10

```
#define CAN_F4R1_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.841 CAN_F4R1_FB11

```
#define CAN_F4R1_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.842 CAN_F4R1_FB12

```
#define CAN_F4R1_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.843 CAN_F4R1_FB13

```
#define CAN_F4R1_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.844 CAN_F4R1_FB14

```
#define CAN_F4R1_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.845 CAN_F4R1_FB15

```
#define CAN_F4R1_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.846 CAN_F4R1_FB16

```
#define CAN_F4R1_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.847 CAN_F4R1_FB17

```
#define CAN_F4R1_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.848 CAN_F4R1_FB18

```
#define CAN_F4R1_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.849 CAN_F4R1_FB19

```
#define CAN_F4R1_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.850 CAN_F4R1_FB2

```
#define CAN_F4R1_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.851 CAN_F4R1_FB20

```
#define CAN_F4R1_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.852 CAN_F4R1_FB21

```
#define CAN_F4R1_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.853 CAN_F4R1_FB22

```
#define CAN_F4R1_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.854 CAN_F4R1_FB23

```
#define CAN_F4R1_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.855 CAN_F4R1_FB24

```
#define CAN_F4R1_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.856 CAN_F4R1_FB25

```
#define CAN_F4R1_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.857 CAN_F4R1_FB26

```
#define CAN_F4R1_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.858 CAN_F4R1_FB27

```
#define CAN_F4R1_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.859 CAN_F4R1_FB28

```
#define CAN_F4R1_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.860 CAN_F4R1_FB29

```
#define CAN_F4R1_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.861 CAN_F4R1_FB3

```
#define CAN_F4R1_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.862 CAN_F4R1_FB30

```
#define CAN_F4R1_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.863 CAN_F4R1_FB31

```
#define CAN_F4R1_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.864 CAN_F4R1_FB4

```
#define CAN_F4R1_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.865 CAN_F4R1_FB5

```
#define CAN_F4R1_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.866 CAN_F4R1_FB6

```
#define CAN_F4R1_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.867 CAN_F4R1_FB7

```
#define CAN_F4R1_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.868 CAN_F4R1_FB8

```
#define CAN_F4R1_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.869 CAN_F4R1_FB9

```
#define CAN_F4R1_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.870 CAN_F4R2_FB0

```
#define CAN_F4R2_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.871 CAN_F4R2_FB1

```
#define CAN_F4R2_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.872 CAN_F4R2_FB10

```
#define CAN_F4R2_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.873 CAN_F4R2_FB11

```
#define CAN_F4R2_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.874 CAN_F4R2_FB12

```
#define CAN_F4R2_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.875 CAN_F4R2_FB13

```
#define CAN_F4R2_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.876 CAN_F4R2_FB14

```
#define CAN_F4R2_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.877 CAN_F4R2_FB15

```
#define CAN_F4R2_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.878 CAN_F4R2_FB16

```
#define CAN_F4R2_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.879 CAN_F4R2_FB17

```
#define CAN_F4R2_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.880 CAN_F4R2_FB18

```
#define CAN_F4R2_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.881 CAN_F4R2_FB19

```
#define CAN_F4R2_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.882 CAN_F4R2_FB2

```
#define CAN_F4R2_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.883 CAN_F4R2_FB20

```
#define CAN_F4R2_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.884 CAN_F4R2_FB21

```
#define CAN_F4R2_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.885 CAN_F4R2_FB22

```
#define CAN_F4R2_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.886 CAN_F4R2_FB23

```
#define CAN_F4R2_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.887 CAN_F4R2_FB24

```
#define CAN_F4R2_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.888 CAN_F4R2_FB25

```
#define CAN_F4R2_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.889 CAN_F4R2_FB26

```
#define CAN_F4R2_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.890 CAN_F4R2_FB27

```
#define CAN_F4R2_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.891 CAN_F4R2_FB28

```
#define CAN_F4R2_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.892 CAN_F4R2_FB29

```
#define CAN_F4R2_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.893 CAN_F4R2_FB3

```
#define CAN_F4R2_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.894 CAN_F4R2_FB30

```
#define CAN_F4R2_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.895 CAN_F4R2_FB31

```
#define CAN_F4R2_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.896 CAN_F4R2_FB4

```
#define CAN_F4R2_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.897 CAN_F4R2_FB5

```
#define CAN_F4R2_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.898 CAN_F4R2_FB6

```
#define CAN_F4R2_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.899 CAN_F4R2_FB7

```
#define CAN_F4R2_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.900 CAN_F4R2_FB8

```
#define CAN_F4R2_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.901 CAN_F4R2_FB9

```
#define CAN_F4R2_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.902 CAN_F5R1_FB0

```
#define CAN_F5R1_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.903 CAN_F5R1_FB1

```
#define CAN_F5R1_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.904 CAN_F5R1_FB10

```
#define CAN_F5R1_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.905 CAN_F5R1_FB11

```
#define CAN_F5R1_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.906 CAN_F5R1_FB12

```
#define CAN_F5R1_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.907 CAN_F5R1_FB13

```
#define CAN_F5R1_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.908 CAN_F5R1_FB14

```
#define CAN_F5R1_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.909 CAN_F5R1_FB15

```
#define CAN_F5R1_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.910 CAN_F5R1_FB16

```
#define CAN_F5R1_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.911 CAN_F5R1_FB17

```
#define CAN_F5R1_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.912 CAN_F5R1_FB18

```
#define CAN_F5R1_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.913 CAN_F5R1_FB19

```
#define CAN_F5R1_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.914 CAN_F5R1_FB2

```
#define CAN_F5R1_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.915 CAN_F5R1_FB20

```
#define CAN_F5R1_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.916 CAN_F5R1_FB21

```
#define CAN_F5R1_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.917 CAN_F5R1_FB22

```
#define CAN_F5R1_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.918 CAN_F5R1_FB23

```
#define CAN_F5R1_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.919 CAN_F5R1_FB24

```
#define CAN_F5R1_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.920 CAN_F5R1_FB25

```
#define CAN_F5R1_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.921 CAN_F5R1_FB26

```
#define CAN_F5R1_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.922 CAN_F5R1_FB27

```
#define CAN_F5R1_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.923 CAN_F5R1_FB28

```
#define CAN_F5R1_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.924 CAN_F5R1_FB29

```
#define CAN_F5R1_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.925 CAN_F5R1_FB3

```
#define CAN_F5R1_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.926 CAN_F5R1_FB30

```
#define CAN_F5R1_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.927 CAN_F5R1_FB31

```
#define CAN_F5R1_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.928 CAN_F5R1_FB4

```
#define CAN_F5R1_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.929 CAN_F5R1_FB5

```
#define CAN_F5R1_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.930 CAN_F5R1_FB6

```
#define CAN_F5R1_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.931 CAN_F5R1_FB7

```
#define CAN_F5R1_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.932 CAN_F5R1_FB8

```
#define CAN_F5R1_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.933 CAN_F5R1_FB9

```
#define CAN_F5R1_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.934 CAN_F5R2_FB0

```
#define CAN_F5R2_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.935 CAN_F5R2_FB1

```
#define CAN_F5R2_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.936 CAN_F5R2_FB10

```
#define CAN_F5R2_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.937 CAN_F5R2_FB11

```
#define CAN_F5R2_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.938 CAN_F5R2_FB12

```
#define CAN_F5R2_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.939 CAN_F5R2_FB13

```
#define CAN_F5R2_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.940 CAN_F5R2_FB14

```
#define CAN_F5R2_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.941 CAN_F5R2_FB15

```
#define CAN_F5R2_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.942 CAN_F5R2_FB16

```
#define CAN_F5R2_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.943 CAN_F5R2_FB17

```
#define CAN_F5R2_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.944 CAN_F5R2_FB18

```
#define CAN_F5R2_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.945 CAN_F5R2_FB19

```
#define CAN_F5R2_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.946 CAN_F5R2_FB2

```
#define CAN_F5R2_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.947 CAN_F5R2_FB20

```
#define CAN_F5R2_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.948 CAN_F5R2_FB21

```
#define CAN_F5R2_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.949 CAN_F5R2_FB22

```
#define CAN_F5R2_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.950 CAN_F5R2_FB23

```
#define CAN_F5R2_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.951 CAN_F5R2_FB24

```
#define CAN_F5R2_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.952 CAN_F5R2_FB25

```
#define CAN_F5R2_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.953 CAN_F5R2_FB26

```
#define CAN_F5R2_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.954 CAN_F5R2_FB27

```
#define CAN_F5R2_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.955 CAN_F5R2_FB28

```
#define CAN_F5R2_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.956 CAN_F5R2_FB29

```
#define CAN_F5R2_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.957 CAN_F5R2_FB3

```
#define CAN_F5R2_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.958 CAN_F5R2_FB30

```
#define CAN_F5R2_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.959 CAN_F5R2_FB31

```
#define CAN_F5R2_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.960 CAN_F5R2_FB4

```
#define CAN_F5R2_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.961 CAN_F5R2_FB5

```
#define CAN_F5R2_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.962 CAN_F5R2_FB6

```
#define CAN_F5R2_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.963 CAN_F5R2_FB7

```
#define CAN_F5R2_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.964 CAN_F5R2_FB8

```
#define CAN_F5R2_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.965 CAN_F5R2_FB9

```
#define CAN_F5R2_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.966 CAN_F6R1_FB0

```
#define CAN_F6R1_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.967 CAN_F6R1_FB1

```
#define CAN_F6R1_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.968 CAN_F6R1_FB10

```
#define CAN_F6R1_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.969 CAN_F6R1_FB11

```
#define CAN_F6R1_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.970 CAN_F6R1_FB12

```
#define CAN_F6R1_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.971 CAN_F6R1_FB13

```
#define CAN_F6R1_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.972 CAN_F6R1_FB14

```
#define CAN_F6R1_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.973 CAN_F6R1_FB15

```
#define CAN_F6R1_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.974 CAN_F6R1_FB16

```
#define CAN_F6R1_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.975 CAN_F6R1_FB17

```
#define CAN_F6R1_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.976 CAN_F6R1_FB18

```
#define CAN_F6R1_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.977 CAN_F6R1_FB19

```
#define CAN_F6R1_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.978 CAN_F6R1_FB2

```
#define CAN_F6R1_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.979 CAN_F6R1_FB20

```
#define CAN_F6R1_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.980 CAN_F6R1_FB21

```
#define CAN_F6R1_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.981 CAN_F6R1_FB22

```
#define CAN_F6R1_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.982 CAN_F6R1_FB23

```
#define CAN_F6R1_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.983 CAN_F6R1_FB24

```
#define CAN_F6R1_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.984 CAN_F6R1_FB25

```
#define CAN_F6R1_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.985 CAN_F6R1_FB26

```
#define CAN_F6R1_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.986 CAN_F6R1_FB27

```
#define CAN_F6R1_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.987 CAN_F6R1_FB28

```
#define CAN_F6R1_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.988 CAN_F6R1_FB29

```
#define CAN_F6R1_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.989 CAN_F6R1_FB3

```
#define CAN_F6R1_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.990 CAN_F6R1_FB30

```
#define CAN_F6R1_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.991 CAN_F6R1_FB31

```
#define CAN_F6R1_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.992 CAN_F6R1_FB4

```
#define CAN_F6R1_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.993 CAN_F6R1_FB5

```
#define CAN_F6R1_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.994 CAN_F6R1_FB6

```
#define CAN_F6R1_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.995 CAN_F6R1_FB7

```
#define CAN_F6R1_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.996 CAN_F6R1_FB8

```
#define CAN_F6R1_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.997 CAN_F6R1_FB9

```
#define CAN_F6R1_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.998 CAN_F6R2_FB0

```
#define CAN_F6R2_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.999 CAN_F6R2_FB1

```
#define CAN_F6R2_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.1000 CAN_F6R2_FB10

```
#define CAN_F6R2_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.1001 CAN_F6R2_FB11

```
#define CAN_F6R2_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.1002 CAN_F6R2_FB12

```
#define CAN_F6R2_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.1003 CAN_F6R2_FB13

```
#define CAN_F6R2_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.1004 CAN_F6R2_FB14

```
#define CAN_F6R2_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.1005 CAN_F6R2_FB15

```
#define CAN_F6R2_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.1006 CAN_F6R2_FB16

```
#define CAN_F6R2_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.1007 CAN_F6R2_FB17

```
#define CAN_F6R2_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.1008 CAN_F6R2_FB18

```
#define CAN_F6R2_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.1009 CAN_F6R2_FB19

```
#define CAN_F6R2_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.1010 CAN_F6R2_FB2

```
#define CAN_F6R2_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.1011 CAN_F6R2_FB20

```
#define CAN_F6R2_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.1012 CAN_F6R2_FB21

```
#define CAN_F6R2_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.1013 CAN_F6R2_FB22

```
#define CAN_F6R2_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.1014 CAN_F6R2_FB23

```
#define CAN_F6R2_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.1015 CAN_F6R2_FB24

```
#define CAN_F6R2_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.1016 CAN_F6R2_FB25

```
#define CAN_F6R2_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.1017 CAN_F6R2_FB26

```
#define CAN_F6R2_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.1018 CAN_F6R2_FB27

```
#define CAN_F6R2_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.1019 CAN_F6R2_FB28

```
#define CAN_F6R2_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.1020 CAN_F6R2_FB29

```
#define CAN_F6R2_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.1021 CAN_F6R2_FB3

```
#define CAN_F6R2_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.1022 CAN_F6R2_FB30

```
#define CAN_F6R2_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.1023 CAN_F6R2_FB31

```
#define CAN_F6R2_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.1024 CAN_F6R2_FB4

```
#define CAN_F6R2_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.1025 CAN_F6R2_FB5

```
#define CAN_F6R2_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.1026 CAN_F6R2_FB6

```
#define CAN_F6R2_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.1027 CAN_F6R2_FB7

```
#define CAN_F6R2_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.1028 CAN_F6R2_FB8

```
#define CAN_F6R2_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.1029 CAN_F6R2_FB9

```
#define CAN_F6R2_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.1030 CAN_F7R1_FB0

```
#define CAN_F7R1_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.1031 CAN_F7R1_FB1

```
#define CAN_F7R1_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.1032 CAN_F7R1_FB10

```
#define CAN_F7R1_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.1033 CAN_F7R1_FB11

```
#define CAN_F7R1_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.1034 CAN_F7R1_FB12

```
#define CAN_F7R1_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.1035 CAN_F7R1_FB13

```
#define CAN_F7R1_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.1036 CAN_F7R1_FB14

```
#define CAN_F7R1_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.1037 CAN_F7R1_FB15

```
#define CAN_F7R1_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.1038 CAN_F7R1_FB16

```
#define CAN_F7R1_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.1039 CAN_F7R1_FB17

```
#define CAN_F7R1_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.1040 CAN_F7R1_FB18

```
#define CAN_F7R1_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.1041 CAN_F7R1_FB19

```
#define CAN_F7R1_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.1042 CAN_F7R1_FB2

```
#define CAN_F7R1_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.1043 CAN_F7R1_FB20

```
#define CAN_F7R1_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.1044 CAN_F7R1_FB21

```
#define CAN_F7R1_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.1045 CAN_F7R1_FB22

```
#define CAN_F7R1_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.1046 CAN_F7R1_FB23

```
#define CAN_F7R1_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.1047 CAN_F7R1_FB24

```
#define CAN_F7R1_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.1048 CAN_F7R1_FB25

```
#define CAN_F7R1_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.1049 CAN_F7R1_FB26

```
#define CAN_F7R1_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.1050 CAN_F7R1_FB27

```
#define CAN_F7R1_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.1051 CAN_F7R1_FB28

```
#define CAN_F7R1_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.1052 CAN_F7R1_FB29

```
#define CAN_F7R1_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.1053 CAN_F7R1_FB3

```
#define CAN_F7R1_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.1054 CAN_F7R1_FB30

```
#define CAN_F7R1_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.1055 CAN_F7R1_FB31

```
#define CAN_F7R1_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.1056 CAN_F7R1_FB4

```
#define CAN_F7R1_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.1057 CAN_F7R1_FB5

```
#define CAN_F7R1_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.1058 CAN_F7R1_FB6

```
#define CAN_F7R1_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.1059 CAN_F7R1_FB7

```
#define CAN_F7R1_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.1060 CAN_F7R1_FB8

```
#define CAN_F7R1_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.1061 CAN_F7R1_FB9

```
#define CAN_F7R1_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.1062 CAN_F7R2_FB0

```
#define CAN_F7R2_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.1063 CAN_F7R2_FB1

```
#define CAN_F7R2_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.1064 CAN_F7R2_FB10

```
#define CAN_F7R2_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.1065 CAN_F7R2_FB11

```
#define CAN_F7R2_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.1066 CAN_F7R2_FB12

```
#define CAN_F7R2_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.1067 CAN_F7R2_FB13

```
#define CAN_F7R2_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.1068 CAN_F7R2_FB14

```
#define CAN_F7R2_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.1069 CAN_F7R2_FB15

```
#define CAN_F7R2_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.1070 CAN_F7R2_FB16

```
#define CAN_F7R2_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.1071 CAN_F7R2_FB17

```
#define CAN_F7R2_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.1072 CAN_F7R2_FB18

```
#define CAN_F7R2_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.1073 CAN_F7R2_FB19

```
#define CAN_F7R2_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.1074 CAN_F7R2_FB2

```
#define CAN_F7R2_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.1075 CAN_F7R2_FB20

```
#define CAN_F7R2_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.1076 CAN_F7R2_FB21

```
#define CAN_F7R2_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.1077 CAN_F7R2_FB22

```
#define CAN_F7R2_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.1078 CAN_F7R2_FB23

```
#define CAN_F7R2_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.1079 CAN_F7R2_FB24

```
#define CAN_F7R2_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.1080 CAN_F7R2_FB25

```
#define CAN_F7R2_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.1081 CAN_F7R2_FB26

```
#define CAN_F7R2_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.1082 CAN_F7R2_FB27

```
#define CAN_F7R2_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.1083 CAN_F7R2_FB28

```
#define CAN_F7R2_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.1084 CAN_F7R2_FB29

```
#define CAN_F7R2_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.1085 CAN_F7R2_FB3

```
#define CAN_F7R2_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.1086 CAN_F7R2_FB30

```
#define CAN_F7R2_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.1087 CAN_F7R2_FB31

```
#define CAN_F7R2_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.1088 CAN_F7R2_FB4

```
#define CAN_F7R2_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.1089 CAN_F7R2_FB5

```
#define CAN_F7R2_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.1090 CAN_F7R2_FB6

```
#define CAN_F7R2_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.1091 CAN_F7R2_FB7

```
#define CAN_F7R2_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.1092 CAN_F7R2_FB8

```
#define CAN_F7R2_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.1093 CAN_F7R2_FB9

```
#define CAN_F7R2_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.1094 CAN_F8R1_FB0

```
#define CAN_F8R1_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.1095 CAN_F8R1_FB1

```
#define CAN_F8R1_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.1096 CAN_F8R1_FB10

```
#define CAN_F8R1_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.1097 CAN_F8R1_FB11

```
#define CAN_F8R1_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.1098 CAN_F8R1_FB12

```
#define CAN_F8R1_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.1099 CAN_F8R1_FB13

```
#define CAN_F8R1_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.1100 CAN_F8R1_FB14

```
#define CAN_F8R1_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.1101 CAN_F8R1_FB15

```
#define CAN_F8R1_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.1102 CAN_F8R1_FB16

```
#define CAN_F8R1_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.1103 CAN_F8R1_FB17

```
#define CAN_F8R1_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.1104 CAN_F8R1_FB18

```
#define CAN_F8R1_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.1105 CAN_F8R1_FB19

```
#define CAN_F8R1_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.1106 CAN_F8R1_FB2

```
#define CAN_F8R1_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.1107 CAN_F8R1_FB20

```
#define CAN_F8R1_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.1108 CAN_F8R1_FB21

```
#define CAN_F8R1_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.1109 CAN_F8R1_FB22

```
#define CAN_F8R1_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.1110 CAN_F8R1_FB23

```
#define CAN_F8R1_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.1111 CAN_F8R1_FB24

```
#define CAN_F8R1_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.1112 CAN_F8R1_FB25

```
#define CAN_F8R1_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.1113 CAN_F8R1_FB26

```
#define CAN_F8R1_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.1114 CAN_F8R1_FB27

```
#define CAN_F8R1_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.1115 CAN_F8R1_FB28

```
#define CAN_F8R1_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.1116 CAN_F8R1_FB29

```
#define CAN_F8R1_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.1117 CAN_F8R1_FB3

```
#define CAN_F8R1_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.1118 CAN_F8R1_FB30

```
#define CAN_F8R1_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.1119 CAN_F8R1_FB31

```
#define CAN_F8R1_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.1120 CAN_F8R1_FB4

```
#define CAN_F8R1_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.1121 CAN_F8R1_FB5

```
#define CAN_F8R1_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.1122 CAN_F8R1_FB6

```
#define CAN_F8R1_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.1123 CAN_F8R1_FB7

```
#define CAN_F8R1_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.1124 CAN_F8R1_FB8

```
#define CAN_F8R1_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.1125 CAN_F8R1_FB9

```
#define CAN_F8R1_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.1126 CAN_F8R2_FB0

```
#define CAN_F8R2_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.1127 CAN_F8R2_FB1

```
#define CAN_F8R2_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.1128 CAN_F8R2_FB10

```
#define CAN_F8R2_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.1129 CAN_F8R2_FB11

```
#define CAN_F8R2_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.1130 CAN_F8R2_FB12

```
#define CAN_F8R2_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.1131 CAN_F8R2_FB13

```
#define CAN_F8R2_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.1132 CAN_F8R2_FB14

```
#define CAN_F8R2_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.1133 CAN_F8R2_FB15

```
#define CAN_F8R2_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.1134 CAN_F8R2_FB16

```
#define CAN_F8R2_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.1135 CAN_F8R2_FB17

```
#define CAN_F8R2_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.1136 CAN_F8R2_FB18

```
#define CAN_F8R2_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.1137 CAN_F8R2_FB19

```
#define CAN_F8R2_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.1138 CAN_F8R2_FB2

```
#define CAN_F8R2_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.1139 CAN_F8R2_FB20

```
#define CAN_F8R2_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.1140 CAN_F8R2_FB21

```
#define CAN_F8R2_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.1141 CAN_F8R2_FB22

```
#define CAN_F8R2_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.1142 CAN_F8R2_FB23

```
#define CAN_F8R2_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.1143 CAN_F8R2_FB24

```
#define CAN_F8R2_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.1144 CAN_F8R2_FB25

```
#define CAN_F8R2_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.1145 CAN_F8R2_FB26

```
#define CAN_F8R2_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.1146 CAN_F8R2_FB27

```
#define CAN_F8R2_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.1147 CAN_F8R2_FB28

```
#define CAN_F8R2_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.1148 CAN_F8R2_FB29

```
#define CAN_F8R2_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.1149 CAN_F8R2_FB3

```
#define CAN_F8R2_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.1150 CAN_F8R2_FB30

```
#define CAN_F8R2_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.1151 CAN_F8R2_FB31

```
#define CAN_F8R2_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.1152 CAN_F8R2_FB4

```
#define CAN_F8R2_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.1153 CAN_F8R2_FB5

```
#define CAN_F8R2_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.1154 CAN_F8R2_FB6

```
#define CAN_F8R2_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.1155 CAN_F8R2_FB7

```
#define CAN_F8R2_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.1156 CAN_F8R2_FB8

```
#define CAN_F8R2_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.1157 CAN_F8R2_FB9

```
#define CAN_F8R2_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.1158 CAN_F9R1_FB0

```
#define CAN_F9R1_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.1159 CAN_F9R1_FB1

```
#define CAN_F9R1_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.1160 CAN_F9R1_FB10

```
#define CAN_F9R1_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.1161 CAN_F9R1_FB11

```
#define CAN_F9R1_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.1162 CAN_F9R1_FB12

```
#define CAN_F9R1_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.1163 CAN_F9R1_FB13

```
#define CAN_F9R1_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.1164 CAN_F9R1_FB14

```
#define CAN_F9R1_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.1165 CAN_F9R1_FB15

```
#define CAN_F9R1_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.1166 CAN_F9R1_FB16

```
#define CAN_F9R1_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.1167 CAN_F9R1_FB17

```
#define CAN_F9R1_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.1168 CAN_F9R1_FB18

```
#define CAN_F9R1_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.1169 CAN_F9R1_FB19

```
#define CAN_F9R1_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.1170 CAN_F9R1_FB2

```
#define CAN_F9R1_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.1171 CAN_F9R1_FB20

```
#define CAN_F9R1_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.1172 CAN_F9R1_FB21

```
#define CAN_F9R1_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.1173 CAN_F9R1_FB22

```
#define CAN_F9R1_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.1174 CAN_F9R1_FB23

```
#define CAN_F9R1_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.1175 CAN_F9R1_FB24

```
#define CAN_F9R1_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.1176 CAN_F9R1_FB25

```
#define CAN_F9R1_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.1177 CAN_F9R1_FB26

```
#define CAN_F9R1_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.1178 CAN_F9R1_FB27

```
#define CAN_F9R1_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.1179 CAN_F9R1_FB28

```
#define CAN_F9R1_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.1180 CAN_F9R1_FB29

```
#define CAN_F9R1_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.1181 CAN_F9R1_FB3

```
#define CAN_F9R1_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.1182 CAN_F9R1_FB30

```
#define CAN_F9R1_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.1183 CAN_F9R1_FB31

```
#define CAN_F9R1_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.1184 CAN_F9R1_FB4

```
#define CAN_F9R1_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.1185 CAN_F9R1_FB5

```
#define CAN_F9R1_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.1186 CAN_F9R1_FB6

```
#define CAN_F9R1_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.1187 CAN_F9R1_FB7

```
#define CAN_F9R1_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.1188 CAN_F9R1_FB8

```
#define CAN_F9R1_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.1189 CAN_F9R1_FB9

```
#define CAN_F9R1_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.1190 CAN_F9R2_FB0

```
#define CAN_F9R2_FB0 ((uint32_t)0x00000001)
```

Filter bit 0

5.154.2.1191 CAN_F9R2_FB1

```
#define CAN_F9R2_FB1 ((uint32_t)0x00000002)
```

Filter bit 1

5.154.2.1192 CAN_F9R2_FB10

```
#define CAN_F9R2_FB10 ((uint32_t)0x00000400)
```

Filter bit 10

5.154.2.1193 CAN_F9R2_FB11

```
#define CAN_F9R2_FB11 ((uint32_t)0x00000800)
```

Filter bit 11

5.154.2.1194 CAN_F9R2_FB12

```
#define CAN_F9R2_FB12 ((uint32_t)0x00001000)
```

Filter bit 12

5.154.2.1195 CAN_F9R2_FB13

```
#define CAN_F9R2_FB13 ((uint32_t)0x00002000)
```

Filter bit 13

5.154.2.1196 CAN_F9R2_FB14

```
#define CAN_F9R2_FB14 ((uint32_t)0x00004000)
```

Filter bit 14

5.154.2.1197 CAN_F9R2_FB15

```
#define CAN_F9R2_FB15 ((uint32_t)0x00008000)
```

Filter bit 15

5.154.2.1198 CAN_F9R2_FB16

```
#define CAN_F9R2_FB16 ((uint32_t)0x00010000)
```

Filter bit 16

5.154.2.1199 CAN_F9R2_FB17

```
#define CAN_F9R2_FB17 ((uint32_t)0x00020000)
```

Filter bit 17

5.154.2.1200 CAN_F9R2_FB18

```
#define CAN_F9R2_FB18 ((uint32_t)0x00040000)
```

Filter bit 18

5.154.2.1201 CAN_F9R2_FB19

```
#define CAN_F9R2_FB19 ((uint32_t)0x00080000)
```

Filter bit 19

5.154.2.1202 CAN_F9R2_FB2

```
#define CAN_F9R2_FB2 ((uint32_t)0x00000004)
```

Filter bit 2

5.154.2.1203 CAN_F9R2_FB20

```
#define CAN_F9R2_FB20 ((uint32_t)0x00100000)
```

Filter bit 20

5.154.2.1204 CAN_F9R2_FB21

```
#define CAN_F9R2_FB21 ((uint32_t)0x00200000)
```

Filter bit 21

5.154.2.1205 CAN_F9R2_FB22

```
#define CAN_F9R2_FB22 ((uint32_t)0x00400000)
```

Filter bit 22

5.154.2.1206 CAN_F9R2_FB23

```
#define CAN_F9R2_FB23 ((uint32_t)0x00800000)
```

Filter bit 23

5.154.2.1207 CAN_F9R2_FB24

```
#define CAN_F9R2_FB24 ((uint32_t)0x01000000)
```

Filter bit 24

5.154.2.1208 CAN_F9R2_FB25

```
#define CAN_F9R2_FB25 ((uint32_t)0x02000000)
```

Filter bit 25

5.154.2.1209 CAN_F9R2_FB26

```
#define CAN_F9R2_FB26 ((uint32_t)0x04000000)
```

Filter bit 26

5.154.2.1210 CAN_F9R2_FB27

```
#define CAN_F9R2_FB27 ((uint32_t)0x08000000)
```

Filter bit 27

5.154.2.1211 CAN_F9R2_FB28

```
#define CAN_F9R2_FB28 ((uint32_t)0x10000000)
```

Filter bit 28

5.154.2.1212 CAN_F9R2_FB29

```
#define CAN_F9R2_FB29 ((uint32_t)0x20000000)
```

Filter bit 29

5.154.2.1213 CAN_F9R2_FB3

```
#define CAN_F9R2_FB3 ((uint32_t)0x00000008)
```

Filter bit 3

5.154.2.1214 CAN_F9R2_FB30

```
#define CAN_F9R2_FB30 ((uint32_t)0x40000000)
```

Filter bit 30

5.154.2.1215 CAN_F9R2_FB31

```
#define CAN_F9R2_FB31 ((uint32_t)0x80000000)
```

Filter bit 31

5.154.2.1216 CAN_F9R2_FB4

```
#define CAN_F9R2_FB4 ((uint32_t)0x00000010)
```

Filter bit 4

5.154.2.1217 CAN_F9R2_FB5

```
#define CAN_F9R2_FB5 ((uint32_t)0x00000020)
```

Filter bit 5

5.154.2.1218 CAN_F9R2_FB6

```
#define CAN_F9R2_FB6 ((uint32_t)0x00000040)
```

Filter bit 6

5.154.2.1219 CAN_F9R2_FB7

```
#define CAN_F9R2_FB7 ((uint32_t)0x00000080)
```

Filter bit 7

5.154.2.1220 CAN_F9R2_FB8

```
#define CAN_F9R2_FB8 ((uint32_t)0x00000100)
```

Filter bit 8

5.154.2.1221 CAN_F9R2_FB9

```
#define CAN_F9R2_FB9 ((uint32_t)0x00000200)
```

Filter bit 9

5.154.2.1222 CAN_FA1R_FACT

```
#define CAN_FA1R_FACT ((uint16_t)0x3FFF)
```

Filter Active

5.154.2.1223 CAN_FA1R_FACT0

```
#define CAN_FA1R_FACT0 ((uint16_t)0x0001)
```

Filter 0 Active

5.154.2.1224 CAN_FA1R_FACT1

```
#define CAN_FA1R_FACT1 ((uint16_t)0x0002)
```

Filter 1 Active

5.154.2.1225 CAN_FA1R_FACT10

```
#define CAN_FA1R_FACT10 ((uint16_t)0x0400)
```

Filter 10 Active

5.154.2.1226 CAN_FA1R_FACT11

```
#define CAN_FA1R_FACT11 ((uint16_t)0x0800)
```

Filter 11 Active

5.154.2.1227 CAN_FA1R_FACT12

```
#define CAN_FA1R_FACT12 ((uint16_t)0x1000)
```

Filter 12 Active

5.154.2.1228 CAN_FA1R_FACT13

```
#define CAN_FA1R_FACT13 ((uint16_t)0x2000)
```

Filter 13 Active

5.154.2.1229 CAN_FA1R_FACT2

```
#define CAN_FA1R_FACT2 ((uint16_t)0x0004)
```

Filter 2 Active

5.154.2.1230 CAN_FA1R_FACT3

```
#define CAN_FA1R_FACT3 ((uint16_t)0x0008)
```

Filter 3 Active

5.154.2.1231 CAN_FA1R_FACT4

```
#define CAN_FA1R_FACT4 ((uint16_t)0x0010)
```

Filter 4 Active

5.154.2.1232 CAN_FA1R_FACT5

```
#define CAN_FA1R_FACT5 ((uint16_t)0x0020)
```

Filter 5 Active

5.154.2.1233 CAN_FA1R_FACT6

```
#define CAN_FA1R_FACT6 ((uint16_t)0x0040)
```

Filter 6 Active

5.154.2.1234 CAN_FA1R_FACT7

```
#define CAN_FA1R_FACT7 ((uint16_t)0x0080)
```

Filter 7 Active

5.154.2.1235 CAN_FA1R_FACT8

```
#define CAN_FA1R_FACT8 ((uint16_t)0x0100)
```

Filter 8 Active

5.154.2.1236 CAN_FA1R_FACT9

```
#define CAN_FA1R_FACT9 ((uint16_t)0x0200)
```

Filter 9 Active

5.154.2.1237 CAN_FFA1R_FFA

```
#define CAN_FFA1R_FFA ((uint16_t)0x3FFF)
```

Filter FIFO Assignment

5.154.2.1238 CAN_FFA1R_FFA0

```
#define CAN_FFA1R_FFA0 ((uint16_t)0x0001)
```

Filter FIFO Assignment for Filter 0

5.154.2.1239 CAN_FFA1R_FFA1

```
#define CAN_FFA1R_FFA1 ((uint16_t)0x0002)
```

Filter FIFO Assignment for Filter 1

5.154.2.1240 CAN_FFA1R_FFA10

```
#define CAN_FFA1R_FFA10 ((uint16_t)0x0400)
```

Filter FIFO Assignment for Filter 10

5.154.2.1241 CAN_FFA1R_FFA11

```
#define CAN_FFA1R_FFA11 ((uint16_t)0x0800)
```

Filter FIFO Assignment for Filter 11

5.154.2.1242 CAN_FFA1R_FFA12

```
#define CAN_FFA1R_FFA12 ((uint16_t)0x1000)
```

Filter FIFO Assignment for Filter 12

5.154.2.1243 CAN_FFA1R_FFA13

```
#define CAN_FFA1R_FFA13 ((uint16_t)0x2000)
```

Filter FIFO Assignment for Filter 13

5.154.2.1244 CAN_FFA1R_FFA2

```
#define CAN_FFA1R_FFA2 ((uint16_t)0x0004)
```

Filter FIFO Assignment for Filter 2

5.154.2.1245 CAN_FFA1R_FFA3

```
#define CAN_FFA1R_FFA3 ((uint16_t)0x0008)
```

Filter FIFO Assignment for Filter 3

5.154.2.1246 CAN_FFA1R_FFA4

```
#define CAN_FFA1R_FFA4 ((uint16_t)0x0010)
```

Filter FIFO Assignment for Filter 4

5.154.2.1247 CAN_FFA1R_FFA5

```
#define CAN_FFA1R_FFA5 ((uint16_t)0x0020)
```

Filter FIFO Assignment for Filter 5

5.154.2.1248 CAN_FFA1R_FFA6

```
#define CAN_FFA1R_FFA6 ((uint16_t)0x0040)
```

Filter FIFO Assignment for Filter 6

5.154.2.1249 CAN_FFA1R_FFA7

```
#define CAN_FFA1R_FFA7 ((uint16_t)0x0080)
```

Filter FIFO Assignment for Filter 7

5.154.2.1250 CAN_FFA1R_FFA8

```
#define CAN_FFA1R_FFA8 ((uint16_t)0x0100)
```

Filter FIFO Assignment for Filter 8

5.154.2.1251 CAN_FFA1R_FFA9

```
#define CAN_FFA1R_FFA9 ((uint16_t)0x0200)
```

Filter FIFO Assignment for Filter 9

5.154.2.1252 CAN_FM1R_FBM

```
#define CAN_FM1R_FBM ((uint16_t)0x3FFF)
```

Filter Mode

5.154.2.1253 CAN_FM1R_FBM0

```
#define CAN_FM1R_FBM0 ((uint16_t)0x0001)
```

Filter Init Mode bit 0

5.154.2.1254 CAN_FM1R_FBM1

```
#define CAN_FM1R_FBM1 ((uint16_t)0x0002)
```

Filter Init Mode bit 1

5.154.2.1255 CAN_FM1R_FBM10

```
#define CAN_FM1R_FBM10 ((uint16_t)0x0400)
```

Filter Init Mode bit 10

5.154.2.1256 CAN_FM1R_FBM11

```
#define CAN_FM1R_FBM11 ((uint16_t)0x0800)
```

Filter Init Mode bit 11

5.154.2.1257 CAN_FM1R_FBM12

```
#define CAN_FM1R_FBM12 ((uint16_t)0x1000)
```

Filter Init Mode bit 12

5.154.2.1258 CAN_FM1R_FBM13

```
#define CAN_FM1R_FBM13 ((uint16_t)0x2000)
```

Filter Init Mode bit 13

5.154.2.1259 CAN_FM1R_FBM2

```
#define CAN_FM1R_FBM2 ((uint16_t)0x0004)
```

Filter Init Mode bit 2

5.154.2.1260 CAN_FM1R_FBM3

```
#define CAN_FM1R_FBM3 ((uint16_t)0x0008)
```

Filter Init Mode bit 3

5.154.2.1261 CAN_FM1R_FBM4

```
#define CAN_FM1R_FBM4 ((uint16_t)0x0010)
```

Filter Init Mode bit 4

5.154.2.1262 CAN_FM1R_FBM5

```
#define CAN_FM1R_FBM5 ((uint16_t)0x0020)
```

Filter Init Mode bit 5

5.154.2.1263 CAN_FM1R_FBM6

```
#define CAN_FM1R_FBM6 ((uint16_t)0x0040)
```

Filter Init Mode bit 6

5.154.2.1264 CAN_FM1R_FBM7

```
#define CAN_FM1R_FBM7 ((uint16_t)0x0080)
```

Filter Init Mode bit 7

5.154.2.1265 CAN_FM1R_FBM8

```
#define CAN_FM1R_FBM8 ((uint16_t)0x0100)
```

Filter Init Mode bit 8

5.154.2.1266 CAN_FM1R_FBM9

```
#define CAN_FM1R_FBM9 ((uint16_t)0x0200)
```

Filter Init Mode bit 9

5.154.2.1267 CAN_FMR_FINIT

```
#define CAN_FMR_FINIT ((uint8_t)0x01)
```

Filter Init Mode

5.154.2.1268 CAN_FS1R_FSC

```
#define CAN_FS1R_FSC ((uint16_t)0x3FFF)
```

Filter Scale Configuration

5.154.2.1269 CAN_FS1R_FSC0

```
#define CAN_FS1R_FSC0 ((uint16_t)0x0001)
```

Filter Scale Configuration bit 0

5.154.2.1270 CAN_FS1R_FSC1

```
#define CAN_FS1R_FSC1 ((uint16_t)0x0002)
```

Filter Scale Configuration bit 1

5.154.2.1271 CAN_FS1R_FSC10

```
#define CAN_FS1R_FSC10 ((uint16_t)0x0400)
```

Filter Scale Configuration bit 10

5.154.2.1272 CAN_FS1R_FSC11

```
#define CAN_FS1R_FSC11 ((uint16_t)0x0800)
```

Filter Scale Configuration bit 11

5.154.2.1273 CAN_FS1R_FSC12

```
#define CAN_FS1R_FSC12 ((uint16_t)0x1000)
```

Filter Scale Configuration bit 12

5.154.2.1274 CAN_FS1R_FSC13

```
#define CAN_FS1R_FSC13 ((uint16_t)0x2000)
```

Filter Scale Configuration bit 13

5.154.2.1275 CAN_FS1R_FSC2

```
#define CAN_FS1R_FSC2 ((uint16_t)0x0004)
```

Filter Scale Configuration bit 2

5.154.2.1276 CAN_FS1R_FSC3

```
#define CAN_FS1R_FSC3 ((uint16_t)0x0008)
```

Filter Scale Configuration bit 3

5.154.2.1277 CAN_FS1R_FSC4

```
#define CAN_FS1R_FSC4 ((uint16_t)0x0010)
```

Filter Scale Configuration bit 4

5.154.2.1278 CAN_FS1R_FSC5

```
#define CAN_FS1R_FSC5 ((uint16_t)0x0020)
```

Filter Scale Configuration bit 5

5.154.2.1279 CAN_FS1R_FSC6

```
#define CAN_FS1R_FSC6 ((uint16_t)0x0040)
```

Filter Scale Configuration bit 6

5.154.2.1280 CAN_FS1R_FSC7

```
#define CAN_FS1R_FSC7 ((uint16_t)0x0080)
```

Filter Scale Configuration bit 7

5.154.2.1281 CAN_FS1R_FSC8

```
#define CAN_FS1R_FSC8 ((uint16_t)0x0100)
```

Filter Scale Configuration bit 8

5.154.2.1282 CAN_FS1R_FSC9

```
#define CAN_FS1R_FSC9 ((uint16_t)0x0200)
```

Filter Scale Configuration bit 9

5.154.2.1283 CAN_IER_BOFIE

```
#define CAN_IER_BOFIE ((uint32_t)0x00000400)
```

Bus-Off Interrupt Enable

5.154.2.1284 CAN_IER_EPVIE

```
#define CAN_IER_EPVIE ((uint32_t)0x00000200)
```

Error Passive Interrupt Enable

5.154.2.1285 CAN_IER_ERRIE

```
#define CAN_IER_ERRIE ((uint32_t)0x00008000)
```

Error Interrupt Enable

5.154.2.1286 CAN_IER_EWGIE

```
#define CAN_IER_EWGIE ((uint32_t)0x00000100)
```

Error Warning Interrupt Enable

5.154.2.1287 CAN_IER_FFIE0

```
#define CAN_IER_FFIE0 ((uint32_t)0x00000004)
```

FIFO Full Interrupt Enable

5.154.2.1288 CAN_IER_FFIE1

```
#define CAN_IER_FFIE1 ((uint32_t)0x00000020)
```

FIFO Full Interrupt Enable

5.154.2.1289 CAN_IER_FMPIE0

```
#define CAN_IER_FMPIE0 ((uint32_t)0x00000002)
```

FIFO Message Pending Interrupt Enable

5.154.2.1290 CAN_IER_FMPIE1

```
#define CAN_IER_FMPIE1 ((uint32_t)0x00000010)
```

FIFO Message Pending Interrupt Enable

5.154.2.1291 CAN_IER_FOVIE0

```
#define CAN_IER_FOVIE0 ((uint32_t)0x00000008)
```

FIFO Overrun Interrupt Enable

5.154.2.1292 CAN_IER_FOVIE1

```
#define CAN_IER_FOVIE1 ((uint32_t)0x00000040)
```

FIFO Overrun Interrupt Enable

5.154.2.1293 CAN_IER_LECIE

```
#define CAN_IER_LECIE ((uint32_t)0x00000800)
```

Last Error Code Interrupt Enable

5.154.2.1294 CAN_IER_SLKIE

```
#define CAN_IER_SLKIE ((uint32_t)0x00020000)
```

Sleep Interrupt Enable

5.154.2.1295 CAN_IER_TMEIE

```
#define CAN_IER_TMEIE ((uint32_t)0x00000001)
```

Transmit Mailbox Empty Interrupt Enable

5.154.2.1296 CAN_IER_WKUIE

```
#define CAN_IER_WKUIE ((uint32_t)0x00010000)
```

Wakeup Interrupt Enable

5.154.2.1297 CAN_MCR_ABOM

```
#define CAN_MCR_ABOM ((uint16_t)0x0040)
```

Automatic Bus-Off Management

5.154.2.1298 CAN_MCR_AWUM

```
#define CAN_MCR_AWUM ((uint16_t)0x0020)
```

Automatic Wakeup Mode

5.154.2.1299 CAN_MCR_INRQ

```
#define CAN_MCR_INRQ ((uint16_t)0x0001)
```

<CAN control and status registers Initialization Request

5.154.2.1300 CAN_MCR_NART

```
#define CAN_MCR_NART ((uint16_t)0x0010)
```

No Automatic Retransmission

5.154.2.1301 CAN_MCR_RESET

```
#define CAN_MCR_RESET ((uint16_t)0x8000)
```

bxCAN software master reset

5.154.2.1302 CAN_MCR_RFLM

```
#define CAN_MCR_RFLM ((uint16_t)0x0008)
```

Receive FIFO Locked Mode

5.154.2.1303 CAN_MCR_SLEEP

```
#define CAN_MCR_SLEEP ((uint16_t)0x0002)
```

Sleep Mode Request

5.154.2.1304 CAN_MCR_TTCM

```
#define CAN_MCR_TTCM ((uint16_t)0x0080)
```

Time Triggered Communication Mode

5.154.2.1305 CAN_MCR_TXFP

```
#define CAN_MCR_TXFP ((uint16_t)0x0004)
```

Transmit FIFO Priority

5.154.2.1306 CAN_MSR_ERRI

```
#define CAN_MSR_ERRI ((uint16_t)0x0004)
```

Error Interrupt

5.154.2.1307 CAN_MSR_INAK

```
#define CAN_MSR_INAK ((uint16_t)0x0001)
```

Initialization Acknowledge

5.154.2.1308 CAN_MSR_RX

```
#define CAN_MSR_RX ((uint16_t)0x0800)
```

CAN Rx Signal

5.154.2.1309 CAN_MSR_RXM

```
#define CAN_MSR_RXM ((uint16_t)0x0200)
```

Receive Mode

5.154.2.1310 CAN_MSR_SAMP

```
#define CAN_MSR_SAMP ((uint16_t)0x0400)
```

Last Sample Point

5.154.2.1311 CAN_MSR_SLAK

```
#define CAN_MSR_SLAK ((uint16_t)0x0002)
```

Sleep Acknowledge

5.154.2.1312 CAN_MSR_SLAKI

```
#define CAN_MSR_SLAKI ((uint16_t)0x0010)
```

Sleep Acknowledge Interrupt

5.154.2.1313 CAN_MSR_TXM

```
#define CAN_MSR_TXM ((uint16_t)0x0100)
```

Transmit Mode

5.154.2.1314 CAN_MSR_WKUI

```
#define CAN_MSR_WKUI ((uint16_t)0x0008)
```

Wakeup Interrupt

5.154.2.1315 CAN_RDH0R_DATA4

```
#define CAN_RDH0R_DATA4 ((uint32_t)0x000000FF)
```

Data byte 4

5.154.2.1316 CAN_RDH0R_DATA5

```
#define CAN_RDH0R_DATA5 ((uint32_t)0x0000FF00)
```

Data byte 5

5.154.2.1317 CAN_RDH0R_DATA6

```
#define CAN_RDH0R_DATA6 ((uint32_t)0x00FF0000)
```

Data byte 6

5.154.2.1318 CAN_RDH0R_DATA7

```
#define CAN_RDH0R_DATA7 ((uint32_t)0xFF000000)
```

Data byte 7

5.154.2.1319 CAN_RDH1R_DATA4

```
#define CAN_RDH1R_DATA4 ((uint32_t)0x000000FF)
```

Data byte 4

5.154.2.1320 CAN_RDH1R_DATA5

```
#define CAN_RDH1R_DATA5 ((uint32_t)0x0000FF00)
```

Data byte 5

5.154.2.1321 CAN_RDH1R_DATA6

```
#define CAN_RDH1R_DATA6 ((uint32_t)0x00FF0000)
```

Data byte 6

5.154.2.1322 CAN_RDH1R_DATA7

```
#define CAN_RDH1R_DATA7 ((uint32_t)0xFF000000)
```

Data byte 7 CAN filter registers

5.154.2.1323 CAN_RDL0R_DATA0

```
#define CAN_RDL0R_DATA0 ((uint32_t)0x000000FF)
```

Data byte 0

5.154.2.1324 CAN_RDL0R_DATA1

```
#define CAN_RDL0R_DATA1 ((uint32_t)0x0000FF00)
```

Data byte 1

5.154.2.1325 CAN_RDL0R_DATA2

```
#define CAN_RDL0R_DATA2 ((uint32_t)0x00FF0000)
```

Data byte 2

5.154.2.1326 CAN_RDL0R_DATA3

```
#define CAN_RDL0R_DATA3 ((uint32_t)0xFF000000)
```

Data byte 3

5.154.2.1327 CAN_RDL1R_DATA0

```
#define CAN_RDL1R_DATA0 ((uint32_t)0x000000FF)
```

Data byte 0

5.154.2.1328 CAN_RDL1R_DATA1

```
#define CAN_RDL1R_DATA1 ((uint32_t)0x0000FF00)
```

Data byte 1

5.154.2.1329 CAN_RDL1R_DATA2

```
#define CAN_RDL1R_DATA2 ((uint32_t)0x00FF0000)
```

Data byte 2

5.154.2.1330 CAN_RDL1R_DATA3

```
#define CAN_RDL1R_DATA3 ((uint32_t)0xFF000000)
```

Data byte 3

5.154.2.1331 CAN_RDT0R_DLC

```
#define CAN_RDT0R_DLC ((uint32_t)0x0000000F)
```

Data Length Code

5.154.2.1332 CAN_RDT0R_FMI

```
#define CAN_RDT0R_FMI ((uint32_t)0x0000FF00)
```

Filter Match Index

5.154.2.1333 CAN_RDT0R_TIME

```
#define CAN_RDT0R_TIME ((uint32_t)0xFFFF0000)
```

Message Time Stamp

5.154.2.1334 CAN_RDT1R_DLC

```
#define CAN_RDT1R_DLC ((uint32_t)0x0000000F)
```

Data Length Code

5.154.2.1335 CAN_RDT1R_FMI

```
#define CAN_RDT1R_FMI ((uint32_t)0x0000FF00)
```

Filter Match Index

5.154.2.1336 CAN_RDT1R_TIME

```
#define CAN_RDT1R_TIME ((uint32_t)0xFFFF0000)
```

Message Time Stamp

5.154.2.1337 CAN_RF0R_FMP0

```
#define CAN_RF0R_FMP0 ((uint8_t)0x03)
```

FIFO 0 Message Pending

5.154.2.1338 CAN_RF0R_FOVR0

```
#define CAN_RF0R_FOVR0 ((uint8_t)0x10)
```

FIFO 0 Overrun

5.154.2.1339 CAN_RF0R_FULL0

```
#define CAN_RF0R_FULL0 ((uint8_t)0x08)
```

FIFO 0 Full

5.154.2.1340 CAN_RF0R_RFOM0

```
#define CAN_RF0R_RFOM0 ((uint8_t)0x20)
```

Release FIFO 0 Output Mailbox

5.154.2.1341 CAN_RF1R_FMP1

```
#define CAN_RF1R_FMP1 ((uint8_t)0x03)
```

FIFO 1 Message Pending

5.154.2.1342 CAN_RF1R_FOVR1

```
#define CAN_RF1R_FOVR1 ((uint8_t)0x10)
```

FIFO 1 Overrun

5.154.2.1343 CAN_RF1R_FULL1

```
#define CAN_RF1R_FULL1 ((uint8_t)0x08)
```

FIFO 1 Full

5.154.2.1344 CAN_RF1R_RFOM1

```
#define CAN_RF1R_RFOM1 ((uint8_t)0x20)
```

Release FIFO 1 Output Mailbox

5.154.2.1345 CAN_RI0R_EXID

```
#define CAN_RI0R_EXID ((uint32_t)0x001FFFF8)
```

Extended Identifier

5.154.2.1346 CAN_RI0R_IDE

```
#define CAN_RI0R_IDE ((uint32_t)0x00000004)
```

Identifier Extension

5.154.2.1347 CAN_RI0R_RTR

```
#define CAN_RI0R_RTR ((uint32_t)0x00000002)
```

Remote Transmission Request

5.154.2.1348 CAN_RI0R_STID

```
#define CAN_RI0R_STID ((uint32_t)0xFFE00000)
```

Standard Identifier or Extended Identifier

5.154.2.1349 CAN_RI1R_EXID

```
#define CAN_RI1R_EXID ((uint32_t)0x001FFFF8)
```

Extended identifier

5.154.2.1350 CAN_RI1R_IDE

```
#define CAN_RI1R_IDE ((uint32_t)0x00000004)
```

Identifier Extension

5.154.2.1351 CAN_RI1R_RTR

```
#define CAN_RI1R_RTR ((uint32_t)0x00000002)
```

Remote Transmission Request

5.154.2.1352 CAN_RI1R_STID

```
#define CAN_RI1R_STID ((uint32_t)0xFFE00000)
```

Standard Identifier or Extended Identifier

5.154.2.1353 CAN_TDH0R_DATA4

```
#define CAN_TDH0R_DATA4 ((uint32_t)0x000000FF)
```

Data byte 4

5.154.2.1354 CAN_TDH0R_DATA5

```
#define CAN_TDH0R_DATA5 ((uint32_t)0x0000FF00)
```

Data byte 5

5.154.2.1355 CAN_TDH0R_DATA6

```
#define CAN_TDH0R_DATA6 ((uint32_t)0x00FF0000)
```

Data byte 6

5.154.2.1356 CAN_TDH0R_DATA7

```
#define CAN_TDH0R_DATA7 ((uint32_t)0xFF000000)
```

Data byte 7

5.154.2.1357 CAN_TDH1R_DATA4

```
#define CAN_TDH1R_DATA4 ((uint32_t)0x000000FF)
```

Data byte 4

5.154.2.1358 CAN_TDH1R_DATA5

```
#define CAN_TDH1R_DATA5 ((uint32_t)0x0000FF00)
```

Data byte 5

5.154.2.1359 CAN_TDH1R_DATA6

```
#define CAN_TDH1R_DATA6 ((uint32_t)0x00FF0000)
```

Data byte 6

5.154.2.1360 CAN_TDH1R_DATA7

```
#define CAN_TDH1R_DATA7 ((uint32_t)0xFF000000)
```

Data byte 7

5.154.2.1361 CAN_TDH2R_DATA4

```
#define CAN_TDH2R_DATA4 ((uint32_t)0x000000FF)
```

Data byte 4

5.154.2.1362 CAN_TDH2R_DATA5

```
#define CAN_TDH2R_DATA5 ((uint32_t)0x0000FF00)
```

Data byte 5

5.154.2.1363 CAN_TDH2R_DATA6

```
#define CAN_TDH2R_DATA6 ((uint32_t)0x00FF0000)
```

Data byte 6

5.154.2.1364 CAN_TDH2R_DATA7

```
#define CAN_TDH2R_DATA7 ((uint32_t)0xFF000000)
```

Data byte 7

5.154.2.1365 CAN_TDL0R_DATA0

```
#define CAN_TDL0R_DATA0 ((uint32_t)0x000000FF)
```

Data byte 0

5.154.2.1366 CAN_TDL0R_DATA1

```
#define CAN_TDL0R_DATA1 ((uint32_t)0x0000FF00)
```

Data byte 1

5.154.2.1367 CAN_TDL0R_DATA2

```
#define CAN_TDL0R_DATA2 ((uint32_t)0x00FF0000)
```

Data byte 2

5.154.2.1368 CAN_TDL0R_DATA3

```
#define CAN_TDL0R_DATA3 ((uint32_t)0xFF000000)
```

Data byte 3

5.154.2.1369 CAN_TDL1R_DATA0

```
#define CAN_TDL1R_DATA0 ((uint32_t)0x000000FF)
```

Data byte 0

5.154.2.1370 CAN_TDL1R_DATA1

```
#define CAN_TDL1R_DATA1 ((uint32_t)0x0000FF00)
```

Data byte 1

5.154.2.1371 CAN_TDL1R_DATA2

```
#define CAN_TDL1R_DATA2 ((uint32_t)0x00FF0000)
```

Data byte 2

5.154.2.1372 CAN_TDL1R_DATA3

```
#define CAN_TDL1R_DATA3 ((uint32_t)0xFF000000)
```

Data byte 3

5.154.2.1373 CAN_TDL2R_DATA0

```
#define CAN_TDL2R_DATA0 ((uint32_t)0x000000FF)
```

Data byte 0

5.154.2.1374 CAN_TDL2R_DATA1

```
#define CAN_TDL2R_DATA1 ((uint32_t)0x0000FF00)
```

Data byte 1

5.154.2.1375 CAN_TDL2R_DATA2

```
#define CAN_TDL2R_DATA2 ((uint32_t)0x00FF0000)
```

Data byte 2

5.154.2.1376 CAN_TDL2R_DATA3

```
#define CAN_TDL2R_DATA3 ((uint32_t)0xFF000000)
```

Data byte 3

5.154.2.1377 CAN_TDT0R_DLC

```
#define CAN_TDT0R_DLC ((uint32_t)0x0000000F)
```

Data Length Code

5.154.2.1378 CAN_TDT0R_TGT

```
#define CAN_TDT0R_TGT ((uint32_t)0x00000100)
```

Transmit Global Time

5.154.2.1379 CAN_TDT0R_TIME

```
#define CAN_TDT0R_TIME ((uint32_t)0xFFFF0000)
```

Message Time Stamp

5.154.2.1380 CAN_TDT1R_DLC

```
#define CAN_TDT1R_DLC ((uint32_t)0x0000000F)
```

Data Length Code

5.154.2.1381 CAN_TDT1R_TGT

```
#define CAN_TDT1R_TGT ((uint32_t)0x00000100)
```

Transmit Global Time

5.154.2.1382 CAN_TDT1R_TIME

```
#define CAN_TDT1R_TIME ((uint32_t)0xFFFF0000)
```

Message Time Stamp

5.154.2.1383 CAN_TDT2R_DLC

```
#define CAN_TDT2R_DLC ((uint32_t)0x0000000F)
```

Data Length Code

5.154.2.1384 CAN_TDT2R_TGT

```
#define CAN_TDT2R_TGT ((uint32_t)0x00000100)
```

Transmit Global Time

5.154.2.1385 CAN_TDT2R_TIME

```
#define CAN_TDT2R_TIME ((uint32_t)0xFFFF0000)
```

Message Time Stamp

5.154.2.1386 CAN_TI0R_EXID

```
#define CAN_TI0R_EXID ((uint32_t)0x001FFFF8)
```

Extended Identifier

5.154.2.1387 CAN_TI0R_IDE

```
#define CAN_TI0R_IDE ((uint32_t)0x00000004)
```

Identifier Extension

5.154.2.1388 CAN_TI0R_RTR

```
#define CAN_TI0R_RTR ((uint32_t)0x00000002)
```

Remote Transmission Request

5.154.2.1389 CAN_TI0R_STID

```
#define CAN_TI0R_STID ((uint32_t)0xFFE00000)
```

Standard Identifier or Extended Identifier

5.154.2.1390 CAN_TI0R_TXRQ

```
#define CAN_TI0R_TXRQ ((uint32_t)0x00000001)
```

Transmit Mailbox Request

5.154.2.1391 CAN_TI1R_EXID

```
#define CAN_TI1R_EXID ((uint32_t)0x001FFFF8)
```

Extended Identifier

5.154.2.1392 CAN_TI1R_IDE

```
#define CAN_TI1R_IDE ((uint32_t)0x00000004)
```

Identifier Extension

5.154.2.1393 CAN_TI1R_RTR

```
#define CAN_TI1R_RTR ((uint32_t)0x00000002)
```

Remote Transmission Request

5.154.2.1394 CAN_TI1R_STID

```
#define CAN_TI1R_STID ((uint32_t)0xFFE00000)
```

Standard Identifier or Extended Identifier

5.154.2.1395 CAN_TI1R_TXRQ

```
#define CAN_TI1R_TXRQ ((uint32_t)0x00000001)
```

Transmit Mailbox Request

5.154.2.1396 CAN_TI2R_EXID

```
#define CAN_TI2R_EXID ((uint32_t)0x001FFFF8)
```

Extended identifier

5.154.2.1397 CAN_TI2R_IDE

```
#define CAN_TI2R_IDE ((uint32_t)0x00000004)
```

Identifier Extension

5.154.2.1398 CAN_TI2R_RTR

```
#define CAN_TI2R_RTR ((uint32_t)0x00000002)
```

Remote Transmission Request

5.154.2.1399 CAN_TI2R_STID

```
#define CAN_TI2R_STID ((uint32_t)0xFFE00000)
```

Standard Identifier or Extended Identifier

5.154.2.1400 CAN_TI2R_TXRQ

```
#define CAN_TI2R_TXRQ ((uint32_t)0x00000001)
```

Transmit Mailbox Request

5.154.2.1401 CAN_TSR_ABRQ0

```
#define CAN_TSR_ABRQ0 ((uint32_t)0x00000080)
```

Abort Request for Mailbox0

5.154.2.1402 CAN_TSR_ABRQ1

```
#define CAN_TSR_ABRQ1 ((uint32_t)0x00008000)
```

Abort Request for Mailbox 1

5.154.2.1403 CAN_TSR_ABRQ2

```
#define CAN_TSR_ABRQ2 ((uint32_t)0x00800000)
```

Abort Request for Mailbox 2

5.154.2.1404 CAN_TSR_ALST0

```
#define CAN_TSR_ALST0 ((uint32_t)0x00000004)
```

Arbitration Lost for Mailbox0

5.154.2.1405 CAN_TSR_ALST1

```
#define CAN_TSR_ALST1 ((uint32_t)0x00000400)
```

Arbitration Lost for Mailbox1

5.154.2.1406 CAN_TSR_ALST2

```
#define CAN_TSR_ALST2 ((uint32_t)0x00040000)
```

Arbitration Lost for mailbox 2

5.154.2.1407 CAN_TSR_CODE

```
#define CAN_TSR_CODE ((uint32_t)0x03000000)
```

Mailbox Code

5.154.2.1408 CAN_TSR_LOW

```
#define CAN_TSR_LOW ((uint32_t)0xE0000000)
```

LOW[2:0] bits

5.154.2.1409 CAN_TSR_LOW0

```
#define CAN_TSR_LOW0 ((uint32_t)0x20000000)
```

Lowest Priority Flag for Mailbox 0

5.154.2.1410 CAN_TSR_LOW1

```
#define CAN_TSR_LOW1 ((uint32_t)0x40000000)
```

Lowest Priority Flag for Mailbox 1

5.154.2.1411 CAN_TSR_LOW2

```
#define CAN_TSR_LOW2 ((uint32_t)0x80000000)
```

Lowest Priority Flag for Mailbox 2

5.154.2.1412 CAN_TSR_RQCP0

```
#define CAN_TSR_RQCP0 ((uint32_t)0x00000001)
```

Request Completed Mailbox0

5.154.2.1413 CAN_TSR_RQCP1

```
#define CAN_TSR_RQCP1 ((uint32_t)0x00000100)
```

Request Completed Mailbox1

5.154.2.1414 CAN_TSR_RQCP2

```
#define CAN_TSR_RQCP2 ((uint32_t)0x00010000)
```

Request Completed Mailbox2

5.154.2.1415 CAN_TSR_TERR0

```
#define CAN_TSR_TERR0 ((uint32_t)0x00000008)
```

Transmission Error of Mailbox0

5.154.2.1416 CAN_TSR_TERR1

```
#define CAN_TSR_TERR1 ((uint32_t)0x00000800)
```

Transmission Error of Mailbox1

5.154.2.1417 CAN_TSR_TERR2

```
#define CAN_TSR_TERR2 ((uint32_t)0x00080000)
```

Transmission Error of Mailbox 2

5.154.2.1418 CAN_TSR_TME

```
#define CAN_TSR_TME ((uint32_t)0x1C000000)
```

TME[2:0] bits

5.154.2.1419 CAN_TSR_TME0

```
#define CAN_TSR_TME0 ((uint32_t)0x04000000)
```

Transmit Mailbox 0 Empty

5.154.2.1420 CAN_TSR_TME1

```
#define CAN_TSR_TME1 ((uint32_t)0x08000000)
```

Transmit Mailbox 1 Empty

5.154.2.1421 CAN_TSR_TME2

```
#define CAN_TSR_TME2 ((uint32_t)0x10000000)
```

Transmit Mailbox 2 Empty

5.154.2.1422 CAN_TSR_TXOK0

```
#define CAN_TSR_TXOK0 ((uint32_t)0x00000002)
```

Transmission OK of Mailbox0

5.154.2.1423 CAN_TSR_TXOK1

```
#define CAN_TSR_TXOK1 ((uint32_t)0x00000200)
```

Transmission OK of Mailbox1

5.154.2.1424 CAN_TSR_TXOK2

```
#define CAN_TSR_TXOK2 ((uint32_t)0x00020000)
```

Transmission OK of Mailbox 2

5.154.2.1425 CRC_CR_RESET

```
#define CRC_CR_RESET ((uint8_t)0x01)
```

RESET bit

5.154.2.1426 CRC_DR_DR

```
#define CRC_DR_DR ((uint32_t)0xFFFFFFFF)
```

Data register bits

5.154.2.1427 CRC_IDR_IDR

```
#define CRC_IDR_IDR ((uint8_t)0xFF)
```

General-purpose 8-bit data register bits

5.154.2.1428 DAC_CR_BOFF1

```
#define DAC_CR_BOFF1 ((uint32_t)0x00000002)
```

DAC channel1 output buffer disable

5.154.2.1429 DAC_CR_BOFF2

```
#define DAC_CR_BOFF2 ((uint32_t)0x00020000)
```

DAC channel2 output buffer disable

5.154.2.1430 DAC_CR_DMAEN1

```
#define DAC_CR_DMAEN1 ((uint32_t)0x00001000)
```

DAC channel1 DMA enable

5.154.2.1431 DAC_CR_DMAEN2

```
#define DAC_CR_DMAEN2 ((uint32_t)0x10000000)
```

DAC channel2 DMA enabled

5.154.2.1432 DAC_CR_EN1

```
#define DAC_CR_EN1 ((uint32_t)0x00000001)
```

DAC channel1 enable

5.154.2.1433 DAC_CR_EN2

```
#define DAC_CR_EN2 ((uint32_t)0x00010000)
```

DAC channel2 enable

5.154.2.1434 DAC_CR_MAMP1

```
#define DAC_CR_MAMP1 ((uint32_t)0x00000F00)
```

MAMP1[3:0] (DAC channel1 Mask/Amplitude selector)

5.154.2.1435 DAC_CR_MAMP1_0

```
#define DAC_CR_MAMP1_0 ((uint32_t)0x00000100)
```

Bit 0

5.154.2.1436 DAC_CR_MAMP1_1

```
#define DAC_CR_MAMP1_1 ((uint32_t)0x00000200)
```

Bit 1

5.154.2.1437 DAC_CR_MAMP1_2

```
#define DAC_CR_MAMP1_2 ((uint32_t)0x00000400)
```

Bit 2

5.154.2.1438 DAC_CR_MAMP1_3

```
#define DAC_CR_MAMP1_3 ((uint32_t)0x00000800)
```

Bit 3

5.154.2.1439 DAC_CR_MAMP2

```
#define DAC_CR_MAMP2 ((uint32_t)0x0F000000)
```

MAMP2[3:0] (DAC channel2 Mask/Amplitude selector)

5.154.2.1440 DAC_CR_MAMP2_0

```
#define DAC_CR_MAMP2_0 ((uint32_t)0x01000000)
```

Bit 0

5.154.2.1441 DAC_CR_MAMP2_1

```
#define DAC_CR_MAMP2_1 ((uint32_t)0x02000000)
```

Bit 1

5.154.2.1442 DAC_CR_MAMP2_2

```
#define DAC_CR_MAMP2_2 ((uint32_t)0x04000000)
```

Bit 2

5.154.2.1443 DAC_CR_MAMP2_3

```
#define DAC_CR_MAMP2_3 ((uint32_t)0x08000000)
```

Bit 3

5.154.2.1444 DAC_CR_TEN1

```
#define DAC_CR_TEN1 ((uint32_t)0x00000004)
```

DAC channel1 Trigger enable

5.154.2.1445 DAC_CR_TEN2

```
#define DAC_CR_TEN2 ((uint32_t)0x00040000)
```

DAC channel2 Trigger enable

5.154.2.1446 DAC_CR_TSEL1

```
#define DAC_CR_TSEL1 ((uint32_t)0x00000038)
```

TSEL1[2:0] (DAC channel1 Trigger selection)

5.154.2.1447 DAC_CR_TSEL1_0

```
#define DAC_CR_TSEL1_0 ((uint32_t)0x00000008)
```

Bit 0

5.154.2.1448 DAC_CR_TSEL1_1

```
#define DAC_CR_TSEL1_1 ((uint32_t)0x00000010)
```

Bit 1

5.154.2.1449 DAC_CR_TSEL1_2

```
#define DAC_CR_TSEL1_2 ((uint32_t)0x00000020)
```

Bit 2

5.154.2.1450 DAC_CR_TSEL2

```
#define DAC_CR_TSEL2 ((uint32_t)0x00380000)
```

TSEL2[2:0] (DAC channel2 Trigger selection)

5.154.2.1451 DAC_CR_TSEL2_0

```
#define DAC_CR_TSEL2_0 ((uint32_t)0x00080000)
```

Bit 0

5.154.2.1452 DAC_CR_TSEL2_1

```
#define DAC_CR_TSEL2_1 ((uint32_t)0x00100000)
```

Bit 1

5.154.2.1453 DAC_CR_TSEL2_2

```
#define DAC_CR_TSEL2_2 ((uint32_t)0x00200000)
```

Bit 2

5.154.2.1454 DAC_CR_WAVE1

```
#define DAC_CR_WAVE1 ((uint32_t)0x000000C0)
```

WAVE1[1:0] (DAC channel1 noise/triangle wave generation enable)

5.154.2.1455 DAC_CR_WAVE1_0

```
#define DAC_CR_WAVE1_0 ((uint32_t)0x00000040)
```

Bit 0

5.154.2.1456 DAC_CR_WAVE1_1

```
#define DAC_CR_WAVE1_1 ((uint32_t)0x00000080)
```

Bit 1

5.154.2.1457 DAC_CR_WAVE2

```
#define DAC_CR_WAVE2 ((uint32_t)0x00C00000)
```

WAVE2[1:0] (DAC channel2 noise/triangle wave generation enable)

5.154.2.1458 DAC_CR_WAVE2_0

```
#define DAC_CR_WAVE2_0 ((uint32_t)0x00400000)
```

Bit 0

5.154.2.1459 DAC_CR_WAVE2_1

```
#define DAC_CR_WAVE2_1 ((uint32_t)0x00800000)
```

Bit 1

5.154.2.1460 DAC_DHR12L1_DACC1DHR

```
#define DAC_DHR12L1_DACC1DHR ((uint16_t)0xFFF0)
```

DAC channel1 12-bit Left aligned data

5.154.2.1461 DAC_DHR12L2_DACC2DHR

```
#define DAC_DHR12L2_DACC2DHR ((uint16_t)0xFFF0)
```

DAC channel2 12-bit Left aligned data

5.154.2.1462 DAC_DHR12LD_DACC1DHR

```
#define DAC_DHR12LD_DACC1DHR ((uint32_t)0x0000FFF0)
```

DAC channel1 12-bit Left aligned data

5.154.2.1463 DAC_DHR12LD_DACC2DHR

```
#define DAC_DHR12LD_DACC2DHR ((uint32_t)0xFFF00000)
```

DAC channel2 12-bit Left aligned data

5.154.2.1464 DAC_DHR12R1_DACC1DHR

```
#define DAC_DHR12R1_DACC1DHR ((uint16_t)0x0FFF)
```

DAC channel1 12-bit Right aligned data

5.154.2.1465 DAC_DHR12R2_DACC2DHR

```
#define DAC_DHR12R2_DACC2DHR ((uint16_t)0x0FFF)
```

DAC channel2 12-bit Right aligned data

5.154.2.1466 DAC_DHR12RD_DACC1DHR

```
#define DAC_DHR12RD_DACC1DHR ((uint32_t)0x00000FFF)
```

DAC channel1 12-bit Right aligned data

5.154.2.1467 DAC_DHR12RD_DACC2DHR

```
#define DAC_DHR12RD_DACC2DHR ((uint32_t)0x0FFF0000)
```

DAC channel2 12-bit Right aligned data

5.154.2.1468 DAC_DHR8R1_DACC1DHR

```
#define DAC_DHR8R1_DACC1DHR ((uint8_t)0xFF)
```

DAC channel1 8-bit Right aligned data

5.154.2.1469 DAC_DHR8R2_DACC2DHR

```
#define DAC_DHR8R2_DACC2DHR ((uint8_t)0xFF)
```

DAC channel2 8-bit Right aligned data

5.154.2.1470 DAC_DHR8RD_DACC1DHR

```
#define DAC_DHR8RD_DACC1DHR ((uint16_t)0x00FF)
```

DAC channel1 8-bit Right aligned data

5.154.2.1471 DAC_DHR8RD_DACC2DHR

```
#define DAC_DHR8RD_DACC2DHR ((uint16_t)0xFF00)
```

DAC channel2 8-bit Right aligned data

5.154.2.1472 DAC_DOR1_DACC1DOR

```
#define DAC_DOR1_DACC1DOR ((uint16_t)0x0FFF)
```

DAC channel1 data output

5.154.2.1473 DAC_DOR2_DACC2DOR

```
#define DAC_DOR2_DACC2DOR ((uint16_t)0x0FFF)
```

DAC channel2 data output

5.154.2.1474 DAC_SR_DMAUDR1

```
#define DAC_SR_DMAUDR1 ((uint32_t)0x00002000)
```

DAC channel1 DMA underrun flag

5.154.2.1475 DAC_SR_DMAUDR2

```
#define DAC_SR_DMAUDR2 ((uint32_t)0x20000000)
```

DAC channel2 DMA underrun flag

5.154.2.1476 DAC_SWTRIGR_SWTRIG1

```
#define DAC_SWTRIGR_SWTRIG1 ((uint8_t)0x01)
```

DAC channel1 software trigger

5.154.2.1477 DAC_SWTRIGR_SWTRIG2

```
#define DAC_SWTRIGR_SWTRIG2 ((uint8_t)0x02)
```

DAC channel2 software trigger

5.154.2.1478 DBGMCU_CR_TRACE_MODE_0

```
#define DBGMCU_CR_TRACE_MODE_0 ((uint32_t)0x00000040)
```

Bit 0

5.154.2.1479 DBGMCU_CR_TRACE_MODE_1

```
#define DBGMCU_CR_TRACE_MODE_1 ((uint32_t)0x00000080)
```

Bit 1

5.154.2.1480 ETH_MACCR_BL

```
#define ETH_MACCR_BL
```

Value:

```
                                                ((uint32_t)0x00000060)  /* Back-off limit: random
    integer number (r) of slot time delays before rescheduling
                                                a transmission attempt during retries after a
    collision: 0 =< r <2^k */
```

5.154.2.1481 EXTI_EMR_MR0

```
#define EXTI_EMR_MR0 ((uint32_t)0x00000001)
```

Event Mask on line 0

5.154.2.1482 EXTI_EMR_MR1

```
#define EXTI_EMR_MR1 ((uint32_t)0x00000002)
```

Event Mask on line 1

5.154.2.1483 EXTI_EMR_MR10

```
#define EXTI_EMR_MR10 ((uint32_t)0x00000400)
```

Event Mask on line 10

5.154.2.1484 EXTI_EMR_MR11

```
#define EXTI_EMR_MR11 ((uint32_t)0x00000800)
```

Event Mask on line 11

5.154.2.1485 EXTI_EMR_MR12

```
#define EXTI_EMR_MR12 ((uint32_t)0x00001000)
```

Event Mask on line 12

5.154.2.1486 EXTI_EMR_MR13

```
#define EXTI_EMR_MR13 ((uint32_t)0x00002000)
```

Event Mask on line 13

5.154.2.1487 EXTI_EMR_MR14

```
#define EXTI_EMR_MR14 ((uint32_t)0x00004000)
```

Event Mask on line 14

5.154.2.1488 EXTI_EMR_MR15

```
#define EXTI_EMR_MR15 ((uint32_t)0x00008000)
```

Event Mask on line 15

5.154.2.1489 EXTI_EMR_MR16

```
#define EXTI_EMR_MR16 ((uint32_t)0x00010000)
```

Event Mask on line 16

5.154.2.1490 EXTI_EMR_MR17

```
#define EXTI_EMR_MR17 ((uint32_t)0x00020000)
```

Event Mask on line 17

5.154.2.1491 EXTI_EMR_MR18

```
#define EXTI_EMR_MR18 ((uint32_t)0x00040000)
```

Event Mask on line 18

5.154.2.1492 EXTI_EMR_MR19

```
#define EXTI_EMR_MR19 ((uint32_t)0x00080000)
```

Event Mask on line 19

5.154.2.1493 EXTI_EMR_MR2

```
#define EXTI_EMR_MR2 ((uint32_t)0x00000004)
```

Event Mask on line 2

5.154.2.1494 EXTI_EMR_MR3

```
#define EXTI_EMR_MR3 ((uint32_t)0x00000008)
```

Event Mask on line 3

5.154.2.1495 EXTI_EMR_MR4

```
#define EXTI_EMR_MR4 ((uint32_t)0x00000010)
```

Event Mask on line 4

5.154.2.1496 EXTI_EMR_MR5

```
#define EXTI_EMR_MR5 ((uint32_t)0x00000020)
```

Event Mask on line 5

5.154.2.1497 EXTI_EMR_MR6

```
#define EXTI_EMR_MR6 ((uint32_t)0x00000040)
```

Event Mask on line 6

5.154.2.1498 EXTI_EMR_MR7

```
#define EXTI_EMR_MR7 ((uint32_t)0x00000080)
```

Event Mask on line 7

5.154.2.1499 EXTI_EMR_MR8

```
#define EXTI_EMR_MR8 ((uint32_t)0x00000100)
```

Event Mask on line 8

5.154.2.1500 EXTI_EMR_MR9

```
#define EXTI_EMR_MR9 ((uint32_t)0x00000200)
```

Event Mask on line 9

5.154.2.1501 EXTI_FTSR_TR0

```
#define EXTI_FTSR_TR0 ((uint32_t)0x00000001)
```

Falling trigger event configuration bit of line 0

5.154.2.1502 EXTI_FTSR_TR1

```
#define EXTI_FTSR_TR1 ((uint32_t)0x00000002)
```

Falling trigger event configuration bit of line 1

5.154.2.1503 EXTI_FTSR_TR10

```
#define EXTI_FTSR_TR10 ((uint32_t)0x00000400)
```

Falling trigger event configuration bit of line 10

5.154.2.1504 EXTI_FTSR_TR11

```
#define EXTI_FTSR_TR11 ((uint32_t)0x00000800)
```

Falling trigger event configuration bit of line 11

5.154.2.1505 EXTI_FTSR_TR12

```
#define EXTI_FTSR_TR12 ((uint32_t)0x00001000)
```

Falling trigger event configuration bit of line 12

5.154.2.1506 EXTI_FTSR_TR13

```
#define EXTI_FTSR_TR13 ((uint32_t)0x00002000)
```

Falling trigger event configuration bit of line 13

5.154.2.1507 EXTI_FTSR_TR14

```
#define EXTI_FTSR_TR14 ((uint32_t)0x00004000)
```

Falling trigger event configuration bit of line 14

5.154.2.1508 EXTI_FTSR_TR15

```
#define EXTI_FTSR_TR15 ((uint32_t)0x00008000)
```

Falling trigger event configuration bit of line 15

5.154.2.1509 EXTI_FTSR_TR16

```
#define EXTI_FTSR_TR16 ((uint32_t)0x00010000)
```

Falling trigger event configuration bit of line 16

5.154.2.1510 EXTI_FTSR_TR17

```
#define EXTI_FTSR_TR17 ((uint32_t)0x00020000)
```

Falling trigger event configuration bit of line 17

5.154.2.1511 EXTI_FTSR_TR18

```
#define EXTI_FTSR_TR18 ((uint32_t)0x00040000)
```

Falling trigger event configuration bit of line 18

5.154.2.1512 EXTI_FTSR_TR19

```
#define EXTI_FTSR_TR19 ((uint32_t)0x00080000)
```

Falling trigger event configuration bit of line 19

5.154.2.1513 EXTI_FTSR_TR2

```
#define EXTI_FTSR_TR2 ((uint32_t)0x00000004)
```

Falling trigger event configuration bit of line 2

5.154.2.1514 EXTI_FTSR_TR3

```
#define EXTI_FTSR_TR3 ((uint32_t)0x00000008)
```

Falling trigger event configuration bit of line 3

5.154.2.1515 EXTI_FTSR_TR4

```
#define EXTI_FTSR_TR4 ((uint32_t)0x00000010)
```

Falling trigger event configuration bit of line 4

5.154.2.1516 EXTI_FTSR_TR5

```
#define EXTI_FTSR_TR5 ((uint32_t)0x00000020)
```

Falling trigger event configuration bit of line 5

5.154.2.1517 EXTI_FTSR_TR6

```
#define EXTI_FTSR_TR6 ((uint32_t)0x00000040)
```

Falling trigger event configuration bit of line 6

5.154.2.1518 EXTI_FTSR_TR7

```
#define EXTI_FTSR_TR7 ((uint32_t)0x00000080)
```

Falling trigger event configuration bit of line 7

5.154.2.1519 EXTI_FTSR_TR8

```
#define EXTI_FTSR_TR8 ((uint32_t)0x00000100)
```

Falling trigger event configuration bit of line 8

5.154.2.1520 EXTI_FTSR_TR9

```
#define EXTI_FTSR_TR9 ((uint32_t)0x00000200)
```

Falling trigger event configuration bit of line 9

5.154.2.1521 EXTI_IMR_MR0

```
#define EXTI_IMR_MR0 ((uint32_t)0x00000001)
```

Interrupt Mask on line 0

5.154.2.1522 EXTI_IMR_MR1

```
#define EXTI_IMR_MR1 ((uint32_t)0x00000002)
```

Interrupt Mask on line 1

5.154.2.1523 EXTI_IMR_MR10

```
#define EXTI_IMR_MR10 ((uint32_t)0x00000400)
```

Interrupt Mask on line 10

5.154.2.1524 EXTI_IMR_MR11

```
#define EXTI_IMR_MR11 ((uint32_t)0x00000800)
```

Interrupt Mask on line 11

5.154.2.1525 EXTI_IMR_MR12

```
#define EXTI_IMR_MR12 ((uint32_t)0x00001000)
```

Interrupt Mask on line 12

5.154.2.1526 EXTI_IMR_MR13

```
#define EXTI_IMR_MR13 ((uint32_t)0x00002000)
```

Interrupt Mask on line 13

5.154.2.1527 EXTI_IMR_MR14

```
#define EXTI_IMR_MR14 ((uint32_t)0x00004000)
```

Interrupt Mask on line 14

5.154.2.1528 EXTI_IMR_MR15

```
#define EXTI_IMR_MR15 ((uint32_t)0x00008000)
```

Interrupt Mask on line 15

5.154.2.1529 EXTI_IMR_MR16

```
#define EXTI_IMR_MR16 ((uint32_t)0x00010000)
```

Interrupt Mask on line 16

5.154.2.1530 EXTI_IMR_MR17

```
#define EXTI_IMR_MR17 ((uint32_t)0x00020000)
```

Interrupt Mask on line 17

5.154.2.1531 EXTI_IMR_MR18

```
#define EXTI_IMR_MR18 ((uint32_t)0x00040000)
```

Interrupt Mask on line 18

5.154.2.1532 EXTI_IMR_MR19

```
#define EXTI_IMR_MR19 ((uint32_t)0x00080000)
```

Interrupt Mask on line 19

5.154.2.1533 EXTI_IMR_MR2

```
#define EXTI_IMR_MR2 ((uint32_t)0x00000004)
```

Interrupt Mask on line 2

5.154.2.1534 EXTI_IMR_MR3

```
#define EXTI_IMR_MR3 ((uint32_t)0x00000008)
```

Interrupt Mask on line 3

5.154.2.1535 EXTI_IMR_MR4

```
#define EXTI_IMR_MR4 ((uint32_t)0x00000010)
```

Interrupt Mask on line 4

5.154.2.1536 EXTI_IMR_MR5

```
#define EXTI_IMR_MR5 ((uint32_t)0x00000020)
```

Interrupt Mask on line 5

5.154.2.1537 EXTI_IMR_MR6

```
#define EXTI_IMR_MR6 ((uint32_t)0x00000040)
```

Interrupt Mask on line 6

5.154.2.1538 EXTI_IMR_MR7

```
#define EXTI_IMR_MR7 ((uint32_t)0x00000080)
```

Interrupt Mask on line 7

5.154.2.1539 EXTI_IMR_MR8

```
#define EXTI_IMR_MR8 ((uint32_t)0x00000100)
```

Interrupt Mask on line 8

5.154.2.1540 EXTI_IMR_MR9

```
#define EXTI_IMR_MR9 ((uint32_t)0x00000200)
```

Interrupt Mask on line 9

5.154.2.1541 EXTI_PR_PR0

```
#define EXTI_PR_PR0 ((uint32_t)0x00000001)
```

Pending bit for line 0

5.154.2.1542 EXTI_PR_PR1

```
#define EXTI_PR_PR1 ((uint32_t)0x00000002)
```

Pending bit for line 1

5.154.2.1543 EXTI_PR_PR10

```
#define EXTI_PR_PR10 ((uint32_t)0x00000400)
```

Pending bit for line 10

5.154.2.1544 EXTI_PR_PR11

```
#define EXTI_PR_PR11 ((uint32_t)0x00000800)
```

Pending bit for line 11

5.154.2.1545 EXTI_PR_PR12

```
#define EXTI_PR_PR12 ((uint32_t)0x00001000)
```

Pending bit for line 12

5.154.2.1546 EXTI_PR_PR13

```
#define EXTI_PR_PR13 ((uint32_t)0x00002000)
```

Pending bit for line 13

5.154.2.1547 EXTI_PR_PR14

```
#define EXTI_PR_PR14 ((uint32_t)0x00004000)
```

Pending bit for line 14

5.154.2.1548 EXTI_PR_PR15

```
#define EXTI_PR_PR15 ((uint32_t)0x00008000)
```

Pending bit for line 15

5.154.2.1549 EXTI_PR_PR16

```
#define EXTI_PR_PR16 ((uint32_t)0x00010000)
```

Pending bit for line 16

5.154.2.1550 EXTI_PR_PR17

```
#define EXTI_PR_PR17 ((uint32_t)0x00020000)
```

Pending bit for line 17

5.154.2.1551 EXTI_PR_PR18

```
#define EXTI_PR_PR18 ((uint32_t)0x00040000)
```

Pending bit for line 18

5.154.2.1552 EXTI_PR_PR19

```
#define EXTI_PR_PR19 ((uint32_t)0x00080000)
```

Pending bit for line 19

5.154.2.1553 EXTI_PR_PR2

```
#define EXTI_PR_PR2 ((uint32_t)0x00000004)
```

Pending bit for line 2

5.154.2.1554 EXTI_PR_PR3

```
#define EXTI_PR_PR3 ((uint32_t)0x00000008)
```

Pending bit for line 3

5.154.2.1555 EXTI_PR_PR4

```
#define EXTI_PR_PR4 ((uint32_t)0x00000010)
```

Pending bit for line 4

5.154.2.1556 EXTI_PR_PR5

```
#define EXTI_PR_PR5 ((uint32_t)0x00000020)
```

Pending bit for line 5

5.154.2.1557 EXTI_PR_PR6

```
#define EXTI_PR_PR6 ((uint32_t)0x00000040)
```

Pending bit for line 6

5.154.2.1558 EXTI_PR_PR7

```
#define EXTI_PR_PR7 ((uint32_t)0x00000080)
```

Pending bit for line 7

5.154.2.1559 EXTI_PR_PR8

```
#define EXTI_PR_PR8 ((uint32_t)0x00000100)
```

Pending bit for line 8

5.154.2.1560 EXTI_PR_PR9

```
#define EXTI_PR_PR9 ((uint32_t)0x00000200)
```

Pending bit for line 9

5.154.2.1561 EXTI_RTSR_TR0

```
#define EXTI_RTSR_TR0 ((uint32_t)0x00000001)
```

Rising trigger event configuration bit of line 0

5.154.2.1562 EXTI_RTSR_TR1

```
#define EXTI_RTSR_TR1 ((uint32_t)0x00000002)
```

Rising trigger event configuration bit of line 1

5.154.2.1563 EXTI_RTSR_TR10

```
#define EXTI_RTSR_TR10 ((uint32_t)0x00000400)
```

Rising trigger event configuration bit of line 10

5.154.2.1564 EXTI_RTSR_TR11

```
#define EXTI_RTSR_TR11 ((uint32_t)0x00000800)
```

Rising trigger event configuration bit of line 11

5.154.2.1565 EXTI_RTSR_TR12

```
#define EXTI_RTSR_TR12 ((uint32_t)0x00001000)
```

Rising trigger event configuration bit of line 12

5.154.2.1566 EXTI_RTSR_TR13

```
#define EXTI_RTSR_TR13 ((uint32_t)0x00002000)
```

Rising trigger event configuration bit of line 13

5.154.2.1567 EXTI_RTSR_TR14

```
#define EXTI_RTSR_TR14 ((uint32_t)0x00004000)
```

Rising trigger event configuration bit of line 14

5.154.2.1568 EXTI_RTSR_TR15

```
#define EXTI_RTSR_TR15 ((uint32_t)0x00008000)
```

Rising trigger event configuration bit of line 15

5.154.2.1569 EXTI_RTSR_TR16

```
#define EXTI_RTSR_TR16 ((uint32_t)0x00010000)
```

Rising trigger event configuration bit of line 16

5.154.2.1570 EXTI_RTSR_TR17

```
#define EXTI_RTSR_TR17 ((uint32_t)0x00020000)
```

Rising trigger event configuration bit of line 17

5.154.2.1571 EXTI_RTSR_TR18

```
#define EXTI_RTSR_TR18 ((uint32_t)0x00040000)
```

Rising trigger event configuration bit of line 18

5.154.2.1572 EXTI_RTSR_TR19

```
#define EXTI_RTSR_TR19 ((uint32_t)0x00080000)
```

Rising trigger event configuration bit of line 19

5.154.2.1573 EXTI_RTSR_TR2

```
#define EXTI_RTSR_TR2 ((uint32_t)0x00000004)
```

Rising trigger event configuration bit of line 2

5.154.2.1574 EXTI_RTSR_TR3

```
#define EXTI_RTSR_TR3 ((uint32_t)0x00000008)
```

Rising trigger event configuration bit of line 3

5.154.2.1575 EXTI_RTSR_TR4

```
#define EXTI_RTSR_TR4 ((uint32_t)0x00000010)
```

Rising trigger event configuration bit of line 4

5.154.2.1576 EXTI_RTSR_TR5

```
#define EXTI_RTSR_TR5 ((uint32_t)0x00000020)
```

Rising trigger event configuration bit of line 5

5.154.2.1577 EXTI_RTSR_TR6

```
#define EXTI_RTSR_TR6 ((uint32_t)0x00000040)
```

Rising trigger event configuration bit of line 6

5.154.2.1578 EXTI_RTSR_TR7

```
#define EXTI_RTSR_TR7 ((uint32_t)0x00000080)
```

Rising trigger event configuration bit of line 7

5.154.2.1579 EXTI_RTSR_TR8

```
#define EXTI_RTSR_TR8 ((uint32_t)0x00000100)
```

Rising trigger event configuration bit of line 8

5.154.2.1580 EXTI_RTSR_TR9

```
#define EXTI_RTSR_TR9 ((uint32_t)0x00000200)
```

Rising trigger event configuration bit of line 9

5.154.2.1581 EXTI_SWIER_SWIER0

```
#define EXTI_SWIER_SWIER0 ((uint32_t)0x00000001)
```

Software Interrupt on line 0

5.154.2.1582 EXTI_SWIER_SWIER1

```
#define EXTI_SWIER_SWIER1 ((uint32_t)0x00000002)
```

Software Interrupt on line 1

5.154.2.1583 EXTI_SWIER_SWIER10

```
#define EXTI_SWIER_SWIER10 ((uint32_t)0x00000400)
```

Software Interrupt on line 10

5.154.2.1584 EXTI_SWIER_SWIER11

```
#define EXTI_SWIER_SWIER11 ((uint32_t)0x00000800)
```

Software Interrupt on line 11

5.154.2.1585 EXTI_SWIER_SWIER12

```
#define EXTI_SWIER_SWIER12 ((uint32_t)0x00001000)
```

Software Interrupt on line 12

5.154.2.1586 EXTI_SWIER_SWIER13

```
#define EXTI_SWIER_SWIER13 ((uint32_t)0x00002000)
```

Software Interrupt on line 13

5.154.2.1587 EXTI_SWIER_SWIER14

```
#define EXTI_SWIER_SWIER14 ((uint32_t)0x00004000)
```

Software Interrupt on line 14

5.154.2.1588 EXTI_SWIER_SWIER15

```
#define EXTI_SWIER_SWIER15 ((uint32_t)0x00008000)
```

Software Interrupt on line 15

5.154.2.1589 EXTI_SWIER_SWIER16

```
#define EXTI_SWIER_SWIER16 ((uint32_t)0x00010000)
```

Software Interrupt on line 16

5.154.2.1590 EXTI_SWIER_SWIER17

```
#define EXTI_SWIER_SWIER17 ((uint32_t)0x00020000)
```

Software Interrupt on line 17

5.154.2.1591 EXTI_SWIER_SWIER18

```
#define EXTI_SWIER_SWIER18 ((uint32_t)0x00040000)
```

Software Interrupt on line 18

5.154.2.1592 EXTI_SWIER_SWIER19

```
#define EXTI_SWIER_SWIER19 ((uint32_t)0x00080000)
```

Software Interrupt on line 19

5.154.2.1593 EXTI_SWIER_SWIER2

```
#define EXTI_SWIER_SWIER2 ((uint32_t)0x00000004)
```

Software Interrupt on line 2

5.154.2.1594 EXTI_SWIER_SWIER3

```
#define EXTI_SWIER_SWIER3 ((uint32_t)0x00000008)
```

Software Interrupt on line 3

5.154.2.1595 EXTI_SWIER_SWIER4

```
#define EXTI_SWIER_SWIER4 ((uint32_t)0x00000010)
```

Software Interrupt on line 4

5.154.2.1596 EXTI_SWIER_SWIER5

```
#define EXTI_SWIER_SWIER5 ((uint32_t)0x00000020)
```

Software Interrupt on line 5

5.154.2.1597 EXTI_SWIER_SWIER6

```
#define EXTI_SWIER_SWIER6 ((uint32_t)0x00000040)
```

Software Interrupt on line 6

5.154.2.1598 EXTI_SWIER_SWIER7

```
#define EXTI_SWIER_SWIER7 ((uint32_t)0x00000080)
```

Software Interrupt on line 7

5.154.2.1599 EXTI_SWIER_SWIER8

```
#define EXTI_SWIER_SWIER8 ((uint32_t)0x00000100)
```

Software Interrupt on line 8

5.154.2.1600 EXTI_SWIER_SWIER9

```
#define EXTI_SWIER_SWIER9 ((uint32_t)0x00000200)
```

Software Interrupt on line 9

5.154.2.1601 FSMC_BCR1_ASYNCWAIT

```
#define FSMC_BCR1_ASYNCWAIT ((uint32_t)0x00008000)
```

Asynchronous wait

5.154.2.1602 FSMC_BCR1_BURSTEN

```
#define FSMC_BCR1_BURSTEN ((uint32_t)0x00000100)
```

Burst enable bit

5.154.2.1603 FSMC_BCR1_CBURSTRW

```
#define FSMC_BCR1_CBURSTRW ((uint32_t)0x00080000)
```

Write burst enable

5.154.2.1604 FSMC_BCR1_EXTMOD

```
#define FSMC_BCR1_EXTMOD ((uint32_t)0x00004000)
```

Extended mode enable

5.154.2.1605 FSMC_BCR1_FACCEN

```
#define FSMC_BCR1_FACCEN ((uint32_t)0x00000040)
```

Flash access enable

5.154.2.1606 FSMC_BCR1_MBKEN

```
#define FSMC_BCR1_MBKEN ((uint32_t)0x00000001)
```

Memory bank enable bit

5.154.2.1607 FSMC_BCR1_MTYP

```
#define FSMC_BCR1_MTYP ((uint32_t)0x0000000C)
```

MTYP[1:0] bits (Memory type)

5.154.2.1608 FSMC_BCR1_MTYP_0

```
#define FSMC_BCR1_MTYP_0 ((uint32_t)0x00000004)
```

Bit 0

5.154.2.1609 FSMC_BCR1_MTYP_1

```
#define FSMC_BCR1_MTYP_1 ((uint32_t)0x00000008)
```

Bit 1

5.154.2.1610 FSMC_BCR1_MUXEN

```
#define FSMC_BCR1_MUXEN ((uint32_t)0x00000002)
```

Address/data multiplexing enable bit

5.154.2.1611 FSMC_BCR1_MWID

```
#define FSMC_BCR1_MWID ((uint32_t)0x00000030)
```

MWID[1:0] bits (Memory data bus width)

5.154.2.1612 FSMC_BCR1_MWID_0

```
#define FSMC_BCR1_MWID_0 ((uint32_t)0x00000010)
```

Bit 0

5.154.2.1613 FSMC_BCR1_MWID_1

```
#define FSMC_BCR1_MWID_1 ((uint32_t)0x00000020)
```

Bit 1

5.154.2.1614 FSMC_BCR1_WAITCFG

```
#define FSMC_BCR1_WAITCFG ((uint32_t)0x00000800)
```

Wait timing configuration

5.154.2.1615 FSMC_BCR1_WAITEN

```
#define FSMC_BCR1_WAITEN ((uint32_t)0x00002000)
```

Wait enable bit

5.154.2.1616 FSMC_BCR1_WAITPOL

```
#define FSMC_BCR1_WAITPOL ((uint32_t)0x00000200)
```

Wait signal polarity bit

5.154.2.1617 FSMC_BCR1_WRAPMOD

```
#define FSMC_BCR1_WRAPMOD ((uint32_t)0x00000400)
```

Wrapped burst mode support

5.154.2.1618 FSMC_BCR1_WREN

```
#define FSMC_BCR1_WREN ((uint32_t)0x00001000)
```

Write enable bit

5.154.2.1619 FSMC_BCR2_ASYNCWAIT

```
#define FSMC_BCR2_ASYNCWAIT ((uint32_t)0x00008000)
```

Asynchronous wait

5.154.2.1620 FSMC_BCR2_BURSTEN

```
#define FSMC_BCR2_BURSTEN ((uint32_t)0x00000100)
```

Burst enable bit

5.154.2.1621 FSMC_BCR2_CBURSTRW

```
#define FSMC_BCR2_CBURSTRW ((uint32_t)0x00080000)
```

Write burst enable

5.154.2.1622 FSMC_BCR2_EXTMOD

```
#define FSMC_BCR2_EXTMOD ((uint32_t)0x00004000)
```

Extended mode enable

5.154.2.1623 FSMC_BCR2_FACCEN

```
#define FSMC_BCR2_FACCEN ((uint32_t)0x00000040)
```

Flash access enable

5.154.2.1624 FSMC_BCR2_MBKEN

```
#define FSMC_BCR2_MBKEN ((uint32_t)0x00000001)
```

Memory bank enable bit

5.154.2.1625 FSMC_BCR2_MTYP

```
#define FSMC_BCR2_MTYP ((uint32_t)0x0000000C)
```

MTYP[1:0] bits (Memory type)

5.154.2.1626 FSMC_BCR2_MTYP_0

```
#define FSMC_BCR2_MTYP_0 ((uint32_t)0x00000004)
```

Bit 0

5.154.2.1627 FSMC_BCR2_MTYP_1

```
#define FSMC_BCR2_MTYP_1 ((uint32_t)0x00000008)
```

Bit 1

5.154.2.1628 FSMC_BCR2_MUXEN

```
#define FSMC_BCR2_MUXEN ((uint32_t)0x00000002)
```

Address/data multiplexing enable bit

5.154.2.1629 FSMC_BCR2_MWID

```
#define FSMC_BCR2_MWID ((uint32_t)0x00000030)
```

MWID[1:0] bits (Memory data bus width)

5.154.2.1630 FSMC_BCR2_MWID_0

```
#define FSMC_BCR2_MWID_0 ((uint32_t)0x00000010)
```

Bit 0

5.154.2.1631 FSMC_BCR2_MWID_1

```
#define FSMC_BCR2_MWID_1 ((uint32_t)0x00000020)
```

Bit 1

5.154.2.1632 FSMC_BCR2_WAITCFG

```
#define FSMC_BCR2_WAITCFG ((uint32_t)0x00000800)
```

Wait timing configuration

5.154.2.1633 FSMC_BCR2_WAITEN

```
#define FSMC_BCR2_WAITEN ((uint32_t)0x00002000)
```

Wait enable bit

5.154.2.1634 FSMC_BCR2_WAITPOL

```
#define FSMC_BCR2_WAITPOL ((uint32_t)0x00000200)
```

Wait signal polarity bit

5.154.2.1635 FSMC_BCR2_WRAPMOD

```
#define FSMC_BCR2_WRAPMOD ((uint32_t)0x00000400)
```

Wrapped burst mode support

5.154.2.1636 FSMC_BCR2_WREN

```
#define FSMC_BCR2_WREN ((uint32_t)0x00001000)
```

Write enable bit

5.154.2.1637 FSMC_BCR3_ASYNCWAIT

```
#define FSMC_BCR3_ASYNCWAIT ((uint32_t)0x00008000)
```

Asynchronous wait

5.154.2.1638 FSMC_BCR3_BURSTEN

```
#define FSMC_BCR3_BURSTEN ((uint32_t)0x00000100)
```

Burst enable bit

5.154.2.1639 FSMC_BCR3_CBURSTRW

```
#define FSMC_BCR3_CBURSTRW ((uint32_t)0x00080000)
```

Write burst enable

5.154.2.1640 FSMC_BCR3_EXTMOD

```
#define FSMC_BCR3_EXTMOD ((uint32_t)0x00004000)
```

Extended mode enable

5.154.2.1641 FSMC_BCR3_FACCEN

```
#define FSMC_BCR3_FACCEN ((uint32_t)0x00000040)
```

Flash access enable

5.154.2.1642 FSMC_BCR3_MBKEN

```
#define FSMC_BCR3_MBKEN ((uint32_t)0x00000001)
```

Memory bank enable bit

5.154.2.1643 FSMC_BCR3_MTYP

```
#define FSMC_BCR3_MTYP ((uint32_t)0x0000000C)
```

MTYP[1:0] bits (Memory type)

5.154.2.1644 FSMC_BCR3_MTYP_0

```
#define FSMC_BCR3_MTYP_0 ((uint32_t)0x00000004)
```

Bit 0

5.154.2.1645 FSMC_BCR3_MTYP_1

```
#define FSMC_BCR3_MTYP_1 ((uint32_t)0x00000008)
```

Bit 1

5.154.2.1646 FSMC_BCR3_MUXEN

```
#define FSMC_BCR3_MUXEN ((uint32_t)0x00000002)
```

Address/data multiplexing enable bit

5.154.2.1647 FSMC_BCR3_MWID

```
#define FSMC_BCR3_MWID ((uint32_t)0x00000030)
```

MWID[1:0] bits (Memory data bus width)

5.154.2.1648 FSMC_BCR3_MWID_0

```
#define FSMC_BCR3_MWID_0 ((uint32_t)0x00000010)
```

Bit 0

5.154.2.1649 FSMC_BCR3_MWID_1

```
#define FSMC_BCR3_MWID_1 ((uint32_t)0x00000020)
```

Bit 1

5.154.2.1650 FSMC_BCR3_WAITCFG

```
#define FSMC_BCR3_WAITCFG ((uint32_t)0x00000800)
```

Wait timing configuration

5.154.2.1651 FSMC_BCR3_WAITEN

```
#define FSMC_BCR3_WAITEN ((uint32_t)0x00002000)
```

Wait enable bit

5.154.2.1652 FSMC_BCR3_WAITPOL

```
#define FSMC_BCR3_WAITPOL ((uint32_t)0x00000200)
```

Wait signal polarity bit.

5.154.2.1653 FSMC_BCR3_WRAPMOD

```
#define FSMC_BCR3_WRAPMOD ((uint32_t)0x00000400)
```

Wrapped burst mode support

5.154.2.1654 FSMC_BCR3_WREN

```
#define FSMC_BCR3_WREN ((uint32_t)0x00001000)
```

Write enable bit

5.154.2.1655 FSMC_BCR4_ASYNCWAIT

```
#define FSMC_BCR4_ASYNCWAIT ((uint32_t)0x00008000)
```

Asynchronous wait

5.154.2.1656 FSMC_BCR4_BURSTEN

```
#define FSMC_BCR4_BURSTEN ((uint32_t)0x00000100)
```

Burst enable bit

5.154.2.1657 FSMC_BCR4_CBURSTRW

```
#define FSMC_BCR4_CBURSTRW ((uint32_t)0x00080000)
```

Write burst enable

5.154.2.1658 FSMC_BCR4_EXTMOD

```
#define FSMC_BCR4_EXTMOD ((uint32_t)0x00004000)
```

Extended mode enable

5.154.2.1659 FSMC_BCR4_FACCEN

```
#define FSMC_BCR4_FACCEN ((uint32_t)0x00000040)
```

Flash access enable

5.154.2.1660 FSMC_BCR4_MBKEN

```
#define FSMC_BCR4_MBKEN ((uint32_t)0x00000001)
```

Memory bank enable bit

5.154.2.1661 FSMC_BCR4_MTYP

```
#define FSMC_BCR4_MTYP ((uint32_t)0x0000000C)
```

MTYP[1:0] bits (Memory type)

5.154.2.1662 FSMC_BCR4_MTYP_0

```
#define FSMC_BCR4_MTYP_0 ((uint32_t)0x00000004)
```

Bit 0

5.154.2.1663 FSMC_BCR4_MTYP_1

```
#define FSMC_BCR4_MTYP_1 ((uint32_t)0x00000008)
```

Bit 1

5.154.2.1664 FSMC_BCR4_MUXEN

```
#define FSMC_BCR4_MUXEN ((uint32_t)0x00000002)
```

Address/data multiplexing enable bit

5.154.2.1665 FSMC_BCR4_MWID

```
#define FSMC_BCR4_MWID ((uint32_t)0x00000030)
```

MWID[1:0] bits (Memory data bus width)

5.154.2.1666 FSMC_BCR4_MWID_0

```
#define FSMC_BCR4_MWID_0 ((uint32_t)0x00000010)
```

Bit 0

5.154.2.1667 FSMC_BCR4_MWID_1

```
#define FSMC_BCR4_MWID_1 ((uint32_t)0x00000020)
```

Bit 1

5.154.2.1668 FSMC_BCR4_WAITCFG

```
#define FSMC_BCR4_WAITCFG ((uint32_t)0x00000800)
```

Wait timing configuration

5.154.2.1669 FSMC_BCR4_WAITEN

```
#define FSMC_BCR4_WAITEN ((uint32_t)0x00002000)
```

Wait enable bit

5.154.2.1670 FSMC_BCR4_WAITPOL

```
#define FSMC_BCR4_WAITPOL ((uint32_t)0x00000200)
```

Wait signal polarity bit

5.154.2.1671 FSMC_BCR4_WRAPMOD

```
#define FSMC_BCR4_WRAPMOD ((uint32_t)0x00000400)
```

Wrapped burst mode support

5.154.2.1672 FSMC_BCR4_WREN

```
#define FSMC_BCR4_WREN ((uint32_t)0x00001000)
```

Write enable bit

5.154.2.1673 FSMC_BTR1_ACCMOD

```
#define FSMC_BTR1_ACCMOD ((uint32_t)0x30000000)
```

ACCMOD[1:0] bits (Access mode)

5.154.2.1674 FSMC_BTR1_ACCMOD_0

```
#define FSMC_BTR1_ACCMOD_0 ((uint32_t)0x10000000)
```

Bit 0

5.154.2.1675 FSMC_BTR1_ACCMOD_1

```
#define FSMC_BTR1_ACCMOD_1 ((uint32_t)0x20000000)
```

Bit 1

5.154.2.1676 FSMC_BTR1_ADDHLD

```
#define FSMC_BTR1_ADDHLD ((uint32_t)0x000000F0)
```

ADDHLD[3:0] bits (Address-hold phase duration)

5.154.2.1677 FSMC_BTR1_ADDHLD_0

```
#define FSMC_BTR1_ADDHLD_0 ((uint32_t)0x00000010)
```

Bit 0

5.154.2.1678 FSMC_BTR1_ADDHLD_1

```
#define FSMC_BTR1_ADDHLD_1 ((uint32_t)0x00000020)
```

Bit 1

5.154.2.1679 FSMC_BTR1_ADDHLD_2

```
#define FSMC_BTR1_ADDHLD_2 ((uint32_t)0x00000040)
```

Bit 2

5.154.2.1680 FSMC_BTR1_ADDHLD_3

```
#define FSMC_BTR1_ADDHLD_3 ((uint32_t)0x00000080)
```

Bit 3

5.154.2.1681 FSMC_BTR1_ADDSET

```
#define FSMC_BTR1_ADDSET ((uint32_t)0x0000000F)
```

ADDSET[3:0] bits (Address setup phase duration)

5.154.2.1682 FSMC_BTR1_ADDSET_0

```
#define FSMC_BTR1_ADDSET_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.1683 FSMC_BTR1_ADDSET_1

```
#define FSMC_BTR1_ADDSET_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.1684 FSMC_BTR1_ADDSET_2

```
#define FSMC_BTR1_ADDSET_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.1685 FSMC_BTR1_ADDSET_3

```
#define FSMC_BTR1_ADDSET_3 ((uint32_t)0x00000008)
```

Bit 3

5.154.2.1686 FSMC_BTR1_BUSTURN

```
#define FSMC_BTR1_BUSTURN ((uint32_t)0x000F0000)
```

BUSTURN[3:0] bits (Bus turnaround phase duration)

5.154.2.1687 FSMC_BTR1_BUSTURN_0

```
#define FSMC_BTR1_BUSTURN_0 ((uint32_t)0x00010000)
```

Bit 0

5.154.2.1688 FSMC_BTR1_BUSTURN_1

```
#define FSMC_BTR1_BUSTURN_1 ((uint32_t)0x00020000)
```

Bit 1

5.154.2.1689 FSMC_BTR1_BUSTURN_2

```
#define FSMC_BTR1_BUSTURN_2 ((uint32_t)0x00040000)
```

Bit 2

5.154.2.1690 FSMC_BTR1_BUSTURN_3

```
#define FSMC_BTR1_BUSTURN_3 ((uint32_t)0x00080000)
```

Bit 3

5.154.2.1691 FSMC_BTR1_CLKDIV

```
#define FSMC_BTR1_CLKDIV ((uint32_t)0x00F00000)
```

CLKDIV[3:0] bits (Clock divide ratio)

5.154.2.1692 FSMC_BTR1_CLKDIV_0

```
#define FSMC_BTR1_CLKDIV_0 ((uint32_t)0x00100000)
```

Bit 0

5.154.2.1693 FSMC_BTR1_CLKDIV_1

```
#define FSMC_BTR1_CLKDIV_1 ((uint32_t)0x00200000)
```

Bit 1

5.154.2.1694 FSMC_BTR1_CLKDIV_2

```
#define FSMC_BTR1_CLKDIV_2 ((uint32_t)0x00400000)
```

Bit 2

5.154.2.1695 FSMC_BTR1_CLKDIV_3

```
#define FSMC_BTR1_CLKDIV_3 ((uint32_t)0x00800000)
```

Bit 3

5.154.2.1696 FSMC_BTR1_DATAST

```
#define FSMC_BTR1_DATAST ((uint32_t)0x0000FF00)
```

DATAST [3:0] bits (Data-phase duration)

5.154.2.1697 FSMC_BTR1_DATAST_0

```
#define FSMC_BTR1_DATAST_0 ((uint32_t)0x00000100)
```

Bit 0

5.154.2.1698 FSMC_BTR1_DATAST_1

```
#define FSMC_BTR1_DATAST_1 ((uint32_t)0x00000200)
```

Bit 1

5.154.2.1699 FSMC_BTR1_DATAST_2

```
#define FSMC_BTR1_DATAST_2 ((uint32_t)0x00000400)
```

Bit 2

5.154.2.1700 FSMC_BTR1_DATAST_3

```
#define FSMC_BTR1_DATAST_3 ((uint32_t)0x00000800)
```

Bit 3

5.154.2.1701 FSMC_BTR1_DATLAT

```
#define FSMC_BTR1_DATLAT ((uint32_t)0x0F000000)
```

DATLA[3:0] bits (Data latency)

5.154.2.1702 FSMC_BTR1_DATLAT_0

```
#define FSMC_BTR1_DATLAT_0 ((uint32_t)0x01000000)
```

Bit 0

5.154.2.1703 FSMC_BTR1_DATLAT_1

```
#define FSMC_BTR1_DATLAT_1 ((uint32_t)0x02000000)
```

Bit 1

5.154.2.1704 FSMC_BTR1_DATLAT_2

```
#define FSMC_BTR1_DATLAT_2 ((uint32_t)0x04000000)
```

Bit 2

5.154.2.1705 FSMC_BTR1_DATLAT_3

```
#define FSMC_BTR1_DATLAT_3 ((uint32_t)0x08000000)
```

Bit 3

5.154.2.1706 FSMC_BTR2_ACCMOD

```
#define FSMC_BTR2_ACCMOD ((uint32_t)0x30000000)
```

ACCMOD[1:0] bits (Access mode)

5.154.2.1707 FSMC_BTR2_ACCMOD_0

```
#define FSMC_BTR2_ACCMOD_0 ((uint32_t)0x10000000)
```

Bit 0

5.154.2.1708 FSMC_BTR2_ACCMOD_1

```
#define FSMC_BTR2_ACCMOD_1 ((uint32_t)0x20000000)
```

Bit 1

5.154.2.1709 FSMC_BTR2_ADDHLD

```
#define FSMC_BTR2_ADDHLD ((uint32_t)0x000000F0)
```

ADDHLD[3:0] bits (Address-hold phase duration)

5.154.2.1710 FSMC_BTR2_ADDHLD_0

```
#define FSMC_BTR2_ADDHLD_0 ((uint32_t)0x00000010)
```

Bit 0

5.154.2.1711 FSMC_BTR2_ADDHLD_1

```
#define FSMC_BTR2_ADDHLD_1 ((uint32_t)0x00000020)
```

Bit 1

5.154.2.1712 FSMC_BTR2_ADDHLD_2

```
#define FSMC_BTR2_ADDHLD_2 ((uint32_t)0x00000040)
```

Bit 2

5.154.2.1713 FSMC_BTR2_ADDHLD_3

```
#define FSMC_BTR2_ADDHLD_3 ((uint32_t)0x00000080)
```

Bit 3

5.154.2.1714 FSMC_BTR2_ADDSET

```
#define FSMC_BTR2_ADDSET ((uint32_t)0x0000000F)
```

ADDSET[3:0] bits (Address setup phase duration)

5.154.2.1715 FSMC_BTR2_ADDSET_0

```
#define FSMC_BTR2_ADDSET_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.1716 FSMC_BTR2_ADDSET_1

```
#define FSMC_BTR2_ADDSET_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.1717 FSMC_BTR2_ADDSET_2

```
#define FSMC_BTR2_ADDSET_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.1718 FSMC_BTR2_ADDSET_3

```
#define FSMC_BTR2_ADDSET_3 ((uint32_t)0x00000008)
```

Bit 3

5.154.2.1719 FSMC_BTR2_BUSTURN

```
#define FSMC_BTR2_BUSTURN ((uint32_t)0x000F0000)
```

BUSTURN[3:0] bits (Bus turnaround phase duration)

5.154.2.1720 FSMC_BTR2_BUSTURN_0

```
#define FSMC_BTR2_BUSTURN_0 ((uint32_t)0x00010000)
```

Bit 0

5.154.2.1721 FSMC_BTR2_BUSTURN_1

```
#define FSMC_BTR2_BUSTURN_1 ((uint32_t)0x00020000)
```

Bit 1

5.154.2.1722 FSMC_BTR2_BUSTURN_2

```
#define FSMC_BTR2_BUSTURN_2 ((uint32_t)0x00040000)
```

Bit 2

5.154.2.1723 FSMC_BTR2_BUSTURN_3

```
#define FSMC_BTR2_BUSTURN_3 ((uint32_t)0x00080000)
```

Bit 3

5.154.2.1724 FSMC_BTR2_CLKDIV

```
#define FSMC_BTR2_CLKDIV ((uint32_t)0x00F00000)
```

CLKDIV[3:0] bits (Clock divide ratio)

5.154.2.1725 FSMC_BTR2_CLKDIV_0

```
#define FSMC_BTR2_CLKDIV_0 ((uint32_t)0x00100000)
```

Bit 0

5.154.2.1726 FSMC_BTR2_CLKDIV_1

```
#define FSMC_BTR2_CLKDIV_1 ((uint32_t)0x00200000)
```

Bit 1

5.154.2.1727 FSMC_BTR2_CLKDIV_2

```
#define FSMC_BTR2_CLKDIV_2 ((uint32_t)0x00400000)
```

Bit 2

5.154.2.1728 FSMC_BTR2_CLKDIV_3

```
#define FSMC_BTR2_CLKDIV_3 ((uint32_t)0x00800000)
```

Bit 3

5.154.2.1729 FSMC_BTR2_DATAST

```
#define FSMC_BTR2_DATAST ((uint32_t)0x0000FF00)
```

DATAST [3:0] bits (Data-phase duration)

5.154.2.1730 FSMC_BTR2_DATAST_0

```
#define FSMC_BTR2_DATAST_0 ((uint32_t)0x00000100)
```

Bit 0

5.154.2.1731 FSMC_BTR2_DATAST_1

```
#define FSMC_BTR2_DATAST_1 ((uint32_t)0x00000200)
```

Bit 1

5.154.2.1732 FSMC_BTR2_DATAST_2

```
#define FSMC_BTR2_DATAST_2 ((uint32_t)0x00000400)
```

Bit 2

5.154.2.1733 FSMC_BTR2_DATAST_3

```
#define FSMC_BTR2_DATAST_3 ((uint32_t)0x00000800)
```

Bit 3

5.154.2.1734 FSMC_BTR2_DATLAT

```
#define FSMC_BTR2_DATLAT ((uint32_t)0x0F000000)
```

DATLA[3:0] bits (Data latency)

5.154.2.1735 FSMC_BTR2_DATLAT_0

```
#define FSMC_BTR2_DATLAT_0 ((uint32_t)0x01000000)
```

Bit 0

5.154.2.1736 FSMC_BTR2_DATLAT_1

```
#define FSMC_BTR2_DATLAT_1 ((uint32_t)0x02000000)
```

Bit 1

5.154.2.1737 FSMC_BTR2_DATLAT_2

```
#define FSMC_BTR2_DATLAT_2 ((uint32_t)0x04000000)
```

Bit 2

5.154.2.1738 FSMC_BTR2_DATLAT_3

```
#define FSMC_BTR2_DATLAT_3 ((uint32_t)0x08000000)
```

Bit 3

5.154.2.1739 FSMC_BTR3_ACCMOD

```
#define FSMC_BTR3_ACCMOD ((uint32_t)0x30000000)
```

ACCMOD[1:0] bits (Access mode)

5.154.2.1740 FSMC_BTR3_ACCMOD_0

```
#define FSMC_BTR3_ACCMOD_0 ((uint32_t)0x10000000)
```

Bit 0

5.154.2.1741 FSMC_BTR3_ACCMOD_1

```
#define FSMC_BTR3_ACCMOD_1 ((uint32_t)0x20000000)
```

Bit 1

5.154.2.1742 FSMC_BTR3_ADDHLD

```
#define FSMC_BTR3_ADDHLD ((uint32_t)0x000000F0)
```

ADDHLD[3:0] bits (Address-hold phase duration)

5.154.2.1743 FSMC_BTR3_ADDHLD_0

```
#define FSMC_BTR3_ADDHLD_0 ((uint32_t)0x00000010)
```

Bit 0

5.154.2.1744 FSMC_BTR3_ADDHLD_1

```
#define FSMC_BTR3_ADDHLD_1 ((uint32_t)0x00000020)
```

Bit 1

5.154.2.1745 FSMC_BTR3_ADDHLD_2

```
#define FSMC_BTR3_ADDHLD_2 ((uint32_t)0x00000040)
```

Bit 2

5.154.2.1746 FSMC_BTR3_ADDHLD_3

```
#define FSMC_BTR3_ADDHLD_3 ((uint32_t)0x00000080)
```

Bit 3

5.154.2.1747 FSMC_BTR3_ADDSET

```
#define FSMC_BTR3_ADDSET ((uint32_t)0x0000000F)
```

ADDSET[3:0] bits (Address setup phase duration)

5.154.2.1748 FSMC_BTR3_ADDSET_0

```
#define FSMC_BTR3_ADDSET_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.1749 FSMC_BTR3_ADDSET_1

```
#define FSMC_BTR3_ADDSET_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.1750 FSMC_BTR3_ADDSET_2

```
#define FSMC_BTR3_ADDSET_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.1751 FSMC_BTR3_ADDSET_3

```
#define FSMC_BTR3_ADDSET_3 ((uint32_t)0x00000008)
```

Bit 3

5.154.2.1752 FSMC_BTR3_BUSTURN

```
#define FSMC_BTR3_BUSTURN ((uint32_t)0x000F0000)
```

BUSTURN[3:0] bits (Bus turnaround phase duration)

5.154.2.1753 FSMC_BTR3_BUSTURN_0

```
#define FSMC_BTR3_BUSTURN_0 ((uint32_t)0x00010000)
```

Bit 0

5.154.2.1754 FSMC_BTR3_BUSTURN_1

```
#define FSMC_BTR3_BUSTURN_1 ((uint32_t)0x00020000)
```

Bit 1

5.154.2.1755 FSMC_BTR3_BUSTURN_2

```
#define FSMC_BTR3_BUSTURN_2 ((uint32_t)0x00040000)
```

Bit 2

5.154.2.1756 FSMC_BTR3_BUSTURN_3

```
#define FSMC_BTR3_BUSTURN_3 ((uint32_t)0x00080000)
```

Bit 3

5.154.2.1757 FSMC_BTR3_CLKDIV

```
#define FSMC_BTR3_CLKDIV ((uint32_t)0x00F00000)
```

CLKDIV[3:0] bits (Clock divide ratio)

5.154.2.1758 FSMC_BTR3_CLKDIV_0

```
#define FSMC_BTR3_CLKDIV_0 ((uint32_t)0x00100000)
```

Bit 0

5.154.2.1759 FSMC_BTR3_CLKDIV_1

```
#define FSMC_BTR3_CLKDIV_1 ((uint32_t)0x00200000)
```

Bit 1

5.154.2.1760 FSMC_BTR3_CLKDIV_2

```
#define FSMC_BTR3_CLKDIV_2 ((uint32_t)0x00400000)
```

Bit 2

5.154.2.1761 FSMC_BTR3_CLKDIV_3

```
#define FSMC_BTR3_CLKDIV_3 ((uint32_t)0x00800000)
```

Bit 3

5.154.2.1762 FSMC_BTR3_DATAST

```
#define FSMC_BTR3_DATAST ((uint32_t)0x0000FF00)
```

DATAST [3:0] bits (Data-phase duration)

5.154.2.1763 FSMC_BTR3_DATAST_0

```
#define FSMC_BTR3_DATAST_0 ((uint32_t)0x00000100)
```

Bit 0

5.154.2.1764 FSMC_BTR3_DATAST_1

```
#define FSMC_BTR3_DATAST_1 ((uint32_t)0x00000200)
```

Bit 1

5.154.2.1765 FSMC_BTR3_DATAST_2

```
#define FSMC_BTR3_DATAST_2 ((uint32_t)0x00000400)
```

Bit 2

5.154.2.1766 FSMC_BTR3_DATAST_3

```
#define FSMC_BTR3_DATAST_3 ((uint32_t)0x00000800)
```

Bit 3

5.154.2.1767 FSMC_BTR3_DATLAT

```
#define FSMC_BTR3_DATLAT ((uint32_t)0x0F000000)
```

DATLA[3:0] bits (Data latency)

5.154.2.1768 FSMC_BTR3_DATLAT_0

```
#define FSMC_BTR3_DATLAT_0 ((uint32_t)0x01000000)
```

Bit 0

5.154.2.1769 FSMC_BTR3_DATLAT_1

```
#define FSMC_BTR3_DATLAT_1 ((uint32_t)0x02000000)
```

Bit 1

5.154.2.1770 FSMC_BTR3_DATLAT_2

```
#define FSMC_BTR3_DATLAT_2 ((uint32_t)0x04000000)
```

Bit 2

5.154.2.1771 FSMC_BTR3_DATLAT_3

```
#define FSMC_BTR3_DATLAT_3 ((uint32_t)0x08000000)
```

Bit 3

5.154.2.1772 FSMC_BTR4_ACCMOD

```
#define FSMC_BTR4_ACCMOD ((uint32_t)0x30000000)
```

ACCMOD[1:0] bits (Access mode)

5.154.2.1773 FSMC_BTR4_ACCMOD_0

```
#define FSMC_BTR4_ACCMOD_0 ((uint32_t)0x10000000)
```

Bit 0

5.154.2.1774 FSMC_BTR4_ACCMOD_1

```
#define FSMC_BTR4_ACCMOD_1 ((uint32_t)0x20000000)
```

Bit 1

5.154.2.1775 FSMC_BTR4_ADDHLD

```
#define FSMC_BTR4_ADDHLD ((uint32_t)0x000000F0)
```

ADDHLD[3:0] bits (Address-hold phase duration)

5.154.2.1776 FSMC_BTR4_ADDHLD_0

```
#define FSMC_BTR4_ADDHLD_0 ((uint32_t)0x00000010)
```

Bit 0

5.154.2.1777 FSMC_BTR4_ADDHLD_1

```
#define FSMC_BTR4_ADDHLD_1 ((uint32_t)0x00000020)
```

Bit 1

5.154.2.1778 FSMC_BTR4_ADDHLD_2

```
#define FSMC_BTR4_ADDHLD_2 ((uint32_t)0x00000040)
```

Bit 2

5.154.2.1779 FSMC_BTR4_ADDHLD_3

```
#define FSMC_BTR4_ADDHLD_3 ((uint32_t)0x00000080)
```

Bit 3

5.154.2.1780 FSMC_BTR4_ADDSET

```
#define FSMC_BTR4_ADDSET ((uint32_t)0x0000000F)
```

ADDSET[3:0] bits (Address setup phase duration)

5.154.2.1781 FSMC_BTR4_ADDSET_0

```
#define FSMC_BTR4_ADDSET_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.1782 FSMC_BTR4_ADDSET_1

```
#define FSMC_BTR4_ADDSET_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.1783 FSMC_BTR4_ADDSET_2

```
#define FSMC_BTR4_ADDSET_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.1784 FSMC_BTR4_ADDSET_3

```
#define FSMC_BTR4_ADDSET_3 ((uint32_t)0x00000008)
```

Bit 3

5.154.2.1785 FSMC_BTR4_BUSTURN

```
#define FSMC_BTR4_BUSTURN ((uint32_t)0x000F0000)
```

BUSTURN[3:0] bits (Bus turnaround phase duration)

5.154.2.1786 FSMC_BTR4_BUSTURN_0

```
#define FSMC_BTR4_BUSTURN_0 ((uint32_t)0x00010000)
```

Bit 0

5.154.2.1787 FSMC_BTR4_BUSTURN_1

```
#define FSMC_BTR4_BUSTURN_1 ((uint32_t)0x00020000)
```

Bit 1

5.154.2.1788 FSMC_BTR4_BUSTURN_2

```
#define FSMC_BTR4_BUSTURN_2 ((uint32_t)0x00040000)
```

Bit 2

5.154.2.1789 FSMC_BTR4_BUSTURN_3

```
#define FSMC_BTR4_BUSTURN_3 ((uint32_t)0x00080000)
```

Bit 3

5.154.2.1790 FSMC_BTR4_CLKDIV

```
#define FSMC_BTR4_CLKDIV ((uint32_t)0x00F00000)
```

CLKDIV[3:0] bits (Clock divide ratio)

5.154.2.1791 FSMC_BTR4_CLKDIV_0

```
#define FSMC_BTR4_CLKDIV_0 ((uint32_t)0x00100000)
```

Bit 0

5.154.2.1792 FSMC_BTR4_CLKDIV_1

```
#define FSMC_BTR4_CLKDIV_1 ((uint32_t)0x00200000)
```

Bit 1

5.154.2.1793 FSMC_BTR4_CLKDIV_2

```
#define FSMC_BTR4_CLKDIV_2 ((uint32_t)0x00400000)
```

Bit 2

5.154.2.1794 FSMC_BTR4_CLKDIV_3

```
#define FSMC_BTR4_CLKDIV_3 ((uint32_t)0x00800000)
```

Bit 3

5.154.2.1795 FSMC_BTR4_DATAST

```
#define FSMC_BTR4_DATAST ((uint32_t)0x0000FF00)
```

DATAST [3:0] bits (Data-phase duration)

5.154.2.1796 FSMC_BTR4_DATAST_0

```
#define FSMC_BTR4_DATAST_0 ((uint32_t)0x00000100)
```

Bit 0

5.154.2.1797 FSMC_BTR4_DATAST_1

```
#define FSMC_BTR4_DATAST_1 ((uint32_t)0x00000200)
```

Bit 1

5.154.2.1798 FSMC_BTR4_DATAST_2

```
#define FSMC_BTR4_DATAST_2 ((uint32_t)0x00000400)
```

Bit 2

5.154.2.1799 FSMC_BTR4_DATAST_3

```
#define FSMC_BTR4_DATAST_3 ((uint32_t)0x00000800)
```

Bit 3

5.154.2.1800 FSMC_BTR4_DATLAT

```
#define FSMC_BTR4_DATLAT ((uint32_t)0x0F000000)
```

DATLA[3:0] bits (Data latency)

5.154.2.1801 FSMC_BTR4_DATLAT_0

```
#define FSMC_BTR4_DATLAT_0 ((uint32_t)0x01000000)
```

Bit 0

5.154.2.1802 FSMC_BTR4_DATLAT_1

```
#define FSMC_BTR4_DATLAT_1 ((uint32_t)0x02000000)
```

Bit 1

5.154.2.1803 FSMC_BTR4_DATLAT_2

```
#define FSMC_BTR4_DATLAT_2 ((uint32_t)0x04000000)
```

Bit 2

5.154.2.1804 FSMC_BTR4_DATLAT_3

```
#define FSMC_BTR4_DATLAT_3 ((uint32_t)0x08000000)
```

Bit 3

5.154.2.1805 FSMC_BWTR1_ACCMOD

```
#define FSMC_BWTR1_ACCMOD ((uint32_t)0x30000000)
```

ACCMOD[1:0] bits (Access mode)

5.154.2.1806 FSMC_BWTR1_ACCMOD_0

```
#define FSMC_BWTR1_ACCMOD_0 ((uint32_t)0x10000000)
```

Bit 0

5.154.2.1807 FSMC_BWTR1_ACCMOD_1

```
#define FSMC_BWTR1_ACCMOD_1 ((uint32_t)0x20000000)
```

Bit 1

5.154.2.1808 FSMC_BWTR1_ADDHLD

```
#define FSMC_BWTR1_ADDHLD ((uint32_t)0x000000F0)
```

ADDHLD[3:0] bits (Address-hold phase duration)

5.154.2.1809 FSMC_BWTR1_ADDHLD_0

```
#define FSMC_BWTR1_ADDHLD_0 ((uint32_t)0x00000010)
```

Bit 0

5.154.2.1810 FSMC_BWTR1_ADDHLD_1

```
#define FSMC_BWTR1_ADDHLD_1 ((uint32_t)0x00000020)
```

Bit 1

5.154.2.1811 FSMC_BWTR1_ADDHLD_2

```
#define FSMC_BWTR1_ADDHLD_2 ((uint32_t)0x00000040)
```

Bit 2

5.154.2.1812 FSMC_BWTR1_ADDHLD_3

```
#define FSMC_BWTR1_ADDHLD_3 ((uint32_t)0x00000080)
```

Bit 3

5.154.2.1813 FSMC_BWTR1_ADDSET

```
#define FSMC_BWTR1_ADDSET ((uint32_t)0x0000000F)
```

ADDSET[3:0] bits (Address setup phase duration)

5.154.2.1814 FSMC_BWTR1_ADDSET_0

```
#define FSMC_BWTR1_ADDSET_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.1815 FSMC_BWTR1_ADDSET_1

```
#define FSMC_BWTR1_ADDSET_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.1816 FSMC_BWTR1_ADDSET_2

```
#define FSMC_BWTR1_ADDSET_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.1817 FSMC_BWTR1_ADDSET_3

```
#define FSMC_BWTR1_ADDSET_3 ((uint32_t)0x00000008)
```

Bit 3

5.154.2.1818 FSMC_BWTR1_CLKDIV

```
#define FSMC_BWTR1_CLKDIV ((uint32_t)0x00F00000)
```

CLKDIV[3:0] bits (Clock divide ratio)

5.154.2.1819 FSMC_BWTR1_CLKDIV_0

```
#define FSMC_BWTR1_CLKDIV_0 ((uint32_t)0x00100000)
```

Bit 0

5.154.2.1820 FSMC_BWTR1_CLKDIV_1

```
#define FSMC_BWTR1_CLKDIV_1 ((uint32_t)0x00200000)
```

Bit 1

5.154.2.1821 FSMC_BWTR1_CLKDIV_2

```
#define FSMC_BWTR1_CLKDIV_2 ((uint32_t)0x00400000)
```

Bit 2

5.154.2.1822 FSMC_BWTR1_CLKDIV_3

```
#define FSMC_BWTR1_CLKDIV_3 ((uint32_t)0x00800000)
```

Bit 3

5.154.2.1823 FSMC_BWTR1_DATAST

```
#define FSMC_BWTR1_DATAST ((uint32_t)0x0000FF00)
```

DATAST [3:0] bits (Data-phase duration)

5.154.2.1824 FSMC_BWTR1_DATAST_0

```
#define FSMC_BWTR1_DATAST_0 ((uint32_t)0x00000100)
```

Bit 0

5.154.2.1825 FSMC_BWTR1_DATAST_1

```
#define FSMC_BWTR1_DATAST_1 ((uint32_t)0x00000200)
```

Bit 1

5.154.2.1826 FSMC_BWTR1_DATAST_2

```
#define FSMC_BWTR1_DATAST_2 ((uint32_t)0x00000400)
```

Bit 2

5.154.2.1827 FSMC_BWTR1_DATAST_3

```
#define FSMC_BWTR1_DATAST_3 ((uint32_t)0x00000800)
```

Bit 3

5.154.2.1828 FSMC_BWTR1_DATLAT

```
#define FSMC_BWTR1_DATLAT ((uint32_t)0x0F000000)
```

DATLA[3:0] bits (Data latency)

5.154.2.1829 FSMC_BWTR1_DATLAT_0

```
#define FSMC_BWTR1_DATLAT_0 ((uint32_t)0x01000000)
```

Bit 0

5.154.2.1830 FSMC_BWTR1_DATLAT_1

```
#define FSMC_BWTR1_DATLAT_1 ((uint32_t)0x02000000)
```

Bit 1

5.154.2.1831 FSMC_BWTR1_DATLAT_2

```
#define FSMC_BWTR1_DATLAT_2 ((uint32_t)0x04000000)
```

Bit 2

5.154.2.1832 FSMC_BWTR1_DATLAT_3

```
#define FSMC_BWTR1_DATLAT_3 ((uint32_t)0x08000000)
```

Bit 3

5.154.2.1833 FSMC_BWTR2_ACCMOD

```
#define FSMC_BWTR2_ACCMOD ((uint32_t)0x30000000)
```

ACCMOD[1:0] bits (Access mode)

5.154.2.1834 FSMC_BWTR2_ACCMOD_0

```
#define FSMC_BWTR2_ACCMOD_0 ((uint32_t)0x10000000)
```

Bit 0

5.154.2.1835 FSMC_BWTR2_ACCMOD_1

```
#define FSMC_BWTR2_ACCMOD_1 ((uint32_t)0x20000000)
```

Bit 1

5.154.2.1836 FSMC_BWTR2_ADDHLD

```
#define FSMC_BWTR2_ADDHLD ((uint32_t)0x000000F0)
```

ADDHLD[3:0] bits (Address-hold phase duration)

5.154.2.1837 FSMC_BWTR2_ADDHLD_0

```
#define FSMC_BWTR2_ADDHLD_0 ((uint32_t)0x00000010)
```

Bit 0

5.154.2.1838 FSMC_BWTR2_ADDHLD_1

```
#define FSMC_BWTR2_ADDHLD_1 ((uint32_t)0x00000020)
```

Bit 1

5.154.2.1839 FSMC_BWTR2_ADDHLD_2

```
#define FSMC_BWTR2_ADDHLD_2 ((uint32_t)0x00000040)
```

Bit 2

5.154.2.1840 FSMC_BWTR2_ADDHLD_3

```
#define FSMC_BWTR2_ADDHLD_3 ((uint32_t)0x00000080)
```

Bit 3

5.154.2.1841 FSMC_BWTR2_ADDSET

```
#define FSMC_BWTR2_ADDSET ((uint32_t)0x0000000F)
```

ADDSET[3:0] bits (Address setup phase duration)

5.154.2.1842 FSMC_BWTR2_ADDSET_0

```
#define FSMC_BWTR2_ADDSET_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.1843 FSMC_BWTR2_ADDSET_1

```
#define FSMC_BWTR2_ADDSET_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.1844 FSMC_BWTR2_ADDSET_2

```
#define FSMC_BWTR2_ADDSET_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.1845 FSMC_BWTR2_ADDSET_3

```
#define FSMC_BWTR2_ADDSET_3 ((uint32_t)0x00000008)
```

Bit 3

5.154.2.1846 FSMC_BWTR2_CLKDIV

```
#define FSMC_BWTR2_CLKDIV ((uint32_t)0x00F00000)
```

CLKDIV[3:0] bits (Clock divide ratio)

5.154.2.1847 FSMC_BWTR2_CLKDIV_0

```
#define FSMC_BWTR2_CLKDIV_0 ((uint32_t)0x00100000)
```

Bit 0

5.154.2.1848 FSMC_BWTR2_CLKDIV_1

```
#define FSMC_BWTR2_CLKDIV_1 ((uint32_t)0x00200000)
```

Bit 1

5.154.2.1849 FSMC_BWTR2_CLKDIV_2

```
#define FSMC_BWTR2_CLKDIV_2 ((uint32_t)0x00400000)
```

Bit 2

5.154.2.1850 FSMC_BWTR2_CLKDIV_3

```
#define FSMC_BWTR2_CLKDIV_3 ((uint32_t)0x00800000)
```

Bit 3

5.154.2.1851 FSMC_BWTR2_DATAST

```
#define FSMC_BWTR2_DATAST ((uint32_t)0x0000FF00)
```

DATAST [3:0] bits (Data-phase duration)

5.154.2.1852 FSMC_BWTR2_DATAST_0

```
#define FSMC_BWTR2_DATAST_0 ((uint32_t)0x00000100)
```

Bit 0

5.154.2.1853 FSMC_BWTR2_DATAST_1

```
#define FSMC_BWTR2_DATAST_1 ((uint32_t)0x00000200)
```

Bit 1

5.154.2.1854 FSMC_BWTR2_DATAST_2

```
#define FSMC_BWTR2_DATAST_2 ((uint32_t)0x00000400)
```

Bit 2

5.154.2.1855 FSMC_BWTR2_DATAST_3

```
#define FSMC_BWTR2_DATAST_3 ((uint32_t)0x00000800)
```

Bit 3

5.154.2.1856 FSMC_BWTR2_DATLAT

```
#define FSMC_BWTR2_DATLAT ((uint32_t)0x0F000000)
```

DATLA[3:0] bits (Data latency)

5.154.2.1857 FSMC_BWTR2_DATLAT_0

```
#define FSMC_BWTR2_DATLAT_0 ((uint32_t)0x01000000)
```

Bit 0

5.154.2.1858 FSMC_BWTR2_DATLAT_1

```
#define FSMC_BWTR2_DATLAT_1 ((uint32_t)0x02000000)
```

Bit 1

5.154.2.1859 FSMC_BWTR2_DATLAT_2

```
#define FSMC_BWTR2_DATLAT_2 ((uint32_t)0x04000000)
```

Bit 2

5.154.2.1860 FSMC_BWTR2_DATLAT_3

```
#define FSMC_BWTR2_DATLAT_3 ((uint32_t)0x08000000)
```

Bit 3

5.154.2.1861 FSMC_BWTR3_ACCMOD

```
#define FSMC_BWTR3_ACCMOD ((uint32_t)0x30000000)
```

ACCMOD[1:0] bits (Access mode)

5.154.2.1862 FSMC_BWTR3_ACCMOD_0

```
#define FSMC_BWTR3_ACCMOD_0 ((uint32_t)0x10000000)
```

Bit 0

5.154.2.1863 FSMC_BWTR3_ACCMOD_1

```
#define FSMC_BWTR3_ACCMOD_1 ((uint32_t)0x20000000)
```

Bit 1

5.154.2.1864 FSMC_BWTR3_ADDHLD

```
#define FSMC_BWTR3_ADDHLD ((uint32_t)0x000000F0)
```

ADDHLD[3:0] bits (Address-hold phase duration)

5.154.2.1865 FSMC_BWTR3_ADDHLD_0

```
#define FSMC_BWTR3_ADDHLD_0 ((uint32_t)0x00000010)
```

Bit 0

5.154.2.1866 FSMC_BWTR3_ADDHLD_1

```
#define FSMC_BWTR3_ADDHLD_1 ((uint32_t)0x00000020)
```

Bit 1

5.154.2.1867 FSMC_BWTR3_ADDHLD_2

```
#define FSMC_BWTR3_ADDHLD_2 ((uint32_t)0x00000040)
```

Bit 2

5.154.2.1868 FSMC_BWTR3_ADDHLD_3

```
#define FSMC_BWTR3_ADDHLD_3 ((uint32_t)0x00000080)
```

Bit 3

5.154.2.1869 FSMC_BWTR3_ADDSET

```
#define FSMC_BWTR3_ADDSET ((uint32_t)0x0000000F)
```

ADDSET[3:0] bits (Address setup phase duration)

5.154.2.1870 FSMC_BWTR3_ADDSET_0

```
#define FSMC_BWTR3_ADDSET_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.1871 FSMC_BWTR3_ADDSET_1

```
#define FSMC_BWTR3_ADDSET_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.1872 FSMC_BWTR3_ADDSET_2

```
#define FSMC_BWTR3_ADDSET_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.1873 FSMC_BWTR3_ADDSET_3

```
#define FSMC_BWTR3_ADDSET_3 ((uint32_t)0x00000008)
```

Bit 3

5.154.2.1874 FSMC_BWTR3_CLKDIV

```
#define FSMC_BWTR3_CLKDIV ((uint32_t)0x00F00000)
```

CLKDIV[3:0] bits (Clock divide ratio)

5.154.2.1875 FSMC_BWTR3_CLKDIV_0

```
#define FSMC_BWTR3_CLKDIV_0 ((uint32_t)0x00100000)
```

Bit 0

5.154.2.1876 FSMC_BWTR3_CLKDIV_1

```
#define FSMC_BWTR3_CLKDIV_1 ((uint32_t)0x00200000)
```

Bit 1

5.154.2.1877 FSMC_BWTR3_CLKDIV_2

```
#define FSMC_BWTR3_CLKDIV_2 ((uint32_t)0x00400000)
```

Bit 2

5.154.2.1878 FSMC_BWTR3_CLKDIV_3

```
#define FSMC_BWTR3_CLKDIV_3 ((uint32_t)0x00800000)
```

Bit 3

5.154.2.1879 FSMC_BWTR3_DATAST

```
#define FSMC_BWTR3_DATAST ((uint32_t)0x0000FF00)
```

DATAST [3:0] bits (Data-phase duration)

5.154.2.1880 FSMC_BWTR3_DATAST_0

```
#define FSMC_BWTR3_DATAST_0 ((uint32_t)0x00000100)
```

Bit 0

5.154.2.1881 FSMC_BWTR3_DATAST_1

```
#define FSMC_BWTR3_DATAST_1 ((uint32_t)0x00000200)
```

Bit 1

5.154.2.1882 FSMC_BWTR3_DATAST_2

```
#define FSMC_BWTR3_DATAST_2 ((uint32_t)0x00000400)
```

Bit 2

5.154.2.1883 FSMC_BWTR3_DATAST_3

```
#define FSMC_BWTR3_DATAST_3 ((uint32_t)0x00000800)
```

Bit 3

5.154.2.1884 FSMC_BWTR3_DATLAT

```
#define FSMC_BWTR3_DATLAT ((uint32_t)0x0F000000)
```

DATLA[3:0] bits (Data latency)

5.154.2.1885 FSMC_BWTR3_DATLAT_0

```
#define FSMC_BWTR3_DATLAT_0 ((uint32_t)0x01000000)
```

Bit 0

5.154.2.1886 FSMC_BWTR3_DATLAT_1

```
#define FSMC_BWTR3_DATLAT_1 ((uint32_t)0x02000000)
```

Bit 1

5.154.2.1887 FSMC_BWTR3_DATLAT_2

```
#define FSMC_BWTR3_DATLAT_2 ((uint32_t)0x04000000)
```

Bit 2

5.154.2.1888 FSMC_BWTR3_DATLAT_3

```
#define FSMC_BWTR3_DATLAT_3 ((uint32_t)0x08000000)
```

Bit 3

5.154.2.1889 FSMC_BWTR4_ACCMOD

```
#define FSMC_BWTR4_ACCMOD ((uint32_t)0x30000000)
```

ACCMOD[1:0] bits (Access mode)

5.154.2.1890 FSMC_BWTR4_ACCMOD_0

```
#define FSMC_BWTR4_ACCMOD_0 ((uint32_t)0x10000000)
```

Bit 0

5.154.2.1891 FSMC_BWTR4_ACCMOD_1

```
#define FSMC_BWTR4_ACCMOD_1 ((uint32_t)0x20000000)
```

Bit 1

5.154.2.1892 FSMC_BWTR4_ADDHLD

```
#define FSMC_BWTR4_ADDHLD ((uint32_t)0x000000F0)
```

ADDHLD[3:0] bits (Address-hold phase duration)

5.154.2.1893 FSMC_BWTR4_ADDHLD_0

```
#define FSMC_BWTR4_ADDHLD_0 ((uint32_t)0x00000010)
```

Bit 0

5.154.2.1894 FSMC_BWTR4_ADDHLD_1

```
#define FSMC_BWTR4_ADDHLD_1 ((uint32_t)0x00000020)
```

Bit 1

5.154.2.1895 FSMC_BWTR4_ADDHLD_2

```
#define FSMC_BWTR4_ADDHLD_2 ((uint32_t)0x00000040)
```

Bit 2

5.154.2.1896 FSMC_BWTR4_ADDHLD_3

```
#define FSMC_BWTR4_ADDHLD_3 ((uint32_t)0x00000080)
```

Bit 3

5.154.2.1897 FSMC_BWTR4_ADDSET

```
#define FSMC_BWTR4_ADDSET ((uint32_t)0x0000000F)
```

ADDSET[3:0] bits (Address setup phase duration)

5.154.2.1898 FSMC_BWTR4_ADDSET_0

```
#define FSMC_BWTR4_ADDSET_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.1899 FSMC_BWTR4_ADDSET_1

```
#define FSMC_BWTR4_ADDSET_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.1900 FSMC_BWTR4_ADDSET_2

```
#define FSMC_BWTR4_ADDSET_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.1901 FSMC_BWTR4_ADDSET_3

```
#define FSMC_BWTR4_ADDSET_3 ((uint32_t)0x00000008)
```

Bit 3

5.154.2.1902 FSMC_BWTR4_CLKDIV

```
#define FSMC_BWTR4_CLKDIV ((uint32_t)0x00F00000)
```

CLKDIV[3:0] bits (Clock divide ratio)

5.154.2.1903 FSMC_BWTR4_CLKDIV_0

```
#define FSMC_BWTR4_CLKDIV_0 ((uint32_t)0x00100000)
```

Bit 0

5.154.2.1904 FSMC_BWTR4_CLKDIV_1

```
#define FSMC_BWTR4_CLKDIV_1 ((uint32_t)0x00200000)
```

Bit 1

5.154.2.1905 FSMC_BWTR4_CLKDIV_2

```
#define FSMC_BWTR4_CLKDIV_2 ((uint32_t)0x00400000)
```

Bit 2

5.154.2.1906 FSMC_BWTR4_CLKDIV_3

```
#define FSMC_BWTR4_CLKDIV_3 ((uint32_t)0x00800000)
```

Bit 3

5.154.2.1907 FSMC_BWTR4_DATAST

```
#define FSMC_BWTR4_DATAST ((uint32_t)0x0000FF00)
```

DATAST [3:0] bits (Data-phase duration)

5.154.2.1908 FSMC_BWTR4_DATAST_0

```
#define FSMC_BWTR4_DATAST_0 ((uint32_t)0x00000100)
```

Bit 0

5.154.2.1909 FSMC_BWTR4_DATAST_1

```
#define FSMC_BWTR4_DATAST_1 ((uint32_t)0x00000200)
```

Bit 1

5.154.2.1910 FSMC_BWTR4_DATAST_2

```
#define FSMC_BWTR4_DATAST_2 ((uint32_t)0x00000400)
```

Bit 2

5.154.2.1911 FSMC_BWTR4_DATAST_3

```
#define FSMC_BWTR4_DATAST_3 ((uint32_t)0x00000800)
```

Bit 3

5.154.2.1912 FSMC_BWTR4_DATLAT

```
#define FSMC_BWTR4_DATLAT ((uint32_t)0x0F000000)
```

DATLA[3:0] bits (Data latency)

5.154.2.1913 FSMC_BWTR4_DATLAT_0

```
#define FSMC_BWTR4_DATLAT_0 ((uint32_t)0x01000000)
```

Bit 0

5.154.2.1914 FSMC_BWTR4_DATLAT_1

```
#define FSMC_BWTR4_DATLAT_1 ((uint32_t)0x02000000)
```

Bit 1

5.154.2.1915 FSMC_BWTR4_DATLAT_2

```
#define FSMC_BWTR4_DATLAT_2 ((uint32_t)0x04000000)
```

Bit 2

5.154.2.1916 FSMC_BWTR4_DATLAT_3

```
#define FSMC_BWTR4_DATLAT_3 ((uint32_t)0x08000000)
```

Bit 3

5.154.2.1917 FSMC_ECCR2_ECC2

```
#define FSMC_ECCR2_ECC2 ((uint32_t)0xFFFFFFFF)
```

ECC result

5.154.2.1918 FSMC_ECCR3_ECC3

```
#define FSMC_ECCR3_ECC3 ((uint32_t)0xFFFFFFFF)
```

ECC result

5.154.2.1919 FSMC_PATT2_ATTHIZ2

```
#define FSMC_PATT2_ATTHIZ2 ((uint32_t)0xFF000000)
```

ATTHIZ2[7:0] bits (Attribute memory 2 databus HiZ time)

5.154.2.1920 FSMC_PATT2_ATTHIZ2_0

```
#define FSMC_PATT2_ATTHIZ2_0 ((uint32_t)0x01000000)
```

Bit 0

5.154.2.1921 FSMC_PATT2_ATTHIZ2_1

```
#define FSMC_PATT2_ATTHIZ2_1 ((uint32_t)0x02000000)
```

Bit 1

5.154.2.1922 FSMC_PATT2_ATTHIZ2_2

```
#define FSMC_PATT2_ATTHIZ2_2 ((uint32_t)0x04000000)
```

Bit 2

5.154.2.1923 FSMC_PATT2_ATTHIZ2_3

```
#define FSMC_PATT2_ATTHIZ2_3 ((uint32_t)0x08000000)
```

Bit 3

5.154.2.1924 FSMC_PATT2_ATTHIZ2_4

```
#define FSMC_PATT2_ATTHIZ2_4 ((uint32_t)0x10000000)
```

Bit 4

5.154.2.1925 FSMC_PATT2_ATTHIZ2_5

```
#define FSMC_PATT2_ATTHIZ2_5 ((uint32_t)0x20000000)
```

Bit 5

5.154.2.1926 FSMC_PATT2_ATTHIZ2_6

```
#define FSMC_PATT2_ATTHIZ2_6 ((uint32_t)0x40000000)
```

Bit 6

5.154.2.1927 FSMC_PATT2_ATTHIZ2_7

```
#define FSMC_PATT2_ATTHIZ2_7 ((uint32_t)0x80000000)
```

Bit 7

5.154.2.1928 FSMC_PATT2_ATTHOLD2

```
#define FSMC_PATT2_ATTHOLD2 ((uint32_t)0x00FF0000)
```

ATTHOLD2[7:0] bits (Attribute memory 2 hold time)

5.154.2.1929 FSMC_PATT2_ATTHOLD2_0

```
#define FSMC_PATT2_ATTHOLD2_0 ((uint32_t)0x00010000)
```

Bit 0

5.154.2.1930 FSMC_PATT2_ATTHOLD2_1

```
#define FSMC_PATT2_ATTHOLD2_1 ((uint32_t)0x00020000)
```

Bit 1

5.154.2.1931 FSMC_PATT2_ATTHOLD2_2

```
#define FSMC_PATT2_ATTHOLD2_2 ((uint32_t)0x00040000)
```

Bit 2

5.154.2.1932 FSMC_PATT2_ATTHOLD2_3

```
#define FSMC_PATT2_ATTHOLD2_3 ((uint32_t)0x00080000)
```

Bit 3

5.154.2.1933 FSMC_PATT2_ATTHOLD2_4

```
#define FSMC_PATT2_ATTHOLD2_4 ((uint32_t)0x00100000)
```

Bit 4

5.154.2.1934 FSMC_PATT2_ATTHOLD2_5

```
#define FSMC_PATT2_ATTHOLD2_5 ((uint32_t)0x00200000)
```

Bit 5

5.154.2.1935 FSMC_PATT2_ATTHOLD2_6

```
#define FSMC_PATT2_ATTHOLD2_6 ((uint32_t)0x00400000)
```

Bit 6

5.154.2.1936 FSMC_PATT2_ATTHOLD2_7

```
#define FSMC_PATT2_ATTHOLD2_7 ((uint32_t)0x00800000)
```

Bit 7

5.154.2.1937 FSMC_PATT2_ATTSET2

```
#define FSMC_PATT2_ATTSET2 ((uint32_t)0x000000FF)
```

ATTSET2[7:0] bits (Attribute memory 2 setup time)

5.154.2.1938 FSMC_PATT2_ATTSET2_0

```
#define FSMC_PATT2_ATTSET2_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.1939 FSMC_PATT2_ATTSET2_1

```
#define FSMC_PATT2_ATTSET2_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.1940 FSMC_PATT2_ATTSET2_2

```
#define FSMC_PATT2_ATTSET2_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.1941 FSMC_PATT2_ATTSET2_3

```
#define FSMC_PATT2_ATTSET2_3 ((uint32_t)0x00000008)
```

Bit 3

5.154.2.1942 FSMC_PATT2_ATTSET2_4

```
#define FSMC_PATT2_ATTSET2_4 ((uint32_t)0x00000010)
```

Bit 4

5.154.2.1943 FSMC_PATT2_ATTSET2_5

```
#define FSMC_PATT2_ATTSET2_5 ((uint32_t)0x00000020)
```

Bit 5

5.154.2.1944 FSMC_PATT2_ATTSET2_6

```
#define FSMC_PATT2_ATTSET2_6 ((uint32_t)0x00000040)
```

Bit 6

5.154.2.1945 FSMC_PATT2_ATTSET2_7

```
#define FSMC_PATT2_ATTSET2_7 ((uint32_t)0x00000080)
```

Bit 7

5.154.2.1946 FSMC_PATT2_ATTWAIT2

```
#define FSMC_PATT2_ATTWAIT2 ((uint32_t)0x0000FF00)
```

ATTWAIT2[7:0] bits (Attribute memory 2 wait time)

5.154.2.1947 FSMC_PATT2_ATTWAIT2_0

```
#define FSMC_PATT2_ATTWAIT2_0 ((uint32_t)0x00000100)
```

Bit 0

5.154.2.1948 FSMC_PATT2_ATTWAIT2_1

```
#define FSMC_PATT2_ATTWAIT2_1 ((uint32_t)0x00000200)
```

Bit 1

5.154.2.1949 FSMC_PATT2_ATTWAIT2_2

```
#define FSMC_PATT2_ATTWAIT2_2 ((uint32_t)0x00000400)
```

Bit 2

5.154.2.1950 FSMC_PATT2_ATTWAIT2_3

```
#define FSMC_PATT2_ATTWAIT2_3 ((uint32_t)0x00000800)
```

Bit 3

5.154.2.1951 FSMC_PATT2_ATTWAIT2_4

```
#define FSMC_PATT2_ATTWAIT2_4 ((uint32_t)0x00001000)
```

Bit 4

5.154.2.1952 FSMC_PATT2_ATTWAIT2_5

```
#define FSMC_PATT2_ATTWAIT2_5 ((uint32_t)0x00002000)
```

Bit 5

5.154.2.1953 FSMC_PATT2_ATTWAIT2_6

```
#define FSMC_PATT2_ATTWAIT2_6 ((uint32_t)0x00004000)
```

Bit 6

5.154.2.1954 FSMC_PATT2_ATTWAIT2_7

```
#define FSMC_PATT2_ATTWAIT2_7 ((uint32_t)0x00008000)
```

Bit 7

5.154.2.1955 FSMC_PATT3_ATTHIZ3

```
#define FSMC_PATT3_ATTHIZ3 ((uint32_t)0xFF000000)
```

ATTHIZ3[7:0] bits (Attribute memory 3 databus HiZ time)

5.154.2.1956 FSMC_PATT3_ATTHIZ3_0

```
#define FSMC_PATT3_ATTHIZ3_0 ((uint32_t)0x01000000)
```

Bit 0

5.154.2.1957 FSMC_PATT3_ATTHIZ3_1

```
#define FSMC_PATT3_ATTHIZ3_1 ((uint32_t)0x02000000)
```

Bit 1

5.154.2.1958 FSMC_PATT3_ATTHIZ3_2

```
#define FSMC_PATT3_ATTHIZ3_2 ((uint32_t)0x04000000)
```

Bit 2

5.154.2.1959 FSMC_PATT3_ATTHIZ3_3

```
#define FSMC_PATT3_ATTHIZ3_3 ((uint32_t)0x08000000)
```

Bit 3

5.154.2.1960 FSMC_PATT3_ATTHIZ3_4

```
#define FSMC_PATT3_ATTHIZ3_4 ((uint32_t)0x10000000)
```

Bit 4

5.154.2.1961 FSMC_PATT3_ATTHIZ3_5

```
#define FSMC_PATT3_ATTHIZ3_5 ((uint32_t)0x20000000)
```

Bit 5

5.154.2.1962 FSMC_PATT3_ATTHIZ3_6

```
#define FSMC_PATT3_ATTHIZ3_6 ((uint32_t)0x40000000)
```

Bit 6

5.154.2.1963 FSMC_PATT3_ATTHIZ3_7

```
#define FSMC_PATT3_ATTHIZ3_7 ((uint32_t)0x80000000)
```

Bit 7

5.154.2.1964 FSMC_PATT3_ATTHOLD3

```
#define FSMC_PATT3_ATTHOLD3 ((uint32_t)0x00FF0000)
```

ATTHOLD3[7:0] bits (Attribute memory 3 hold time)

5.154.2.1965 FSMC_PATT3_ATTHOLD3_0

```
#define FSMC_PATT3_ATTHOLD3_0 ((uint32_t)0x00010000)
```

Bit 0

5.154.2.1966 FSMC_PATT3_ATTHOLD3_1

```
#define FSMC_PATT3_ATTHOLD3_1 ((uint32_t)0x00020000)
```

Bit 1

5.154.2.1967 FSMC_PATT3_ATTHOLD3_2

```
#define FSMC_PATT3_ATTHOLD3_2 ((uint32_t)0x00040000)
```

Bit 2

5.154.2.1968 FSMC_PATT3_ATTHOLD3_3

```
#define FSMC_PATT3_ATTHOLD3_3 ((uint32_t)0x00080000)
```

Bit 3

5.154.2.1969 FSMC_PATT3_ATTHOLD3_4

```
#define FSMC_PATT3_ATTHOLD3_4 ((uint32_t)0x00100000)
```

Bit 4

5.154.2.1970 FSMC_PATT3_ATTHOLD3_5

```
#define FSMC_PATT3_ATTHOLD3_5 ((uint32_t)0x00200000)
```

Bit 5

5.154.2.1971 FSMC_PATT3_ATTHOLD3_6

```
#define FSMC_PATT3_ATTHOLD3_6 ((uint32_t)0x00400000)
```

Bit 6

5.154.2.1972 FSMC_PATT3_ATTHOLD3_7

```
#define FSMC_PATT3_ATTHOLD3_7 ((uint32_t)0x00800000)
```

Bit 7

5.154.2.1973 FSMC_PATT3_ATTSET3

```
#define FSMC_PATT3_ATTSET3 ((uint32_t)0x000000FF)
```

ATTSET3[7:0] bits (Attribute memory 3 setup time)

5.154.2.1974 FSMC_PATT3_ATTSET3_0

```
#define FSMC_PATT3_ATTSET3_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.1975 FSMC_PATT3_ATTSET3_1

```
#define FSMC_PATT3_ATTSET3_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.1976 FSMC_PATT3_ATTSET3_2

```
#define FSMC_PATT3_ATTSET3_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.1977 FSMC_PATT3_ATTSET3_3

```
#define FSMC_PATT3_ATTSET3_3 ((uint32_t)0x00000008)
```

Bit 3

5.154.2.1978 FSMC_PATT3_ATTSET3_4

```
#define FSMC_PATT3_ATTSET3_4 ((uint32_t)0x00000010)
```

Bit 4

5.154.2.1979 FSMC_PATT3_ATTSET3_5

```
#define FSMC_PATT3_ATTSET3_5 ((uint32_t)0x00000020)
```

Bit 5

5.154.2.1980 FSMC_PATT3_ATTSET3_6

```
#define FSMC_PATT3_ATTSET3_6 ((uint32_t)0x00000040)
```

Bit 6

5.154.2.1981 FSMC_PATT3_ATTSET3_7

```
#define FSMC_PATT3_ATTSET3_7 ((uint32_t)0x00000080)
```

Bit 7

5.154.2.1982 FSMC_PATT3_ATTWAIT3

```
#define FSMC_PATT3_ATTWAIT3 ((uint32_t)0x0000FF00)
```

ATTWAIT3[7:0] bits (Attribute memory 3 wait time)

5.154.2.1983 FSMC_PATT3_ATTWAIT3_0

```
#define FSMC_PATT3_ATTWAIT3_0 ((uint32_t)0x00000100)
```

Bit 0

5.154.2.1984 FSMC_PATT3_ATTWAIT3_1

```
#define FSMC_PATT3_ATTWAIT3_1 ((uint32_t)0x00000200)
```

Bit 1

5.154.2.1985 FSMC_PATT3_ATTWAIT3_2

```
#define FSMC_PATT3_ATTWAIT3_2 ((uint32_t)0x00000400)
```

Bit 2

5.154.2.1986 FSMC_PATT3_ATTWAIT3_3

```
#define FSMC_PATT3_ATTWAIT3_3 ((uint32_t)0x00000800)
```

Bit 3

5.154.2.1987 FSMC_PATT3_ATTWAIT3_4

```
#define FSMC_PATT3_ATTWAIT3_4 ((uint32_t)0x00001000)
```

Bit 4

5.154.2.1988 FSMC_PATT3_ATTWAIT3_5

```
#define FSMC_PATT3_ATTWAIT3_5 ((uint32_t)0x00002000)
```

Bit 5

5.154.2.1989 FSMC_PATT3_ATTWAIT3_6

```
#define FSMC_PATT3_ATTWAIT3_6 ((uint32_t)0x00004000)
```

Bit 6

5.154.2.1990 FSMC_PATT3_ATTWAIT3_7

```
#define FSMC_PATT3_ATTWAIT3_7 ((uint32_t)0x00008000)
```

Bit 7

5.154.2.1991 FSMC_PATT4_ATTHIZ4

```
#define FSMC_PATT4_ATTHIZ4 ((uint32_t)0xFF000000)
```

ATTHIZ4[7:0] bits (Attribute memory 4 databus HiZ time)

5.154.2.1992 FSMC_PATT4_ATTHIZ4_0

```
#define FSMC_PATT4_ATTHIZ4_0 ((uint32_t)0x01000000)
```

Bit 0

5.154.2.1993 FSMC_PATT4_ATTHIZ4_1

```
#define FSMC_PATT4_ATTHIZ4_1 ((uint32_t)0x02000000)
```

Bit 1

5.154.2.1994 FSMC_PATT4_ATTHIZ4_2

```
#define FSMC_PATT4_ATTHIZ4_2 ((uint32_t)0x04000000)
```

Bit 2

5.154.2.1995 FSMC_PATT4_ATTHIZ4_3

```
#define FSMC_PATT4_ATTHIZ4_3 ((uint32_t)0x08000000)
```

Bit 3

5.154.2.1996 FSMC_PATT4_ATTHIZ4_4

```
#define FSMC_PATT4_ATTHIZ4_4 ((uint32_t)0x10000000)
```

Bit 4

5.154.2.1997 FSMC_PATT4_ATTHIZ4_5

```
#define FSMC_PATT4_ATTHIZ4_5 ((uint32_t)0x20000000)
```

Bit 5

5.154.2.1998 FSMC_PATT4_ATTHIZ4_6

```
#define FSMC_PATT4_ATTHIZ4_6 ((uint32_t)0x40000000)
```

Bit 6

5.154.2.1999 FSMC_PATT4_ATTHIZ4_7

```
#define FSMC_PATT4_ATTHIZ4_7 ((uint32_t)0x80000000)
```

Bit 7

5.154.2.2000 FSMC_PATT4_ATTHOLD4

```
#define FSMC_PATT4_ATTHOLD4 ((uint32_t)0x00FF0000)
```

ATTHOLD4[7:0] bits (Attribute memory 4 hold time)

5.154.2.2001 FSMC_PATT4_ATTHOLD4_0

```
#define FSMC_PATT4_ATTHOLD4_0 ((uint32_t)0x00010000)
```

Bit 0

5.154.2.2002 FSMC_PATT4_ATTHOLD4_1

```
#define FSMC_PATT4_ATTHOLD4_1 ((uint32_t)0x00020000)
```

Bit 1

5.154.2.2003 FSMC_PATT4_ATTHOLD4_2

```
#define FSMC_PATT4_ATTHOLD4_2 ((uint32_t)0x00040000)
```

Bit 2

5.154.2.2004 FSMC_PATT4_ATTHOLD4_3

```
#define FSMC_PATT4_ATTHOLD4_3 ((uint32_t)0x00080000)
```

Bit 3

5.154.2.2005 FSMC_PATT4_ATTHOLD4_4

```
#define FSMC_PATT4_ATTHOLD4_4 ((uint32_t)0x00100000)
```

Bit 4

5.154.2.2006 FSMC_PATT4_ATTHOLD4_5

```
#define FSMC_PATT4_ATTHOLD4_5 ((uint32_t)0x00200000)
```

Bit 5

5.154.2.2007 FSMC_PATT4_ATTHOLD4_6

```
#define FSMC_PATT4_ATTHOLD4_6 ((uint32_t)0x00400000)
```

Bit 6

5.154.2.2008 FSMC_PATT4_ATTHOLD4_7

```
#define FSMC_PATT4_ATTHOLD4_7 ((uint32_t)0x00800000)
```

Bit 7

5.154.2.2009 FSMC_PATT4_ATTSET4

```
#define FSMC_PATT4_ATTSET4 ((uint32_t)0x000000FF)
```

ATTSET4[7:0] bits (Attribute memory 4 setup time)

5.154.2.2010 FSMC_PATT4_ATTSET4_0

```
#define FSMC_PATT4_ATTSET4_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.2011 FSMC_PATT4_ATTSET4_1

```
#define FSMC_PATT4_ATTSET4_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.2012 FSMC_PATT4_ATTSET4_2

```
#define FSMC_PATT4_ATTSET4_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.2013 FSMC_PATT4_ATTSET4_3

```
#define FSMC_PATT4_ATTSET4_3 ((uint32_t)0x00000008)
```

Bit 3

5.154.2.2014 FSMC_PATT4_ATTSET4_4

```
#define FSMC_PATT4_ATTSET4_4 ((uint32_t)0x00000010)
```

Bit 4

5.154.2.2015 FSMC_PATT4_ATTSET4_5

```
#define FSMC_PATT4_ATTSET4_5 ((uint32_t)0x00000020)
```

Bit 5

5.154.2.2016 FSMC_PATT4_ATTSET4_6

```
#define FSMC_PATT4_ATTSET4_6 ((uint32_t)0x00000040)
```

Bit 6

5.154.2.2017 FSMC_PATT4_ATTSET4_7

```
#define FSMC_PATT4_ATTSET4_7 ((uint32_t)0x00000080)
```

Bit 7

5.154.2.2018 FSMC_PATT4_ATTWAIT4

```
#define FSMC_PATT4_ATTWAIT4 ((uint32_t)0x0000FF00)
```

ATTWAIT4[7:0] bits (Attribute memory 4 wait time)

5.154.2.2019 FSMC_PATT4_ATTWAIT4_0

```
#define FSMC_PATT4_ATTWAIT4_0 ((uint32_t)0x00000100)
```

Bit 0

5.154.2.2020 FSMC_PATT4_ATTWAIT4_1

```
#define FSMC_PATT4_ATTWAIT4_1 ((uint32_t)0x00000200)
```

Bit 1

5.154.2.2021 FSMC_PATT4_ATTWAIT4_2

```
#define FSMC_PATT4_ATTWAIT4_2 ((uint32_t)0x00000400)
```

Bit 2

5.154.2.2022 FSMC_PATT4_ATTWAIT4_3

```
#define FSMC_PATT4_ATTWAIT4_3 ((uint32_t)0x00000800)
```

Bit 3

5.154.2.2023 FSMC_PATT4_ATTWAIT4_4

```
#define FSMC_PATT4_ATTWAIT4_4 ((uint32_t)0x00001000)
```

Bit 4

5.154.2.2024 FSMC_PATT4_ATTWAIT4_5

```
#define FSMC_PATT4_ATTWAIT4_5 ((uint32_t)0x00002000)
```

Bit 5

5.154.2.2025 FSMC_PATT4_ATTWAIT4_6

```
#define FSMC_PATT4_ATTWAIT4_6 ((uint32_t)0x00004000)
```

Bit 6

5.154.2.2026 FSMC_PATT4_ATTWAIT4_7

```
#define FSMC_PATT4_ATTWAIT4_7 ((uint32_t)0x00008000)
```

Bit 7

5.154.2.2027 FSMC_PCR2_ECCEN

```
#define FSMC_PCR2_ECCEN ((uint32_t)0x00000040)
```

ECC computation logic enable bit

5.154.2.2028 FSMC_PCR2_ECCPS

```
#define FSMC_PCR2_ECCPS ((uint32_t)0x000E0000)
```

ECCPS[1:0] bits (ECC page size)

5.154.2.2029 FSMC_PCR2_ECCPS_0

```
#define FSMC_PCR2_ECCPS_0 ((uint32_t)0x00020000)
```

Bit 0

5.154.2.2030 FSMC_PCR2_ECCPS_1

```
#define FSMC_PCR2_ECCPS_1 ((uint32_t)0x00040000)
```

Bit 1

5.154.2.2031 FSMC_PCR2_ECCPS_2

```
#define FSMC_PCR2_ECCPS_2 ((uint32_t)0x00080000)
```

Bit 2

5.154.2.2032 FSMC_PCR2_PBKEN

```
#define FSMC_PCR2_PBKEN ((uint32_t)0x00000004)
```

PC Card/NAND Flash memory bank enable bit

5.154.2.2033 FSMC_PCR2_PTYP

```
#define FSMC_PCR2_PTYP ((uint32_t)0x00000008)
```

Memory type

5.154.2.2034 FSMC_PCR2_PWAITEN

```
#define FSMC_PCR2_PWAITEN ((uint32_t)0x00000002)
```

Wait feature enable bit

5.154.2.2035 FSMC_PCR2_PWID

```
#define FSMC_PCR2_PWID ((uint32_t)0x00000030)
```

PWID[1:0] bits (NAND Flash databus width)

5.154.2.2036 FSMC_PCR2_PWID_0

```
#define FSMC_PCR2_PWID_0 ((uint32_t)0x00000010)
```

Bit 0

5.154.2.2037 FSMC_PCR2_PWID_1

```
#define FSMC_PCR2_PWID_1 ((uint32_t)0x00000020)
```

Bit 1

5.154.2.2038 FSMC_PCR2_TAR

```
#define FSMC_PCR2_TAR ((uint32_t)0x0001E000)
```

TAR[3:0] bits (ALE to RE delay)

5.154.2.2039 FSMC_PCR2_TAR_0

```
#define FSMC_PCR2_TAR_0 ((uint32_t)0x00002000)
```

Bit 0

5.154.2.2040 FSMC_PCR2_TAR_1

```
#define FSMC_PCR2_TAR_1 ((uint32_t)0x00004000)
```

Bit 1

5.154.2.2041 FSMC_PCR2_TAR_2

```
#define FSMC_PCR2_TAR_2 ((uint32_t)0x00008000)
```

Bit 2

5.154.2.2042 FSMC_PCR2_TAR_3

```
#define FSMC_PCR2_TAR_3 ((uint32_t)0x00010000)
```

Bit 3

5.154.2.2043 FSMC_PCR2_TCLR

```
#define FSMC_PCR2_TCLR ((uint32_t)0x00001E00)
```

TCLR[3:0] bits (CLE to RE delay)

5.154.2.2044 FSMC_PCR2_TCLR_0

```
#define FSMC_PCR2_TCLR_0 ((uint32_t)0x00000200)
```

Bit 0

5.154.2.2045 FSMC_PCR2_TCLR_1

```
#define FSMC_PCR2_TCLR_1 ((uint32_t)0x00000400)
```

Bit 1

5.154.2.2046 FSMC_PCR2_TCLR_2

```
#define FSMC_PCR2_TCLR_2 ((uint32_t)0x00000800)
```

Bit 2

5.154.2.2047 FSMC_PCR2_TCLR_3

```
#define FSMC_PCR2_TCLR_3 ((uint32_t)0x00001000)
```

Bit 3

5.154.2.2048 FSMC_PCR3_ECCEN

```
#define FSMC_PCR3_ECCEN ((uint32_t)0x00000040)
```

ECC computation logic enable bit

5.154.2.2049 FSMC_PCR3_ECCPS

```
#define FSMC_PCR3_ECCPS ((uint32_t)0x000E0000)
```

ECCPS[2:0] bits (ECC page size)

5.154.2.2050 FSMC_PCR3_ECCPS_0

```
#define FSMC_PCR3_ECCPS_0 ((uint32_t)0x00020000)
```

Bit 0

5.154.2.2051 FSMC_PCR3_ECCPS_1

```
#define FSMC_PCR3_ECCPS_1 ((uint32_t)0x00040000)
```

Bit 1

5.154.2.2052 FSMC_PCR3_ECCPS_2

```
#define FSMC_PCR3_ECCPS_2 ((uint32_t)0x00080000)
```

Bit 2

5.154.2.2053 FSMC_PCR3_PBKEN

```
#define FSMC_PCR3_PBKEN ((uint32_t)0x00000004)
```

PC Card/NAND Flash memory bank enable bit

5.154.2.2054 FSMC_PCR3_PTYP

```
#define FSMC_PCR3_PTYP ((uint32_t)0x00000008)
```

Memory type

5.154.2.2055 FSMC_PCR3_PWAITEN

```
#define FSMC_PCR3_PWAITEN ((uint32_t)0x00000002)
```

Wait feature enable bit

5.154.2.2056 FSMC_PCR3_PWID

```
#define FSMC_PCR3_PWID ((uint32_t)0x00000030)
```

PWID[1:0] bits (NAND Flash databus width)

5.154.2.2057 FSMC_PCR3_PWID_0

```
#define FSMC_PCR3_PWID_0 ((uint32_t)0x00000010)
```

Bit 0

5.154.2.2058 FSMC_PCR3_PWID_1

```
#define FSMC_PCR3_PWID_1 ((uint32_t)0x00000020)
```

Bit 1

5.154.2.2059 FSMC_PCR3_TAR

```
#define FSMC_PCR3_TAR ((uint32_t)0x0001E000)
```

TAR[3:0] bits (ALE to RE delay)

5.154.2.2060 FSMC_PCR3_TAR_0

```
#define FSMC_PCR3_TAR_0 ((uint32_t)0x00002000)
```

Bit 0

5.154.2.2061 FSMC_PCR3_TAR_1

```
#define FSMC_PCR3_TAR_1 ((uint32_t)0x00004000)
```

Bit 1

5.154.2.2062 FSMC_PCR3_TAR_2

```
#define FSMC_PCR3_TAR_2 ((uint32_t)0x00008000)
```

Bit 2

5.154.2.2063 FSMC_PCR3_TAR_3

```
#define FSMC_PCR3_TAR_3 ((uint32_t)0x00010000)
```

Bit 3

5.154.2.2064 FSMC_PCR3_TCLR

```
#define FSMC_PCR3_TCLR ((uint32_t)0x00001E00)
```

TCLR[3:0] bits (CLE to RE delay)

5.154.2.2065 FSMC_PCR3_TCLR_0

```
#define FSMC_PCR3_TCLR_0 ((uint32_t)0x00000200)
```

Bit 0

5.154.2.2066 FSMC_PCR3_TCLR_1

```
#define FSMC_PCR3_TCLR_1 ((uint32_t)0x00000400)
```

Bit 1

5.154.2.2067 FSMC_PCR3_TCLR_2

```
#define FSMC_PCR3_TCLR_2 ((uint32_t)0x00000800)
```

Bit 2

5.154.2.2068 FSMC_PCR3_TCLR_3

```
#define FSMC_PCR3_TCLR_3 ((uint32_t)0x00001000)
```

Bit 3

5.154.2.2069 FSMC_PCR4_ECCEN

```
#define FSMC_PCR4_ECCEN ((uint32_t)0x00000040)
```

ECC computation logic enable bit

5.154.2.2070 FSMC_PCR4_ECCPS

```
#define FSMC_PCR4_ECCPS ((uint32_t)0x000E0000)
```

ECCPS[2:0] bits (ECC page size)

5.154.2.2071 FSMC_PCR4_ECCPS_0

```
#define FSMC_PCR4_ECCPS_0 ((uint32_t)0x00020000)
```

Bit 0

5.154.2.2072 FSMC_PCR4_ECCPS_1

```
#define FSMC_PCR4_ECCPS_1 ((uint32_t)0x00040000)
```

Bit 1

5.154.2.2073 FSMC_PCR4_ECCPS_2

```
#define FSMC_PCR4_ECCPS_2 ((uint32_t)0x00080000)
```

Bit 2

5.154.2.2074 FSMC_PCR4_PBKEN

```
#define FSMC_PCR4_PBKEN ((uint32_t)0x00000004)
```

PC Card/NAND Flash memory bank enable bit

5.154.2.2075 FSMC_PCR4_PTYP

```
#define FSMC_PCR4_PTYP ((uint32_t)0x00000008)
```

Memory type

5.154.2.2076 FSMC_PCR4_PWAITEN

```
#define FSMC_PCR4_PWAITEN ((uint32_t)0x00000002)
```

Wait feature enable bit

5.154.2.2077 FSMC_PCR4_PWID

```
#define FSMC_PCR4_PWID ((uint32_t)0x00000030)
```

PWID[1:0] bits (NAND Flash databus width)

5.154.2.2078 FSMC_PCR4_PWID_0

```
#define FSMC_PCR4_PWID_0 ((uint32_t)0x00000010)
```

Bit 0

5.154.2.2079 FSMC_PCR4_PWID_1

```
#define FSMC_PCR4_PWID_1 ((uint32_t)0x00000020)
```

Bit 1

5.154.2.2080 FSMC_PCR4_TAR

```
#define FSMC_PCR4_TAR ((uint32_t)0x0001E000)
```

TAR[3:0] bits (ALE to RE delay)

5.154.2.2081 FSMC_PCR4_TAR_0

```
#define FSMC_PCR4_TAR_0 ((uint32_t)0x00002000)
```

Bit 0

5.154.2.2082 FSMC_PCR4_TAR_1

```
#define FSMC_PCR4_TAR_1 ((uint32_t)0x00004000)
```

Bit 1

5.154.2.2083 FSMC_PCR4_TAR_2

```
#define FSMC_PCR4_TAR_2 ((uint32_t)0x00008000)
```

Bit 2

5.154.2.2084 FSMC_PCR4_TAR_3

```
#define FSMC_PCR4_TAR_3 ((uint32_t)0x00010000)
```

Bit 3

5.154.2.2085 FSMC_PCR4_TCLR

```
#define FSMC_PCR4_TCLR ((uint32_t)0x00001E00)
```

TCLR[3:0] bits (CLE to RE delay)

5.154.2.2086 FSMC_PCR4_TCLR_0

```
#define FSMC_PCR4_TCLR_0 ((uint32_t)0x00000200)
```

Bit 0

5.154.2.2087 FSMC_PCR4_TCLR_1

```
#define FSMC_PCR4_TCLR_1 ((uint32_t)0x00000400)
```

Bit 1

5.154.2.2088 FSMC_PCR4_TCLR_2

```
#define FSMC_PCR4_TCLR_2 ((uint32_t)0x00000800)
```

Bit 2

5.154.2.2089 FSMC_PCR4_TCLR_3

```
#define FSMC_PCR4_TCLR_3 ((uint32_t)0x00001000)
```

Bit 3

5.154.2.2090 FSMC_PIO4_IOHIZ4

```
#define FSMC_PIO4_IOHIZ4 ((uint32_t)0xFF000000)
```

IOHIZ4[7:0] bits (I/O 4 databus HiZ time)

5.154.2.2091 FSMC_PIO4_IOHIZ4_0

```
#define FSMC_PIO4_IOHIZ4_0 ((uint32_t)0x01000000)
```

Bit 0

5.154.2.2092 FSMC_PIO4_IOHIZ4_1

```
#define FSMC_PIO4_IOHIZ4_1 ((uint32_t)0x02000000)
```

Bit 1

5.154.2.2093 FSMC_PIO4_IOHIZ4_2

```
#define FSMC_PIO4_IOHIZ4_2 ((uint32_t)0x04000000)
```

Bit 2

5.154.2.2094 FSMC_PIO4_IOHIZ4_3

```
#define FSMC_PIO4_IOHIZ4_3 ((uint32_t)0x08000000)
```

Bit 3

5.154.2.2095 FSMC_PIO4_IOHIZ4_4

```
#define FSMC_PIO4_IOHIZ4_4 ((uint32_t)0x10000000)
```

Bit 4

5.154.2.2096 FSMC_PIO4_IOHIZ4_5

```
#define FSMC_PIO4_IOHIZ4_5 ((uint32_t)0x20000000)
```

Bit 5

5.154.2.2097 FSMC_PIO4_IOHIZ4_6

```
#define FSMC_PIO4_IOHIZ4_6 ((uint32_t)0x40000000)
```

Bit 6

5.154.2.2098 FSMC_PIO4_IOHIZ4_7

```
#define FSMC_PIO4_IOHIZ4_7 ((uint32_t)0x80000000)
```

Bit 7

5.154.2.2099 FSMC_PIO4_IOHOLD4

```
#define FSMC_PIO4_IOHOLD4 ((uint32_t)0x00FF0000)
```

IOHOLD4[7:0] bits (I/O 4 hold time)

5.154.2.2100 FSMC_PIO4_IOHOLD4_0

```
#define FSMC_PIO4_IOHOLD4_0 ((uint32_t)0x00010000)
```

Bit 0

5.154.2.2101 FSMC_PIO4_IOHOLD4_1

```
#define FSMC_PIO4_IOHOLD4_1 ((uint32_t)0x00020000)
```

Bit 1

5.154.2.2102 FSMC_PIO4_IOHOLD4_2

```
#define FSMC_PIO4_IOHOLD4_2 ((uint32_t)0x00040000)
```

Bit 2

5.154.2.2103 FSMC_PIO4_IOHOLD4_3

```
#define FSMC_PIO4_IOHOLD4_3 ((uint32_t)0x00080000)
```

Bit 3

5.154.2.2104 FSMC_PIO4_IOHOLD4_4

```
#define FSMC_PIO4_IOHOLD4_4 ((uint32_t)0x00100000)
```

Bit 4

5.154.2.2105 FSMC_PIO4_IOHOLD4_5

```
#define FSMC_PIO4_IOHOLD4_5 ((uint32_t)0x00200000)
```

Bit 5

5.154.2.2106 FSMC_PIO4_IOHOLD4_6

```
#define FSMC_PIO4_IOHOLD4_6 ((uint32_t)0x00400000)
```

Bit 6

5.154.2.2107 FSMC_PIO4_IOHOLD4_7

```
#define FSMC_PIO4_IOHOLD4_7 ((uint32_t)0x00800000)
```

Bit 7

5.154.2.2108 FSMC_PIO4_IOSET4

```
#define FSMC_PIO4_IOSET4 ((uint32_t)0x000000FF)
```

IOSET4[7:0] bits (I/O 4 setup time)

5.154.2.2109 FSMC_PIO4_IOSET4_0

```
#define FSMC_PIO4_IOSET4_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.2110 FSMC_PIO4_IOSET4_1

```
#define FSMC_PIO4_IOSET4_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.2111 FSMC_PIO4_IOSET4_2

```
#define FSMC_PIO4_IOSET4_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.2112 FSMC_PIO4_IOSET4_3

```
#define FSMC_PIO4_IOSET4_3 ((uint32_t)0x00000008)
```

Bit 3

5.154.2.2113 FSMC_PIO4_IOSET4_4

```
#define FSMC_PIO4_IOSET4_4 ((uint32_t)0x00000010)
```

Bit 4

5.154.2.2114 FSMC_PIO4_IOSET4_5

```
#define FSMC_PIO4_IOSET4_5 ((uint32_t)0x00000020)
```

Bit 5

5.154.2.2115 FSMC_PIO4_IOSET4_6

```
#define FSMC_PIO4_IOSET4_6 ((uint32_t)0x00000040)
```

Bit 6

5.154.2.2116 FSMC_PIO4_IOSET4_7

```
#define FSMC_PIO4_IOSET4_7 ((uint32_t)0x00000080)
```

Bit 7

5.154.2.2117 FSMC_PIO4_IOWAIT4

```
#define FSMC_PIO4_IOWAIT4 ((uint32_t)0x0000FF00)
```

IOWAIT4[7:0] bits (I/O 4 wait time)

5.154.2.2118 FSMC_PIO4_IOWAIT4_0

```
#define FSMC_PIO4_IOWAIT4_0 ((uint32_t)0x00000100)
```

Bit 0

5.154.2.2119 FSMC_PIO4_IOWAIT4_1

```
#define FSMC_PIO4_IOWAIT4_1 ((uint32_t)0x00000200)
```

Bit 1

5.154.2.2120 FSMC_PIO4_IOWAIT4_2

```
#define FSMC_PIO4_IOWAIT4_2 ((uint32_t)0x00000400)
```

Bit 2

5.154.2.2121 FSMC_PIO4_IOWAIT4_3

```
#define FSMC_PIO4_IOWAIT4_3 ((uint32_t)0x00000800)
```

Bit 3

5.154.2.2122 FSMC_PIO4_IOWAIT4_4

```
#define FSMC_PIO4_IOWAIT4_4 ((uint32_t)0x00001000)
```

Bit 4

5.154.2.2123 FSMC_PIO4_IOWAIT4_5

```
#define FSMC_PIO4_IOWAIT4_5 ((uint32_t)0x00002000)
```

Bit 5

5.154.2.2124 FSMC_PIO4_IOWAIT4_6

```
#define FSMC_PIO4_IOWAIT4_6 ((uint32_t)0x00004000)
```

Bit 6

5.154.2.2125 FSMC_PIO4_IOWAIT4_7

```
#define FSMC_PIO4_IOWAIT4_7 ((uint32_t)0x00008000)
```

Bit 7

5.154.2.2126 FSMC_PMEM2_MEMHIZ2

```
#define FSMC_PMEM2_MEMHIZ2 ((uint32_t)0xFF000000)
```

MEMHIZ2[7:0] bits (Common memory 2 databus HiZ time)

5.154.2.2127 FSMC_PMEM2_MEMHIZ2_0

```
#define FSMC_PMEM2_MEMHIZ2_0 ((uint32_t)0x01000000)
```

Bit 0

5.154.2.2128 FSMC_PMEM2_MEMHIZ2_1

```
#define FSMC_PMEM2_MEMHIZ2_1 ((uint32_t)0x02000000)
```

Bit 1

5.154.2.2129 FSMC_PMEM2_MEMHIZ2_2

```
#define FSMC_PMEM2_MEMHIZ2_2 ((uint32_t)0x04000000)
```

Bit 2

5.154.2.2130 FSMC_PMEM2_MEMHIZ2_3

```
#define FSMC_PMEM2_MEMHIZ2_3 ((uint32_t)0x08000000)
```

Bit 3

5.154.2.2131 FSMC_PMEM2_MEMHIZ2_4

```
#define FSMC_PMEM2_MEMHIZ2_4 ((uint32_t)0x10000000)
```

Bit 4

5.154.2.2132 FSMC_PMEM2_MEMHIZ2_5

```
#define FSMC_PMEM2_MEMHIZ2_5 ((uint32_t)0x20000000)
```

Bit 5

5.154.2.2133 FSMC_PMEM2_MEMHIZ2_6

```
#define FSMC_PMEM2_MEMHIZ2_6 ((uint32_t)0x40000000)
```

Bit 6

5.154.2.2134 FSMC_PMEM2_MEMHIZ2_7

```
#define FSMC_PMEM2_MEMHIZ2_7 ((uint32_t)0x80000000)
```

Bit 7

5.154.2.2135 FSMC_PMEM2_MEMHOLD2

```
#define FSMC_PMEM2_MEMHOLD2 ((uint32_t)0x00FF0000)
```

MEMHOLD2[7:0] bits (Common memory 2 hold time)

5.154.2.2136 FSMC_PMEM2_MEMHOLD2_0

```
#define FSMC_PMEM2_MEMHOLD2_0 ((uint32_t)0x00010000)
```

Bit 0

5.154.2.2137 FSMC_PMEM2_MEMHOLD2_1

```
#define FSMC_PMEM2_MEMHOLD2_1 ((uint32_t)0x00020000)
```

Bit 1

5.154.2.2138 FSMC_PMEM2_MEMHOLD2_2

```
#define FSMC_PMEM2_MEMHOLD2_2 ((uint32_t)0x00040000)
```

Bit 2

5.154.2.2139 FSMC_PMEM2_MEMHOLD2_3

```
#define FSMC_PMEM2_MEMHOLD2_3 ((uint32_t)0x00080000)
```

Bit 3

5.154.2.2140 FSMC_PMEM2_MEMHOLD2_4

```
#define FSMC_PMEM2_MEMHOLD2_4 ((uint32_t)0x00100000)
```

Bit 4

5.154.2.2141 FSMC_PMEM2_MEMHOLD2_5

```
#define FSMC_PMEM2_MEMHOLD2_5 ((uint32_t)0x00200000)
```

Bit 5

5.154.2.2142 FSMC_PMEM2_MEMHOLD2_6

```
#define FSMC_PMEM2_MEMHOLD2_6 ((uint32_t)0x00400000)
```

Bit 6

5.154.2.2143 FSMC_PMEM2_MEMHOLD2_7

```
#define FSMC_PMEM2_MEMHOLD2_7 ((uint32_t)0x00800000)
```

Bit 7

5.154.2.2144 FSMC_PMEM2_MEMSET2

```
#define FSMC_PMEM2_MEMSET2 ((uint32_t)0x000000FF)
```

MEMSET2[7:0] bits (Common memory 2 setup time)

5.154.2.2145 FSMC_PMEM2_MEMSET2_0

```
#define FSMC_PMEM2_MEMSET2_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.2146 FSMC_PMEM2_MEMSET2_1

```
#define FSMC_PMEM2_MEMSET2_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.2147 FSMC_PMEM2_MEMSET2_2

```
#define FSMC_PMEM2_MEMSET2_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.2148 FSMC_PMEM2_MEMSET2_3

```
#define FSMC_PMEM2_MEMSET2_3 ((uint32_t)0x00000008)
```

Bit 3

5.154.2.2149 FSMC_PMEM2_MEMSET2_4

```
#define FSMC_PMEM2_MEMSET2_4 ((uint32_t)0x00000010)
```

Bit 4

5.154.2.2150 FSMC_PMEM2_MEMSET2_5

```
#define FSMC_PMEM2_MEMSET2_5 ((uint32_t)0x00000020)
```

Bit 5

5.154.2.2151 FSMC_PMEM2_MEMSET2_6

```
#define FSMC_PMEM2_MEMSET2_6 ((uint32_t)0x00000040)
```

Bit 6

5.154.2.2152 FSMC_PMEM2_MEMSET2_7

```
#define FSMC_PMEM2_MEMSET2_7 ((uint32_t)0x00000080)
```

Bit 7

5.154.2.2153 FSMC_PMEM2_MEMWAIT2

```
#define FSMC_PMEM2_MEMWAIT2 ((uint32_t)0x0000FF00)
```

MEMWAIT2[7:0] bits (Common memory 2 wait time)

5.154.2.2154 FSMC_PMEM2_MEMWAIT2_0

```
#define FSMC_PMEM2_MEMWAIT2_0 ((uint32_t)0x00000100)
```

Bit 0

5.154.2.2155 FSMC_PMEM2_MEMWAIT2_1

```
#define FSMC_PMEM2_MEMWAIT2_1 ((uint32_t)0x00000200)
```

Bit 1

5.154.2.2156 FSMC_PMEM2_MEMWAIT2_2

```
#define FSMC_PMEM2_MEMWAIT2_2 ((uint32_t)0x00000400)
```

Bit 2

5.154.2.2157 FSMC_PMEM2_MEMWAIT2_3

```
#define FSMC_PMEM2_MEMWAIT2_3 ((uint32_t)0x00000800)
```

Bit 3

5.154.2.2158 FSMC_PMEM2_MEMWAIT2_4

```
#define FSMC_PMEM2_MEMWAIT2_4 ((uint32_t)0x00001000)
```

Bit 4

5.154.2.2159 FSMC_PMEM2_MEMWAIT2_5

```
#define FSMC_PMEM2_MEMWAIT2_5 ((uint32_t)0x00002000)
```

Bit 5

5.154.2.2160 FSMC_PMEM2_MEMWAIT2_6

```
#define FSMC_PMEM2_MEMWAIT2_6 ((uint32_t)0x00004000)
```

Bit 6

5.154.2.2161 FSMC_PMEM2_MEMWAIT2_7

```
#define FSMC_PMEM2_MEMWAIT2_7 ((uint32_t)0x00008000)
```

Bit 7

5.154.2.2162 FSMC_PMEM3_MEMHIZ3

```
#define FSMC_PMEM3_MEMHIZ3 ((uint32_t)0xFF000000)
```

MEMHIZ3[7:0] bits (Common memory 3 databus HiZ time)

5.154.2.2163 FSMC_PMEM3_MEMHIZ3_0

```
#define FSMC_PMEM3_MEMHIZ3_0 ((uint32_t)0x01000000)
```

Bit 0

5.154.2.2164 FSMC_PMEM3_MEMHIZ3_1

```
#define FSMC_PMEM3_MEMHIZ3_1 ((uint32_t)0x02000000)
```

Bit 1

5.154.2.2165 FSMC_PMEM3_MEMHIZ3_2

```
#define FSMC_PMEM3_MEMHIZ3_2 ((uint32_t)0x04000000)
```

Bit 2

5.154.2.2166 FSMC_PMEM3_MEMHIZ3_3

```
#define FSMC_PMEM3_MEMHIZ3_3 ((uint32_t)0x08000000)
```

Bit 3

5.154.2.2167 FSMC_PMEM3_MEMHIZ3_4

```
#define FSMC_PMEM3_MEMHIZ3_4 ((uint32_t)0x10000000)
```

Bit 4

5.154.2.2168 FSMC_PMEM3_MEMHIZ3_5

```
#define FSMC_PMEM3_MEMHIZ3_5 ((uint32_t)0x20000000)
```

Bit 5

5.154.2.2169 FSMC_PMEM3_MEMHIZ3_6

```
#define FSMC_PMEM3_MEMHIZ3_6 ((uint32_t)0x40000000)
```

Bit 6

5.154.2.2170 FSMC_PMEM3_MEMHIZ3_7

```
#define FSMC_PMEM3_MEMHIZ3_7 ((uint32_t)0x80000000)
```

Bit 7

5.154.2.2171 FSMC_PMEM3_MEMHOLD3

```
#define FSMC_PMEM3_MEMHOLD3 ((uint32_t)0x00FF0000)
```

MEMHOLD3[7:0] bits (Common memory 3 hold time)

5.154.2.2172 FSMC_PMEM3_MEMHOLD3_0

```
#define FSMC_PMEM3_MEMHOLD3_0 ((uint32_t)0x00010000)
```

Bit 0

5.154.2.2173 FSMC_PMEM3_MEMHOLD3_1

```
#define FSMC_PMEM3_MEMHOLD3_1 ((uint32_t)0x00020000)
```

Bit 1

5.154.2.2174 FSMC_PMEM3_MEMHOLD3_2

```
#define FSMC_PMEM3_MEMHOLD3_2 ((uint32_t)0x00040000)
```

Bit 2

5.154.2.2175 FSMC_PMEM3_MEMHOLD3_3

```
#define FSMC_PMEM3_MEMHOLD3_3 ((uint32_t)0x00080000)
```

Bit 3

5.154.2.2176 FSMC_PMEM3_MEMHOLD3_4

```
#define FSMC_PMEM3_MEMHOLD3_4 ((uint32_t)0x00100000)
```

Bit 4

5.154.2.2177 FSMC_PMEM3_MEMHOLD3_5

```
#define FSMC_PMEM3_MEMHOLD3_5 ((uint32_t)0x00200000)
```

Bit 5

5.154.2.2178 FSMC_PMEM3_MEMHOLD3_6

```
#define FSMC_PMEM3_MEMHOLD3_6 ((uint32_t)0x00400000)
```

Bit 6

5.154.2.2179 FSMC_PMEM3_MEMHOLD3_7

```
#define FSMC_PMEM3_MEMHOLD3_7 ((uint32_t)0x00800000)
```

Bit 7

5.154.2.2180 FSMC_PMEM3_MEMSET3

```
#define FSMC_PMEM3_MEMSET3 ((uint32_t)0x000000FF)
```

MEMSET3[7:0] bits (Common memory 3 setup time)

5.154.2.2181 FSMC_PMEM3_MEMSET3_0

```
#define FSMC_PMEM3_MEMSET3_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.2182 FSMC_PMEM3_MEMSET3_1

```
#define FSMC_PMEM3_MEMSET3_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.2183 FSMC_PMEM3_MEMSET3_2

```
#define FSMC_PMEM3_MEMSET3_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.2184 FSMC_PMEM3_MEMSET3_3

```
#define FSMC_PMEM3_MEMSET3_3 ((uint32_t)0x00000008)
```

Bit 3

5.154.2.2185 FSMC_PMEM3_MEMSET3_4

```
#define FSMC_PMEM3_MEMSET3_4 ((uint32_t)0x00000010)
```

Bit 4

5.154.2.2186 FSMC_PMEM3_MEMSET3_5

```
#define FSMC_PMEM3_MEMSET3_5 ((uint32_t)0x00000020)
```

Bit 5

5.154.2.2187 FSMC_PMEM3_MEMSET3_6

```
#define FSMC_PMEM3_MEMSET3_6 ((uint32_t)0x00000040)
```

Bit 6

5.154.2.2188 FSMC_PMEM3_MEMSET3_7

```
#define FSMC_PMEM3_MEMSET3_7 ((uint32_t)0x00000080)
```

Bit 7

5.154.2.2189 FSMC_PMEM3_MEMWAIT3

```
#define FSMC_PMEM3_MEMWAIT3 ((uint32_t)0x0000FF00)
```

MEMWAIT3[7:0] bits (Common memory 3 wait time)

5.154.2.2190 FSMC_PMEM3_MEMWAIT3_0

```
#define FSMC_PMEM3_MEMWAIT3_0 ((uint32_t)0x00000100)
```

Bit 0

5.154.2.2191 FSMC_PMEM3_MEMWAIT3_1

```
#define FSMC_PMEM3_MEMWAIT3_1 ((uint32_t)0x00000200)
```

Bit 1

5.154.2.2192 FSMC_PMEM3_MEMWAIT3_2

```
#define FSMC_PMEM3_MEMWAIT3_2 ((uint32_t)0x00000400)
```

Bit 2

5.154.2.2193 FSMC_PMEM3_MEMWAIT3_3

```
#define FSMC_PMEM3_MEMWAIT3_3 ((uint32_t)0x00000800)
```

Bit 3

5.154.2.2194 FSMC_PMEM3_MEMWAIT3_4

```
#define FSMC_PMEM3_MEMWAIT3_4 ((uint32_t)0x00001000)
```

Bit 4

5.154.2.2195 FSMC_PMEM3_MEMWAIT3_5

```
#define FSMC_PMEM3_MEMWAIT3_5 ((uint32_t)0x00002000)
```

Bit 5

5.154.2.2196 FSMC_PMEM3_MEMWAIT3_6

```
#define FSMC_PMEM3_MEMWAIT3_6 ((uint32_t)0x00004000)
```

Bit 6

5.154.2.2197 FSMC_PMEM3_MEMWAIT3_7

```
#define FSMC_PMEM3_MEMWAIT3_7 ((uint32_t)0x00008000)
```

Bit 7

5.154.2.2198 FSMC_PMEM4_MEMHIZ4

```
#define FSMC_PMEM4_MEMHIZ4 ((uint32_t)0xFF000000)
```

MEMHIZ4[7:0] bits (Common memory 4 databus HiZ time)

5.154.2.2199 FSMC_PMEM4_MEMHIZ4_0

```
#define FSMC_PMEM4_MEMHIZ4_0 ((uint32_t)0x01000000)
```

Bit 0

5.154.2.2200 FSMC_PMEM4_MEMHIZ4_1

```
#define FSMC_PMEM4_MEMHIZ4_1 ((uint32_t)0x02000000)
```

Bit 1

5.154.2.2201 FSMC_PMEM4_MEMHIZ4_2

```
#define FSMC_PMEM4_MEMHIZ4_2 ((uint32_t)0x04000000)
```

Bit 2

5.154.2.2202 FSMC_PMEM4_MEMHIZ4_3

```
#define FSMC_PMEM4_MEMHIZ4_3 ((uint32_t)0x08000000)
```

Bit 3

5.154.2.2203 FSMC_PMEM4_MEMHIZ4_4

```
#define FSMC_PMEM4_MEMHIZ4_4 ((uint32_t)0x10000000)
```

Bit 4

5.154.2.2204 FSMC_PMEM4_MEMHIZ4_5

```
#define FSMC_PMEM4_MEMHIZ4_5 ((uint32_t)0x20000000)
```

Bit 5

5.154.2.2205 FSMC_PMEM4_MEMHIZ4_6

```
#define FSMC_PMEM4_MEMHIZ4_6 ((uint32_t)0x40000000)
```

Bit 6

5.154.2.2206 FSMC_PMEM4_MEMHIZ4_7

```
#define FSMC_PMEM4_MEMHIZ4_7 ((uint32_t)0x80000000)
```

Bit 7

5.154.2.2207 FSMC_PMEM4_MEMHOLD4

```
#define FSMC_PMEM4_MEMHOLD4 ((uint32_t)0x00FF0000)
```

MEMHOLD4[7:0] bits (Common memory 4 hold time)

5.154.2.2208 FSMC_PMEM4_MEMHOLD4_0

```
#define FSMC_PMEM4_MEMHOLD4_0 ((uint32_t)0x00010000)
```

Bit 0

5.154.2.2209 FSMC_PMEM4_MEMHOLD4_1

```
#define FSMC_PMEM4_MEMHOLD4_1 ((uint32_t)0x00020000)
```

Bit 1

5.154.2.2210 FSMC_PMEM4_MEMHOLD4_2

```
#define FSMC_PMEM4_MEMHOLD4_2 ((uint32_t)0x00040000)
```

Bit 2

5.154.2.2211 FSMC_PMEM4_MEMHOLD4_3

```
#define FSMC_PMEM4_MEMHOLD4_3 ((uint32_t)0x00080000)
```

Bit 3

5.154.2.2212 FSMC_PMEM4_MEMHOLD4_4

```
#define FSMC_PMEM4_MEMHOLD4_4 ((uint32_t)0x00100000)
```

Bit 4

5.154.2.2213 FSMC_PMEM4_MEMHOLD4_5

```
#define FSMC_PMEM4_MEMHOLD4_5 ((uint32_t)0x00200000)
```

Bit 5

5.154.2.2214 FSMC_PMEM4_MEMHOLD4_6

```
#define FSMC_PMEM4_MEMHOLD4_6 ((uint32_t)0x00400000)
```

Bit 6

5.154.2.2215 FSMC_PMEM4_MEMHOLD4_7

```
#define FSMC_PMEM4_MEMHOLD4_7 ((uint32_t)0x00800000)
```

Bit 7

5.154.2.2216 FSMC_PMEM4_MEMSET4

```
#define FSMC_PMEM4_MEMSET4 ((uint32_t)0x000000FF)
```

MEMSET4[7:0] bits (Common memory 4 setup time)

5.154.2.2217 FSMC_PMEM4_MEMSET4_0

```
#define FSMC_PMEM4_MEMSET4_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.2218 FSMC_PMEM4_MEMSET4_1

```
#define FSMC_PMEM4_MEMSET4_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.2219 FSMC_PMEM4_MEMSET4_2

```
#define FSMC_PMEM4_MEMSET4_2 ((uint32_t)0x00000004)
```

Bit 2

5.154.2.2220 FSMC_PMEM4_MEMSET4_3

```
#define FSMC_PMEM4_MEMSET4_3 ((uint32_t)0x00000008)
```

Bit 3

5.154.2.2221 FSMC_PMEM4_MEMSET4_4

```
#define FSMC_PMEM4_MEMSET4_4 ((uint32_t)0x00000010)
```

Bit 4

5.154.2.2222 FSMC_PMEM4_MEMSET4_5

```
#define FSMC_PMEM4_MEMSET4_5 ((uint32_t)0x00000020)
```

Bit 5

5.154.2.2223 FSMC_PMEM4_MEMSET4_6

```
#define FSMC_PMEM4_MEMSET4_6 ((uint32_t)0x00000040)
```

Bit 6

5.154.2.2224 FSMC_PMEM4_MEMSET4_7

```
#define FSMC_PMEM4_MEMSET4_7 ((uint32_t)0x00000080)
```

Bit 7

5.154.2.2225 FSMC_PMEM4_MEMWAIT4

```
#define FSMC_PMEM4_MEMWAIT4 ((uint32_t)0x0000FF00)
```

MEMWAIT4[7:0] bits (Common memory 4 wait time)

5.154.2.2226 FSMC_PMEM4_MEMWAIT4_0

```
#define FSMC_PMEM4_MEMWAIT4_0 ((uint32_t)0x00000100)
```

Bit 0

5.154.2.2227 FSMC_PMEM4_MEMWAIT4_1

```
#define FSMC_PMEM4_MEMWAIT4_1 ((uint32_t)0x00000200)
```

Bit 1

5.154.2.2228 FSMC_PMEM4_MEMWAIT4_2

```
#define FSMC_PMEM4_MEMWAIT4_2 ((uint32_t)0x00000400)
```

Bit 2

5.154.2.2229 FSMC_PMEM4_MEMWAIT4_3

```
#define FSMC_PMEM4_MEMWAIT4_3 ((uint32_t)0x00000800)
```

Bit 3

5.154.2.2230 FSMC_PMEM4_MEMWAIT4_4

```
#define FSMC_PMEM4_MEMWAIT4_4 ((uint32_t)0x00001000)
```

Bit 4

5.154.2.2231 FSMC_PMEM4_MEMWAIT4_5

```
#define FSMC_PMEM4_MEMWAIT4_5 ((uint32_t)0x00002000)
```

Bit 5

5.154.2.2232 FSMC_PMEM4_MEMWAIT4_6

```
#define FSMC_PMEM4_MEMWAIT4_6 ((uint32_t)0x00004000)
```

Bit 6

5.154.2.2233 FSMC_PMEM4_MEMWAIT4_7

```
#define FSMC_PMEM4_MEMWAIT4_7 ((uint32_t)0x00008000)
```

Bit 7

5.154.2.2234 FSMC_SR2_FEMPT

```
#define FSMC_SR2_FEMPT ((uint8_t)0x40)
```

FIFO empty

5.154.2.2235 FSMC_SR2_IFEN

```
#define FSMC_SR2_IFEN ((uint8_t)0x20)
```

Interrupt Falling Edge detection Enable bit

5.154.2.2236 FSMC_SR2_IFS

```
#define FSMC_SR2_IFS ((uint8_t)0x04)
```

Interrupt Falling Edge status

5.154.2.2237 FSMC_SR2_ILEN

```
#define FSMC_SR2_ILEN ((uint8_t)0x10)
```

Interrupt Level detection Enable bit

5.154.2.2238 FSMC_SR2_ILS

```
#define FSMC_SR2_ILS ((uint8_t)0x02)
```

Interrupt Level status

5.154.2.2239 FSMC_SR2_IREN

```
#define FSMC_SR2_IREN ((uint8_t)0x08)
```

Interrupt Rising Edge detection Enable bit

5.154.2.2240 FSMC_SR2_IRS

```
#define FSMC_SR2_IRS ((uint8_t)0x01)
```

Interrupt Rising Edge status

5.154.2.2241 FSMC_SR3_FEMPT

```
#define FSMC_SR3_FEMPT ((uint8_t)0x40)
```

FIFO empty

5.154.2.2242 FSMC_SR3_IFEN

```
#define FSMC_SR3_IFEN ((uint8_t)0x20)
```

Interrupt Falling Edge detection Enable bit

5.154.2.2243 FSMC_SR3_IFS

```
#define FSMC_SR3_IFS ((uint8_t)0x04)
```

Interrupt Falling Edge status

5.154.2.2244 FSMC_SR3_ILEN

```
#define FSMC_SR3_ILEN ((uint8_t)0x10)
```

Interrupt Level detection Enable bit

5.154.2.2245 FSMC_SR3_ILS

```
#define FSMC_SR3_ILS ((uint8_t)0x02)
```

Interrupt Level status

5.154.2.2246 FSMC_SR3_IREN

```
#define FSMC_SR3_IREN ((uint8_t)0x08)
```

Interrupt Rising Edge detection Enable bit

5.154.2.2247 FSMC_SR3_IRS

```
#define FSMC_SR3_IRS ((uint8_t)0x01)
```

Interrupt Rising Edge status

5.154.2.2248 FSMC_SR4_FEMPT

```
#define FSMC_SR4_FEMPT ((uint8_t)0x40)
```

FIFO empty

5.154.2.2249 FSMC_SR4_IFEN

```
#define FSMC_SR4_IFEN ((uint8_t)0x20)
```

Interrupt Falling Edge detection Enable bit

5.154.2.2250 FSMC_SR4_IFS

```
#define FSMC_SR4_IFS ((uint8_t)0x04)
```

Interrupt Falling Edge status

5.154.2.2251 FSMC_SR4_ILEN

```
#define FSMC_SR4_ILEN ((uint8_t)0x10)
```

Interrupt Level detection Enable bit

5.154.2.2252 FSMC_SR4_ILS

```
#define FSMC_SR4_ILS ((uint8_t)0x02)
```

Interrupt Level status

5.154.2.2253 FSMC_SR4_IREN

```
#define FSMC_SR4_IREN ((uint8_t)0x08)
```

Interrupt Rising Edge detection Enable bit

5.154.2.2254 FSMC_SR4_IRS

```
#define FSMC_SR4_IRS ((uint8_t)0x01)
```

Interrupt Rising Edge status

5.154.2.2255 I2C_CCR_CCR

```
#define I2C_CCR_CCR ((uint16_t)0x0FFF)
```

Clock Control Register in Fast/Standard mode (Master mode)

5.154.2.2256 I2C_CCR_DUTY

```
#define I2C_CCR_DUTY ((uint16_t)0x4000)
```

Fast Mode Duty Cycle

5.154.2.2257 I2C_CCR_FS

```
#define I2C_CCR_FS ((uint16_t)0x8000)
```

I2C Master Mode Selection

5.154.2.2258 I2C_CR1_ACK

```
#define I2C_CR1_ACK ((uint16_t)0x0400)
```

Acknowledge Enable

5.154.2.2259 I2C_CR1_ALERT

```
#define I2C_CR1_ALERT ((uint16_t)0x2000)
```

SMBus Alert

5.154.2.2260 I2C_CR1_ENARP

```
#define I2C_CR1_ENARP ((uint16_t)0x0010)
```

ARP Enable

5.154.2.2261 I2C_CR1_ENGC

```
#define I2C_CR1_ENGC ((uint16_t)0x0040)
```

General Call Enable

5.154.2.2262 I2C_CR1_ENPEC

```
#define I2C_CR1_ENPEC ((uint16_t)0x0020)
```

PEC Enable

5.154.2.2263 I2C_CR1_NOSTRETCH

```
#define I2C_CR1_NOSTRETCH ((uint16_t)0x0080)
```

Clock Stretching Disable (Slave mode)

5.154.2.2264 I2C_CR1_PE

```
#define I2C_CR1_PE ((uint16_t)0x0001)
```

Peripheral Enable

5.154.2.2265 I2C_CR1_PEC

```
#define I2C_CR1_PEC ((uint16_t)0x1000)
```

Packet Error Checking

5.154.2.2266 I2C_CR1_POS

```
#define I2C_CR1_POS ((uint16_t)0x0800)
```

Acknowledge/PEC Position (for data reception)

5.154.2.2267 I2C_CR1_SMBTYPE

```
#define I2C_CR1_SMBTYPE ((uint16_t)0x0008)
```

SMBus Type

5.154.2.2268 I2C_CR1_SMBUS

```
#define I2C_CR1_SMBUS ((uint16_t)0x0002)
```

SMBus Mode

5.154.2.2269 I2C_CR1_START

```
#define I2C_CR1_START ((uint16_t)0x0100)
```

Start Generation

5.154.2.2270 I2C_CR1_STOP

```
#define I2C_CR1_STOP ((uint16_t)0x0200)
```

Stop Generation

5.154.2.2271 I2C_CR1_SWRST

```
#define I2C_CR1_SWRST ((uint16_t)0x8000)
```

Software Reset

5.154.2.2272 I2C_CR2_DMAEN

```
#define I2C_CR2_DMAEN ((uint16_t)0x0800)
```

DMA Requests Enable

5.154.2.2273 I2C_CR2_FREQ

```
#define I2C_CR2_FREQ ((uint16_t)0x003F)
```

FREQ[5:0] bits (Peripheral Clock Frequency)

5.154.2.2274 I2C_CR2_FREQ_0

```
#define I2C_CR2_FREQ_0 ((uint16_t)0x0001)
```

Bit 0

5.154.2.2275 I2C_CR2_FREQ_1

```
#define I2C_CR2_FREQ_1 ((uint16_t)0x0002)
```

Bit 1

5.154.2.2276 I2C_CR2_FREQ_2

```
#define I2C_CR2_FREQ_2 ((uint16_t)0x0004)
```

Bit 2

5.154.2.2277 I2C_CR2_FREQ_3

```
#define I2C_CR2_FREQ_3 ((uint16_t)0x0008)
```

Bit 3

5.154.2.2278 I2C_CR2_FREQ_4

```
#define I2C_CR2_FREQ_4 ((uint16_t)0x0010)
```

Bit 4

5.154.2.2279 I2C_CR2_FREQ_5

```
#define I2C_CR2_FREQ_5 ((uint16_t)0x0020)
```

Bit 5

5.154.2.2280 I2C_CR2_ITBUFEN

```
#define I2C_CR2_ITBUFEN ((uint16_t)0x0400)
```

Buffer Interrupt Enable

5.154.2.2281 I2C_CR2_ITERREN

```
#define I2C_CR2_ITERREN ((uint16_t)0x0100)
```

Error Interrupt Enable

5.154.2.2282 I2C_CR2_ITEVTEN

```
#define I2C_CR2_ITEVTEN ((uint16_t)0x0200)
```

Event Interrupt Enable

5.154.2.2283 I2C_CR2_LAST

```
#define I2C_CR2_LAST ((uint16_t)0x1000)
```

DMA Last Transfer

5.154.2.2284 I2C_DR_DR

```
#define I2C_DR_DR ((uint8_t)0xFF)
```

8-bit Data Register

5.154.2.2285 I2C_OAR1_ADD0

```
#define I2C_OAR1_ADD0 ((uint16_t)0x0001)
```

Bit 0

5.154.2.2286 I2C_OAR1_ADD1

```
#define I2C_OAR1_ADD1 ((uint16_t)0x0002)
```

Bit 1

5.154.2.2287 I2C_OAR1_ADD1_7

```
#define I2C_OAR1_ADD1_7 ((uint16_t)0x00FE)
```

Interface Address

5.154.2.2288 I2C_OAR1_ADD2

```
#define I2C_OAR1_ADD2 ((uint16_t)0x0004)
```

Bit 2

5.154.2.2289 I2C_OAR1_ADD3

```
#define I2C_OAR1_ADD3 ((uint16_t)0x0008)
```

Bit 3

5.154.2.2290 I2C_OAR1_ADD4

```
#define I2C_OAR1_ADD4 ((uint16_t)0x0010)
```

Bit 4

5.154.2.2291 I2C_OAR1_ADD5

```
#define I2C_OAR1_ADD5 ((uint16_t)0x0020)
```

Bit 5

5.154.2.2292 I2C_OAR1_ADD6

```
#define I2C_OAR1_ADD6 ((uint16_t)0x0040)
```

Bit 6

5.154.2.2293 I2C_OAR1_ADD7

```
#define I2C_OAR1_ADD7 ((uint16_t)0x0080)
```

Bit 7

5.154.2.2294 I2C_OAR1_ADD8

```
#define I2C_OAR1_ADD8 ((uint16_t)0x0100)
```

Bit 8

5.154.2.2295 I2C_OAR1_ADD8_9

```
#define I2C_OAR1_ADD8_9 ((uint16_t)0x0300)
```

Interface Address

5.154.2.2296 I2C_OAR1_ADD9

```
#define I2C_OAR1_ADD9 ((uint16_t)0x0200)
```

Bit 9

5.154.2.2297 I2C_OAR1_ADDMODE

```
#define I2C_OAR1_ADDMODE ((uint16_t)0x8000)
```

Addressing Mode (Slave mode)

5.154.2.2298 I2C_OAR2_ADD2

```
#define I2C_OAR2_ADD2 ((uint8_t)0xFE)
```

Interface address

5.154.2.2299 I2C_OAR2_ENDUAL

```
#define I2C_OAR2_ENDUAL ((uint8_t)0x01)
```

Dual addressing mode enable

5.154.2.2300 I2C_SR1_ADD10

```
#define I2C_SR1_ADD10 ((uint16_t)0x0008)
```

10-bit header sent (Master mode)

5.154.2.2301 I2C_SR1_ADDR

```
#define I2C_SR1_ADDR ((uint16_t)0x0002)
```

Address sent (master mode)/matched (slave mode)

5.154.2.2302 I2C_SR1_AF

```
#define I2C_SR1_AF ((uint16_t)0x0400)
```

Acknowledge Failure

5.154.2.2303 I2C_SR1_ARLO

```
#define I2C_SR1_ARLO ((uint16_t)0x0200)
```

Arbitration Lost (master mode)

5.154.2.2304 I2C_SR1_BERR

```
#define I2C_SR1_BERR ((uint16_t)0x0100)
```

Bus Error

5.154.2.2305 I2C_SR1_BTF

```
#define I2C_SR1_BTF ((uint16_t)0x0004)
```

Byte Transfer Finished

5.154.2.2306 I2C_SR1_OVR

```
#define I2C_SR1_OVR ((uint16_t)0x0800)
```

Overrun/Underrun

5.154.2.2307 I2C_SR1_PECERR

```
#define I2C_SR1_PECERR ((uint16_t)0x1000)
```

PEC Error in reception

5.154.2.2308 I2C_SR1_RXNE

```
#define I2C_SR1_RXNE ((uint16_t)0x0040)
```

Data Register not Empty (receivers)

5.154.2.2309 I2C_SR1_SB

```
#define I2C_SR1_SB ((uint16_t)0x0001)
```

Start Bit (Master mode)

5.154.2.2310 I2C_SR1_SMBALERT

```
#define I2C_SR1_SMBALERT ((uint16_t)0x8000)
```

SMBus Alert

5.154.2.2311 I2C_SR1_STOPF

```
#define I2C_SR1_STOPF ((uint16_t)0x0010)
```

Stop detection (Slave mode)

5.154.2.2312 I2C_SR1_TIMEOUT

```
#define I2C_SR1_TIMEOUT ((uint16_t)0x4000)
```

Timeout or Tlow Error

5.154.2.2313 I2C_SR1_TXE

```
#define I2C_SR1_TXE ((uint16_t)0x0080)
```

Data Register Empty (transmitters)

5.154.2.2314 I2C_SR2_BUSY

```
#define I2C_SR2_BUSY ((uint16_t)0x0002)
```

Bus Busy

5.154.2.2315 I2C_SR2_DUALF

```
#define I2C_SR2_DUALF ((uint16_t)0x0080)
```

Dual Flag (Slave mode)

5.154.2.2316 I2C_SR2_GENCALL

```
#define I2C_SR2_GENCALL ((uint16_t)0x0010)
```

General Call Address (Slave mode)

5.154.2.2317 I2C_SR2_MSL

```
#define I2C_SR2_MSL ((uint16_t)0x0001)
```

Master/Slave

5.154.2.2318 I2C_SR2_PEC

```
#define I2C_SR2_PEC ((uint16_t)0xFF00)
```

Packet Error Checking Register

5.154.2.2319 I2C_SR2_SMBDEFAULT

```
#define I2C_SR2_SMBDEFAULT ((uint16_t)0x0020)
```

SMBus Device Default Address (Slave mode)

5.154.2.2320 I2C_SR2_SMBHOST

```
#define I2C_SR2_SMBHOST ((uint16_t)0x0040)
```

SMBus Host Header (Slave mode)

5.154.2.2321 I2C_SR2_TRA

```
#define I2C_SR2_TRA ((uint16_t)0x0004)
```

Transmitter/Receiver

5.154.2.2322 I2C_TRISE_TRISE

```
#define I2C_TRISE_TRISE ((uint8_t)0x3F)
```

Maximum Rise Time in Fast/Standard mode (Master mode)

5.154.2.2323 IWDG_KR_KEY

```
#define IWDG_KR_KEY ((uint16_t)0xFFFF)
```

Key value (write only, read 0000h)

5.154.2.2324 IWDG_PR_PR

```
#define IWDG_PR_PR ((uint8_t)0x07)
```

PR[2:0] (Prescaler divider)

5.154.2.2325 IWDG_PR_PR_0

```
#define IWDG_PR_PR_0 ((uint8_t)0x01)
```

Bit 0

5.154.2.2326 IWDG_PR_PR_1

```
#define IWDG_PR_PR_1 ((uint8_t)0x02)
```

Bit 1

5.154.2.2327 IWDG_PR_PR_2

```
#define IWDG_PR_PR_2 ((uint8_t)0x04)
```

Bit 2

5.154.2.2328 IWDG_RLR_RL

```
#define IWDG_RLR_RL ((uint16_t)0x0FFF)
```

Watchdog counter reload value

5.154.2.2329 IWDG_SR_PVU

```
#define IWDG_SR_PVU ((uint8_t)0x01)
```

Watchdog prescaler value update

5.154.2.2330 IWDG_SR_RVU

```
#define IWDG_SR_RVU ((uint8_t)0x02)
```

Watchdog counter reload value update

5.154.2.2331 PWR_CR_CSBF

```
#define PWR_CR_CSBF ((uint16_t)0x0008)
```

Clear Standby Flag

5.154.2.2332 PWR_CR_CWUF

```
#define PWR_CR_CWUF ((uint16_t)0x0004)
```

Clear Wakeup Flag

5.154.2.2333 PWR_CR_DBP

```
#define PWR_CR_DBP ((uint16_t)0x0100)
```

Disable Backup Domain write protection

5.154.2.2334 PWR_CR_FPDS

```
#define PWR_CR_FPDS ((uint16_t)0x0200)
```

Flash power down in Stop mode

5.154.2.2335 PWR_CR_LPDS

```
#define PWR_CR_LPDS ((uint16_t)0x0001)
```

Low-Power Deepsleep

5.154.2.2336 PWR_CR_PDDS

```
#define PWR_CR_PDDS ((uint16_t)0x0002)
```

Power Down Deepsleep

5.154.2.2337 PWR_CR_PLS

```
#define PWR_CR_PLS ((uint16_t)0x00E0)
```

PLS[2:0] bits (PVD Level Selection)

5.154.2.2338 PWR_CR_PLS_0

```
#define PWR_CR_PLS_0 ((uint16_t)0x0020)
```

Bit 0

5.154.2.2339 PWR_CR_PLS_1

```
#define PWR_CR_PLS_1 ((uint16_t)0x0040)
```

Bit 1

5.154.2.2340 PWR_CR_PLS_2

```
#define PWR_CR_PLS_2 ((uint16_t)0x0080)
```

Bit 2 PVD level configuration

5.154.2.2341 PWR_CR_PLS_LEV0

```
#define PWR_CR_PLS_LEV0 ((uint16_t)0x0000)
```

PVD level 0

5.154.2.2342 PWR_CR_PLS_LEV1

```
#define PWR_CR_PLS_LEV1 ((uint16_t)0x0020)
```

PVD level 1

5.154.2.2343 PWR_CR_PLS_LEV2

```
#define PWR_CR_PLS_LEV2 ((uint16_t)0x0040)
```

PVD level 2

5.154.2.2344 PWR_CR_PLS_LEV3

```
#define PWR_CR_PLS_LEV3 ((uint16_t)0x0060)
```

PVD level 3

5.154.2.2345 PWR_CR_PLS_LEV4

```
#define PWR_CR_PLS_LEV4 ((uint16_t)0x0080)
```

PVD level 4

5.154.2.2346 PWR_CR_PLS_LEV5

```
#define PWR_CR_PLS_LEV5 ((uint16_t)0x00A0)
```

PVD level 5

5.154.2.2347 PWR_CR_PLS_LEV6

```
#define PWR_CR_PLS_LEV6 ((uint16_t)0x00C0)
```

PVD level 6

5.154.2.2348 PWR_CR_PLS_LEV7

```
#define PWR_CR_PLS_LEV7 ((uint16_t)0x00E0)
```

PVD level 7

5.154.2.2349 PWR_CR_PVDE

```
#define PWR_CR_PVDE ((uint16_t)0x0010)
```

Power Voltage Detector Enable

5.154.2.2350 PWR_CR_VOS

```
#define PWR_CR_VOS ((uint16_t)0x4000)
```

Regulator voltage scaling output selection

5.154.2.2351 PWR_CSR_BRE

```
#define PWR_CSR_BRE ((uint16_t)0x0200)
```

Backup regulator enable

5.154.2.2352 PWR_CSR_BRR

```
#define PWR_CSR_BRR ((uint16_t)0x0008)
```

Backup regulator ready

5.154.2.2353 PWR_CSR_EWUP

```
#define PWR_CSR_EWUP ((uint16_t)0x0100)
```

Enable WKUP pin

5.154.2.2354 PWR_CSR_PVDO

```
#define PWR_CSR_PVDO ((uint16_t)0x0004)
```

PVD Output

5.154.2.2355 PWR_CSR_SBF

```
#define PWR_CSR_SBF ((uint16_t)0x0002)
```

Standby Flag

5.154.2.2356 PWR_CSR_VOSRDY

```
#define PWR_CSR_VOSRDY ((uint16_t)0x4000)
```

Regulator voltage scaling output selection ready

5.154.2.2357 PWR_CSR_WUF

```
#define PWR_CSR_WUF ((uint16_t)0x0001)
```

Wakeup Flag

5.154.2.2358 RCC_CFGR_HPRE

```
#define RCC_CFGR_HPRE ((uint32_t)0x000000F0)
```

HPRE[3:0] bits (AHB prescaler)

5.154.2.2359 RCC_CFGR_HPRE_0

```
#define RCC_CFGR_HPRE_0 ((uint32_t)0x00000010)
```

Bit 0

5.154.2.2360 RCC_CFGR_HPRE_1

```
#define RCC_CFGR_HPRE_1 ((uint32_t)0x00000020)
```

Bit 1

5.154.2.2361 RCC_CFGR_HPRE_2

```
#define RCC_CFGR_HPRE_2 ((uint32_t)0x00000040)
```

Bit 2

5.154.2.2362 RCC_CFGR_HPRE_3

```
#define RCC_CFGR_HPRE_3 ((uint32_t)0x00000080)
```

Bit 3

5.154.2.2363 RCC_CFGR_HPRE_DIV1

```
#define RCC_CFGR_HPRE_DIV1 ((uint32_t)0x00000000)
```

SYSCLK not divided

5.154.2.2364 RCC_CFGR_HPRE_DIV128

```
#define RCC_CFGR_HPRE_DIV128 ((uint32_t)0x000000D0)
```

SYSCLK divided by 128

5.154.2.2365 RCC_CFGR_HPRE_DIV16

```
#define RCC_CFGR_HPRE_DIV16 ((uint32_t)0x000000B0)
```

SYSCLK divided by 16

5.154.2.2366 RCC_CFGR_HPRE_DIV2

```
#define RCC_CFGR_HPRE_DIV2 ((uint32_t)0x00000080)
```

SYSCLK divided by 2

5.154.2.2367 RCC_CFGR_HPRE_DIV256

```
#define RCC_CFGR_HPRE_DIV256 ((uint32_t)0x000000E0)
```

SYSCLK divided by 256

5.154.2.2368 RCC_CFGR_HPRE_DIV4

```
#define RCC_CFGR_HPRE_DIV4 ((uint32_t)0x00000090)
```

SYSCLK divided by 4

5.154.2.2369 RCC_CFGR_HPRE_DIV512

```
#define RCC_CFGR_HPRE_DIV512 ((uint32_t)0x000000F0)
```

SYSCLK divided by 512 PPRE1 configuration

5.154.2.2370 RCC_CFGR_HPRE_DIV64

```
#define RCC_CFGR_HPRE_DIV64 ((uint32_t)0x000000C0)
```

SYSCLK divided by 64

5.154.2.2371 RCC_CFGR_HPRE_DIV8

```
#define RCC_CFGR_HPRE_DIV8 ((uint32_t)0x000000A0)
```

SYSCLK divided by 8

5.154.2.2372 RCC_CFGR_PPRE1

```
#define RCC_CFGR_PPRE1 ((uint32_t)0x00001C00)
```

PRE1[2:0] bits (APB1 prescaler)

5.154.2.2373 RCC_CFGR_PPRE1_0

```
#define RCC_CFGR_PPRE1_0 ((uint32_t)0x00000400)
```

Bit 0

5.154.2.2374 RCC_CFGR_PPRE1_1

```
#define RCC_CFGR_PPRE1_1 ((uint32_t)0x00000800)
```

Bit 1

5.154.2.2375 RCC_CFGR_PPRE1_2

```
#define RCC_CFGR_PPRE1_2 ((uint32_t)0x00001000)
```

Bit 2

5.154.2.2376 RCC_CFGR_PPRE1_DIV1

```
#define RCC_CFGR_PPRE1_DIV1 ((uint32_t)0x00000000)
```

HCLK not divided

5.154.2.2377 RCC_CFGR_PPRE1_DIV16

```
#define RCC_CFGR_PPRE1_DIV16 ((uint32_t)0x00001C00)
```

HCLK divided by 16 PPRE2 configuration

5.154.2.2378 RCC_CFGR_PPRE1_DIV2

```
#define RCC_CFGR_PPRE1_DIV2 ((uint32_t)0x00001000)
```

HCLK divided by 2

5.154.2.2379 RCC_CFGR_PPRE1_DIV4

```
#define RCC_CFGR_PPRE1_DIV4 ((uint32_t)0x00001400)
```

HCLK divided by 4

5.154.2.2380 RCC_CFGR_PPRE1_DIV8

```
#define RCC_CFGR_PPRE1_DIV8 ((uint32_t)0x00001800)
```

HCLK divided by 8

5.154.2.2381 RCC_CFGR_PPRE2

```
#define RCC_CFGR_PPRE2 ((uint32_t)0x0000E000)
```

PRE2[2:0] bits (APB2 prescaler)

5.154.2.2382 RCC_CFGR_PPRE2_0

```
#define RCC_CFGR_PPRE2_0 ((uint32_t)0x00002000)
```

Bit 0

5.154.2.2383 RCC_CFGR_PPRE2_1

```
#define RCC_CFGR_PPRE2_1 ((uint32_t)0x00004000)
```

Bit 1

5.154.2.2384 RCC_CFGR_PPRE2_2

```
#define RCC_CFGR_PPRE2_2 ((uint32_t)0x00008000)
```

Bit 2

5.154.2.2385 RCC_CFGR_PPRE2_DIV1

```
#define RCC_CFGR_PPRE2_DIV1 ((uint32_t)0x00000000)
```

HCLK not divided

5.154.2.2386 RCC_CFGR_PPRE2_DIV16

```
#define RCC_CFGR_PPRE2_DIV16 ((uint32_t)0x0000E000)
```

HCLK divided by 16 RTCPRE configuration

5.154.2.2387 RCC_CFGR_PPRE2_DIV2

```
#define RCC_CFGR_PPRE2_DIV2 ((uint32_t)0x00008000)
```

HCLK divided by 2

5.154.2.2388 RCC_CFGR_PPRE2_DIV4

```
#define RCC_CFGR_PPRE2_DIV4 ((uint32_t)0x0000A000)
```

HCLK divided by 4

5.154.2.2389 RCC_CFGR_PPRE2_DIV8

```
#define RCC_CFGR_PPRE2_DIV8 ((uint32_t)0x0000C000)
```

HCLK divided by 8

5.154.2.2390 RCC_CFGR_RTCPRE_4

```
#define RCC_CFGR_RTCPRE_4 ((uint32_t)0x00100000)
```

MCO1 configuration

5.154.2.2391 RCC_CFGR_SW

```
#define RCC_CFGR_SW ((uint32_t)0x00000003)
```

< SW configuration SW[1:0] bits (System clock Switch)

5.154.2.2392 RCC_CFGR_SW_0

```
#define RCC_CFGR_SW_0 ((uint32_t)0x00000001)
```

Bit 0

5.154.2.2393 RCC_CFGR_SW_1

```
#define RCC_CFGR_SW_1 ((uint32_t)0x00000002)
```

Bit 1

5.154.2.2394 RCC_CFGR_SW_HSE

```
#define RCC_CFGR_SW_HSE ((uint32_t)0x00000001)
```

HSE selected as system clock

5.154.2.2395 RCC_CFGR_SW_HSI

```
#define RCC_CFGR_SW_HSI ((uint32_t)0x00000000)
```

HSI selected as system clock

5.154.2.2396 RCC_CFGR_SW_PLL

```
#define RCC_CFGR_SW_PLL ((uint32_t)0x00000002)
```

PLL selected as system clock SWS configuration

5.154.2.2397 RCC_CFGR_SWS

```
#define RCC_CFGR_SWS ((uint32_t)0x0000000C)
```

SWS[1:0] bits (System Clock Switch Status)

5.154.2.2398 RCC_CFGR_SWS_0

```
#define RCC_CFGR_SWS_0 ((uint32_t)0x00000004)
```

Bit 0

5.154.2.2399 RCC_CFGR_SWS_1

```
#define RCC_CFGR_SWS_1 ((uint32_t)0x00000008)
```

Bit 1

5.154.2.2400 RCC_CFGR_SWS_HSE

```
#define RCC_CFGR_SWS_HSE ((uint32_t)0x00000004)
```

HSE oscillator used as system clock

5.154.2.2401 RCC_CFGR_SWS_HSI

```
#define RCC_CFGR_SWS_HSI ((uint32_t)0x00000000)
```

HSI oscillator used as system clock

5.154.2.2402 RCC_CFGR_SWS_PLL

```
#define RCC_CFGR_SWS_PLL ((uint32_t)0x00000008)
```

PLL used as system clock HPRE configuration

5.154.2.2403 RCC_CR_HSICAL_0

```
#define RCC_CR_HSICAL_0 ((uint32_t)0x00000100)
```

Bit 0

5.154.2.2404 RCC_CR_HSICAL_1

```
#define RCC_CR_HSICAL_1 ((uint32_t)0x00000200)
```

Bit 1

5.154.2.2405 RCC_CR_HSICAL_2

```
#define RCC_CR_HSICAL_2 ((uint32_t)0x00000400)
```

Bit 2

5.154.2.2406 RCC_CR_HSICAL_3

```
#define RCC_CR_HSICAL_3 ((uint32_t)0x00000800)
```

Bit 3

5.154.2.2407 RCC_CR_HSICAL_4

```
#define RCC_CR_HSICAL_4 ((uint32_t)0x00001000)
```

Bit 4

5.154.2.2408 RCC_CR_HSICAL_5

```
#define RCC_CR_HSICAL_5 ((uint32_t)0x00002000)
```

Bit 5

5.154.2.2409 RCC_CR_HSICAL_6

```
#define RCC_CR_HSICAL_6 ((uint32_t)0x00004000)
```

Bit 6

5.154.2.2410 RCC_CR_HSICAL_7

```
#define RCC_CR_HSICAL_7 ((uint32_t)0x00008000)
```

Bit 7

5.154.2.2411 RCC_CR_HSITRIM_0

```
#define RCC_CR_HSITRIM_0 ((uint32_t)0x00000008)
```

Bit 0

5.154.2.2412 RCC_CR_HSITRIM_1

```
#define RCC_CR_HSITRIM_1 ((uint32_t)0x00000010)
```

Bit 1

5.154.2.2413 RCC_CR_HSITRIM_2

```
#define RCC_CR_HSITRIM_2 ((uint32_t)0x00000020)
```

Bit 2

5.154.2.2414 RCC_CR_HSITRIM_3

```
#define RCC_CR_HSITRIM_3 ((uint32_t)0x00000040)
```

Bit 3

5.154.2.2415 RCC_CR_HSITRIM_4

```
#define RCC_CR_HSITRIM_4 ((uint32_t)0x00000080)
```

Bit 4

5.154.2.2416 SDIO_ARG_CMDARG

```
#define SDIO_ARG_CMDARG ((uint32_t)0xFFFFFFFF)
```

Command argument

5.154.2.2417 SDIO_CLKCR_BYPASS

```
#define SDIO_CLKCR_BYPASS ((uint16_t)0x0400)
```

Clock divider bypass enable bit

5.154.2.2418 SDIO_CLKCR_CLKDIV

```
#define SDIO_CLKCR_CLKDIV ((uint16_t)0x00FF)
```

Clock divide factor

5.154.2.2419 SDIO_CLKCR_CLKEN

```
#define SDIO_CLKCR_CLKEN ((uint16_t)0x0100)
```

Clock enable bit

5.154.2.2420 SDIO_CLKCR_HWFC_EN

```
#define SDIO_CLKCR_HWFC_EN ((uint16_t)0x4000)
```

HW Flow Control enable

5.154.2.2421 SDIO_CLKCR_NEGEDGE

```
#define SDIO_CLKCR_NEGEDGE ((uint16_t)0x2000)
```

SDIO_CK dephasing selection bit

5.154.2.2422 SDIO_CLKCR_PWRSAV

```
#define SDIO_CLKCR_PWRSAV ((uint16_t)0x0200)
```

Power saving configuration bit

5.154.2.2423 SDIO_CLKCR_WIDBUS

```
#define SDIO_CLKCR_WIDBUS ((uint16_t)0x1800)
```

WIDBUS[1:0] bits (Wide bus mode enable bit)

5.154.2.2424 SDIO_CLKCR_WIDBUS_0

```
#define SDIO_CLKCR_WIDBUS_0 ((uint16_t)0x0800)
```

Bit 0

5.154.2.2425 SDIO_CLKCR_WIDBUS_1

```
#define SDIO_CLKCR_WIDBUS_1 ((uint16_t)0x1000)
```

Bit 1

5.154.2.2426 SDIO_CMD_CEATACMD

```
#define SDIO_CMD_CEATACMD ((uint16_t)0x4000)
```

CE-ATA command

5.154.2.2427 SDIO_CMD_CMDINDEX

```
#define SDIO_CMD_CMDINDEX ((uint16_t)0x003F)
```

Command Index

5.154.2.2428 SDIO_CMD_CPSMEN

```
#define SDIO_CMD_CPSMEN ((uint16_t)0x0400)
```

Command path state machine (CPSM) Enable bit

5.154.2.2429 SDIO_CMD_ENCMDCOMPL

```
#define SDIO_CMD_ENCMDCOMPL ((uint16_t)0x1000)
```

Enable CMD completion

5.154.2.2430 SDIO_CMD_NIEN

```
#define SDIO_CMD_NIEN ((uint16_t)0x2000)
```

Not Interrupt Enable

5.154.2.2431 SDIO_CMD_SDIOSUSPEND

```
#define SDIO_CMD_SDIOSUSPEND ((uint16_t)0x0800)
```

SD I/O suspend command

5.154.2.2432 SDIO_CMD_WAITINT

```
#define SDIO_CMD_WAITINT ((uint16_t)0x0100)
```

CPSM Waits for Interrupt Request

5.154.2.2433 SDIO_CMD_WAITPEND

```
#define SDIO_CMD_WAITPEND ((uint16_t)0x0200)
```

CPSM Waits for ends of data transfer (CmdPend internal signal)

5.154.2.2434 SDIO_CMD_WAITRESP

```
#define SDIO_CMD_WAITRESP ((uint16_t)0x00C0)
```

WAITRESP[1:0] bits (Wait for response bits)

5.154.2.2435 SDIO_CMD_WAITRESP_0

```
#define SDIO_CMD_WAITRESP_0 ((uint16_t)0x0040)
```

Bit 0

5.154.2.2436 SDIO_CMD_WAITRESP_1

```
#define SDIO_CMD_WAITRESP_1 ((uint16_t)0x0080)
```

Bit 1

5.154.2.2437 SDIO_DCOUNT_DATACOUNT

```
#define SDIO_DCOUNT_DATACOUNT ((uint32_t)0x01FFFFFF)
```

Data count value

5.154.2.2438 SDIO_DCTRL_DBLOCKSIZE

```
#define SDIO_DCTRL_DBLOCKSIZE ((uint16_t)0x00F0)
```

DBLOCKSIZE[3:0] bits (Data block size)

5.154.2.2439 SDIO_DCTRL_DBLOCKSIZE_0

```
#define SDIO_DCTRL_DBLOCKSIZE_0 ((uint16_t)0x0010)
```

Bit 0

5.154.2.2440 SDIO_DCTRL_DBLOCKSIZE_1

```
#define SDIO_DCTRL_DBLOCKSIZE_1 ((uint16_t)0x0020)
```

Bit 1

5.154.2.2441 SDIO_DCTRL_DBLOCKSIZE_2

```
#define SDIO_DCTRL_DBLOCKSIZE_2 ((uint16_t)0x0040)
```

Bit 2

5.154.2.2442 SDIO_DCTRL_DBLOCKSIZE_3

```
#define SDIO_DCTRL_DBLOCKSIZE_3 ((uint16_t)0x0080)
```

Bit 3

5.154.2.2443 SDIO_DCTRL_DMAEN

```
#define SDIO_DCTRL_DMAEN ((uint16_t)0x0008)
```

DMA enabled bit

5.154.2.2444 SDIO_DCTRL_DTDIR

```
#define SDIO_DCTRL_DTDIR ((uint16_t)0x0002)
```

Data transfer direction selection

5.154.2.2445 SDIO_DCTRL_DTEN

```
#define SDIO_DCTRL_DTEN ((uint16_t)0x0001)
```

Data transfer enabled bit

5.154.2.2446 SDIO_DCTRL_DTMODE

```
#define SDIO_DCTRL_DTMODE ((uint16_t)0x0004)
```

Data transfer mode selection

5.154.2.2447 SDIO_DCTRL_RWMOD

```
#define SDIO_DCTRL_RWMOD ((uint16_t)0x0400)
```

Read wait mode

5.154.2.2448 SDIO_DCTRL_RWSTART

```
#define SDIO_DCTRL_RWSTART ((uint16_t)0x0100)
```

Read wait start

5.154.2.2449 SDIO_DCTRL_RWSTOP

```
#define SDIO_DCTRL_RWSTOP ((uint16_t)0x0200)
```

Read wait stop

5.154.2.2450 SDIO_DCTRL_SDIOEN

```
#define SDIO_DCTRL_SDIOEN ((uint16_t)0x0800)
```

SD I/O enable functions

5.154.2.2451 SDIO_DLEN_DATALENGTH

```
#define SDIO_DLEN_DATALENGTH ((uint32_t)0x01FFFFFF)
```

Data length value

5.154.2.2452 SDIO_DTIMER_DATATIME

```
#define SDIO_DTIMER_DATATIME ((uint32_t)0xFFFFFFFF)
```

Data timeout period.

5.154.2.2453 SDIO_FIFO_FIFODATA

```
#define SDIO_FIFO_FIFODATA ((uint32_t)0xFFFFFFFF)
```

Receive and transmit FIFO data

5.154.2.2454 SDIO_FIFOCNT_FIFOCOUNT

```
#define SDIO_FIFOCNT_FIFOCOUNT ((uint32_t)0x00FFFFFF)
```

Remaining number of words to be written to or read from the FIFO

5.154.2.2455 SDIO_ICR_CCRCFAILC

```
#define SDIO_ICR_CCRCFAILC ((uint32_t)0x00000001)
```

CCRCFAIL flag clear bit

5.154.2.2456 SDIO_ICR_CEATAENDC

```
#define SDIO_ICR_CEATAENDC ((uint32_t)0x00800000)
```

CEATAEND flag clear bit

5.154.2.2457 SDIO_ICR_CMDRENDC

```
#define SDIO_ICR_CMDRENDC ((uint32_t)0x00000040)
```

CMDREND flag clear bit

5.154.2.2458 SDIO_ICR_CMDSENTC

```
#define SDIO_ICR_CMDSENTC ((uint32_t)0x00000080)
```

CMDSENT flag clear bit

5.154.2.2459 SDIO_ICR_CTIMEOUTC

```
#define SDIO_ICR_CTIMEOUTC ((uint32_t)0x00000004)
```

CTIMEOUT flag clear bit

5.154.2.2460 SDIO_ICR_DATAENDC

```
#define SDIO_ICR_DATAENDC ((uint32_t)0x00000100)
```

DATAEND flag clear bit

5.154.2.2461 SDIO_ICR_DBCKENDC

```
#define SDIO_ICR_DBCKENDC ((uint32_t)0x00000400)
```

DBCKEND flag clear bit

5.154.2.2462 SDIO_ICR_DCRCFAILC

```
#define SDIO_ICR_DCRCFAILC ((uint32_t)0x00000002)
```

DCRCFAIL flag clear bit

5.154.2.2463 SDIO_ICR_DTIMEOUTC

```
#define SDIO_ICR_DTIMEOUTC ((uint32_t)0x00000008)
```

DTIMEOUT flag clear bit

5.154.2.2464 SDIO_ICR_RXOVERRC

```
#define SDIO_ICR_RXOVERRC ((uint32_t)0x00000020)
```

RXOVERR flag clear bit

5.154.2.2465 SDIO_ICR_SDIOITC

```
#define SDIO_ICR_SDIOITC ((uint32_t)0x00400000)
```

SDIOIT flag clear bit

5.154.2.2466 SDIO_ICR_STBITERRC

```
#define SDIO_ICR_STBITERRC ((uint32_t)0x00000200)
```

STBITERR flag clear bit

5.154.2.2467 SDIO_ICR_TXUNDERRC

```
#define SDIO_ICR_TXUNDERRC ((uint32_t)0x00000010)
```

TXUNDERR flag clear bit

5.154.2.2468 SDIO_MASK_CCRCFAILIE

```
#define SDIO_MASK_CCRCFAILIE ((uint32_t)0x00000001)
```

Command CRC Fail Interrupt Enable

5.154.2.2469 SDIO_MASK_CEATAENDIE

```
#define SDIO_MASK_CEATAENDIE ((uint32_t)0x00800000)
```

CE-ATA command completion signal received Interrupt Enable

5.154.2.2470 SDIO_MASK_CMDACTIE

```
#define SDIO_MASK_CMDACTIE ((uint32_t)0x00000800)
```

CCommand Acting Interrupt Enable

5.154.2.2471 SDIO_MASK_CMDRENDIE

```
#define SDIO_MASK_CMDRENDIE ((uint32_t)0x00000040)
```

Command Response Received Interrupt Enable

5.154.2.2472 SDIO_MASK_CMDSENTIE

```
#define SDIO_MASK_CMDSENTIE ((uint32_t)0x00000080)
```

Command Sent Interrupt Enable

5.154.2.2473 SDIO_MASK_CTIMEOUTIE

```
#define SDIO_MASK_CTIMEOUTIE ((uint32_t)0x00000004)
```

Command TimeOut Interrupt Enable

5.154.2.2474 SDIO_MASK_DATAENDIE

```
#define SDIO_MASK_DATAENDIE ((uint32_t)0x00000100)
```

Data End Interrupt Enable

5.154.2.2475 SDIO_MASK_DBCKENDIE

```
#define SDIO_MASK_DBCKENDIE ((uint32_t)0x00000400)
```

Data Block End Interrupt Enable

5.154.2.2476 SDIO_MASK_DCRCFAILIE

```
#define SDIO_MASK_DCRCFAILIE ((uint32_t)0x00000002)
```

Data CRC Fail Interrupt Enable

5.154.2.2477 SDIO_MASK_DTIMEOUTIE

```
#define SDIO_MASK_DTIMEOUTIE ((uint32_t)0x00000008)
```

Data TimeOut Interrupt Enable

5.154.2.2478 SDIO_MASK_RXACTIE

```
#define SDIO_MASK_RXACTIE ((uint32_t)0x00002000)
```

Data receive acting interrupt enabled

5.154.2.2479 SDIO_MASK_RXDAVLIE

```
#define SDIO_MASK_RXDAVLIE ((uint32_t)0x00200000)
```

Data available in Rx FIFO interrupt Enable

5.154.2.2480 SDIO_MASK_RXFIFOEIE

```
#define SDIO_MASK_RXFIFOEIE ((uint32_t)0x00080000)
```

Rx FIFO Empty interrupt Enable

5.154.2.2481 SDIO_MASK_RXFIFOFIE

```
#define SDIO_MASK_RXFIFOFIE ((uint32_t)0x00020000)
```

Rx FIFO Full interrupt Enable

5.154.2.2482 SDIO_MASK_RXFIFOHFIE

```
#define SDIO_MASK_RXFIFOHFIE ((uint32_t)0x00008000)
```

Rx FIFO Half Full interrupt Enable

5.154.2.2483 SDIO_MASK_RXOVERRIE

```
#define SDIO_MASK_RXOVERRIE ((uint32_t)0x00000020)
```

Rx FIFO OverRun Error Interrupt Enable

5.154.2.2484 SDIO_MASK_SDIOITIE

```
#define SDIO_MASK_SDIOITIE ((uint32_t)0x00400000)
```

SDIO Mode Interrupt Received interrupt Enable

5.154.2.2485 SDIO_MASK_STBITERRIE

```
#define SDIO_MASK_STBITERRIE ((uint32_t)0x00000200)
```

Start Bit Error Interrupt Enable

5.154.2.2486 SDIO_MASK_TXACTIE

```
#define SDIO_MASK_TXACTIE ((uint32_t)0x00001000)
```

Data Transmit Acting Interrupt Enable

5.154.2.2487 SDIO_MASK_TXDAVLIE

```
#define SDIO_MASK_TXDAVLIE ((uint32_t)0x00100000)
```

Data available in Tx FIFO interrupt Enable

5.154.2.2488 SDIO_MASK_TXFIFOEIE

```
#define SDIO_MASK_TXFIFOEIE ((uint32_t)0x00040000)
```

Tx FIFO Empty interrupt Enable

5.154.2.2489 SDIO_MASK_TXFIFOFIE

```
#define SDIO_MASK_TXFIFOFIE ((uint32_t)0x00010000)
```

Tx FIFO Full interrupt Enable

5.154.2.2490 SDIO_MASK_TXFIFOHEIE

```
#define SDIO_MASK_TXFIFOHEIE ((uint32_t)0x00004000)
```

Tx FIFO Half Empty interrupt Enable

5.154.2.2491 SDIO_MASK_TXUNDERRIE

```
#define SDIO_MASK_TXUNDERRIE ((uint32_t)0x00000010)
```

Tx FIFO UnderRun Error Interrupt Enable

5.154.2.2492 SDIO_POWER_PWRCTRL

```
#define SDIO_POWER_PWRCTRL ((uint8_t)0x03)
```

PWRCTRL[1:0] bits (Power supply control bits)

5.154.2.2493 SDIO_POWER_PWRCTRL_0

```
#define SDIO_POWER_PWRCTRL_0 ((uint8_t)0x01)
```

Bit 0

5.154.2.2494 SDIO_POWER_PWRCTRL_1

```
#define SDIO_POWER_PWRCTRL_1 ((uint8_t)0x02)
```

Bit 1

5.154.2.2495 SDIO_RESP0_CARDSTATUS0

```
#define SDIO_RESP0_CARDSTATUS0 ((uint32_t)0xFFFFFFFF)
```

Card Status

5.154.2.2496 SDIO_RESP1_CARDSTATUS1

```
#define SDIO_RESP1_CARDSTATUS1 ((uint32_t)0xFFFFFFFF)
```

Card Status

5.154.2.2497 SDIO_RESP2_CARDSTATUS2

```
#define SDIO_RESP2_CARDSTATUS2 ((uint32_t)0xFFFFFFFF)
```

Card Status

5.154.2.2498 SDIO_RESP3_CARDSTATUS3

```
#define SDIO_RESP3_CARDSTATUS3 ((uint32_t)0xFFFFFFFF)
```

Card Status

5.154.2.2499 SDIO_RESP4_CARDSTATUS4

```
#define SDIO_RESP4_CARDSTATUS4 ((uint32_t)0xFFFFFFFF)
```

Card Status

5.154.2.2500 SDIO_RESPCMD_RESPCMD

```
#define SDIO_RESPCMD_RESPCMD ((uint8_t)0x3F)
```

Response command index

5.154.2.2501 SDIO_STA_CCRCFAIL

```
#define SDIO_STA_CCRCFAIL ((uint32_t)0x00000001)
```

Command response received (CRC check failed)

5.154.2.2502 SDIO_STA_CEATAEND

```
#define SDIO_STA_CEATAEND ((uint32_t)0x00800000)
```

CE-ATA command completion signal received for CMD61

5.154.2.2503 SDIO_STA_CMDACT

```
#define SDIO_STA_CMDACT ((uint32_t)0x00000800)
```

Command transfer in progress

5.154.2.2504 SDIO_STA_CMDREND

```
#define SDIO_STA_CMDREND ((uint32_t)0x00000040)
```

Command response received (CRC check passed)

5.154.2.2505 SDIO_STA_CMDSENT

```
#define SDIO_STA_CMDSENT ((uint32_t)0x00000080)
```

Command sent (no response required)

5.154.2.2506 SDIO_STA_CTIMEOUT

```
#define SDIO_STA_CTIMEOUT ((uint32_t)0x00000004)
```

Command response timeout

5.154.2.2507 SDIO_STA_DATAEND

```
#define SDIO_STA_DATAEND ((uint32_t)0x00000100)
```

Data end (data counter, SDIDCOUNT, is zero)

5.154.2.2508 SDIO_STA_DBCKEND

```
#define SDIO_STA_DBCKEND ((uint32_t)0x00000400)
```

Data block sent/received (CRC check passed)

5.154.2.2509 SDIO_STA_DCRCFAIL

```
#define SDIO_STA_DCRCFAIL ((uint32_t)0x00000002)
```

Data block sent/received (CRC check failed)

5.154.2.2510 SDIO_STA_DTIMEOUT

```
#define SDIO_STA_DTIMEOUT ((uint32_t)0x00000008)
```

Data timeout

5.154.2.2511 SDIO_STA_RXACT

```
#define SDIO_STA_RXACT ((uint32_t)0x00002000)
```

Data receive in progress

5.154.2.2512 SDIO_STA_RXDAVL

```
#define SDIO_STA_RXDAVL ((uint32_t)0x00200000)
```

Data available in receive FIFO

5.154.2.2513 SDIO_STA_RXFIFOE

```
#define SDIO_STA_RXFIFOE ((uint32_t)0x00080000)
```

Receive FIFO empty

5.154.2.2514 SDIO_STA_RXFIFOF

```
#define SDIO_STA_RXFIFOF ((uint32_t)0x00020000)
```

Receive FIFO full

5.154.2.2515 SDIO_STA_RXFIFOHF

```
#define SDIO_STA_RXFIFOHF ((uint32_t)0x00008000)
```

Receive FIFO Half Full: there are at least 8 words in the FIFO

5.154.2.2516 SDIO_STA_RXOVERR

```
#define SDIO_STA_RXOVERR ((uint32_t)0x00000020)
```

Received FIFO overrun error

5.154.2.2517 SDIO_STA_SDIOIT

```
#define SDIO_STA_SDIOIT ((uint32_t)0x00400000)
```

SDIO interrupt received

5.154.2.2518 SDIO_STA_STBITERR

```
#define SDIO_STA_STBITERR ((uint32_t)0x00000200)
```

Start bit not detected on all data signals in wide bus mode

5.154.2.2519 SDIO_STA_TXACT

```
#define SDIO_STA_TXACT ((uint32_t)0x00001000)
```

Data transmit in progress

5.154.2.2520 SDIO_STA_TXDAVL

```
#define SDIO_STA_TXDAVL ((uint32_t)0x00100000)
```

Data available in transmit FIFO

5.154.2.2521 SDIO_STA_TXFIFOE

```
#define SDIO_STA_TXFIFOE ((uint32_t)0x00040000)
```

Transmit FIFO empty

5.154.2.2522 SDIO_STA_TXFIFOF

```
#define SDIO_STA_TXFIFOF ((uint32_t)0x00010000)
```

Transmit FIFO full

5.154.2.2523 SDIO_STA_TXFIFOHE

```
#define SDIO_STA_TXFIFOHE ((uint32_t)0x00004000)
```

Transmit FIFO Half Empty: at least 8 words can be written into the FIFO

5.154.2.2524 SDIO_STA_TXUNDERR

```
#define SDIO_STA_TXUNDERR ((uint32_t)0x00000010)
```

Transmit FIFO underrun error

5.154.2.2525 SPI_CR1_BIDIMODE

```
#define SPI_CR1_BIDIMODE ((uint16_t)0x8000)
```

Bidirectional data mode enable

5.154.2.2526 SPI_CR1_BIDIOE

```
#define SPI_CR1_BIDIOE ((uint16_t)0x4000)
```

Output enable in bidirectional mode

5.154.2.2527 SPI_CR1_BR

```
#define SPI_CR1_BR ((uint16_t)0x0038)
```

BR[2:0] bits (Baud Rate Control)

5.154.2.2528 SPI_CR1_BR_0

```
#define SPI_CR1_BR_0 ((uint16_t)0x0008)
```

Bit 0

5.154.2.2529 SPI_CR1_BR_1

```
#define SPI_CR1_BR_1 ((uint16_t)0x0010)
```

Bit 1

5.154.2.2530 SPI_CR1_BR_2

```
#define SPI_CR1_BR_2 ((uint16_t)0x0020)
```

Bit 2

5.154.2.2531 SPI_CR1_CPHA

```
#define SPI_CR1_CPHA ((uint16_t)0x0001)
```

Clock Phase

5.154.2.2532 SPI_CR1_CPOL

```
#define SPI_CR1_CPOL ((uint16_t)0x0002)
```

Clock Polarity

5.154.2.2533 SPI_CR1_CRCEN

```
#define SPI_CR1_CRCEN ((uint16_t)0x2000)
```

Hardware CRC calculation enable

5.154.2.2534 SPI_CR1_CRCNEXT

```
#define SPI_CR1_CRCNEXT ((uint16_t)0x1000)
```

Transmit CRC next

5.154.2.2535 SPI_CR1_DFF

```
#define SPI_CR1_DFF ((uint16_t)0x0800)
```

Data Frame Format

5.154.2.2536 SPI_CR1_LSBFIRST

```
#define SPI_CR1_LSBFIRST ((uint16_t)0x0080)
```

Frame Format

5.154.2.2537 SPI_CR1_MSTR

```
#define SPI_CR1_MSTR ((uint16_t)0x0004)
```

Master Selection

5.154.2.2538 SPI_CR1_RXONLY

```
#define SPI_CR1_RXONLY ((uint16_t)0x0400)
```

Receive only

5.154.2.2539 SPI_CR1_SPE

```
#define SPI_CR1_SPE ((uint16_t)0x0040)
```

SPI Enable

5.154.2.2540 SPI_CR1_SSI

```
#define SPI_CR1_SSI ((uint16_t)0x0100)
```

Internal slave select

5.154.2.2541 SPI_CR1_SSM

```
#define SPI_CR1_SSM ((uint16_t)0x0200)
```

Software slave management

5.154.2.2542 SPI_CR2_ERRIE

```
#define SPI_CR2_ERRIE ((uint8_t)0x20)
```

Error Interrupt Enable

5.154.2.2543 SPI_CR2_RXDMAEN

```
#define SPI_CR2_RXDMAEN ((uint8_t)0x01)
```

Rx Buffer DMA Enable

5.154.2.2544 SPI_CR2_RXNEIE

```
#define SPI_CR2_RXNEIE ((uint8_t)0x40)
```

RX buffer Not Empty Interrupt Enable

5.154.2.2545 SPI_CR2_SSOE

```
#define SPI_CR2_SSOE ((uint8_t)0x04)
```

SS Output Enable

5.154.2.2546 SPI_CR2_TXDMAEN

```
#define SPI_CR2_TXDMAEN ((uint8_t)0x02)
```

Tx Buffer DMA Enable

5.154.2.2547 SPI_CR2_TXEIE

```
#define SPI_CR2_TXEIE ((uint8_t)0x80)
```

Tx buffer Empty Interrupt Enable

5.154.2.2548 SPI_CRCPR_CRCPOLY

```
#define SPI_CRCPR_CRCPOLY ((uint16_t)0xFFFF)
```

CRC polynomial register

5.154.2.2549 SPI_DR_DR

```
#define SPI_DR_DR ((uint16_t)0xFFFF)
```

Data Register

5.154.2.2550 SPI_I2SCFGR_CHLEN

```
#define SPI_I2SCFGR_CHLEN ((uint16_t)0x0001)
```

Channel length (number of bits per audio channel)

5.154.2.2551 SPI_I2SCFGR_CKPOL

```
#define SPI_I2SCFGR_CKPOL ((uint16_t)0x0008)
```

steady state clock polarity

5.154.2.2552 SPI_I2SCFGR_DATLEN

```
#define SPI_I2SCFGR_DATLEN ((uint16_t)0x0006)
```

DATLEN[1:0] bits (Data length to be transferred)

5.154.2.2553 SPI_I2SCFGR_DATLEN_0

```
#define SPI_I2SCFGR_DATLEN_0 ((uint16_t)0x0002)
```

Bit 0

5.154.2.2554 SPI_I2SCFGR_DATLEN_1

```
#define SPI_I2SCFGR_DATLEN_1 ((uint16_t)0x0004)
```

Bit 1

5.154.2.2555 SPI_I2SCFGR_I2SCFG

```
#define SPI_I2SCFGR_I2SCFG ((uint16_t)0x0300)
```

I2SCFG[1:0] bits (I2S configuration mode)

5.154.2.2556 SPI_I2SCFGR_I2SCFG_0

```
#define SPI_I2SCFGR_I2SCFG_0 ((uint16_t)0x0100)
```

Bit 0

5.154.2.2557 SPI_I2SCFGR_I2SCFG_1

```
#define SPI_I2SCFGR_I2SCFG_1 ((uint16_t)0x0200)
```

Bit 1

5.154.2.2558 SPI_I2SCFGR_I2SE

```
#define SPI_I2SCFGR_I2SE ((uint16_t)0x0400)
```

I2S Enable

5.154.2.2559 SPI_I2SCFGR_I2SMOD

```
#define SPI_I2SCFGR_I2SMOD ((uint16_t)0x0800)
```

I2S mode selection

5.154.2.2560 SPI_I2SCFGR_I2SSTD

```
#define SPI_I2SCFGR_I2SSTD ((uint16_t)0x0030)
```

I2SSTD[1:0] bits (I2S standard selection)

5.154.2.2561 SPI_I2SCFGR_I2SSTD_0

```
#define SPI_I2SCFGR_I2SSTD_0 ((uint16_t)0x0010)
```

Bit 0

5.154.2.2562 SPI_I2SCFGR_I2SSTD_1

```
#define SPI_I2SCFGR_I2SSTD_1 ((uint16_t)0x0020)
```

Bit 1

5.154.2.2563 SPI_I2SCFGR_PCMSYNC

```
#define SPI_I2SCFGR_PCMSYNC ((uint16_t)0x0080)
```

PCM frame synchronization

5.154.2.2564 SPI_I2SPR_I2SDIV

```
#define SPI_I2SPR_I2SDIV ((uint16_t)0x00FF)
```

I2S Linear prescaler

5.154.2.2565 SPI_I2SPR_MCKOE

```
#define SPI_I2SPR_MCKOE ((uint16_t)0x0200)
```

Master Clock Output Enable

5.154.2.2566 SPI_I2SPR_ODD

```
#define SPI_I2SPR_ODD ((uint16_t)0x0100)
```

Odd factor for the prescaler

5.154.2.2567 SPI_RXCRCR_RXCRC

```
#define SPI_RXCRCR_RXCRC ((uint16_t)0xFFFF)
```

Rx CRC Register

5.154.2.2568 SPI_SR_BSY

```
#define SPI_SR_BSY ((uint8_t)0x80)
```

Busy flag

5.154.2.2569 SPI_SR_CHSIDE

```
#define SPI_SR_CHSIDE ((uint8_t)0x04)
```

Channel side

5.154.2.2570 SPI_SR_CRCERR

```
#define SPI_SR_CRCERR ((uint8_t)0x10)
```

CRC Error flag

5.154.2.2571 SPI_SR_MODF

```
#define SPI_SR_MODF ((uint8_t)0x20)
```

Mode fault

5.154.2.2572 SPI_SR_OVR

```
#define SPI_SR_OVR ((uint8_t)0x40)
```

Overrun flag

5.154.2.2573 SPI_SR_RXNE

```
#define SPI_SR_RXNE ((uint8_t)0x01)
```

Receive buffer Not Empty

5.154.2.2574 SPI_SR_TXE

```
#define SPI_SR_TXE ((uint8_t)0x02)
```

Transmit buffer Empty

5.154.2.2575 SPI_SR_UDR

```
#define SPI_SR_UDR ((uint8_t)0x08)
```

Underrun flag

5.154.2.2576 SPI_TXCRCR_TXCRC

```
#define SPI_TXCRCR_TXCRC ((uint16_t)0xFFFF)
```

Tx CRC Register

5.154.2.2577 SYSCFG_CMPCR_CMP_PD

```
#define SYSCFG_CMPCR_CMP_PD ((uint32_t)0x00000001)
```

Compensation cell ready flag

5.154.2.2578 SYSCFG_CMPCR_READY

```
#define SYSCFG_CMPCR_READY ((uint32_t)0x00000100)
```

Compensation cell power-down

5.154.2.2579 SYSCFG_EXTICR1_EXTI0

```
#define SYSCFG_EXTICR1_EXTI0 ((uint16_t)0x000F)
```

EXTI 0 configuration

5.154.2.2580 SYSCFG_EXTICR1_EXTI0_PA

```
#define SYSCFG_EXTICR1_EXTI0_PA ((uint16_t)0x0000)
```

EXTI0 configuration

PA[0] pin

5.154.2.2581 SYSCFG_EXTICR1_EXTI0_PB

```
#define SYSCFG_EXTICR1_EXTI0_PB ((uint16_t)0x0001)
```

PB[0] pin

5.154.2.2582 SYSCFG_EXTICR1_EXTI0_PC

```
#define SYSCFG_EXTICR1_EXTI0_PC ((uint16_t)0x0002)
```

PC[0] pin

5.154.2.2583 SYSCFG_EXTICR1_EXTI0_PD

```
#define SYSCFG_EXTICR1_EXTI0_PD ((uint16_t)0x0003)
```

PD[0] pin

5.154.2.2584 SYSCFG_EXTICR1_EXTI0_PE

```
#define SYSCFG_EXTICR1_EXTI0_PE ((uint16_t)0x0004)
```

PE[0] pin

5.154.2.2585 SYSCFG_EXTICR1_EXTI0_PF

```
#define SYSCFG_EXTICR1_EXTI0_PF ((uint16_t)0x0005)
```

PF[0] pin

5.154.2.2586 SYSCFG_EXTICR1_EXTI0_PG

```
#define SYSCFG_EXTICR1_EXTI0_PG ((uint16_t)0x0006)
```

PG[0] pin

5.154.2.2587 SYSCFG_EXTICR1_EXTI0_PH

```
#define SYSCFG_EXTICR1_EXTI0_PH ((uint16_t)0x0007)
```

PH[0] pin

5.154.2.2588 SYSCFG_EXTICR1_EXTI0_PI

```
#define SYSCFG_EXTICR1_EXTI0_PI ((uint16_t)0x0008)
```

PI[0] pin

5.154.2.2589 SYSCFG_EXTICR1_EXTI1

```
#define SYSCFG_EXTICR1_EXTI1 ((uint16_t)0x00F0)
```

EXTI 1 configuration

5.154.2.2590 SYSCFG_EXTICR1_EXTI1_PA

```
#define SYSCFG_EXTICR1_EXTI1_PA ((uint16_t)0x0000)
```

EXTI1 configuration

PA[1] pin

5.154.2.2591 SYSCFG_EXTICR1_EXTI1_PB

```
#define SYSCFG_EXTICR1_EXTI1_PB ((uint16_t)0x0010)
```

PB[1] pin

5.154.2.2592 SYSCFG_EXTICR1_EXTI1_PC

```
#define SYSCFG_EXTICR1_EXTI1_PC ((uint16_t)0x0020)
```

PC[1] pin

5.154.2.2593 SYSCFG_EXTICR1_EXTI1_PD

```
#define SYSCFG_EXTICR1_EXTI1_PD ((uint16_t)0x0030)
```

PD[1] pin

5.154.2.2594 SYSCFG_EXTICR1_EXTI1_PE

```
#define SYSCFG_EXTICR1_EXTI1_PE ((uint16_t)0x0040)
```

PE[1] pin

5.154.2.2595 SYSCFG_EXTICR1_EXTI1_PF

```
#define SYSCFG_EXTICR1_EXTI1_PF ((uint16_t)0x0050)
```

PF[1] pin

5.154.2.2596 SYSCFG_EXTICR1_EXTI1_PG

```
#define SYSCFG_EXTICR1_EXTI1_PG ((uint16_t)0x0060)
```

PG[1] pin

5.154.2.2597 SYSCFG_EXTICR1_EXTI1_PH

```
#define SYSCFG_EXTICR1_EXTI1_PH ((uint16_t)0x0070)
```

PH[1] pin

5.154.2.2598 SYSCFG_EXTICR1_EXTI1_PI

```
#define SYSCFG_EXTICR1_EXTI1_PI ((uint16_t)0x0080)
```

PI[1] pin

5.154.2.2599 SYSCFG_EXTICR1_EXTI2

```
#define SYSCFG_EXTICR1_EXTI2 ((uint16_t)0x0F00)
```

EXTI 2 configuration

5.154.2.2600 SYSCFG_EXTICR1_EXTI2_PA

```
#define SYSCFG_EXTICR1_EXTI2_PA ((uint16_t)0x0000)
```

EXTI2 configuration

PA[2] pin

5.154.2.2601 SYSCFG_EXTICR1_EXTI2_PB

```
#define SYSCFG_EXTICR1_EXTI2_PB ((uint16_t)0x0100)
```

PB[2] pin

5.154.2.2602 SYSCFG_EXTICR1_EXTI2_PC

```
#define SYSCFG_EXTICR1_EXTI2_PC ((uint16_t)0x0200)
```

PC[2] pin

5.154.2.2603 SYSCFG_EXTICR1_EXTI2_PD

```
#define SYSCFG_EXTICR1_EXTI2_PD ((uint16_t)0x0300)
```

PD[2] pin

5.154.2.2604 SYSCFG_EXTICR1_EXTI2_PE

```
#define SYSCFG_EXTICR1_EXTI2_PE ((uint16_t)0x0400)
```

PE[2] pin

5.154.2.2605 SYSCFG_EXTICR1_EXTI2_PF

```
#define SYSCFG_EXTICR1_EXTI2_PF ((uint16_t)0x0500)
```

PF[2] pin

5.154.2.2606 SYSCFG_EXTICR1_EXTI2_PG

```
#define SYSCFG_EXTICR1_EXTI2_PG ((uint16_t)0x0600)
```

PG[2] pin

5.154.2.2607 SYSCFG_EXTICR1_EXTI2_PH

```
#define SYSCFG_EXTICR1_EXTI2_PH ((uint16_t)0x0700)
```

PH[2] pin

5.154.2.2608 SYSCFG_EXTICR1_EXTI2_PI

```
#define SYSCFG_EXTICR1_EXTI2_PI ((uint16_t)0x0800)
```

PI[2] pin

5.154.2.2609 SYSCFG_EXTICR1_EXTI3

```
#define SYSCFG_EXTICR1_EXTI3 ((uint16_t)0xF000)
```

EXTI 3 configuration

5.154.2.2610 SYSCFG_EXTICR1_EXTI3_PA

```
#define SYSCFG_EXTICR1_EXTI3_PA ((uint16_t)0x0000)
```

EXTI3 configuration

PA[3] pin

5.154.2.2611 SYSCFG_EXTICR1_EXTI3_PB

```
#define SYSCFG_EXTICR1_EXTI3_PB ((uint16_t)0x1000)
```

PB[3] pin

5.154.2.2612 SYSCFG_EXTICR1_EXTI3_PC

```
#define SYSCFG_EXTICR1_EXTI3_PC ((uint16_t)0x2000)
```

PC[3] pin

5.154.2.2613 SYSCFG_EXTICR1_EXTI3_PD

```
#define SYSCFG_EXTICR1_EXTI3_PD ((uint16_t)0x3000)
```

PD[3] pin

5.154.2.2614 SYSCFG_EXTICR1_EXTI3_PE

```
#define SYSCFG_EXTICR1_EXTI3_PE ((uint16_t)0x4000)
```

PE[3] pin

5.154.2.2615 SYSCFG_EXTICR1_EXTI3_PF

```
#define SYSCFG_EXTICR1_EXTI3_PF ((uint16_t)0x5000)
```

PF[3] pin

5.154.2.2616 SYSCFG_EXTICR1_EXTI3_PG

```
#define SYSCFG_EXTICR1_EXTI3_PG ((uint16_t)0x6000)
```

PG[3] pin

5.154.2.2617 SYSCFG_EXTICR1_EXTI3_PH

```
#define SYSCFG_EXTICR1_EXTI3_PH ((uint16_t)0x7000)
```

PH[3] pin

5.154.2.2618 SYSCFG_EXTICR1_EXTI3_PI

```
#define SYSCFG_EXTICR1_EXTI3_PI ((uint16_t)0x8000)
```

PI[3] pin

5.154.2.2619 SYSCFG_EXTICR2_EXTI4

```
#define SYSCFG_EXTICR2_EXTI4 ((uint16_t)0x000F)
```

EXTI 4 configuration

5.154.2.2620 SYSCFG_EXTICR2_EXTI4_PA

```
#define SYSCFG_EXTICR2_EXTI4_PA ((uint16_t)0x0000)
```

EXTI4 configuration

PA[4] pin

5.154.2.2621 SYSCFG_EXTICR2_EXTI4_PB

```
#define SYSCFG_EXTICR2_EXTI4_PB ((uint16_t)0x0001)
```

PB[4] pin

5.154.2.2622 SYSCFG_EXTICR2_EXTI4_PC

```
#define SYSCFG_EXTICR2_EXTI4_PC ((uint16_t)0x0002)
```

PC[4] pin

5.154.2.2623 SYSCFG_EXTICR2_EXTI4_PD

```
#define SYSCFG_EXTICR2_EXTI4_PD ((uint16_t)0x0003)
```

PD[4] pin

5.154.2.2624 SYSCFG_EXTICR2_EXTI4_PE

```
#define SYSCFG_EXTICR2_EXTI4_PE ((uint16_t)0x0004)
```

PE[4] pin

5.154.2.2625 SYSCFG_EXTICR2_EXTI4_PF

```
#define SYSCFG_EXTICR2_EXTI4_PF ((uint16_t)0x0005)
```

PF[4] pin

5.154.2.2626 SYSCFG_EXTICR2_EXTI4_PG

```
#define SYSCFG_EXTICR2_EXTI4_PG ((uint16_t)0x0006)
```

PG[4] pin

5.154.2.2627 SYSCFG_EXTICR2_EXTI4_PH

```
#define SYSCFG_EXTICR2_EXTI4_PH ((uint16_t)0x0007)
```

PH[4] pin

5.154.2.2628 SYSCFG_EXTICR2_EXTI4_PI

```
#define SYSCFG_EXTICR2_EXTI4_PI ((uint16_t)0x0008)
```

PI[4] pin

5.154.2.2629 SYSCFG_EXTICR2_EXTI5

```
#define SYSCFG_EXTICR2_EXTI5 ((uint16_t)0x00F0)
```

EXTI 5 configuration

5.154.2.2630 SYSCFG_EXTICR2_EXTI5_PA

```
#define SYSCFG_EXTICR2_EXTI5_PA ((uint16_t)0x0000)
```

EXTI5 configuration

PA[5] pin

5.154.2.2631 SYSCFG_EXTICR2_EXTI5_PB

```
#define SYSCFG_EXTICR2_EXTI5_PB ((uint16_t)0x0010)
```

PB[5] pin

5.154.2.2632 SYSCFG_EXTICR2_EXTI5_PC

```
#define SYSCFG_EXTICR2_EXTI5_PC ((uint16_t)0x0020)
```

PC[5] pin

5.154.2.2633 SYSCFG_EXTICR2_EXTI5_PD

```
#define SYSCFG_EXTICR2_EXTI5_PD ((uint16_t)0x0030)
```

PD[5] pin

5.154.2.2634 SYSCFG_EXTICR2_EXTI5_PE

```
#define SYSCFG_EXTICR2_EXTI5_PE ((uint16_t)0x0040)
```

PE[5] pin

5.154.2.2635 SYSCFG_EXTICR2_EXTI5_PF

```
#define SYSCFG_EXTICR2_EXTI5_PF ((uint16_t)0x0050)
```

PF[5] pin

5.154.2.2636 SYSCFG_EXTICR2_EXTI5_PG

```
#define SYSCFG_EXTICR2_EXTI5_PG ((uint16_t)0x0060)
```

PG[5] pin

5.154.2.2637 SYSCFG_EXTICR2_EXTI5_PH

```
#define SYSCFG_EXTICR2_EXTI5_PH ((uint16_t)0x0070)
```

PH[5] pin

5.154.2.2638 SYSCFG_EXTICR2_EXTI5_PI

```
#define SYSCFG_EXTICR2_EXTI5_PI ((uint16_t)0x0080)
```

PI[5] pin

5.154.2.2639 SYSCFG_EXTICR2_EXTI6

```
#define SYSCFG_EXTICR2_EXTI6 ((uint16_t)0x0F00)
```

EXTI 6 configuration

5.154.2.2640 SYSCFG_EXTICR2_EXTI6_PA

```
#define SYSCFG_EXTICR2_EXTI6_PA ((uint16_t)0x0000)
```

EXTI6 configuration

PA[6] pin

5.154.2.2641 SYSCFG_EXTICR2_EXTI6_PB

```
#define SYSCFG_EXTICR2_EXTI6_PB ((uint16_t)0x0100)
```

PB[6] pin

5.154.2.2642 SYSCFG_EXTICR2_EXTI6_PC

```
#define SYSCFG_EXTICR2_EXTI6_PC ((uint16_t)0x0200)
```

PC[6] pin

5.154.2.2643 SYSCFG_EXTICR2_EXTI6_PD

```
#define SYSCFG_EXTICR2_EXTI6_PD ((uint16_t)0x0300)
```

PD[6] pin

5.154.2.2644 SYSCFG_EXTICR2_EXTI6_PE

```
#define SYSCFG_EXTICR2_EXTI6_PE ((uint16_t)0x0400)
```

PE[6] pin

5.154.2.2645 SYSCFG_EXTICR2_EXTI6_PF

```
#define SYSCFG_EXTICR2_EXTI6_PF ((uint16_t)0x0500)
```

PF[6] pin

5.154.2.2646 SYSCFG_EXTICR2_EXTI6_PG

```
#define SYSCFG_EXTICR2_EXTI6_PG ((uint16_t)0x0600)
```

PG[6] pin

5.154.2.2647 SYSCFG_EXTICR2_EXTI6_PH

```
#define SYSCFG_EXTICR2_EXTI6_PH ((uint16_t)0x0700)
```

PH[6] pin

5.154.2.2648 SYSCFG_EXTICR2_EXTI6_PI

```
#define SYSCFG_EXTICR2_EXTI6_PI ((uint16_t)0x0800)
```

PI[6] pin

5.154.2.2649 SYSCFG_EXTICR2_EXTI7

```
#define SYSCFG_EXTICR2_EXTI7 ((uint16_t)0xF000)
```

EXTI 7 configuration

5.154.2.2650 SYSCFG_EXTICR2_EXTI7_PA

```
#define SYSCFG_EXTICR2_EXTI7_PA ((uint16_t)0x0000)
```

EXTI7 configuration

PA[7] pin

5.154.2.2651 SYSCFG_EXTICR2_EXTI7_PB

```
#define SYSCFG_EXTICR2_EXTI7_PB ((uint16_t)0x1000)
```

PB[7] pin

5.154.2.2652 SYSCFG_EXTICR2_EXTI7_PC

```
#define SYSCFG_EXTICR2_EXTI7_PC ((uint16_t)0x2000)
```

PC[7] pin

5.154.2.2653 SYSCFG_EXTICR2_EXTI7_PD

```
#define SYSCFG_EXTICR2_EXTI7_PD ((uint16_t)0x3000)
```

PD[7] pin

5.154.2.2654 SYSCFG_EXTICR2_EXTI7_PE

```
#define SYSCFG_EXTICR2_EXTI7_PE ((uint16_t)0x4000)
```

PE[7] pin

5.154.2.2655 SYSCFG_EXTICR2_EXTI7_PF

```
#define SYSCFG_EXTICR2_EXTI7_PF ((uint16_t)0x5000)
```

PF[7] pin

5.154.2.2656 SYSCFG_EXTICR2_EXTI7_PG

```
#define SYSCFG_EXTICR2_EXTI7_PG ((uint16_t)0x6000)
```

PG[7] pin

5.154.2.2657 SYSCFG_EXTICR2_EXTI7_PH

```
#define SYSCFG_EXTICR2_EXTI7_PH ((uint16_t)0x7000)
```

PH[7] pin

5.154.2.2658 SYSCFG_EXTICR2_EXTI7_PI

```
#define SYSCFG_EXTICR2_EXTI7_PI ((uint16_t)0x8000)
```

PI[7] pin

5.154.2.2659 SYSCFG_EXTICR3_EXTI10

```
#define SYSCFG_EXTICR3_EXTI10 ((uint16_t)0x0F00)
```

EXTI 10 configuration

5.154.2.2660 SYSCFG_EXTICR3_EXTI10_PA

```
#define SYSCFG_EXTICR3_EXTI10_PA ((uint16_t)0x0000)
```

EXTI10 configuration

PA[10] pin

5.154.2.2661 SYSCFG_EXTICR3_EXTI10_PB

```
#define SYSCFG_EXTICR3_EXTI10_PB ((uint16_t)0x0100)
```

PB[10] pin

5.154.2.2662 SYSCFG_EXTICR3_EXTI10_PC

```
#define SYSCFG_EXTICR3_EXTI10_PC ((uint16_t)0x0200)
```

PC[10] pin

5.154.2.2663 SYSCFG_EXTICR3_EXTI10_PD

```
#define SYSCFG_EXTICR3_EXTI10_PD ((uint16_t)0x0300)
```

PD[10] pin

5.154.2.2664 SYSCFG_EXTICR3_EXTI10_PE

```
#define SYSCFG_EXTICR3_EXTI10_PE ((uint16_t)0x0400)
```

PE[10] pin

5.154.2.2665 SYSCFG_EXTICR3_EXTI10_PF

```
#define SYSCFG_EXTICR3_EXTI10_PF ((uint16_t)0x0500)
```

PF[10] pin

5.154.2.2666 SYSCFG_EXTICR3_EXTI10_PG

```
#define SYSCFG_EXTICR3_EXTI10_PG ((uint16_t)0x0600)
```

PG[10] pin

5.154.2.2667 SYSCFG_EXTICR3_EXTI10_PH

```
#define SYSCFG_EXTICR3_EXTI10_PH ((uint16_t)0x0700)
```

PH[10] pin

5.154.2.2668 SYSCFG_EXTICR3_EXTI10_PI

```
#define SYSCFG_EXTICR3_EXTI10_PI ((uint16_t)0x0800)
```

PI[10] pin

5.154.2.2669 SYSCFG_EXTICR3_EXTI11

```
#define SYSCFG_EXTICR3_EXTI11 ((uint16_t)0xF000)
```

EXTI 11 configuration

5.154.2.2670 SYSCFG_EXTICR3_EXTI11_PA

```
#define SYSCFG_EXTICR3_EXTI11_PA ((uint16_t)0x0000)
```

EXTI11 configuration

PA[11] pin

5.154.2.2671 SYSCFG_EXTICR3_EXTI11_PB

```
#define SYSCFG_EXTICR3_EXTI11_PB ((uint16_t)0x1000)
```

PB[11] pin

5.154.2.2672 SYSCFG_EXTICR3_EXTI11_PC

```
#define SYSCFG_EXTICR3_EXTI11_PC ((uint16_t)0x2000)
```

PC[11] pin

5.154.2.2673 SYSCFG_EXTICR3_EXTI11_PD

```
#define SYSCFG_EXTICR3_EXTI11_PD ((uint16_t)0x3000)
```

PD[11] pin

5.154.2.2674 SYSCFG_EXTICR3_EXTI11_PE

```
#define SYSCFG_EXTICR3_EXTI11_PE ((uint16_t)0x4000)
```

PE[11] pin

5.154.2.2675 SYSCFG_EXTICR3_EXTI11_PF

```
#define SYSCFG_EXTICR3_EXTI11_PF ((uint16_t)0x5000)
```

PF[11] pin

5.154.2.2676 SYSCFG_EXTICR3_EXTI11_PG

```
#define SYSCFG_EXTICR3_EXTI11_PG ((uint16_t)0x6000)
```

PG[11] pin

5.154.2.2677 SYSCFG_EXTICR3_EXTI11_PH

```
#define SYSCFG_EXTICR3_EXTI11_PH ((uint16_t)0x7000)
```

PH[11] pin

5.154.2.2678 SYSCFG_EXTICR3_EXTI11_PI

```
#define SYSCFG_EXTICR3_EXTI11_PI ((uint16_t)0x8000)
```

PI[11] pin

5.154.2.2679 SYSCFG_EXTICR3_EXTI12_PH

```
#define SYSCFG_EXTICR3_EXTI12_PH ((uint16_t)0x0007)
```

PH[12] pin

5.154.2.2680 SYSCFG_EXTICR3_EXTI13_PH

```
#define SYSCFG_EXTICR3_EXTI13_PH ((uint16_t)0x0070)
```

PH[13] pin

5.154.2.2681 SYSCFG_EXTICR3_EXTI14_PH

```
#define SYSCFG_EXTICR3_EXTI14_PH ((uint16_t)0x0700)
```

PH[14] pin

5.154.2.2682 SYSCFG_EXTICR3_EXTI15_PH

```
#define SYSCFG_EXTICR3_EXTI15_PH ((uint16_t)0x7000)
```

PH[15] pin

5.154.2.2683 SYSCFG_EXTICR3_EXTI8

```
#define SYSCFG_EXTICR3_EXTI8 ((uint16_t)0x000F)
```

EXTI 8 configuration

5.154.2.2684 SYSCFG_EXTICR3_EXTI8_PA

```
#define SYSCFG_EXTICR3_EXTI8_PA ((uint16_t)0x0000)
```

EXTI8 configuration

PA[8] pin

5.154.2.2685 SYSCFG_EXTICR3_EXTI8_PB

```
#define SYSCFG_EXTICR3_EXTI8_PB ((uint16_t)0x0001)
```

PB[8] pin

5.154.2.2686 SYSCFG_EXTICR3_EXTI8_PC

```
#define SYSCFG_EXTICR3_EXTI8_PC ((uint16_t)0x0002)
```

PC[8] pin

5.154.2.2687 SYSCFG_EXTICR3_EXTI8_PD

```
#define SYSCFG_EXTICR3_EXTI8_PD ((uint16_t)0x0003)
```

PD[8] pin

5.154.2.2688 SYSCFG_EXTICR3_EXTI8_PE

```
#define SYSCFG_EXTICR3_EXTI8_PE ((uint16_t)0x0004)
```

PE[8] pin

5.154.2.2689 SYSCFG_EXTICR3_EXTI8_PF

```
#define SYSCFG_EXTICR3_EXTI8_PF ((uint16_t)0x0005)
```

PF[8] pin

5.154.2.2690 SYSCFG_EXTICR3_EXTI8_PG

```
#define SYSCFG_EXTICR3_EXTI8_PG ((uint16_t)0x0006)
```

PG[8] pin

5.154.2.2691 SYSCFG_EXTICR3_EXTI8_PH

```
#define SYSCFG_EXTICR3_EXTI8_PH ((uint16_t)0x0007)
```

PH[8] pin

5.154.2.2692 SYSCFG_EXTICR3_EXTI8_PI

```
#define SYSCFG_EXTICR3_EXTI8_PI ((uint16_t)0x0008)
```

PI[8] pin

5.154.2.2693 SYSCFG_EXTICR3_EXTI9

```
#define SYSCFG_EXTICR3_EXTI9 ((uint16_t)0x00F0)
```

EXTI 9 configuration

5.154.2.2694 SYSCFG_EXTICR3_EXTI9_PA

```
#define SYSCFG_EXTICR3_EXTI9_PA ((uint16_t)0x0000)
```

EXTI9 configuration

PA[9] pin

5.154.2.2695 SYSCFG_EXTICR3_EXTI9_PB

```
#define SYSCFG_EXTICR3_EXTI9_PB ((uint16_t)0x0010)
```

PB[9] pin

5.154.2.2696 SYSCFG_EXTICR3_EXTI9_PC

```
#define SYSCFG_EXTICR3_EXTI9_PC ((uint16_t)0x0020)
```

PC[9] pin

5.154.2.2697 SYSCFG_EXTICR3_EXTI9_PD

```
#define SYSCFG_EXTICR3_EXTI9_PD ((uint16_t)0x0030)
```

PD[9] pin

5.154.2.2698 SYSCFG_EXTICR3_EXTI9_PE

```
#define SYSCFG_EXTICR3_EXTI9_PE ((uint16_t)0x0040)
```

PE[9] pin

5.154.2.2699 SYSCFG_EXTICR3_EXTI9_PF

```
#define SYSCFG_EXTICR3_EXTI9_PF ((uint16_t)0x0050)
```

PF[9] pin

5.154.2.2700 SYSCFG_EXTICR3_EXTI9_PG

```
#define SYSCFG_EXTICR3_EXTI9_PG ((uint16_t)0x0060)
```

PG[9] pin

5.154.2.2701 SYSCFG_EXTICR3_EXTI9_PH

```
#define SYSCFG_EXTICR3_EXTI9_PH ((uint16_t)0x0070)
```

PH[9] pin

5.154.2.2702 SYSCFG_EXTICR3_EXTI9_PI

```
#define SYSCFG_EXTICR3_EXTI9_PI ((uint16_t)0x0080)
```

PI[9] pin

5.154.2.2703 SYSCFG_EXTICR4_EXTI12

```
#define SYSCFG_EXTICR4_EXTI12 ((uint16_t)0x000F)
```

EXTI 12 configuration

5.154.2.2704 SYSCFG_EXTICR4_EXTI12_PA

```
#define SYSCFG_EXTICR4_EXTI12_PA ((uint16_t)0x0000)
```

EXTI12 configuration

PA[12] pin

5.154.2.2705 SYSCFG_EXTICR4_EXTI12_PB

```
#define SYSCFG_EXTICR4_EXTI12_PB ((uint16_t)0x0001)
```

PB[12] pin

5.154.2.2706 SYSCFG_EXTICR4_EXTI12_PC

```
#define SYSCFG_EXTICR4_EXTI12_PC ((uint16_t)0x0002)
```

PC[12] pin

5.154.2.2707 SYSCFG_EXTICR4_EXTI12_PD

```
#define SYSCFG_EXTICR4_EXTI12_PD ((uint16_t)0x0003)
```

PD[12] pin

5.154.2.2708 SYSCFG_EXTICR4_EXTI12_PE

```
#define SYSCFG_EXTICR4_EXTI12_PE ((uint16_t)0x0004)
```

PE[12] pin

5.154.2.2709 SYSCFG_EXTICR4_EXTI12_PF

```
#define SYSCFG_EXTICR4_EXTI12_PF ((uint16_t)0x0005)
```

PF[12] pin

5.154.2.2710 SYSCFG_EXTICR4_EXTI12_PG

```
#define SYSCFG_EXTICR4_EXTI12_PG ((uint16_t)0x0006)
```

PG[12] pin

5.154.2.2711 SYSCFG_EXTICR4_EXTI13

```
#define SYSCFG_EXTICR4_EXTI13 ((uint16_t)0x00F0)
```

EXTI 13 configuration

5.154.2.2712 SYSCFG_EXTICR4_EXTI13_PA

```
#define SYSCFG_EXTICR4_EXTI13_PA ((uint16_t)0x0000)
```

EXTI13 configuration

PA[13] pin

5.154.2.2713 SYSCFG_EXTICR4_EXTI13_PB

```
#define SYSCFG_EXTICR4_EXTI13_PB ((uint16_t)0x0010)
```

PB[13] pin

5.154.2.2714 SYSCFG_EXTICR4_EXTI13_PC

```
#define SYSCFG_EXTICR4_EXTI13_PC ((uint16_t)0x0020)
```

PC[13] pin

5.154.2.2715 SYSCFG_EXTICR4_EXTI13_PD

```
#define SYSCFG_EXTICR4_EXTI13_PD ((uint16_t)0x0030)
```

PD[13] pin

5.154.2.2716 SYSCFG_EXTICR4_EXTI13_PE

```
#define SYSCFG_EXTICR4_EXTI13_PE ((uint16_t)0x0040)
```

PE[13] pin

5.154.2.2717 SYSCFG_EXTICR4_EXTI13_PF

```
#define SYSCFG_EXTICR4_EXTI13_PF ((uint16_t)0x0050)
```

PF[13] pin

5.154.2.2718 SYSCFG_EXTICR4_EXTI13_PG

```
#define SYSCFG_EXTICR4_EXTI13_PG ((uint16_t)0x0060)
```

PG[13] pin

5.154.2.2719 SYSCFG_EXTICR4_EXTI14

```
#define SYSCFG_EXTICR4_EXTI14 ((uint16_t)0x0F00)
```

EXTI 14 configuration

5.154.2.2720 SYSCFG_EXTICR4_EXTI14_PA

```
#define SYSCFG_EXTICR4_EXTI14_PA ((uint16_t)0x0000)
```

EXTI14 configuration

PA[14] pin

5.154.2.2721 SYSCFG_EXTICR4_EXTI14_PB

```
#define SYSCFG_EXTICR4_EXTI14_PB ((uint16_t)0x0100)
```

PB[14] pin

5.154.2.2722 SYSCFG_EXTICR4_EXTI14_PC

```
#define SYSCFG_EXTICR4_EXTI14_PC ((uint16_t)0x0200)
```

PC[14] pin

5.154.2.2723 SYSCFG_EXTICR4_EXTI14_PD

```
#define SYSCFG_EXTICR4_EXTI14_PD ((uint16_t)0x0300)
```

PD[14] pin

5.154.2.2724 SYSCFG_EXTICR4_EXTI14_PE

```
#define SYSCFG_EXTICR4_EXTI14_PE ((uint16_t)0x0400)
```

PE[14] pin

5.154.2.2725 SYSCFG_EXTICR4_EXTI14_PF

```
#define SYSCFG_EXTICR4_EXTI14_PF ((uint16_t)0x0500)
```

PF[14] pin

5.154.2.2726 SYSCFG_EXTICR4_EXTI14_PG

```
#define SYSCFG_EXTICR4_EXTI14_PG ((uint16_t)0x0600)
```

PG[14] pin

5.154.2.2727 SYSCFG_EXTICR4_EXTI15

```
#define SYSCFG_EXTICR4_EXTI15 ((uint16_t)0xF000)
```

EXTI 15 configuration

5.154.2.2728 SYSCFG_EXTICR4_EXTI15_PA

```
#define SYSCFG_EXTICR4_EXTI15_PA ((uint16_t)0x0000)
```

EXTI15 configuration

PA[15] pin

5.154.2.2729 SYSCFG_EXTICR4_EXTI15_PB

```
#define SYSCFG_EXTICR4_EXTI15_PB ((uint16_t)0x1000)
```

PB[15] pin

5.154.2.2730 SYSCFG_EXTICR4_EXTI15_PC

```
#define SYSCFG_EXTICR4_EXTI15_PC ((uint16_t)0x2000)
```

PC[15] pin

5.154.2.2731 SYSCFG_EXTICR4_EXTI15_PD

```
#define SYSCFG_EXTICR4_EXTI15_PD ((uint16_t)0x3000)
```

PD[15] pin

5.154.2.2732 SYSCFG_EXTICR4_EXTI15_PE

```
#define SYSCFG_EXTICR4_EXTI15_PE ((uint16_t)0x4000)
```

PE[15] pin

5.154.2.2733 SYSCFG_EXTICR4_EXTI15_PF

```
#define SYSCFG_EXTICR4_EXTI15_PF ((uint16_t)0x5000)
```

PF[15] pin

5.154.2.2734 SYSCFG_EXTICR4_EXTI15_PG

```
#define SYSCFG_EXTICR4_EXTI15_PG ((uint16_t)0x6000)
```

PG[15] pin

5.154.2.2735 SYSCFG_MEMRMP_MEM_MODE

```
#define SYSCFG_MEMRMP_MEM_MODE ((uint32_t)0x00000003)
```

SYSCFG_Memory Remap Config

5.154.2.2736 SYSCFG_PMC_MII_RMII_SEL

```
#define SYSCFG_PMC_MII_RMII_SEL ((uint32_t)0x00800000)
```

Ethernet PHY interface selection

5.154.2.2737 TIM_ARR_ARR

```
#define TIM_ARR_ARR ((uint16_t)0xFFFF)
```

actual auto-reload Value

5.154.2.2738 TIM_BDTR_AOE

```
#define TIM_BDTR_AOE ((uint16_t)0x4000)
```

Automatic Output enable

5.154.2.2739 TIM_BDTR_BKE

```
#define TIM_BDTR_BKE ((uint16_t)0x1000)
```

Break enable

5.154.2.2740 TIM_BDTR_BKP

```
#define TIM_BDTR_BKP ((uint16_t)0x2000)
```

Break Polarity

5.154.2.2741 TIM_BDTR_DTG

```
#define TIM_BDTR_DTG ((uint16_t)0x00FF)
```

DTG[0:7] bits (Dead-Time Generator set-up)

5.154.2.2742 TIM_BDTR_DTG_0

```
#define TIM_BDTR_DTG_0 ((uint16_t)0x0001)
```

Bit 0

5.154.2.2743 TIM_BDTR_DTG_1

```
#define TIM_BDTR_DTG_1 ((uint16_t)0x0002)
```

Bit 1

5.154.2.2744 TIM_BDTR_DTG_2

```
#define TIM_BDTR_DTG_2 ((uint16_t)0x0004)
```

Bit 2

5.154.2.2745 TIM_BDTR_DTG_3

```
#define TIM_BDTR_DTG_3 ((uint16_t)0x0008)
```

Bit 3

5.154.2.2746 TIM_BDTR_DTG_4

```
#define TIM_BDTR_DTG_4 ((uint16_t)0x0010)
```

Bit 4

5.154.2.2747 TIM_BDTR_DTG_5

```
#define TIM_BDTR_DTG_5 ((uint16_t)0x0020)
```

Bit 5

5.154.2.2748 TIM_BDTR_DTG_6

```
#define TIM_BDTR_DTG_6 ((uint16_t)0x0040)
```

Bit 6

5.154.2.2749 TIM_BDTR_DTG_7

```
#define TIM_BDTR_DTG_7 ((uint16_t)0x0080)
```

Bit 7

5.154.2.2750 TIM_BDTR_LOCK

```
#define TIM_BDTR_LOCK ((uint16_t)0x0300)
```

LOCK[1:0] bits (Lock Configuration)

5.154.2.2751 TIM_BDTR_LOCK_0

```
#define TIM_BDTR_LOCK_0 ((uint16_t)0x0100)
```

Bit 0

5.154.2.2752 TIM_BDTR_LOCK_1

```
#define TIM_BDTR_LOCK_1 ((uint16_t)0x0200)
```

Bit 1

5.154.2.2753 TIM_BDTR_MOE

```
#define TIM_BDTR_MOE ((uint16_t)0x8000)
```

Main Output enable

5.154.2.2754 TIM_BDTR_OSSI

```
#define TIM_BDTR_OSSI ((uint16_t)0x0400)
```

Off-State Selection for Idle mode

5.154.2.2755 TIM_BDTR_OSSR

```
#define TIM_BDTR_OSSR ((uint16_t)0x0800)
```

Off-State Selection for Run mode

5.154.2.2756 TIM_CCER_CC1E

```
#define TIM_CCER_CC1E ((uint16_t)0x0001)
```

Capture/Compare 1 output enable

5.154.2.2757 TIM_CCER_CC1NE

```
#define TIM_CCER_CC1NE ((uint16_t)0x0004)
```

Capture/Compare 1 Complementary output enable

5.154.2.2758 TIM_CCER_CC1NP

```
#define TIM_CCER_CC1NP ((uint16_t)0x0008)
```

Capture/Compare 1 Complementary output Polarity

5.154.2.2759 TIM_CCER_CC1P

```
#define TIM_CCER_CC1P ((uint16_t)0x0002)
```

Capture/Compare 1 output Polarity

5.154.2.2760 TIM_CCER_CC2E

```
#define TIM_CCER_CC2E ((uint16_t)0x0010)
```

Capture/Compare 2 output enable

5.154.2.2761 TIM_CCER_CC2NE

```
#define TIM_CCER_CC2NE ((uint16_t)0x0040)
```

Capture/Compare 2 Complementary output enable

5.154.2.2762 TIM_CCER_CC2NP

```
#define TIM_CCER_CC2NP ((uint16_t)0x0080)
```

Capture/Compare 2 Complementary output Polarity

5.154.2.2763 TIM_CCER_CC2P

```
#define TIM_CCER_CC2P ((uint16_t)0x0020)
```

Capture/Compare 2 output Polarity

5.154.2.2764 TIM_CCER_CC3E

```
#define TIM_CCER_CC3E ((uint16_t)0x0100)
```

Capture/Compare 3 output enable

5.154.2.2765 TIM_CCER_CC3NE

```
#define TIM_CCER_CC3NE ((uint16_t)0x0400)
```

Capture/Compare 3 Complementary output enable

5.154.2.2766 TIM_CCER_CC3NP

```
#define TIM_CCER_CC3NP ((uint16_t)0x0800)
```

Capture/Compare 3 Complementary output Polarity

5.154.2.2767 TIM_CCER_CC3P

```
#define TIM_CCER_CC3P ((uint16_t)0x0200)
```

Capture/Compare 3 output Polarity

5.154.2.2768 TIM_CCER_CC4E

```
#define TIM_CCER_CC4E ((uint16_t)0x1000)
```

Capture/Compare 4 output enable

5.154.2.2769 TIM_CCER_CC4NP

```
#define TIM_CCER_CC4NP ((uint16_t)0x8000)
```

Capture/Compare 4 Complementary output Polarity

5.154.2.2770 TIM_CCER_CC4P

```
#define TIM_CCER_CC4P ((uint16_t)0x2000)
```

Capture/Compare 4 output Polarity

5.154.2.2771 TIM_CCMR1_CC1S

```
#define TIM_CCMR1_CC1S ((uint16_t)0x0003)
```

CC1S[1:0] bits (Capture/Compare 1 Selection)

5.154.2.2772 TIM_CCMR1_CC1S_0

```
#define TIM_CCMR1_CC1S_0 ((uint16_t)0x0001)
```

Bit 0

5.154.2.2773 TIM_CCMR1_CC1S_1

```
#define TIM_CCMR1_CC1S_1 ((uint16_t)0x0002)
```

Bit 1

5.154.2.2774 TIM_CCMR1_CC2S

```
#define TIM_CCMR1_CC2S ((uint16_t)0x0300)
```

CC2S[1:0] bits (Capture/Compare 2 Selection)

5.154.2.2775 TIM_CCMR1_CC2S_0

```
#define TIM_CCMR1_CC2S_0 ((uint16_t)0x0100)
```

Bit 0

5.154.2.2776 TIM_CCMR1_CC2S_1

```
#define TIM_CCMR1_CC2S_1 ((uint16_t)0x0200)
```

Bit 1

5.154.2.2777 TIM_CCMR1_IC1F

```
#define TIM_CCMR1_IC1F ((uint16_t)0x00F0)
```

IC1F[3:0] bits (Input Capture 1 Filter)

5.154.2.2778 TIM_CCMR1_IC1F_0

```
#define TIM_CCMR1_IC1F_0 ((uint16_t)0x0010)
```

Bit 0

5.154.2.2779 TIM_CCMR1_IC1F_1

```
#define TIM_CCMR1_IC1F_1 ((uint16_t)0x0020)
```

Bit 1

5.154.2.2780 TIM_CCMR1_IC1F_2

```
#define TIM_CCMR1_IC1F_2 ((uint16_t)0x0040)
```

Bit 2

5.154.2.2781 TIM_CCMR1_IC1F_3

```
#define TIM_CCMR1_IC1F_3 ((uint16_t)0x0080)
```

Bit 3

5.154.2.2782 TIM_CCMR1_IC1PSC

```
#define TIM_CCMR1_IC1PSC ((uint16_t)0x000C)
```

IC1PSC[1:0] bits (Input Capture 1 Prescaler)

5.154.2.2783 TIM_CCMR1_IC1PSC_0

```
#define TIM_CCMR1_IC1PSC_0 ((uint16_t)0x0004)
```

Bit 0

5.154.2.2784 TIM_CCMR1_IC1PSC_1

```
#define TIM_CCMR1_IC1PSC_1 ((uint16_t)0x0008)
```

Bit 1

5.154.2.2785 TIM_CCMR1_IC2F

```
#define TIM_CCMR1_IC2F ((uint16_t)0xF000)
```

IC2F[3:0] bits (Input Capture 2 Filter)

5.154.2.2786 TIM_CCMR1_IC2F_0

```
#define TIM_CCMR1_IC2F_0 ((uint16_t)0x1000)
```

Bit 0

5.154.2.2787 TIM_CCMR1_IC2F_1

```
#define TIM_CCMR1_IC2F_1 ((uint16_t)0x2000)
```

Bit 1

5.154.2.2788 TIM_CCMR1_IC2F_2

```
#define TIM_CCMR1_IC2F_2 ((uint16_t)0x4000)
```

Bit 2

5.154.2.2789 TIM_CCMR1_IC2F_3

```
#define TIM_CCMR1_IC2F_3 ((uint16_t)0x8000)
```

Bit 3

5.154.2.2790 TIM_CCMR1_IC2PSC

```
#define TIM_CCMR1_IC2PSC ((uint16_t)0x0C00)
```

IC2PSC[1:0] bits (Input Capture 2 Prescaler)

5.154.2.2791 TIM_CCMR1_IC2PSC_0

```
#define TIM_CCMR1_IC2PSC_0 ((uint16_t)0x0400)
```

Bit 0

5.154.2.2792 TIM_CCMR1_IC2PSC_1

```
#define TIM_CCMR1_IC2PSC_1 ((uint16_t)0x0800)
```

Bit 1

5.154.2.2793 TIM_CCMR1_OC1CE

```
#define TIM_CCMR1_OC1CE ((uint16_t)0x0080)
```

Output Compare 1Clear Enable

5.154.2.2794 TIM_CCMR1_OC1FE

```
#define TIM_CCMR1_OC1FE ((uint16_t)0x0004)
```

Output Compare 1 Fast enable

5.154.2.2795 TIM_CCMR1_OC1M

```
#define TIM_CCMR1_OC1M ((uint16_t)0x0070)
```

OC1M[2:0] bits (Output Compare 1 Mode)

5.154.2.2796 TIM_CCMR1_OC1M_0

```
#define TIM_CCMR1_OC1M_0 ((uint16_t)0x0010)
```

Bit 0

5.154.2.2797 TIM_CCMR1_OC1M_1

```
#define TIM_CCMR1_OC1M_1 ((uint16_t)0x0020)
```

Bit 1

5.154.2.2798 TIM_CCMR1_OC1M_2

```
#define TIM_CCMR1_OC1M_2 ((uint16_t)0x0040)
```

Bit 2

5.154.2.2799 TIM_CCMR1_OC1PE

```
#define TIM_CCMR1_OC1PE ((uint16_t)0x0008)
```

Output Compare 1 Preload enable

5.154.2.2800 TIM_CCMR1_OC2CE

```
#define TIM_CCMR1_OC2CE ((uint16_t)0x8000)
```

Output Compare 2 Clear Enable

5.154.2.2801 TIM_CCMR1_OC2FE

```
#define TIM_CCMR1_OC2FE ((uint16_t)0x0400)
```

Output Compare 2 Fast enable

5.154.2.2802 TIM_CCMR1_OC2M

```
#define TIM_CCMR1_OC2M ((uint16_t)0x7000)
```

OC2M[2:0] bits (Output Compare 2 Mode)

5.154.2.2803 TIM_CCMR1_OC2M_0

```
#define TIM_CCMR1_OC2M_0 ((uint16_t)0x1000)
```

Bit 0

5.154.2.2804 TIM_CCMR1_OC2M_1

```
#define TIM_CCMR1_OC2M_1 ((uint16_t)0x2000)
```

Bit 1

5.154.2.2805 TIM_CCMR1_OC2M_2

```
#define TIM_CCMR1_OC2M_2 ((uint16_t)0x4000)
```

Bit 2

5.154.2.2806 TIM_CCMR1_OC2PE

```
#define TIM_CCMR1_OC2PE ((uint16_t)0x0800)
```

Output Compare 2 Preload enable

5.154.2.2807 TIM_CCMR2_CC3S

```
#define TIM_CCMR2_CC3S ((uint16_t)0x0003)
```

CC3S[1:0] bits (Capture/Compare 3 Selection)

5.154.2.2808 TIM_CCMR2_CC3S_0

```
#define TIM_CCMR2_CC3S_0 ((uint16_t)0x0001)
```

Bit 0

5.154.2.2809 TIM_CCMR2_CC3S_1

```
#define TIM_CCMR2_CC3S_1 ((uint16_t)0x0002)
```

Bit 1

5.154.2.2810 TIM_CCMR2_CC4S

```
#define TIM_CCMR2_CC4S ((uint16_t)0x0300)
```

CC4S[1:0] bits (Capture/Compare 4 Selection)

5.154.2.2811 TIM_CCMR2_CC4S_0

```
#define TIM_CCMR2_CC4S_0 ((uint16_t)0x0100)
```

Bit 0

5.154.2.2812 TIM_CCMR2_CC4S_1

```
#define TIM_CCMR2_CC4S_1 ((uint16_t)0x0200)
```

Bit 1

5.154.2.2813 TIM_CCMR2_IC3F

```
#define TIM_CCMR2_IC3F ((uint16_t)0x00F0)
```

IC3F[3:0] bits (Input Capture 3 Filter)

5.154.2.2814 TIM_CCMR2_IC3F_0

```
#define TIM_CCMR2_IC3F_0 ((uint16_t)0x0010)
```

Bit 0

5.154.2.2815 TIM_CCMR2_IC3F_1

```
#define TIM_CCMR2_IC3F_1 ((uint16_t)0x0020)
```

Bit 1

5.154.2.2816 TIM_CCMR2_IC3F_2

```
#define TIM_CCMR2_IC3F_2 ((uint16_t)0x0040)
```

Bit 2

5.154.2.2817 TIM_CCMR2_IC3F_3

```
#define TIM_CCMR2_IC3F_3 ((uint16_t)0x0080)
```

Bit 3

5.154.2.2818 TIM_CCMR2_IC3PSC

```
#define TIM_CCMR2_IC3PSC ((uint16_t)0x000C)
```

IC3PSC[1:0] bits (Input Capture 3 Prescaler)

5.154.2.2819 TIM_CCMR2_IC3PSC_0

```
#define TIM_CCMR2_IC3PSC_0 ((uint16_t)0x0004)
```

Bit 0

5.154.2.2820 TIM_CCMR2_IC3PSC_1

```
#define TIM_CCMR2_IC3PSC_1 ((uint16_t)0x0008)
```

Bit 1

5.154.2.2821 TIM_CCMR2_IC4F

```
#define TIM_CCMR2_IC4F ((uint16_t)0xF000)
```

IC4F[3:0] bits (Input Capture 4 Filter)

5.154.2.2822 TIM_CCMR2_IC4F_0

```
#define TIM_CCMR2_IC4F_0 ((uint16_t)0x1000)
```

Bit 0

5.154.2.2823 TIM_CCMR2_IC4F_1

```
#define TIM_CCMR2_IC4F_1 ((uint16_t)0x2000)
```

Bit 1

5.154.2.2824 TIM_CCMR2_IC4F_2

```
#define TIM_CCMR2_IC4F_2 ((uint16_t)0x4000)
```

Bit 2

5.154.2.2825 TIM_CCMR2_IC4F_3

```
#define TIM_CCMR2_IC4F_3 ((uint16_t)0x8000)
```

Bit 3

5.154.2.2826 TIM_CCMR2_IC4PSC

```
#define TIM_CCMR2_IC4PSC ((uint16_t)0x0C00)
```

IC4PSC[1:0] bits (Input Capture 4 Prescaler)

5.154.2.2827 TIM_CCMR2_IC4PSC_0

```
#define TIM_CCMR2_IC4PSC_0 ((uint16_t)0x0400)
```

Bit 0

5.154.2.2828 TIM_CCMR2_IC4PSC_1

```
#define TIM_CCMR2_IC4PSC_1 ((uint16_t)0x0800)
```

Bit 1

5.154.2.2829 TIM_CCMR2_OC3CE

```
#define TIM_CCMR2_OC3CE ((uint16_t)0x0080)
```

Output Compare 3 Clear Enable

5.154.2.2830 TIM_CCMR2_OC3FE

```
#define TIM_CCMR2_OC3FE ((uint16_t)0x0004)
```

Output Compare 3 Fast enable

5.154.2.2831 TIM_CCMR2_OC3M

```
#define TIM_CCMR2_OC3M ((uint16_t)0x0070)
```

OC3M[2:0] bits (Output Compare 3 Mode)

5.154.2.2832 TIM_CCMR2_OC3M_0

```
#define TIM_CCMR2_OC3M_0 ((uint16_t)0x0010)
```

Bit 0

5.154.2.2833 TIM_CCMR2_OC3M_1

```
#define TIM_CCMR2_OC3M_1 ((uint16_t)0x0020)
```

Bit 1

5.154.2.2834 TIM_CCMR2_OC3M_2

```
#define TIM_CCMR2_OC3M_2 ((uint16_t)0x0040)
```

Bit 2

5.154.2.2835 TIM_CCMR2_OC3PE

```
#define TIM_CCMR2_OC3PE ((uint16_t)0x0008)
```

Output Compare 3 Preload enable

5.154.2.2836 TIM_CCMR2_OC4CE

```
#define TIM_CCMR2_OC4CE ((uint16_t)0x8000)
```

Output Compare 4 Clear Enable

5.154.2.2837 TIM_CCMR2_OC4FE

```
#define TIM_CCMR2_OC4FE ((uint16_t)0x0400)
```

Output Compare 4 Fast enable

5.154.2.2838 TIM_CCMR2_OC4M

```
#define TIM_CCMR2_OC4M ((uint16_t)0x7000)
```

OC4M[2:0] bits (Output Compare 4 Mode)

5.154.2.2839 TIM_CCMR2_OC4M_0

```
#define TIM_CCMR2_OC4M_0 ((uint16_t)0x1000)
```

Bit 0

5.154.2.2840 TIM_CCMR2_OC4M_1

```
#define TIM_CCMR2_OC4M_1 ((uint16_t)0x2000)
```

Bit 1

5.154.2.2841 TIM_CCMR2_OC4M_2

```
#define TIM_CCMR2_OC4M_2 ((uint16_t)0x4000)
```

Bit 2

5.154.2.2842 TIM_CCMR2_OC4PE

```
#define TIM_CCMR2_OC4PE ((uint16_t)0x0800)
```

Output Compare 4 Preload enable

5.154.2.2843 TIM_CCR1_CCR1

```
#define TIM_CCR1_CCR1 ((uint16_t)0xFFFF)
```

Capture/Compare 1 Value

5.154.2.2844 TIM_CCR2_CCR2

```
#define TIM_CCR2_CCR2 ((uint16_t)0xFFFF)
```

Capture/Compare 2 Value

5.154.2.2845 TIM_CCR3_CCR3

```
#define TIM_CCR3_CCR3 ((uint16_t)0xFFFF)
```

Capture/Compare 3 Value

5.154.2.2846 TIM_CCR4_CCR4

```
#define TIM_CCR4_CCR4 ((uint16_t)0xFFFF)
```

Capture/Compare 4 Value

5.154.2.2847 TIM_CNT_CNT

```
#define TIM_CNT_CNT ((uint16_t)0xFFFF)
```

Counter Value

5.154.2.2848 TIM_CR1_ARPE

```
#define TIM_CR1_ARPE ((uint16_t)0x0080)
```

Auto-reload preload enable

5.154.2.2849 TIM_CR1_CEN

```
#define TIM_CR1_CEN ((uint16_t)0x0001)
```

Counter enable

5.154.2.2850 TIM_CR1_CKD

```
#define TIM_CR1_CKD ((uint16_t)0x0300)
```

CKD[1:0] bits (clock division)

5.154.2.2851 TIM_CR1_CKD_0

```
#define TIM_CR1_CKD_0 ((uint16_t)0x0100)
```

Bit 0

5.154.2.2852 TIM_CR1_CKD_1

```
#define TIM_CR1_CKD_1 ((uint16_t)0x0200)
```

Bit 1

5.154.2.2853 TIM_CR1_CMS

```
#define TIM_CR1_CMS ((uint16_t)0x0060)
```

CMS[1:0] bits (Center-aligned mode selection)

5.154.2.2854 TIM_CR1_CMS_0

```
#define TIM_CR1_CMS_0 ((uint16_t)0x0020)
```

Bit 0

5.154.2.2855 TIM_CR1_CMS_1

```
#define TIM_CR1_CMS_1 ((uint16_t)0x0040)
```

Bit 1

5.154.2.2856 TIM_CR1_DIR

```
#define TIM_CR1_DIR ((uint16_t)0x0010)
```

Direction

5.154.2.2857 TIM_CR1_OPM

```
#define TIM_CR1_OPM ((uint16_t)0x0008)
```

One pulse mode

5.154.2.2858 TIM_CR1_UDIS

```
#define TIM_CR1_UDIS ((uint16_t)0x0002)
```

Update disable

5.154.2.2859 TIM_CR1_URS

```
#define TIM_CR1_URS ((uint16_t)0x0004)
```

Update request source

5.154.2.2860 TIM_CR2_CCDS

```
#define TIM_CR2_CCDS ((uint16_t)0x0008)
```

Capture/Compare DMA Selection

5.154.2.2861 TIM_CR2_CCPC

```
#define TIM_CR2_CCPC ((uint16_t)0x0001)
```

Capture/Compare Preloaded Control

5.154.2.2862 TIM_CR2_CCUS

```
#define TIM_CR2_CCUS ((uint16_t)0x0004)
```

Capture/Compare Control Update Selection

5.154.2.2863 TIM_CR2_MMS

```
#define TIM_CR2_MMS ((uint16_t)0x0070)
```

MMS[2:0] bits (Master Mode Selection)

5.154.2.2864 TIM_CR2_MMS_0

```
#define TIM_CR2_MMS_0 ((uint16_t)0x0010)
```

Bit 0

5.154.2.2865 TIM_CR2_MMS_1

```
#define TIM_CR2_MMS_1 ((uint16_t)0x0020)
```

Bit 1

5.154.2.2866 TIM_CR2_MMS_2

```
#define TIM_CR2_MMS_2 ((uint16_t)0x0040)
```

Bit 2

5.154.2.2867 TIM_CR2_OIS1

```
#define TIM_CR2_OIS1 ((uint16_t)0x0100)
```

Output Idle state 1 (OC1 output)

5.154.2.2868 TIM_CR2_OIS1N

```
#define TIM_CR2_OIS1N ((uint16_t)0x0200)
```

Output Idle state 1 (OC1N output)

5.154.2.2869 TIM_CR2_OIS2

```
#define TIM_CR2_OIS2 ((uint16_t)0x0400)
```

Output Idle state 2 (OC2 output)

5.154.2.2870 TIM_CR2_OIS2N

```
#define TIM_CR2_OIS2N ((uint16_t)0x0800)
```

Output Idle state 2 (OC2N output)

5.154.2.2871 TIM_CR2_OIS3

```
#define TIM_CR2_OIS3 ((uint16_t)0x1000)
```

Output Idle state 3 (OC3 output)

5.154.2.2872 TIM_CR2_OIS3N

```
#define TIM_CR2_OIS3N ((uint16_t)0x2000)
```

Output Idle state 3 (OC3N output)

5.154.2.2873 TIM_CR2_OIS4

```
#define TIM_CR2_OIS4 ((uint16_t)0x4000)
```

Output Idle state 4 (OC4 output)

5.154.2.2874 TIM_CR2_TI1S

```
#define TIM_CR2_TI1S ((uint16_t)0x0080)
```

TI1 Selection

5.154.2.2875 TIM_DCR_DBA

```
#define TIM_DCR_DBA ((uint16_t)0x001F)
```

DBA[4:0] bits (DMA Base Address)

5.154.2.2876 TIM_DCR_DBA_0

```
#define TIM_DCR_DBA_0 ((uint16_t)0x0001)
```

Bit 0

5.154.2.2877 TIM_DCR_DBA_1

```
#define TIM_DCR_DBA_1 ((uint16_t)0x0002)
```

Bit 1

5.154.2.2878 TIM_DCR_DBA_2

```
#define TIM_DCR_DBA_2 ((uint16_t)0x0004)
```

Bit 2

5.154.2.2879 TIM_DCR_DBA_3

```
#define TIM_DCR_DBA_3 ((uint16_t)0x0008)
```

Bit 3

5.154.2.2880 TIM_DCR_DBA_4

```
#define TIM_DCR_DBA_4 ((uint16_t)0x0010)
```

Bit 4

5.154.2.2881 TIM_DCR_DBL

```
#define TIM_DCR_DBL ((uint16_t)0x1F00)
```

DBL[4:0] bits (DMA Burst Length)

5.154.2.2882 TIM_DCR_DBL_0

```
#define TIM_DCR_DBL_0 ((uint16_t)0x0100)
```

Bit 0

5.154.2.2883 TIM_DCR_DBL_1

```
#define TIM_DCR_DBL_1 ((uint16_t)0x0200)
```

Bit 1

5.154.2.2884 TIM_DCR_DBL_2

```
#define TIM_DCR_DBL_2 ((uint16_t)0x0400)
```

Bit 2

5.154.2.2885 TIM_DCR_DBL_3

```
#define TIM_DCR_DBL_3 ((uint16_t)0x0800)
```

Bit 3

5.154.2.2886 TIM_DCR_DBL_4

```
#define TIM_DCR_DBL_4 ((uint16_t)0x1000)
```

Bit 4

5.154.2.2887 TIM_DIER_BIE

```
#define TIM_DIER_BIE ((uint16_t)0x0080)
```

Break interrupt enable

5.154.2.2888 TIM_DIER_CC1DE

```
#define TIM_DIER_CC1DE ((uint16_t)0x0200)
```

Capture/Compare 1 DMA request enable

5.154.2.2889 TIM_DIER_CC1IE

```
#define TIM_DIER_CC1IE ((uint16_t)0x0002)
```

Capture/Compare 1 interrupt enable

5.154.2.2890 TIM_DIER_CC2DE

```
#define TIM_DIER_CC2DE ((uint16_t)0x0400)
```

Capture/Compare 2 DMA request enable

5.154.2.2891 TIM_DIER_CC2IE

```
#define TIM_DIER_CC2IE ((uint16_t)0x0004)
```

Capture/Compare 2 interrupt enable

5.154.2.2892 TIM_DIER_CC3DE

```
#define TIM_DIER_CC3DE ((uint16_t)0x0800)
```

Capture/Compare 3 DMA request enable

5.154.2.2893 TIM_DIER_CC3IE

```
#define TIM_DIER_CC3IE ((uint16_t)0x0008)
```

Capture/Compare 3 interrupt enable

5.154.2.2894 TIM_DIER_CC4DE

```
#define TIM_DIER_CC4DE ((uint16_t)0x1000)
```

Capture/Compare 4 DMA request enable

5.154.2.2895 TIM_DIER_CC4IE

```
#define TIM_DIER_CC4IE ((uint16_t)0x0010)
```

Capture/Compare 4 interrupt enable

5.154.2.2896 TIM_DIER_COMDE

```
#define TIM_DIER_COMDE ((uint16_t)0x2000)
```

COM DMA request enable

5.154.2.2897 TIM_DIER_COMIE

```
#define TIM_DIER_COMIE ((uint16_t)0x0020)
```

COM interrupt enable

5.154.2.2898 TIM_DIER_TDE

```
#define TIM_DIER_TDE ((uint16_t)0x4000)
```

Trigger DMA request enable

5.154.2.2899 TIM_DIER_TIE

```
#define TIM_DIER_TIE ((uint16_t)0x0040)
```

Trigger interrupt enable

5.154.2.2900 TIM_DIER_UDE

```
#define TIM_DIER_UDE ((uint16_t)0x0100)
```

Update DMA request enable

5.154.2.2901 TIM_DIER_UIE

```
#define TIM_DIER_UIE ((uint16_t)0x0001)
```

Update interrupt enable

5.154.2.2902 TIM_DMAR_DMAB

```
#define TIM_DMAR_DMAB ((uint16_t)0xFFFF)
```

DMA register for burst accesses

5.154.2.2903 TIM_EGR_BG

```
#define TIM_EGR_BG ((uint8_t)0x80)
```

Break Generation

5.154.2.2904 TIM_EGR_CC1G

```
#define TIM_EGR_CC1G ((uint8_t)0x02)
```

Capture/Compare 1 Generation

5.154.2.2905 TIM_EGR_CC2G

```
#define TIM_EGR_CC2G ((uint8_t)0x04)
```

Capture/Compare 2 Generation

5.154.2.2906 TIM_EGR_CC3G

```
#define TIM_EGR_CC3G ((uint8_t)0x08)
```

Capture/Compare 3 Generation

5.154.2.2907 TIM_EGR_CC4G

```
#define TIM_EGR_CC4G ((uint8_t)0x10)
```

Capture/Compare 4 Generation

5.154.2.2908 TIM_EGR_COMG

```
#define TIM_EGR_COMG ((uint8_t)0x20)
```

Capture/Compare Control Update Generation

5.154.2.2909 TIM_EGR_TG

```
#define TIM_EGR_TG ((uint8_t)0x40)
```

Trigger Generation

5.154.2.2910 TIM_EGR_UG

```
#define TIM_EGR_UG ((uint8_t)0x01)
```

Update Generation

5.154.2.2911 TIM_OR_ITR1_RMP

```
#define TIM_OR_ITR1_RMP ((uint16_t)0x0C00)
```

ITR1_RMP[1:0] bits (TIM2 Internal trigger 1 remap)

5.154.2.2912 TIM_OR_ITR1_RMP_0

```
#define TIM_OR_ITR1_RMP_0 ((uint16_t)0x0400)
```

Bit 0

5.154.2.2913 TIM_OR_ITR1_RMP_1

```
#define TIM_OR_ITR1_RMP_1 ((uint16_t)0x0800)
```

Bit 1

5.154.2.2914 TIM_OR_TI4_RMP

```
#define TIM_OR_TI4_RMP ((uint16_t)0x00C0)
```

TI4_RMP[1:0] bits (TIM5 Input 4 remap)

5.154.2.2915 TIM_OR_TI4_RMP_0

```
#define TIM_OR_TI4_RMP_0 ((uint16_t)0x0040)
```

Bit 0

5.154.2.2916 TIM_OR_TI4_RMP_1

```
#define TIM_OR_TI4_RMP_1 ((uint16_t)0x0080)
```

Bit 1

5.154.2.2917 TIM_PSC_PSC

```
#define TIM_PSC_PSC ((uint16_t)0xFFFF)
```

Prescaler Value

5.154.2.2918 TIM_RCR_REP

```
#define TIM_RCR_REP ((uint8_t)0xFF)
```

Repetition Counter Value

5.154.2.2919 TIM_SMCR_ECE

```
#define TIM_SMCR_ECE ((uint16_t)0x4000)
```

External clock enable

5.154.2.2920 TIM_SMCR_ETF

```
#define TIM_SMCR_ETF ((uint16_t)0x0F00)
```

ETF[3:0] bits (External trigger filter)

5.154.2.2921 TIM_SMCR_ETF_0

```
#define TIM_SMCR_ETF_0 ((uint16_t)0x0100)
```

Bit 0

5.154.2.2922 TIM_SMCR_ETF_1

```
#define TIM_SMCR_ETF_1 ((uint16_t)0x0200)
```

Bit 1

5.154.2.2923 TIM_SMCR_ETF_2

```
#define TIM_SMCR_ETF_2 ((uint16_t)0x0400)
```

Bit 2

5.154.2.2924 TIM_SMCR_ETF_3

```
#define TIM_SMCR_ETF_3 ((uint16_t)0x0800)
```

Bit 3

5.154.2.2925 TIM_SMCR_ETP

```
#define TIM_SMCR_ETP ((uint16_t)0x8000)
```

External trigger polarity

5.154.2.2926 TIM_SMCR_ETPS

```
#define TIM_SMCR_ETPS ((uint16_t)0x3000)
```

ETPS[1:0] bits (External trigger prescaler)

5.154.2.2927 TIM_SMCR_ETPS_0

```
#define TIM_SMCR_ETPS_0 ((uint16_t)0x1000)
```

Bit 0

5.154.2.2928 TIM_SMCR_ETPS_1

```
#define TIM_SMCR_ETPS_1 ((uint16_t)0x2000)
```

Bit 1

5.154.2.2929 TIM_SMCR_MSM

```
#define TIM_SMCR_MSM ((uint16_t)0x0080)
```

Master/slave mode

5.154.2.2930 TIM_SMCR_SMS

```
#define TIM_SMCR_SMS ((uint16_t)0x0007)
```

SMS[2:0] bits (Slave mode selection)

5.154.2.2931 TIM_SMCR_SMS_0

```
#define TIM_SMCR_SMS_0 ((uint16_t)0x0001)
```

Bit 0

5.154.2.2932 TIM_SMCR_SMS_1

```
#define TIM_SMCR_SMS_1 ((uint16_t)0x0002)
```

Bit 1

5.154.2.2933 TIM_SMCR_SMS_2

```
#define TIM_SMCR_SMS_2 ((uint16_t)0x0004)
```

Bit 2

5.154.2.2934 TIM_SMCR_TS

```
#define TIM_SMCR_TS ((uint16_t)0x0070)
```

TS[2:0] bits (Trigger selection)

5.154.2.2935 TIM_SMCR_TS_0

```
#define TIM_SMCR_TS_0 ((uint16_t)0x0010)
```

Bit 0

5.154.2.2936 TIM_SMCR_TS_1

```
#define TIM_SMCR_TS_1 ((uint16_t)0x0020)
```

Bit 1

5.154.2.2937 TIM_SMCR_TS_2

```
#define TIM_SMCR_TS_2 ((uint16_t)0x0040)
```

Bit 2

5.154.2.2938 TIM_SR_BIF

```
#define TIM_SR_BIF ((uint16_t)0x0080)
```

Break interrupt Flag

5.154.2.2939 TIM_SR_CC1IF

```
#define TIM_SR_CC1IF ((uint16_t)0x0002)
```

Capture/Compare 1 interrupt Flag

5.154.2.2940 TIM_SR_CC1OF

```
#define TIM_SR_CC1OF ((uint16_t)0x0200)
```

Capture/Compare 1 Overcapture Flag

5.154.2.2941 TIM_SR_CC2IF

```
#define TIM_SR_CC2IF ((uint16_t)0x0004)
```

Capture/Compare 2 interrupt Flag

5.154.2.2942 TIM_SR_CC2OF

```
#define TIM_SR_CC2OF ((uint16_t)0x0400)
```

Capture/Compare 2 Overcapture Flag

5.154.2.2943 TIM_SR_CC3IF

```
#define TIM_SR_CC3IF ((uint16_t)0x0008)
```

Capture/Compare 3 interrupt Flag

5.154.2.2944 TIM_SR_CC3OF

```
#define TIM_SR_CC3OF ((uint16_t)0x0800)
```

Capture/Compare 3 Overcapture Flag

5.154.2.2945 TIM_SR_CC4IF

```
#define TIM_SR_CC4IF ((uint16_t)0x0010)
```

Capture/Compare 4 interrupt Flag

5.154.2.2946 TIM_SR_CC4OF

```
#define TIM_SR_CC4OF ((uint16_t)0x1000)
```

Capture/Compare 4 Overcapture Flag

5.154.2.2947 TIM_SR_COMIF

```
#define TIM_SR_COMIF ((uint16_t)0x0020)
```

COM interrupt Flag

5.154.2.2948 TIM_SR_TIF

```
#define TIM_SR_TIF ((uint16_t)0x0040)
```

Trigger interrupt Flag

5.154.2.2949 TIM_SR_UIF

```
#define TIM_SR_UIF ((uint16_t)0x0001)
```

Update interrupt Flag

5.154.2.2950 USART_BRR_DIV_Fraction

```
#define USART_BRR_DIV_Fraction ((uint16_t)0x000F)
```

Fraction of USARTDIV

5.154.2.2951 USART_BRR_DIV_Mantissa

```
#define USART_BRR_DIV_Mantissa ((uint16_t)0xFFF0)
```

Mantissa of USARTDIV

5.154.2.2952 USART_CR1_IDLEIE

```
#define USART_CR1_IDLEIE ((uint16_t)0x0010)
```

IDLE Interrupt Enable

5.154.2.2953 USART_CR1_M

```
#define USART_CR1_M ((uint16_t)0x1000)
```

Word length

5.154.2.2954 USART_CR1_OVER8

```
#define USART_CR1_OVER8 ((uint16_t)0x8000)
```

USART Oversampling by 8 enable

5.154.2.2955 USART_CR1_PCE

```
#define USART_CR1_PCE ((uint16_t)0x0400)
```

Parity Control Enable

5.154.2.2956 USART_CR1_PEIE

```
#define USART_CR1_PEIE ((uint16_t)0x0100)
```

PE Interrupt Enable

5.154.2.2957 USART_CR1_PS

```
#define USART_CR1_PS ((uint16_t)0x0200)
```

Parity Selection

5.154.2.2958 USART_CR1_RE

```
#define USART_CR1_RE ((uint16_t)0x0004)
```

Receiver Enable

5.154.2.2959 USART_CR1_RWU

```
#define USART_CR1_RWU ((uint16_t)0x0002)
```

Receiver wakeup

5.154.2.2960 USART_CR1_RXNEIE

```
#define USART_CR1_RXNEIE ((uint16_t)0x0020)
```

RXNE Interrupt Enable

5.154.2.2961 USART_CR1_SBK

```
#define USART_CR1_SBK ((uint16_t)0x0001)
```

Send Break

5.154.2.2962 USART_CR1_TCIE

```
#define USART_CR1_TCIE ((uint16_t)0x0040)
```

Transmission Complete Interrupt Enable

5.154.2.2963 USART_CR1_TE

```
#define USART_CR1_TE ((uint16_t)0x0008)
```

Transmitter Enable

5.154.2.2964 USART_CR1_TXEIE

```
#define USART_CR1_TXEIE ((uint16_t)0x0080)
```

PE Interrupt Enable

5.154.2.2965 USART_CR1_UE

```
#define USART_CR1_UE ((uint16_t)0x2000)
```

USART Enable

5.154.2.2966 USART_CR1_WAKE

```
#define USART_CR1_WAKE ((uint16_t)0x0800)
```

Wakeup method

5.154.2.2967 USART_CR2_ADD

```
#define USART_CR2_ADD ((uint16_t)0x000F)
```

Address of the USART node

5.154.2.2968 USART_CR2_CLKEN

```
#define USART_CR2_CLKEN ((uint16_t)0x0800)
```

Clock Enable

5.154.2.2969 USART_CR2_CPHA

```
#define USART_CR2_CPHA ((uint16_t)0x0200)
```

Clock Phase

5.154.2.2970 USART_CR2_CPOL

```
#define USART_CR2_CPOL ((uint16_t)0x0400)
```

Clock Polarity

5.154.2.2971 USART_CR2_LBCL

```
#define USART_CR2_LBCL ((uint16_t)0x0100)
```

Last Bit Clock pulse

5.154.2.2972 USART_CR2_LBDIE

```
#define USART_CR2_LBDIE ((uint16_t)0x0040)
```

LIN Break Detection Interrupt Enable

5.154.2.2973 USART_CR2_LBDL

```
#define USART_CR2_LBDL ((uint16_t)0x0020)
```

LIN Break Detection Length

5.154.2.2974 USART_CR2_LINEN

```
#define USART_CR2_LINEN ((uint16_t)0x4000)
```

LIN mode enable

5.154.2.2975 USART_CR2_STOP

```
#define USART_CR2_STOP ((uint16_t)0x3000)
```

STOP[1:0] bits (STOP bits)

5.154.2.2976 USART_CR2_STOP_0

```
#define USART_CR2_STOP_0 ((uint16_t)0x1000)
```

Bit 0

5.154.2.2977 USART_CR2_STOP_1

```
#define USART_CR2_STOP_1 ((uint16_t)0x2000)
```

Bit 1

5.154.2.2978 USART_CR3_CTSE

```
#define USART_CR3_CTSE ((uint16_t)0x0200)
```

CTS Enable

5.154.2.2979 USART_CR3_CTSIE

```
#define USART_CR3_CTSIE ((uint16_t)0x0400)
```

CTS Interrupt Enable

5.154.2.2980 USART_CR3_DMAR

```
#define USART_CR3_DMAR ((uint16_t)0x0040)
```

DMA Enable Receiver

5.154.2.2981 USART_CR3_DMAT

```
#define USART_CR3_DMAT ((uint16_t)0x0080)
```

DMA Enable Transmitter

5.154.2.2982 USART_CR3_EIE

```
#define USART_CR3_EIE ((uint16_t)0x0001)
```

Error Interrupt Enable

5.154.2.2983 USART_CR3_HDSEL

```
#define USART_CR3_HDSEL ((uint16_t)0x0008)
```

Half-Duplex Selection

5.154.2.2984 USART_CR3_IREN

```
#define USART_CR3_IREN ((uint16_t)0x0002)
```

IrDA mode Enable

5.154.2.2985 USART_CR3_IRLP

```
#define USART_CR3_IRLP ((uint16_t)0x0004)
```

IrDA Low-Power

5.154.2.2986 USART_CR3_NACK

```
#define USART_CR3_NACK ((uint16_t)0x0010)
```

Smartcard NACK enable

5.154.2.2987 USART_CR3_ONEBIT

```
#define USART_CR3_ONEBIT ((uint16_t)0x0800)
```

USART One bit method enable

5.154.2.2988 USART_CR3_RTSE

```
#define USART_CR3_RTSE ((uint16_t)0x0100)
```

RTS Enable

5.154.2.2989 USART_CR3_SCEN

```
#define USART_CR3_SCEN ((uint16_t)0x0020)
```

Smartcard mode enable

5.154.2.2990 USART_DR_DR

```
#define USART_DR_DR ((uint16_t)0x01FF)
```

Data value

5.154.2.2991 USART_GTPR_GT

```
#define USART_GTPR_GT ((uint16_t)0xFF00)
```

Guard time value

5.154.2.2992 USART_GTPR_PSC

```
#define USART_GTPR_PSC ((uint16_t)0x00FF)
```

PSC[7:0] bits (Prescaler value)

5.154.2.2993 USART_GTPR_PSC_0

```
#define USART_GTPR_PSC_0 ((uint16_t)0x0001)
```

Bit 0

5.154.2.2994 USART_GTPR_PSC_1

```
#define USART_GTPR_PSC_1 ((uint16_t)0x0002)
```

Bit 1

5.154.2.2995 USART_GTPR_PSC_2

```
#define USART_GTPR_PSC_2 ((uint16_t)0x0004)
```

Bit 2

5.154.2.2996 USART_GTPR_PSC_3

```
#define USART_GTPR_PSC_3 ((uint16_t)0x0008)
```

Bit 3

5.154.2.2997 USART_GTPR_PSC_4

```
#define USART_GTPR_PSC_4 ((uint16_t)0x0010)
```

Bit 4

5.154.2.2998 USART_GTPR_PSC_5

```
#define USART_GTPR_PSC_5 ((uint16_t)0x0020)
```

Bit 5

5.154.2.2999 USART_GTPR_PSC_6

```
#define USART_GTPR_PSC_6 ((uint16_t)0x0040)
```

Bit 6

5.154.2.3000 USART_GTPR_PSC_7

```
#define USART_GTPR_PSC_7 ((uint16_t)0x0080)
```

Bit 7

5.154.2.3001 USART_SR_CTS

```
#define USART_SR_CTS ((uint16_t)0x0200)
```

CTS Flag

5.154.2.3002 USART_SR_FE

```
#define USART_SR_FE ((uint16_t)0x0002)
```

Framing Error

5.154.2.3003 USART_SR_IDLE

```
#define USART_SR_IDLE ((uint16_t)0x0010)
```

IDLE line detected

5.154.2.3004 USART_SR_LBD

```
#define USART_SR_LBD ((uint16_t)0x0100)
```

LIN Break Detection Flag

5.154.2.3005 USART_SR_NE

```
#define USART_SR_NE ((uint16_t)0x0004)
```

Noise Error Flag

5.154.2.3006 USART_SR_ORE

```
#define USART_SR_ORE ((uint16_t)0x0008)
```

OverRun Error

5.154.2.3007 USART_SR_PE

```
#define USART_SR_PE ((uint16_t)0x0001)
```

Parity Error

5.154.2.3008 USART_SR_RXNE

```
#define USART_SR_RXNE ((uint16_t)0x0020)
```

Read Data Register Not Empty

5.154.2.3009 USART_SR_TC

```
#define USART_SR_TC ((uint16_t)0x0040)
```

Transmission Complete

5.154.2.3010 USART_SR_TXE

```
#define USART_SR_TXE ((uint16_t)0x0080)
```

Transmit Data Register Empty

5.154.2.3011 WWDG_CFR_EWI

```
#define WWDG_CFR_EWI ((uint16_t)0x0200)
```

Early Wakeup Interrupt

5.154.2.3012 WWDG_CFR_W

```
#define WWDG_CFR_W ((uint16_t)0x007F)
```

W[6:0] bits (7-bit window value)

5.154.2.3013 WWDG_CFR_W0

```
#define WWDG_CFR_W0 ((uint16_t)0x0001)
```

Bit 0

5.154.2.3014 WWDG_CFR_W1

```
#define WWDG_CFR_W1 ((uint16_t)0x0002)
```

Bit 1

5.154.2.3015 WWDG_CFR_W2

```
#define WWDG_CFR_W2 ((uint16_t)0x0004)
```

Bit 2

5.154.2.3016 WWDG_CFR_W3

```
#define WWDG_CFR_W3 ((uint16_t)0x0008)
```

Bit 3

5.154.2.3017 WWDG_CFR_W4

```
#define WWDG_CFR_W4 ((uint16_t)0x0010)
```

Bit 4

5.154.2.3018 WWDG_CFR_W5

```
#define WWDG_CFR_W5 ((uint16_t)0x0020)
```

Bit 5

5.154.2.3019 WWDG_CFR_W6

```
#define WWDG_CFR_W6 ((uint16_t)0x0040)
```

Bit 6

5.154.2.3020 WWDG_CFR_WDGTB

```
#define WWDG_CFR_WDGTB ((uint16_t)0x0180)
```

WDGTB[1:0] bits (Timer Base)

5.154.2.3021 WWDG_CFR_WDGTB0

```
#define WWDG_CFR_WDGTB0 ((uint16_t)0x0080)
```

Bit 0

5.154.2.3022 WWDG_CFR_WDGTB1

```
#define WWDG_CFR_WDGTB1 ((uint16_t)0x0100)
```

Bit 1

5.154.2.3023 WWDG_CR_T

```
#define WWDG_CR_T ((uint8_t)0x7F)
```

T[6:0] bits (7-Bit counter (MSB to LSB))

5.154.2.3024 WWDG_CR_T0

```
#define WWDG_CR_T0 ((uint8_t)0x01)
```

Bit 0

5.154.2.3025 WWDG_CR_T1

```
#define WWDG_CR_T1 ((uint8_t)0x02)
```

Bit 1

5.154.2.3026 WWDG_CR_T2

```
#define WWDG_CR_T2 ((uint8_t)0x04)
```

Bit 2

5.154.2.3027 WWDG_CR_T3

```
#define WWDG_CR_T3 ((uint8_t)0x08)
```

Bit 3

5.154.2.3028 WWDG_CR_T4

```
#define WWDG_CR_T4 ((uint8_t)0x10)
```

Bit 4

5.154.2.3029 WWDG_CR_T5

```
#define WWDG_CR_T5 ((uint8_t)0x20)
```

Bit 5

5.154.2.3030 WWDG_CR_T6

```
#define WWDG_CR_T6 ((uint8_t)0x40)
```

Bit 6

5.154.2.3031 WWDG_CR_WDGA

```
#define WWDG_CR_WDGA ((uint8_t)0x80)
```

Activation bit

5.154.2.3032 WWDG_SR_EWIF

```
#define WWDG_SR_EWIF ((uint8_t)0x01)
```

Early Wakeup Interrupt Flag

5.155 Exported_macro

Macros

- #define **SET_BIT**(REG, BIT) ((REG) |= (BIT))
- #define **CLEAR_BIT**(REG, BIT) ((REG) &= ~(BIT))
- #define **READ_BIT**(REG, BIT) ((REG) & (BIT))
- #define **CLEAR_REG**(REG) ((REG) = (0x0))
- #define **WRITE_REG**(REG, VAL) ((REG) = (VAL))
- #define **READ_REG**(REG) ((REG))
- #define **MODIFY_REG**(REG, CLEARMASK, SETMASK) WRITE_REG((REG), (((READ_REG(REG)) & (~(CLEARMASK))) | (SETMASK)))

5.155.1 Detailed Description

5.156 Stm32f4xx_system

**

Modules

- STM32F4xx_System_Includes

  Define to prevent recursive inclusion.
- STM32F4xx_System_Exported_types
- STM32F4xx_System_Exported_Constants
- STM32F4xx_System_Exported_Macros
- STM32F4xx_System_Exported_Functions
- STM32F4xx_System_Private_Includes

  **
- STM32F4xx_System_Private_TypesDefinitions

  **
- STM32F4xx_System_Private_Defines

  **
- STM32F4xx_System_Private_Macros
- STM32F4xx_System_Private_Variables
- STM32F4xx_System_Private_FunctionPrototypes
- STM32F4xx_System_Private_Functions

5.156.1 Detailed Description

**

5.157 STM32F4xx_System_Includes

Define to prevent recursive inclusion.

Define to prevent recursive inclusion.

5.158 STM32F4xx_System_Exported_types

Variables

- uint32_t SystemCoreClock

5.158.1 Detailed Description

5.158.2 Variable Documentation

5.158.2.1 SystemCoreClock

```
uint32_t SystemCoreClock  [extern]
```

System Clock Frequency (Core Clock)

5.159 STM32F4xx_System_Exported_Constants

5.160 STM32F4xx_System_Exported_Macros

5.161 STM32F4xx_System_Exported_Functions

Functions

- void SystemInit (void)

  Setup the microcontroller system Initialize the Embedded Flash Interface, the PLL and update the SystemFrequency variable.
- void SystemCoreClockUpdate (void)

  Update SystemCoreClock variable according to Clock Register Values. The SystemCoreClock variable contains the core clock (HCLK), it can be used by the user application to setup the SysTick timer or configure other parameters.

5.161.1 Detailed Description

5.161.2 Function Documentation

5.161.2.1 SystemCoreClockUpdate()

```
void SystemCoreClockUpdate (
            void  )
```

Update SystemCoreClock variable according to Clock Register Values. The SystemCoreClock variable contains the core clock (HCLK), it can be used by the user application to setup the SysTick timer or configure other parameters.

Note

Each time the core clock (HCLK) changes, this function must be called to update SystemCoreClock variable value. Otherwise, any configuration based on this variable will be incorrect.

- The system frequency computed by this function is not the real frequency in the chip. It is calculated based on the predefined constant and the selected clock source:

- If SYSCLK source is HSI, SystemCoreClock will contain the HSI_VALUE(∗)
- If SYSCLK source is HSE, SystemCoreClock will contain the HSE_VALUE(∗∗)
- If SYSCLK source is PLL, SystemCoreClock will contain the HSE_VALUE(∗∗) or HSI_VALUE(∗) multiplied/divided by the PLL factors.

(∗) HSI_VALUE is a constant defined in stm32f4xx.h file (default value 16 MHz) but the real value may vary depending on the variations in voltage and temperature.

(∗∗) HSE_VALUE is a constant defined in stm32f4xx.h file (default value 25 MHz), user has to ensure that HSE_↩ VALUE is same as the real frequency of the crystal used. Otherwise, this function may have wrong result.

- The result of this function could be not correct when using fractional value for HSE crystal.

Parameters

None	

Return values

None	

5.161.2.2 SystemInit()

```
void SystemInit (
             void  )
```

Setup the microcontroller system Initialize the Embedded Flash Interface, the PLL and update the SystemFrequency variable.

Parameters

None	

Return values

None	

5.162 STM32F4xx_System_Private_Includes

**

**

5.163 STM32F4xx_System_Private_TypesDefinitions

**

**

**

5.164 STM32F4xx_System_Private_Defines

**

Macros

- #define VECT_TAB_OFFSET 0x00
- #define **PLL_M** 8
- #define **PLL_N** 336
- #define **PLL_P** 2
- #define **PLL_Q** 7

5.164.1 Detailed Description

**

**

5.164.2 Macro Definition Documentation

5.164.2.1 VECT_TAB_OFFSET

```
#define VECT_TAB_OFFSET 0x00
```

< Uncomment the following line if you need to use external SRAM mounted on STM324xG_EVAL board as data memory

< Uncomment the following line if you need to relocate your vector Table in Internal SRAM. Vector Table base offset field. This value must be a multiple of 0x200.

5.165 STM32F4xx_System_Private_Macros

5.166 STM32F4xx_System_Private_Variables

Variables

- uint32_t SystemCoreClock = 168000000
- __I uint8_t **AHBPrescTable** [16] = {0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 3, 4, 6, 7, 8, 9}

5.166.1 Detailed Description

5.166.2 Variable Documentation

5.166.2.1 SystemCoreClock

```
uint32_t SystemCoreClock = 168000000
```

System Clock Frequency (Core Clock)

5.167 STM32F4xx_System_Private_FunctionPrototypes

5.167.1 Detailed Description

5.168 STM32F4xx_System_Private_Functions

Functions

- void SystemInit (void)

  Setup the microcontroller system Initialize the Embedded Flash Interface, the PLL and update the SystemFrequency variable.
- void SystemCoreClockUpdate (void)

  Update SystemCoreClock variable according to Clock Register Values. The SystemCoreClock variable contains the core clock (HCLK), it can be used by the user application to setup the SysTick timer or configure other parameters.

5.168.1 Detailed Description

5.168.2 Function Documentation

5.168.2.1 SystemCoreClockUpdate()

```
void SystemCoreClockUpdate (
            void  )
```

Update SystemCoreClock variable according to Clock Register Values. The SystemCoreClock variable contains the core clock (HCLK), it can be used by the user application to setup the SysTick timer or configure other parameters.

Note

Each time the core clock (HCLK) changes, this function must be called to update SystemCoreClock variable value. Otherwise, any configuration based on this variable will be incorrect.

- The system frequency computed by this function is not the real frequency in the chip. It is calculated based on the predefined constant and the selected clock source:

- If SYSCLK source is HSI, SystemCoreClock will contain the HSI_VALUE(∗)
- If SYSCLK source is HSE, SystemCoreClock will contain the HSE_VALUE(∗∗)
- If SYSCLK source is PLL, SystemCoreClock will contain the HSE_VALUE(∗∗) or HSI_VALUE(∗) multiplied/divided by the PLL factors.

(∗) HSI_VALUE is a constant defined in stm32f4xx.h file (default value 16 MHz) but the real value may vary depending on the variations in voltage and temperature.

(∗∗) HSE_VALUE is a constant defined in stm32f4xx.h file (default value 25 MHz), user has to ensure that HSE_↩ VALUE is same as the real frequency of the crystal used. Otherwise, this function may have wrong result.

- The result of this function could be not correct when using fractional value for HSE crystal.

Parameters

None	

Return values

None	

5.168.2.2 SystemInit()

```
void SystemInit (
             void  )
```

Setup the microcontroller system Initialize the Embedded Flash Interface, the PLL and update the SystemFrequency variable.

Parameters

None	

Return values

None	

Chapter 6

Data Structure Documentation

6.1 ADC_Common_TypeDef Struct Reference

Data Fields

- __IO uint32_t CSR
- __IO uint32_t CCR
- __IO uint32_t CDR

6.1.1 Field Documentation

6.1.1.1 CCR

```
__IO uint32_t CCR
```

ADC common control register, Address offset: ADC1 base address + 0x304

6.1.1.2 CDR

```
__IO uint32_t CDR
```

ADC common regular data register for dual AND triple modes, Address offset: ADC1 base address + 0x308

6.1.1.3 CSR

```
__IO uint32_t CSR
```

ADC Common status register, Address offset: ADC1 base address + 0x300

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.2 ADC_TypeDef Struct Reference

Analog to Digital Converter

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t SR
- __IO uint32_t CR1
- __IO uint32_t CR2
- __IO uint32_t SMPR1
- __IO uint32_t SMPR2
- __IO uint32_t JOFR1
- __IO uint32_t JOFR2
- __IO uint32_t JOFR3
- __IO uint32_t JOFR4
- __IO uint32_t HTR
- __IO uint32_t LTR
- __IO uint32_t SQR1
- __IO uint32_t SQR2
- __IO uint32_t SQR3
- __IO uint32_t JSQR
- __IO uint32_t JDR1
- __IO uint32_t JDR2
- __IO uint32_t JDR3
- __IO uint32_t JDR4
- __IO uint32_t DR

6.2.1 Detailed Description

Analog to Digital Converter

6.2.2 Field Documentation

6.2.2.1 CR1

```
__IO uint32_t CR1
```

ADC control register 1, Address offset: 0x04

6.2.2.2 CR2

```
__IO uint32_t CR2
```

ADC control register 2, Address offset: 0x08

6.2.2.3 DR

```
__IO uint32_t DR
```

ADC regular data register, Address offset: 0x4C

6.2.2.4 HTR

```
__IO uint32_t HTR
```

ADC watchdog higher threshold register, Address offset: 0x24

6.2.2.5 JDR1

```
__IO uint32_t JDR1
```

ADC injected data register 1, Address offset: 0x3C

6.2.2.6 JDR2

```
__IO uint32_t JDR2
```

ADC injected data register 2, Address offset: 0x40

6.2.2.7 JDR3

```
__IO uint32_t JDR3
```

ADC injected data register 3, Address offset: 0x44

6.2.2.8 JDR4

```
__IO uint32_t JDR4
```

ADC injected data register 4, Address offset: 0x48

6.2.2.9 JOFR1

```
__IO uint32_t JOFR1
```

ADC injected channel data offset register 1, Address offset: 0x14

6.2.2.10 JOFR2

```
__IO uint32_t JOFR2
```

ADC injected channel data offset register 2, Address offset: 0x18

6.2.2.11 JOFR3

```
__IO uint32_t JOFR3
```

ADC injected channel data offset register 3, Address offset: 0x1C

6.2.2.12 JOFR4

```
__IO uint32_t JOFR4
```

ADC injected channel data offset register 4, Address offset: 0x20

6.2.2.13 JSQR

```
__IO uint32_t JSQR
```

ADC injected sequence register, Address offset: 0x38

6.2.2.14 LTR

```
__IO uint32_t LTR
```

ADC watchdog lower threshold register, Address offset: 0x28

6.2.2.15 SMPR1

```
__IO uint32_t SMPR1
```

ADC sample time register 1, Address offset: 0x0C

6.2.2.16 SMPR2

```
__IO uint32_t SMPR2
```

ADC sample time register 2, Address offset: 0x10

6.2.2.17 SQR1

```
__IO uint32_t SQR1
```

ADC regular sequence register 1, Address offset: 0x2C

6.2.2.18 SQR2

```
__IO uint32_t SQR2
```

ADC regular sequence register 2, Address offset: 0x30

6.2.2.19 SQR3

```
__IO uint32_t SQR3
```

ADC regular sequence register 3, Address offset: 0x34

6.2.2.20 SR

```
__IO uint32_t SR
```

ADC status register, Address offset: 0x00

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.3 APSR_Type Union Reference

Union type to access the Application Program Status Register (APSR).

```
#include <core_cm4.h>
```

Data Fields

- struct {
  uint32_t _reserved0:16
  uint32_t GE:4
  uint32_t _reserved1:7
  uint32_t Q:1
  uint32_t V:1
  uint32_t C:1
  uint32_t Z:1
  uint32_t N:1
  } b
- uint32_t w

6.3.1 Detailed Description

Union type to access the Application Program Status Register (APSR).

The documentation for this union was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/core_cm4.h

6.4 CAN_FIFOMailBox_TypeDef Struct Reference

Controller Area Network FIFOMailBox.

`#include <stm32f4xx.h>`

Data Fields

- __IO uint32_t RIR
- __IO uint32_t RDTR
- __IO uint32_t RDLR
- __IO uint32_t RDHR

6.4.1 Detailed Description

Controller Area Network FIFOMailBox.

6.4.2 Field Documentation

6.4.2.1 RDHR

`__IO uint32_t RDHR`

CAN receive FIFO mailbox data high register

6.4.2.2 RDLR

`__IO uint32_t RDLR`

CAN receive FIFO mailbox data low register

6.4.2.3 RDTR

`__IO uint32_t RDTR`

CAN receive FIFO mailbox data length control and time stamp register

6.4.2.4 RIR

`__IO uint32_t RIR`

CAN receive FIFO mailbox identifier register

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.5 CAN_FilterRegister_TypeDef Struct Reference

Controller Area Network FilterRegister.

`#include <stm32f4xx.h>`

Data Fields

- __IO uint32_t FR1
- __IO uint32_t FR2

6.5.1 Detailed Description

Controller Area Network FilterRegister.

6.5.2 Field Documentation

6.5.2.1 FR1

`__IO uint32_t FR1`

CAN Filter bank register 1

6.5.2.2 FR2

`__IO uint32_t FR2`

CAN Filter bank register 1

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.6 CAN_TxMailBox_TypeDef Struct Reference

Controller Area Network TxMailBox.

`#include <stm32f4xx.h>`

Data Fields

- __IO uint32_t TIR
- __IO uint32_t TDTR
- __IO uint32_t TDLR
- __IO uint32_t TDHR

6.6.1 Detailed Description

Controller Area Network TxMailBox.

6.6.2 Field Documentation

6.6.2.1 TDHR

```
__IO uint32_t TDHR
```

CAN mailbox data high register

6.6.2.2 TDLR

```
__IO uint32_t TDLR
```

CAN mailbox data low register

6.6.2.3 TDTR

```
__IO uint32_t TDTR
```

CAN mailbox data length control and time stamp register

6.6.2.4 TIR

```
__IO uint32_t TIR
```

CAN TX mailbox identifier register

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.7 CAN_TypeDef Struct Reference

Controller Area Network.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t MCR
- __IO uint32_t MSR
- __IO uint32_t TSR
- __IO uint32_t RF0R
- __IO uint32_t RF1R
- __IO uint32_t IER
- __IO uint32_t ESR
- __IO uint32_t BTR
- uint32_t RESERVED0 [88]
- CAN_TxMailBox_TypeDef sTxMailBox [3]
- CAN_FIFOMailBox_TypeDef sFIFOMailBox [2]
- uint32_t RESERVED1 [12]
- __IO uint32_t FMR
- __IO uint32_t FM1R
- uint32_t RESERVED2
- __IO uint32_t FS1R
- uint32_t RESERVED3
- __IO uint32_t FFA1R
- uint32_t RESERVED4
- __IO uint32_t FA1R
- uint32_t RESERVED5 [8]
- CAN_FilterRegister_TypeDef sFilterRegister [28]

6.7.1 Detailed Description

Controller Area Network.

6.7.2 Field Documentation

6.7.2.1 BTR

```
__IO uint32_t BTR
```

CAN bit timing register, Address offset: 0x1C

6.7.2.2 ESR

```
__IO uint32_t ESR
```

CAN error status register, Address offset: 0x18

6.7.2.3 FA1R

```
__IO uint32_t FA1R
```

CAN filter activation register, Address offset: 0x21C

6.7.2.4 FFA1R

```
__IO uint32_t FFA1R
```

CAN filter FIFO assignment register, Address offset: 0x214

6.7.2.5 FM1R

```
__IO uint32_t FM1R
```

CAN filter mode register, Address offset: 0x204

6.7.2.6 FMR

```
__IO uint32_t FMR
```

CAN filter master register, Address offset: 0x200

6.7.2.7 FS1R

```
__IO uint32_t FS1R
```

CAN filter scale register, Address offset: 0x20C

6.7.2.8 IER

```
__IO uint32_t IER
```

CAN interrupt enable register, Address offset: 0x14

6.7.2.9 MCR

```
__IO uint32_t MCR
```

CAN master control register, Address offset: 0x00

6.7.2.10 MSR

```
__IO uint32_t MSR
```

CAN master status register, Address offset: 0x04

6.7.2.11 RESERVED0

```
uint32_t RESERVED0[88]
```

Reserved, 0x020 - 0x17F

6.7.2.12 RESERVED1

```
uint32_t RESERVED1[12]
```

Reserved, 0x1D0 - 0x1FF

6.7.2.13 RESERVED2

```
uint32_t RESERVED2
```

Reserved, 0x208

6.7.2.14 RESERVED3

```
uint32_t RESERVED3
```

Reserved, 0x210

6.7.2.15 RESERVED4

```
uint32_t RESERVED4
```

Reserved, 0x218

6.7.2.16 RESERVED5

```
uint32_t RESERVED5[8]
```

Reserved, 0x220-0x23F

6.7.2.17 RF0R

```
__IO uint32_t RF0R
```

CAN receive FIFO 0 register, Address offset: 0x0C

6.7.2.18 RF1R

```
__IO uint32_t RF1R
```

CAN receive FIFO 1 register, Address offset: 0x10

6.7.2.19 sFIFOMailBox

```
CAN_FIFOMailBox_TypeDef sFIFOMailBox[2]
```

CAN FIFO MailBox, Address offset: 0x1B0 - 0x1CC

6.7.2.20 sFilterRegister

```
CAN_FilterRegister_TypeDef sFilterRegister[28]
```

CAN Filter Register, Address offset: 0x240-0x31C

6.7.2.21 sTxMailBox

```
CAN_TxMailBox_TypeDef sTxMailBox[3]
```

CAN Tx MailBox, Address offset: 0x180 - 0x1AC

6.7.2.22 TSR

```
__IO uint32_t TSR
```

CAN transmit status register, Address offset: 0x08

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.8 CONTROL_Type Union Reference

Union type to access the Control Registers (CONTROL).

```
#include <core_cm4.h>
```

Data Fields

- struct {
  uint32_t nPRIV:1
  uint32_t SPSEL:1
  uint32_t FPCA:1
  uint32_t _reserved0:29
  } b
- uint32_t w

6.8.1 Detailed Description

Union type to access the Control Registers (CONTROL).

The documentation for this union was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/core_cm4.h

6.9 CoreDebug_Type Struct Reference

Structure type to access the Core Debug Register (CoreDebug).

```
#include <core_cm4.h>
```

Data Fields

- __IO uint32_t DHCSR
- __O uint32_t DCRSR
- __IO uint32_t DCRDR
- __IO uint32_t DEMCR

6.9.1 Detailed Description

Structure type to access the Core Debug Register (CoreDebug).

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/core_cm4.h

6.10 CRC_TypeDef Struct Reference

CRC calculation unit.

`#include <stm32f4xx.h>`

Data Fields

- __IO uint32_t DR
- __IO uint8_t IDR
- uint8_t RESERVED0
- uint16_t RESERVED1
- __IO uint32_t CR

6.10.1 Detailed Description

CRC calculation unit.

6.10.2 Field Documentation

6.10.2.1 CR

`__IO uint32_t CR`

CRC Control register, Address offset: 0x08

6.10.2.2 DR

`__IO uint32_t DR`

CRC Data register, Address offset: 0x00

6.10.2.3 IDR

`__IO uint8_t IDR`

CRC Independent data register, Address offset: 0x04

6.10.2.4 RESERVED0

`uint8_t RESERVED0`

Reserved, 0x05

6.10.2.5 RESERVED1

```
uint16_t RESERVED1
```

Reserved, 0x06

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.11 CRYP_TypeDef Struct Reference

Crypto Processor.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t CR
- __IO uint32_t SR
- __IO uint32_t DR
- __IO uint32_t DOUT
- __IO uint32_t DMACR
- __IO uint32_t IMSCR
- __IO uint32_t RISR
- __IO uint32_t MISR
- __IO uint32_t K0LR
- __IO uint32_t K0RR
- __IO uint32_t K1LR
- __IO uint32_t K1RR
- __IO uint32_t K2LR
- __IO uint32_t K2RR
- __IO uint32_t K3LR
- __IO uint32_t K3RR
- __IO uint32_t IV0LR
- __IO uint32_t IV0RR
- __IO uint32_t IV1LR
- __IO uint32_t IV1RR

6.11.1 Detailed Description

Crypto Processor.

6.11.2 Field Documentation

6.11.2.1 CR

```
__IO uint32_t CR
```

CRYP control register, Address offset: 0x00

6.11.2.2 DMACR

```
__IO uint32_t DMACR
```

CRYP DMA control register, Address offset: 0x10

6.11.2.3 DOUT

```
__IO uint32_t DOUT
```

CRYP data output register, Address offset: 0x0C

6.11.2.4 DR

```
__IO uint32_t DR
```

CRYP data input register, Address offset: 0x08

6.11.2.5 IMSCR

```
__IO uint32_t IMSCR
```

CRYP interrupt mask set/clear register, Address offset: 0x14

6.11.2.6 IV0LR

```
__IO uint32_t IV0LR
```

CRYP initialization vector left-word register 0, Address offset: 0x40

6.11.2.7 IV0RR

```
__IO uint32_t IV0RR
```

CRYP initialization vector right-word register 0, Address offset: 0x44

6.11.2.8 IV1LR

```
__IO uint32_t IV1LR
```

CRYP initialization vector left-word register 1, Address offset: 0x48

6.11.2.9 IV1RR

```
__IO uint32_t IV1RR
```

CRYP initialization vector right-word register 1, Address offset: 0x4C

6.11.2.10 K0LR

```
__IO uint32_t K0LR
```

CRYP key left register 0, Address offset: 0x20

6.11.2.11 K0RR

```
__IO uint32_t K0RR
```

CRYP key right register 0, Address offset: 0x24

6.11.2.12 K1LR

```
__IO uint32_t K1LR
```

CRYP key left register 1, Address offset: 0x28

6.11.2.13 K1RR

```
__IO uint32_t K1RR
```

CRYP key right register 1, Address offset: 0x2C

6.11.2.14 K2LR

```
__IO uint32_t K2LR
```

CRYP key left register 2, Address offset: 0x30

6.11.2.15 K2RR

```
__IO uint32_t K2RR
```

CRYP key right register 2, Address offset: 0x34

6.11.2.16 K3LR

```
__IO uint32_t K3LR
```

CRYP key left register 3, Address offset: 0x38

6.11.2.17 K3RR

```
__IO uint32_t K3RR
```

CRYP key right register 3, Address offset: 0x3C

6.11.2.18 MISR

```
__IO uint32_t MISR
```

CRYP masked interrupt status register, Address offset: 0x1C

6.11.2.19 RISR

```
__IO uint32_t RISR
```

CRYP raw interrupt status register, Address offset: 0x18

6.11.2.20 SR

```
__IO uint32_t SR
```

CRYP status register, Address offset: 0x04

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.12 DAC_TypeDef Struct Reference

Digital to Analog Converter.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t CR
- __IO uint32_t SWTRIGR
- __IO uint32_t DHR12R1
- __IO uint32_t DHR12L1
- __IO uint32_t DHR8R1
- __IO uint32_t DHR12R2
- __IO uint32_t DHR12L2
- __IO uint32_t DHR8R2
- __IO uint32_t DHR12RD
- __IO uint32_t DHR12LD
- __IO uint32_t DHR8RD
- __IO uint32_t DOR1
- __IO uint32_t DOR2
- __IO uint32_t SR

6.12.1 Detailed Description

Digital to Analog Converter.

6.12.2 Field Documentation

6.12.2.1 CR

```
__IO uint32_t CR
```

DAC control register, Address offset: 0x00

6.12.2.2 DHR12L1

```
__IO uint32_t DHR12L1
```

DAC channel1 12-bit left aligned data holding register, Address offset: 0x0C

6.12.2.3 DHR12L2

```
__IO uint32_t DHR12L2
```

DAC channel2 12-bit left aligned data holding register, Address offset: 0x18

6.12.2.4 DHR12LD

```
__IO uint32_t DHR12LD
```

DUAL DAC 12-bit left aligned data holding register, Address offset: 0x24

6.12.2.5 DHR12R1

```
__IO uint32_t DHR12R1
```

DAC channel1 12-bit right-aligned data holding register, Address offset: 0x08

6.12.2.6 DHR12R2

```
__IO uint32_t DHR12R2
```

DAC channel2 12-bit right aligned data holding register, Address offset: 0x14

6.12.2.7 DHR12RD

```
__IO uint32_t DHR12RD
```

Dual DAC 12-bit right-aligned data holding register, Address offset: 0x20

6.12.2.8 DHR8R1

```
__IO uint32_t DHR8R1
```

DAC channel1 8-bit right aligned data holding register, Address offset: 0x10

6.12.2.9 DHR8R2

```
__IO uint32_t DHR8R2
```

DAC channel2 8-bit right-aligned data holding register, Address offset: 0x1C

6.12.2.10 DHR8RD

```
__IO uint32_t DHR8RD
```

DUAL DAC 8-bit right aligned data holding register, Address offset: 0x28

6.12.2.11 DOR1

```
__IO uint32_t DOR1
```

DAC channel1 data output register, Address offset: 0x2C

6.12.2.12 DOR2

```
__IO uint32_t DOR2
```

DAC channel2 data output register, Address offset: 0x30

6.12.2.13 SR

```
__IO uint32_t SR
```

DAC status register, Address offset: 0x34

6.12.2.14 SWTRIGR

```
__IO uint32_t SWTRIGR
```

DAC software trigger register, Address offset: 0x04

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.13 DBGMCU_TypeDef Struct Reference

Debug MCU.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t IDCODE
- __IO uint32_t CR
- __IO uint32_t APB1FZ
- __IO uint32_t APB2FZ

6.13.1 Detailed Description

Debug MCU.

6.13.2 Field Documentation

6.13.2.1 APB1FZ

```
__IO uint32_t APB1FZ
```

Debug MCU APB1 freeze register, Address offset: 0x08

6.13.2.2 APB2FZ

```
__IO uint32_t APB2FZ
```

Debug MCU APB2 freeze register, Address offset: 0x0C

6.13.2.3 CR

```
__IO uint32_t CR
```

Debug MCU configuration register, Address offset: 0x04

6.13.2.4 IDCODE

```
__IO uint32_t IDCODE
```

MCU device ID code, Address offset: 0x00

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.14 DCMI_TypeDef Struct Reference

DCMI.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t CR
- __IO uint32_t SR
- __IO uint32_t RISR
- __IO uint32_t IER
- __IO uint32_t MISR
- __IO uint32_t ICR
- __IO uint32_t ESCR
- __IO uint32_t ESUR
- __IO uint32_t CWSTRTR
- __IO uint32_t CWSIZER
- __IO uint32_t DR

6.14.1 Detailed Description

DCMI.

6.14.2 Field Documentation

6.14.2.1 CR

```
__IO uint32_t CR
```

DCMI control register 1, Address offset: 0x00

6.14.2.2 CWSIZER

```
__IO uint32_t CWSIZER
```

DCMI crop window size, Address offset: 0x24

6.14.2.3 CWSTRTR

```
__IO uint32_t CWSTRTR
```

DCMI crop window start, Address offset: 0x20

6.14.2.4 DR

```
__IO uint32_t DR
```

DCMI data register, Address offset: 0x28

6.14.2.5 ESCR

```
__IO uint32_t ESCR
```

DCMI embedded synchronization code register, Address offset: 0x18

6.14.2.6 ESUR

```
__IO uint32_t ESUR
```

DCMI embedded synchronization unmask register, Address offset: 0x1C

6.14.2.7 ICR

```
__IO uint32_t ICR
```

DCMI interrupt clear register, Address offset: 0x14

6.14.2.8 IER

```
__IO uint32_t IER
```

DCMI interrupt enable register, Address offset: 0x0C

6.14.2.9 MISR

```
__IO uint32_t MISR
```

DCMI masked interrupt status register, Address offset: 0x10

6.14.2.10 RISR

```
__IO uint32_t RISR
```

DCMI raw interrupt status register, Address offset: 0x08

6.14.2.11 SR

```
__IO uint32_t SR
```

DCMI status register, Address offset: 0x04

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.15 DMA_InitTypeDef Struct Reference

DMA Init structure definition.

```
#include <stm32f4xx_dma.h>
```

Data Fields

- uint32_t DMA_Channel
- uint32_t DMA_PeripheralBaseAddr
- uint32_t DMA_Memory0BaseAddr
- uint32_t DMA_DIR
- uint32_t DMA_BufferSize
- uint32_t DMA_PeripheralInc
- uint32_t DMA_MemoryInc
- uint32_t DMA_PeripheralDataSize
- uint32_t DMA_MemoryDataSize
- uint32_t DMA_Mode
- uint32_t DMA_Priority
- uint32_t DMA_FIFOMode
- uint32_t DMA_FIFOThreshold
- uint32_t DMA_MemoryBurst
- uint32_t DMA_PeripheralBurst

6.15.1 Detailed Description

DMA Init structure definition.

6.15.2 Field Documentation

6.15.2.1 DMA_BufferSize

```
uint32_t DMA_BufferSize
```

Specifies the buffer size, in data unit, of the specified Stream. The data unit is equal to the configuration set in DMA_PeripheralDataSize or DMA_MemoryDataSize members depending in the transfer direction.

6.15.2.2 DMA_Channel

```
uint32_t DMA_Channel
```

Specifies the channel used for the specified stream. This parameter can be a value of DMA_channel

6.15.2.3 DMA_DIR

```
uint32_t DMA_DIR
```

Specifies if the data will be transferred from memory to peripheral, from memory to memory or from peripheral to memory. This parameter can be a value of DMA_data_transfer_direction

6.15.2.4 DMA_FIFOMode

```
uint32_t DMA_FIFOMode
```

Specifies if the FIFO mode or Direct mode will be used for the specified Stream. This parameter can be a value of DMA_fifo_direct_mode

Note

The Direct mode (FIFO mode disabled) cannot be used if the memory-to-memory data transfer is configured on the selected Stream

6.15.2.5 DMA_FIFOThreshold

```
uint32_t DMA_FIFOThreshold
```

Specifies the FIFO threshold level. This parameter can be a value of DMA_fifo_threshold_level

6.15.2.6 DMA_Memory0BaseAddr

```
uint32_t DMA_Memory0BaseAddr
```

Specifies the memory 0 base address for DMAy Streamx. This memory is the default memory used when double buffer mode is not enabled.

6.15.2.7 DMA_MemoryBurst

```
uint32_t DMA_MemoryBurst
```

Specifies the Burst transfer configuration for the memory transfers. It specifies the amount of data to be transferred in a single non interruptable transaction. This parameter can be a value of DMA_memory_burst

Note

The burst mode is possible only if the address Increment mode is enabled.

6.15.2.8 DMA_MemoryDataSize

```
uint32_t DMA_MemoryDataSize
```

Specifies the Memory data width. This parameter can be a value of DMA_memory_data_size

6.15.2.9 DMA_MemoryInc

```
uint32_t DMA_MemoryInc
```

Specifies whether the memory address register should be incremented or not. This parameter can be a value of DMA_memory_incremented_mode

6.15.2.10 DMA_Mode

```
uint32_t DMA_Mode
```

Specifies the operation mode of the DMAy Streamx. This parameter can be a value of DMA_circular_normal_mode

Note

The circular buffer mode cannot be used if the memory-to-memory data transfer is configured on the selected Stream

6.15.2.11 DMA_PeripheralBaseAddr

```
uint32_t DMA_PeripheralBaseAddr
```

Specifies the peripheral base address for DMAy Streamx.

6.15.2.12 DMA_PeripheralBurst

```
uint32_t DMA_PeripheralBurst
```

Specifies the Burst transfer configuration for the peripheral transfers. It specifies the amount of data to be transferred in a single non interruptable transaction. This parameter can be a value of DMA_peripheral_burst

Note

The burst mode is possible only if the address Increment mode is enabled.

6.15.2.13 DMA_PeripheralDataSize

```
uint32_t DMA_PeripheralDataSize
```

Specifies the Peripheral data width. This parameter can be a value of DMA_peripheral_data_size

6.15.2.14 DMA_PeripheralInc

```
uint32_t DMA_PeripheralInc
```

Specifies whether the Peripheral address register should be incremented or not. This parameter can be a value of DMA_peripheral_incremented_mode

6.15.2.15 DMA_Priority

```
uint32_t DMA_Priority
```

Specifies the software priority for the DMAy Streamx. This parameter can be a value of DMA_priority_level

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx_dma.h

6.16 DMA_Stream_TypeDef Struct Reference

DMA Controller.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t CR
- __IO uint32_t NDTR
- __IO uint32_t PAR
- __IO uint32_t M0AR
- __IO uint32_t M1AR
- __IO uint32_t FCR

6.16.1 Detailed Description

DMA Controller.

6.16.2 Field Documentation

6.16.2.1 CR

```
__IO uint32_t CR
```

DMA stream x configuration register

6.16.2.2 FCR

```
__IO uint32_t FCR
```

DMA stream x FIFO control register

6.16.2.3 M0AR

```
__IO uint32_t M0AR
```

DMA stream x memory 0 address register

6.16.2.4 M1AR

```
__IO uint32_t M1AR
```

DMA stream x memory 1 address register

6.16.2.5 NDTR

```
__IO uint32_t NDTR
```

DMA stream x number of data register

6.16.2.6 PAR

```
__IO uint32_t PAR
```

DMA stream x peripheral address register

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.17 DMA_TypeDef Struct Reference

Data Fields

- __IO uint32_t LISR
- __IO uint32_t HISR
- __IO uint32_t LIFCR
- __IO uint32_t HIFCR

6.17.1 Field Documentation

6.17.1.1 HIFCR

```
__IO uint32_t HIFCR
```

DMA high interrupt flag clear register, Address offset: 0x0C

6.17.1.2 HISR

```
__IO uint32_t HISR
```

DMA high interrupt status register, Address offset: 0x04

6.17.1.3 LIFCR

```
__IO uint32_t LIFCR
```

DMA low interrupt flag clear register, Address offset: 0x08

6.17.1.4 LISR

```
__IO uint32_t LISR
```

DMA low interrupt status register, Address offset: 0x00

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.18 ETH_TypeDef Struct Reference

Ethernet MAC.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t **MACCR**
- __IO uint32_t **MACFFR**
- __IO uint32_t **MACHTHR**
- __IO uint32_t **MACHTLR**
- __IO uint32_t **MACMIIAR**
- __IO uint32_t **MACMIIDR**
- __IO uint32_t **MACFCR**
- __IO uint32_t **MACVLANTR**
- uint32_t **RESERVED0** [2]
- __IO uint32_t **MACRWUFFR**
- __IO uint32_t **MACPMTCSR**
- uint32_t **RESERVED1** [2]
- __IO uint32_t **MACSR**
- __IO uint32_t **MACIMR**
- __IO uint32_t **MACA0HR**
- __IO uint32_t **MACA0LR**
- __IO uint32_t **MACA1HR**
- __IO uint32_t **MACA1LR**
- __IO uint32_t **MACA2HR**
- __IO uint32_t **MACA2LR**
- __IO uint32_t **MACA3HR**
- __IO uint32_t **MACA3LR**
- uint32_t **RESERVED2** [40]
- __IO uint32_t **MMCCR**
- __IO uint32_t **MMCRIR**
- __IO uint32_t **MMCTIR**
- __IO uint32_t **MMCRIMR**
- __IO uint32_t **MMCTIMR**
- uint32_t **RESERVED3** [14]
- __IO uint32_t **MMCTGFSCCR**
- __IO uint32_t **MMCTGFMSCCR**
- uint32_t **RESERVED4** [5]
- __IO uint32_t **MMCTGFCR**

- uint32_t **RESERVED5** [10]
- __IO uint32_t **MMCRFCECR**
- __IO uint32_t **MMCRFAECR**
- uint32_t **RESERVED6** [10]
- __IO uint32_t **MMCRGUFCR**
- uint32_t **RESERVED7** [334]
- __IO uint32_t **PTPTSCR**
- __IO uint32_t **PTPSSIR**
- __IO uint32_t **PTPTSHR**
- __IO uint32_t **PTPTSLR**
- __IO uint32_t **PTPTSHUR**
- __IO uint32_t **PTPTSLUR**
- __IO uint32_t **PTPTSAR**
- __IO uint32_t **PTPTTHR**
- __IO uint32_t **PTPTTLR**
- __IO uint32_t **RESERVED8**
- __IO uint32_t **PTPTSSR**
- uint32_t **RESERVED9** [565]
- __IO uint32_t **DMABMR**
- __IO uint32_t **DMATPDR**
- __IO uint32_t **DMARPDR**
- __IO uint32_t **DMARDLAR**
- __IO uint32_t **DMATDLAR**
- __IO uint32_t **DMASR**
- __IO uint32_t **DMAOMR**
- __IO uint32_t **DMAIER**
- __IO uint32_t **DMAMFBOCR**
- __IO uint32_t **DMARSWTR**
- uint32_t **RESERVED10** [8]
- __IO uint32_t **DMACHTDR**
- __IO uint32_t **DMACHRDR**
- __IO uint32_t **DMACHTBAR**
- __IO uint32_t **DMACHRBAR**

6.18.1 Detailed Description

Ethernet MAC.

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.19 EXTI_TypeDef Struct Reference

External Interrupt/Event Controller.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t IMR
- __IO uint32_t EMR
- __IO uint32_t RTSR
- __IO uint32_t FTSR
- __IO uint32_t SWIER
- __IO uint32_t PR

6.19.1 Detailed Description

External Interrupt/Event Controller.

6.19.2 Field Documentation

6.19.2.1 EMR

```
__IO uint32_t EMR
```

EXTI Event mask register, Address offset: 0x04

6.19.2.2 FTSR

```
__IO uint32_t FTSR
```

EXTI Falling trigger selection register, Address offset: 0x0C

6.19.2.3 IMR

```
__IO uint32_t IMR
```

EXTI Interrupt mask register, Address offset: 0x00

6.19.2.4 PR

```
__IO uint32_t PR
```

EXTI Pending register, Address offset: 0x14

6.19.2.5 RTSR

```
__IO uint32_t RTSR
```

EXTI Rising trigger selection register, Address offset: 0x08

6.19.2.6 SWIER

```
__IO uint32_t SWIER
```

EXTI Software interrupt event register, Address offset: 0x10

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.20 FLASH_TypeDef Struct Reference

FLASH Registers.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t ACR
- __IO uint32_t KEYR
- __IO uint32_t OPTKEYR
- __IO uint32_t SR
- __IO uint32_t CR
- __IO uint32_t OPTCR

6.20.1 Detailed Description

FLASH Registers.

6.20.2 Field Documentation

6.20.2.1 ACR

```
__IO uint32_t ACR
```

FLASH access control register, Address offset: 0x00

6.20.2.2 CR

```
__IO uint32_t CR
```

FLASH control register, Address offset: 0x10

6.20.2.3 KEYR

```
__IO uint32_t KEYR
```

FLASH key register, Address offset: 0x04

6.20.2.4 OPTCR

```
__IO uint32_t OPTCR
```

FLASH option control register, Address offset: 0x14

6.20.2.5 OPTKEYR

```
__IO uint32_t OPTKEYR
```

FLASH option key register, Address offset: 0x08

6.20.2.6 SR

```
__IO uint32_t SR
```

FLASH status register, Address offset: 0x0C

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.21 FSMC_Bank1_TypeDef Struct Reference

Flexible Static Memory Controller.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t BTCR [8]

6.21.1 Detailed Description

Flexible Static Memory Controller.

6.21.2 Field Documentation

6.21.2.1 BTCR

```
__IO uint32_t BTCR[8]
```

NOR/PSRAM chip-select control register(BCR) and chip-select timing register(BTR), Address offset: 0x00-1C

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.22 FSMC_Bank1E_TypeDef Struct Reference

Flexible Static Memory Controller Bank1E.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t BWTR [7]

6.22.1 Detailed Description

Flexible Static Memory Controller Bank1E.

6.22.2 Field Documentation

6.22.2.1 BWTR

```
__IO uint32_t BWTR[7]
```

NOR/PSRAM write timing registers, Address offset: 0x104-0x11C

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.23 FSMC_Bank2_TypeDef Struct Reference

Flexible Static Memory Controller Bank2.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t PCR2
- __IO uint32_t SR2
- __IO uint32_t PMEM2
- __IO uint32_t PATT2
- uint32_t RESERVED0
- __IO uint32_t ECCR2

6.23.1 Detailed Description

Flexible Static Memory Controller Bank2.

6.23.2 Field Documentation

6.23.2.1 ECCR2

```
__IO uint32_t ECCR2
```

NAND Flash ECC result registers 2, Address offset: 0x74

6.23.2.2 PATT2

```
__IO uint32_t PATT2
```

NAND Flash Attribute memory space timing register 2, Address offset: 0x6C

6.23.2.3 PCR2

```
__IO uint32_t PCR2
```

NAND Flash control register 2, Address offset: 0x60

6.23.2.4 PMEM2

```
__IO uint32_t PMEM2
```

NAND Flash Common memory space timing register 2, Address offset: 0x68

6.23.2.5 RESERVED0

```
uint32_t RESERVED0
```

Reserved, 0x70

6.23.2.6 SR2

```
__IO uint32_t SR2
```

NAND Flash FIFO status and interrupt register 2, Address offset: 0x64

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.24 FSMC_Bank3_TypeDef Struct Reference

Flexible Static Memory Controller Bank3.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t PCR3
- __IO uint32_t SR3
- __IO uint32_t PMEM3
- __IO uint32_t PATT3
- uint32_t RESERVED0
- __IO uint32_t ECCR3

6.24.1 Detailed Description

Flexible Static Memory Controller Bank3.

6.24.2 Field Documentation

6.24.2.1 ECCR3

```
__IO uint32_t ECCR3
```

NAND Flash ECC result registers 3, Address offset: 0x94

6.24.2.2 PATT3

```
__IO uint32_t PATT3
```

NAND Flash Attribute memory space timing register 3, Address offset: 0x8C

6.24.2.3 PCR3

```
__IO uint32_t PCR3
```

NAND Flash control register 3, Address offset: 0x80

6.24.2.4 PMEM3

```
__IO uint32_t PMEM3
```

NAND Flash Common memory space timing register 3, Address offset: 0x88

6.24.2.5 RESERVED0

```
uint32_t RESERVED0
```

Reserved, 0x90

6.24.2.6 SR3

```
__IO uint32_t SR3
```

NAND Flash FIFO status and interrupt register 3, Address offset: 0x84

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.25 FSMC_Bank4_TypeDef Struct Reference

Flexible Static Memory Controller Bank4.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t PCR4
- __IO uint32_t SR4
- __IO uint32_t PMEM4
- __IO uint32_t PATT4
- __IO uint32_t PIO4

6.25.1 Detailed Description

Flexible Static Memory Controller Bank4.

6.25.2 Field Documentation

6.25.2.1 PATT4

```
__IO uint32_t PATT4
```

PC Card Attribute memory space timing register 4, Address offset: 0xAC

6.25.2.2 PCR4

```
__IO uint32_t PCR4
```

PC Card control register 4, Address offset: 0xA0

6.25.2.3 PIO4

```
__IO uint32_t PIO4
```

PC Card I/O space timing register 4, Address offset: 0xB0

6.25.2.4 PMEM4

```
__IO uint32_t PMEM4
```

PC Card Common memory space timing register 4, Address offset: 0xA8

6.25.2.5 SR4

```
__IO uint32_t SR4
```

PC Card FIFO status and interrupt register 4, Address offset: 0xA4

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.26 GPIO_InitTypeDef Struct Reference

GPIO Init structure definition

```
#include <stm32f4xx_gpio.h>
```

Data Fields

- uint32_t GPIO_Pin
- GPIOMode_TypeDef GPIO_Mode
- GPIOSpeed_TypeDef GPIO_Speed
- GPIOOType_TypeDef GPIO_OType
- GPIOPuPd_TypeDef GPIO_PuPd

6.26.1 Detailed Description

GPIO Init structure definition

6.26.2 Field Documentation

6.26.2.1 GPIO_Mode

```
GPIOMode_TypeDef GPIO_Mode
```

Specifies the operating mode for the selected pins. This parameter can be a value of GPIOMode_TypeDef

6.26.2.2 GPIO_OType

```
GPIOOType_TypeDef GPIO_OType
```

Specifies the operating output type for the selected pins. This parameter can be a value of GPIOOType_TypeDef

6.26.2.3 GPIO_Pin

```
uint32_t GPIO_Pin
```

Specifies the GPIO pins to be configured. This parameter can be any value of GPIO_pins_define

6.26.2.4 GPIO_PuPd

```
GPIOPuPd_TypeDef GPIO_PuPd
```

Specifies the operating Pull-up/Pull down for the selected pins. This parameter can be a value of GPIOPuPd_TypeDef

6.26.2.5 GPIO_Speed

```
GPIOSpeed_TypeDef GPIO_Speed
```

Specifies the speed for the selected pins. This parameter can be a value of GPIOSpeed_TypeDef

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx_gpio.h

6.27 GPIO_TypeDef Struct Reference

General Purpose I/O.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t MODER
- __IO uint32_t OTYPER
- __IO uint32_t OSPEEDR
- __IO uint32_t PUPDR
- __IO uint32_t IDR
- __IO uint32_t ODR
- __IO uint16_t BSRRL
- __IO uint16_t BSRRH
- __IO uint32_t LCKR
- __IO uint32_t AFR [2]

6.27.1 Detailed Description

General Purpose I/O.

6.27.2 Field Documentation

6.27.2.1 AFR

```
__IO uint32_t AFR[2]
```

GPIO alternate function registers, Address offset: 0x20-0x24

6.27.2.2 BSRRH

```
__IO uint16_t BSRRH
```

GPIO port bit set/reset high register, Address offset: 0x1A

6.27.2.3 BSRRL

```
__IO uint16_t BSRRL
```

GPIO port bit set/reset low register, Address offset: 0x18

6.27.2.4 IDR

```
__IO uint32_t IDR
```

GPIO port input data register, Address offset: 0x10

6.27.2.5 LCKR

```
__IO uint32_t LCKR
```

GPIO port configuration lock register, Address offset: 0x1C

6.27.2.6 MODER

```
__IO uint32_t MODER
```

GPIO port mode register, Address offset: 0x00

6.27.2.7 ODR

```
__IO uint32_t ODR
```

GPIO port output data register, Address offset: 0x14

6.27.2.8 OSPEEDR

```
__IO uint32_t OSPEEDR
```

GPIO port output speed register, Address offset: 0x08

6.27.2.9 OTYPER

```
__IO uint32_t OTYPER
```

GPIO port output type register, Address offset: 0x04

6.27.2.10 PUPDR

```
__IO uint32_t PUPDR
```

GPIO port pull-up/pull-down register, Address offset: 0x0C

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.28 HASH_TypeDef Struct Reference

HASH.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t CR
- __IO uint32_t DIN
- __IO uint32_t STR
- __IO uint32_t HR [5]
- __IO uint32_t IMR
- __IO uint32_t SR
- uint32_t RESERVED [52]
- __IO uint32_t CSR [51]

6.28.1 Detailed Description

HASH.

6.28.2 Field Documentation

6.28.2.1 CR

```
__IO uint32_t CR
```

HASH control register, Address offset: 0x00

6.28.2.2 CSR

```
__IO uint32_t CSR[51]
```

HASH context swap registers, Address offset: 0x0F8-0x1C0

6.28.2.3 DIN

```
__IO uint32_t DIN
```

HASH data input register, Address offset: 0x04

6.28.2.4 HR

```
__IO uint32_t HR[5]
```

HASH digest registers, Address offset: 0x0C-0x1C

6.28.2.5 IMR

```
__IO uint32_t IMR
```

HASH interrupt enable register, Address offset: 0x20

6.28.2.6 RESERVED

```
uint32_t RESERVED[52]
```

Reserved, 0x28-0xF4

6.28.2.7 SR

```
__IO uint32_t SR
```

HASH status register, Address offset: 0x24

6.28.2.8 STR

```
__IO uint32_t STR
```

HASH start register, Address offset: 0x08

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.29 I2C_TypeDef Struct Reference

Inter-integrated Circuit Interface.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint16_t CR1
- uint16_t RESERVED0
- __IO uint16_t CR2
- uint16_t RESERVED1
- __IO uint16_t OAR1
- uint16_t RESERVED2
- __IO uint16_t OAR2
- uint16_t RESERVED3
- __IO uint16_t DR
- uint16_t RESERVED4
- __IO uint16_t SR1
- uint16_t RESERVED5
- __IO uint16_t SR2
- uint16_t RESERVED6
- __IO uint16_t CCR
- uint16_t RESERVED7
- __IO uint16_t TRISE
- uint16_t RESERVED8

6.29.1 Detailed Description

Inter-integrated Circuit Interface.

6.29.2 Field Documentation

6.29.2.1 CCR

```
__IO uint16_t CCR
```

I2C Clock control register, Address offset: 0x1C

6.29.2.2 CR1

```
__IO uint16_t CR1
```

I2C Control register 1, Address offset: 0x00

6.29.2.3 CR2

```
__IO uint16_t CR2
```

I2C Control register 2, Address offset: 0x04

6.29.2.4 DR

```
__IO uint16_t DR
```

I2C Data register, Address offset: 0x10

6.29.2.5 OAR1

```
__IO uint16_t OAR1
```

I2C Own address register 1, Address offset: 0x08

6.29.2.6 OAR2

```
__IO uint16_t OAR2
```

I2C Own address register 2, Address offset: 0x0C

6.29.2.7 RESERVED0

```
uint16_t RESERVED0
```

Reserved, 0x02

6.29.2.8 RESERVED1

```
uint16_t RESERVED1
```

Reserved, 0x06

6.29.2.9 RESERVED2

```
uint16_t RESERVED2
```

Reserved, 0x0A

6.29.2.10 RESERVED3

```
uint16_t RESERVED3
```

Reserved, 0x0E

6.29.2.11 RESERVED4

```
uint16_t RESERVED4
```

Reserved, 0x12

6.29.2.12 RESERVED5

```
uint16_t RESERVED5
```

Reserved, 0x16

6.29.2.13 RESERVED6

```
uint16_t RESERVED6
```

Reserved, 0x1A

6.29.2.14 RESERVED7

```
uint16_t RESERVED7
```

Reserved, 0x1E

6.29.2.15 RESERVED8

```
uint16_t RESERVED8
```

Reserved, 0x22

6.29.2.16 SR1

```
__IO uint16_t SR1
```

I2C Status register 1, Address offset: 0x14

6.29.2.17 SR2

```
__IO uint16_t SR2
```

I2C Status register 2, Address offset: 0x18

6.29.2.18 TRISE

```
__IO uint16_t TRISE
```

I2C TRISE register, Address offset: 0x20

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.30 IPSR_Type Union Reference

Union type to access the Interrupt Program Status Register (IPSR).

```
#include <core_cm4.h>
```

Data Fields

- struct {
  uint32_t ISR:9
  uint32_t _reserved0:23
  } b
- uint32_t w

6.30.1 Detailed Description

Union type to access the Interrupt Program Status Register (IPSR).

The documentation for this union was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/core_cm4.h

6.31 ITM_Type Struct Reference

Structure type to access the Instrumentation Trace Macrocell Register (ITM).

```
#include <core_cm4.h>
```

Data Fields

- union {
  __O uint8_t u8
  __O uint16_t u16
  __O uint32_t u32
  } PORT [32]
- uint32_t **RESERVED0** [864]
- __IO uint32_t TER
- uint32_t **RESERVED1** [15]
- __IO uint32_t TPR
- uint32_t **RESERVED2** [15]
- __IO uint32_t TCR

6.31.1 Detailed Description

Structure type to access the Instrumentation Trace Macrocell Register (ITM).

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/core_cm4.h

6.32 IWDG_TypeDef Struct Reference

Independent WATCHDOG.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t KR
- __IO uint32_t PR
- __IO uint32_t RLR
- __IO uint32_t SR

6.32.1 Detailed Description

Independent WATCHDOG.

6.32.2 Field Documentation

6.32.2.1 KR

```
__IO uint32_t KR
```

IWDG Key register, Address offset: 0x00

6.32.2.2 PR

```
__IO uint32_t PR
```

IWDG Prescaler register, Address offset: 0x04

6.32.2.3 RLR

```
__IO uint32_t RLR
```

IWDG Reload register, Address offset: 0x08

6.32.2.4 SR

```
__IO uint32_t SR
```

IWDG Status register, Address offset: 0x0C

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.33 NVIC_InitTypeDef Struct Reference

NVIC Init Structure definition

```
#include <misc.h>
```

Data Fields

- uint8_t NVIC_IRQChannel
- uint8_t NVIC_IRQChannelPreemptionPriority
- uint8_t NVIC_IRQChannelSubPriority
- FunctionalState NVIC_IRQChannelCmd

6.33.1 Detailed Description

NVIC Init Structure definition

6.33.2 Field Documentation

6.33.2.1 NVIC_IRQChannel

```
uint8_t NVIC_IRQChannel
```

Specifies the IRQ channel to be enabled or disabled. This parameter can be an enumerator of IRQn_Type enumeration (For the complete STM32 Devices IRQ Channels list, please refer to stm32f4xx.h file)

6.33.2.2 NVIC_IRQChannelCmd

```
FunctionalState NVIC_IRQChannelCmd
```

Specifies whether the IRQ channel defined in NVIC_IRQChannel will be enabled or disabled. This parameter can be set either to ENABLE or DISABLE

6.33.2.3 NVIC_IRQChannelPreemptionPriority

```
uint8_t NVIC_IRQChannelPreemptionPriority
```

Specifies the pre-emption priority for the IRQ channel specified in NVIC_IRQChannel. This parameter can be a value between 0 and 15 as described in the table MISC_NVIC_Priority_Table A lower priority value indicates a higher priority

6.33.2.4 NVIC_IRQChannelSubPriority

```
uint8_t NVIC_IRQChannelSubPriority
```

Specifies the subpriority level for the IRQ channel specified in NVIC_IRQChannel. This parameter can be a value between 0 and 15 as described in the table MISC_NVIC_Priority_Table A lower priority value indicates a higher priority

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/misc.h

6.34 NVIC_Type Struct Reference

Structure type to access the Nested Vectored Interrupt Controller (NVIC).

```
#include <core_cm4.h>
```

Data Fields

- __IO uint32_t ISER [8]
- uint32_t **RESERVED0** [24]
- __IO uint32_t ICER [8]
- uint32_t **RSERVED1** [24]
- __IO uint32_t ISPR [8]
- uint32_t **RESERVED2** [24]
- __IO uint32_t ICPR [8]
- uint32_t **RESERVED3** [24]
- __IO uint32_t IABR [8]
- uint32_t **RESERVED4** [56]
- __IO uint8_t IP [240]
- uint32_t **RESERVED5** [644]
- __O uint32_t STIR

6.34.1 Detailed Description

Structure type to access the Nested Vectored Interrupt Controller (NVIC).

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/core_cm4.h

6.35 PARSE_STORAGE Struct Reference

Data Fields

- int **number_store** [3]
- int **color**
- int **loop_iterator**
- int **var_store** [6]
- int **err_code**

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/logic_layer.h

6.36 PWR_TypeDef Struct Reference

Power Control.

`#include <stm32f4xx.h>`

Data Fields

- __IO uint32_t CR
- __IO uint32_t CSR

6.36.1 Detailed Description

Power Control.

6.36.2 Field Documentation

6.36.2.1 CR

```
__IO uint32_t CR
```

PWR power control register, Address offset: 0x00

6.36.2.2 CSR

```
__IO uint32_t CSR
```

PWR power control/status register, Address offset: 0x04

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.37 RCC_ClocksTypeDef Struct Reference

Data Fields

- uint32_t SYSCLK_Frequency
- uint32_t HCLK_Frequency
- uint32_t PCLK1_Frequency
- uint32_t PCLK2_Frequency

6.37.1 Field Documentation

6.37.1.1 HCLK_Frequency

```
uint32_t HCLK_Frequency
```

HCLK clock frequency expressed in Hz

6.37.1.2 PCLK1_Frequency

```
uint32_t PCLK1_Frequency
```

PCLK1 clock frequency expressed in Hz

6.37.1.3 PCLK2_Frequency

```
uint32_t PCLK2_Frequency
```

PCLK2 clock frequency expressed in Hz

6.37.1.4 SYSCLK_Frequency

```
uint32_t SYSCLK_Frequency
```

SYSCLK clock frequency expressed in Hz

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx_rcc.h

6.38 RCC_TypeDef Struct Reference

Reset and Clock Control.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t CR
- __IO uint32_t PLLCFGR
- __IO uint32_t CFGR
- __IO uint32_t CIR
- __IO uint32_t AHB1RSTR
- __IO uint32_t AHB2RSTR
- __IO uint32_t AHB3RSTR
- uint32_t RESERVED0
- __IO uint32_t APB1RSTR
- __IO uint32_t APB2RSTR
- uint32_t RESERVED1 [2]
- __IO uint32_t AHB1ENR
- __IO uint32_t AHB2ENR
- __IO uint32_t AHB3ENR
- uint32_t RESERVED2
- __IO uint32_t APB1ENR
- __IO uint32_t APB2ENR
- uint32_t RESERVED3 [2]
- __IO uint32_t AHB1LPENR
- __IO uint32_t AHB2LPENR
- __IO uint32_t AHB3LPENR
- uint32_t RESERVED4
- __IO uint32_t APB1LPENR
- __IO uint32_t APB2LPENR
- uint32_t RESERVED5 [2]
- __IO uint32_t BDCR
- __IO uint32_t CSR
- uint32_t RESERVED6 [2]
- __IO uint32_t SSCGR
- __IO uint32_t PLLI2SCFGR

6.38.1 Detailed Description

Reset and Clock Control.

6.38.2 Field Documentation

6.38.2.1 AHB1ENR

```
__IO uint32_t AHB1ENR
```

RCC AHB1 peripheral clock register, Address offset: 0x30

6.38.2.2 AHB1LPENR

```
__IO uint32_t AHB1LPENR
```

RCC AHB1 peripheral clock enable in low power mode register, Address offset: 0x50

6.38.2.3 AHB1RSTR

```
__IO uint32_t AHB1RSTR
```

RCC AHB1 peripheral reset register, Address offset: 0x10

6.38.2.4 AHB2ENR

```
__IO uint32_t AHB2ENR
```

RCC AHB2 peripheral clock register, Address offset: 0x34

6.38.2.5 AHB2LPENR

```
__IO uint32_t AHB2LPENR
```

RCC AHB2 peripheral clock enable in low power mode register, Address offset: 0x54

6.38.2.6 AHB2RSTR

```
__IO uint32_t AHB2RSTR
```

RCC AHB2 peripheral reset register, Address offset: 0x14

6.38.2.7 AHB3ENR

```
__IO uint32_t AHB3ENR
```

RCC AHB3 peripheral clock register, Address offset: 0x38

6.38.2.8 AHB3LPENR

```
__IO uint32_t AHB3LPENR
```

RCC AHB3 peripheral clock enable in low power mode register, Address offset: 0x58

6.38.2.9 AHB3RSTR

```
__IO uint32_t AHB3RSTR
```

RCC AHB3 peripheral reset register, Address offset: 0x18

6.38.2.10 APB1ENR

```
__IO uint32_t APB1ENR
```

RCC APB1 peripheral clock enable register, Address offset: 0x40

6.38.2.11 APB1LPENR

```
__IO uint32_t APB1LPENR
```

RCC APB1 peripheral clock enable in low power mode register, Address offset: 0x60

6.38.2.12 APB1RSTR

```
__IO uint32_t APB1RSTR
```

RCC APB1 peripheral reset register, Address offset: 0x20

6.38.2.13 APB2ENR

```
__IO uint32_t APB2ENR
```

RCC APB2 peripheral clock enable register, Address offset: 0x44

6.38.2.14 APB2LPENR

```
__IO uint32_t APB2LPENR
```

RCC APB2 peripheral clock enable in low power mode register, Address offset: 0x64

6.38.2.15 APB2RSTR

```
__IO uint32_t APB2RSTR
```

RCC APB2 peripheral reset register, Address offset: 0x24

6.38.2.16 BDCR

```
__IO uint32_t BDCR
```

RCC Backup domain control register, Address offset: 0x70

6.38.2.17 CFGR

```
__IO uint32_t CFGR
```

RCC clock configuration register, Address offset: 0x08

6.38.2.18 CIR

```
__IO uint32_t CIR
```

RCC clock interrupt register, Address offset: 0x0C

6.38.2.19 CR

```
__IO uint32_t CR
```

RCC clock control register, Address offset: 0x00

6.38.2.20 CSR

```
__IO uint32_t CSR
```

RCC clock control & status register, Address offset: 0x74

6.38.2.21 PLLCFGR

```
__IO uint32_t PLLCFGR
```

RCC PLL configuration register, Address offset: 0x04

6.38.2.22 PLLI2SCFGR

```
__IO uint32_t PLLI2SCFGR
```

RCC PLLI2S configuration register, Address offset: 0x84

6.38.2.23 RESERVED0

```
uint32_t RESERVED0
```

Reserved, 0x1C

6.38.2.24 RESERVED1

```
uint32_t RESERVED1[2]
```

Reserved, 0x28-0x2C

6.38.2.25 RESERVED2

```
uint32_t RESERVED2
```

Reserved, 0x3C

6.38.2.26 RESERVED3

```
uint32_t RESERVED3[2]
```

Reserved, 0x48-0x4C

6.38.2.27 RESERVED4

```
uint32_t RESERVED4
```

Reserved, 0x5C

6.38.2.28 RESERVED5

```
uint32_t RESERVED5[2]
```

Reserved, 0x68-0x6C

6.38.2.29 RESERVED6

```
uint32_t RESERVED6[2]
```

Reserved, 0x78-0x7C

6.38.2.30 SSCGR

```
__IO uint32_t SSCGR
```

RCC spread spectrum clock generation register, Address offset: 0x80

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.39 RNG_TypeDef Struct Reference

HASH.

`#include <stm32f4xx.h>`

Data Fields

- __IO uint32_t CR
- __IO uint32_t SR
- __IO uint32_t DR

6.39.1 Detailed Description

HASH.

6.39.2 Field Documentation

6.39.2.1 CR

`__IO uint32_t CR`

RNG control register, Address offset: 0x00

6.39.2.2 DR

`__IO uint32_t DR`

RNG data register, Address offset: 0x08

6.39.2.3 SR

`__IO uint32_t SR`

RNG status register, Address offset: 0x04

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.40 RTC_TypeDef Struct Reference

Real-Time Clock.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t TR
- __IO uint32_t DR
- __IO uint32_t CR
- __IO uint32_t ISR
- __IO uint32_t PRER
- __IO uint32_t WUTR
- __IO uint32_t CALIBR
- __IO uint32_t ALRMAR
- __IO uint32_t ALRMBR
- __IO uint32_t WPR
- __IO uint32_t SSR
- __IO uint32_t SHIFTR
- __IO uint32_t TSTR
- __IO uint32_t TSDR
- __IO uint32_t TSSSR
- __IO uint32_t CALR
- __IO uint32_t TAFCR
- __IO uint32_t ALRMASSR
- __IO uint32_t ALRMBSSR
- uint32_t RESERVED7
- __IO uint32_t BKP0R
- __IO uint32_t BKP1R
- __IO uint32_t BKP2R
- __IO uint32_t BKP3R
- __IO uint32_t BKP4R
- __IO uint32_t BKP5R
- __IO uint32_t BKP6R
- __IO uint32_t BKP7R
- __IO uint32_t BKP8R
- __IO uint32_t BKP9R
- __IO uint32_t BKP10R
- __IO uint32_t BKP11R
- __IO uint32_t BKP12R
- __IO uint32_t BKP13R
- __IO uint32_t BKP14R
- __IO uint32_t BKP15R
- __IO uint32_t BKP16R
- __IO uint32_t BKP17R
- __IO uint32_t BKP18R
- __IO uint32_t BKP19R

6.40.1 Detailed Description

Real-Time Clock.

6.40.2 Field Documentation

6.40.2.1 ALRMAR

```
__IO uint32_t ALRMAR
```

RTC alarm A register, Address offset: 0x1C

6.40.2.2 ALRMASSR

```
__IO uint32_t ALRMASSR
```

RTC alarm A sub second register, Address offset: 0x44

6.40.2.3 ALRMBR

```
__IO uint32_t ALRMBR
```

RTC alarm B register, Address offset: 0x20

6.40.2.4 ALRMBSSR

```
__IO uint32_t ALRMBSSR
```

RTC alarm B sub second register, Address offset: 0x48

6.40.2.5 BKP0R

```
__IO uint32_t BKP0R
```

RTC backup register 1, Address offset: 0x50

6.40.2.6 BKP10R

```
__IO uint32_t BKP10R
```

RTC backup register 10, Address offset: 0x78

6.40.2.7 BKP11R

```
__IO uint32_t BKP11R
```

RTC backup register 11, Address offset: 0x7C

6.40.2.8 BKP12R

```
__IO uint32_t BKP12R
```

RTC backup register 12, Address offset: 0x80

6.40.2.9 BKP13R

```
__IO uint32_t BKP13R
```

RTC backup register 13, Address offset: 0x84

6.40.2.10 BKP14R

```
__IO uint32_t BKP14R
```

RTC backup register 14, Address offset: 0x88

6.40.2.11 BKP15R

```
__IO uint32_t BKP15R
```

RTC backup register 15, Address offset: 0x8C

6.40.2.12 BKP16R

```
__IO uint32_t BKP16R
```

RTC backup register 16, Address offset: 0x90

6.40.2.13 BKP17R

```
__IO uint32_t BKP17R
```

RTC backup register 17, Address offset: 0x94

6.40.2.14 BKP18R

```
__IO uint32_t BKP18R
```

RTC backup register 18, Address offset: 0x98

6.40.2.15 BKP19R

```
__IO uint32_t BKP19R
```

RTC backup register 19, Address offset: 0x9C

6.40.2.16 BKP1R

```
__IO uint32_t BKP1R
```

RTC backup register 1, Address offset: 0x54

6.40.2.17 BKP2R

```
__IO uint32_t BKP2R
```

RTC backup register 2, Address offset: 0x58

6.40.2.18 BKP3R

```
__IO uint32_t BKP3R
```

RTC backup register 3, Address offset: 0x5C

6.40.2.19 BKP4R

```
__IO uint32_t BKP4R
```

RTC backup register 4, Address offset: 0x60

6.40.2.20 BKP5R

```
__IO uint32_t BKP5R
```

RTC backup register 5, Address offset: 0x64

6.40.2.21 BKP6R

```
__IO uint32_t BKP6R
```

RTC backup register 6, Address offset: 0x68

6.40.2.22 BKP7R

```
__IO uint32_t BKP7R
```

RTC backup register 7, Address offset: 0x6C

6.40.2.23 BKP8R

```
__IO uint32_t BKP8R
```

RTC backup register 8, Address offset: 0x70

6.40.2.24 BKP9R

```
__IO uint32_t BKP9R
```

RTC backup register 9, Address offset: 0x74

6.40.2.25 CALIBR

```
__IO uint32_t CALIBR
```

RTC calibration register, Address offset: 0x18

6.40.2.26 CALR

```
__IO uint32_t CALR
```

RTC calibration register, Address offset: 0x3C

6.40.2.27 CR

```
__IO uint32_t CR
```

RTC control register, Address offset: 0x08

6.40.2.28 DR

```
__IO uint32_t DR
```

RTC date register, Address offset: 0x04

6.40.2.29 ISR

```
__IO uint32_t ISR
```

RTC initialization and status register, Address offset: 0x0C

6.40.2.30 PRER

```
__IO uint32_t PRER
```

RTC prescaler register, Address offset: 0x10

6.40.2.31 RESERVED7

```
uint32_t RESERVED7
```

Reserved, 0x4C

6.40.2.32 SHIFTR

```
__IO uint32_t SHIFTR
```

RTC shift control register, Address offset: 0x2C

6.40.2.33 SSR

```
__IO uint32_t SSR
```

RTC sub second register, Address offset: 0x28

6.40.2.34 TAFCR

```
__IO uint32_t TAFCR
```

RTC tamper and alternate function configuration register, Address offset: 0x40

6.40.2.35 TR

```
__IO uint32_t TR
```

RTC time register, Address offset: 0x00

6.40.2.36 TSDR

```
__IO uint32_t TSDR
```

RTC time stamp date register, Address offset: 0x34

6.40.2.37 TSSSR

```
__IO uint32_t TSSSR
```

RTC time-stamp sub second register, Address offset: 0x38

6.40.2.38 TSTR

```
__IO uint32_t TSTR
```

RTC time stamp time register, Address offset: 0x30

6.40.2.39 WPR

```
__IO uint32_t WPR
```

RTC write protection register, Address offset: 0x24

6.40.2.40 WUTR

```
__IO uint32_t WUTR
```

RTC wakeup timer register, Address offset: 0x14

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.41 SCB_Type Struct Reference

Structure type to access the System Control Block (SCB).

```
#include <core_cm4.h>
```

Data Fields

- __I uint32_t CPUID
- __IO uint32_t ICSR
- __IO uint32_t VTOR
- __IO uint32_t AIRCR
- __IO uint32_t SCR
- __IO uint32_t CCR
- __IO uint8_t SHP [12]
- __IO uint32_t SHCSR
- __IO uint32_t CFSR
- __IO uint32_t HFSR
- __IO uint32_t DFSR
- __IO uint32_t MMFAR
- __IO uint32_t BFAR
- __IO uint32_t AFSR
- __I uint32_t PFR [2]
- __I uint32_t DFR
- __I uint32_t ADR
- __I uint32_t MMFR [4]
- __I uint32_t ISAR [5]
- uint32_t **RESERVED0** [5]
- __IO uint32_t CPACR

6.41.1 Detailed Description

Structure type to access the System Control Block (SCB).

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/core_cm4.h

6.42 SCnSCB_Type Struct Reference

Structure type to access the System Control and ID Register not in the SCB.

`#include <core_cm4.h>`

Data Fields

- uint32_t **RESERVED0** [1]
- __I uint32_t ICTR
- __IO uint32_t ACTLR

6.42.1 Detailed Description

Structure type to access the System Control and ID Register not in the SCB.

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/core_cm4.h

6.43 SDIO_TypeDef Struct Reference

SD host Interface.

`#include <stm32f4xx.h>`

Data Fields

- __IO uint32_t POWER
- __IO uint32_t CLKCR
- __IO uint32_t ARG
- __IO uint32_t CMD
- __I uint32_t RESPCMD
- __I uint32_t RESP1
- __I uint32_t RESP2
- __I uint32_t RESP3
- __I uint32_t RESP4
- __IO uint32_t DTIMER
- __IO uint32_t DLEN
- __IO uint32_t DCTRL
- __I uint32_t DCOUNT
- __I uint32_t STA
- __IO uint32_t ICR
- __IO uint32_t MASK
- uint32_t RESERVED0 [2]
- __I uint32_t FIFOCNT
- uint32_t RESERVED1 [13]
- __IO uint32_t FIFO

6.43.1 Detailed Description

SD host Interface.

6.43.2 Field Documentation

6.43.2.1 ARG

```
__IO uint32_t ARG
```

SDIO argument register, Address offset: 0x08

6.43.2.2 CLKCR

```
__IO uint32_t CLKCR
```

SDI clock control register, Address offset: 0x04

6.43.2.3 CMD

```
__IO uint32_t CMD
```

SDIO command register, Address offset: 0x0C

6.43.2.4 DCOUNT

```
__I uint32_t DCOUNT
```

SDIO data counter register, Address offset: 0x30

6.43.2.5 DCTRL

```
__IO uint32_t DCTRL
```

SDIO data control register, Address offset: 0x2C

6.43.2.6 DLEN

```
__IO uint32_t DLEN
```

SDIO data length register, Address offset: 0x28

6.43.2.7 DTIMER

```
__IO uint32_t DTIMER
```

SDIO data timer register, Address offset: 0x24

6.43.2.8 FIFO

```
__IO uint32_t FIFO
```

SDIO data FIFO register, Address offset: 0x80

6.43.2.9 FIFOCNT

```
__I uint32_t FIFOCNT
```

SDIO FIFO counter register, Address offset: 0x48

6.43.2.10 ICR

```
__IO uint32_t ICR
```

SDIO interrupt clear register, Address offset: 0x38

6.43.2.11 MASK

```
__IO uint32_t MASK
```

SDIO mask register, Address offset: 0x3C

6.43.2.12 POWER

```
__IO uint32_t POWER
```

SDIO power control register, Address offset: 0x00

6.43.2.13 RESERVED0

```
uint32_t RESERVED0[2]
```

Reserved, 0x40-0x44

6.43.2.14 RESERVED1

```
uint32_t RESERVED1[13]
```

Reserved, 0x4C-0x7C

6.43.2.15 RESP1

```
__I uint32_t RESP1
```

SDIO response 1 register, Address offset: 0x14

6.43.2.16 RESP2

```
__I uint32_t RESP2
```

SDIO response 2 register, Address offset: 0x18

6.43.2.17 RESP3

```
__I uint32_t RESP3
```

SDIO response 3 register, Address offset: 0x1C

6.43.2.18 RESP4

```
__I uint32_t RESP4
```

SDIO response 4 register, Address offset: 0x20

6.43.2.19 RESPCMD

```
__I uint32_t RESPCMD
```

SDIO command response register, Address offset: 0x10

6.43.2.20 STA

```
__I uint32_t STA
```

SDIO status register, Address offset: 0x34

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.44 SPI_TypeDef Struct Reference

Serial Peripheral Interface.

`#include <stm32f4xx.h>`

Data Fields

- __IO uint16_t CR1
- uint16_t RESERVED0
- __IO uint16_t CR2
- uint16_t RESERVED1
- __IO uint16_t SR
- uint16_t RESERVED2
- __IO uint16_t DR
- uint16_t RESERVED3
- __IO uint16_t CRCPR
- uint16_t RESERVED4
- __IO uint16_t RXCRCR
- uint16_t RESERVED5
- __IO uint16_t TXCRCR
- uint16_t RESERVED6
- __IO uint16_t I2SCFGR
- uint16_t RESERVED7
- __IO uint16_t I2SPR
- uint16_t RESERVED8

6.44.1 Detailed Description

Serial Peripheral Interface.

6.44.2 Field Documentation

6.44.2.1 CR1

`__IO uint16_t CR1`

SPI control register 1 (not used in I2S mode), Address offset: 0x00

6.44.2.2 CR2

`__IO uint16_t CR2`

SPI control register 2, Address offset: 0x04

6.44.2.3 CRCPR

```
__IO uint16_t CRCPR
```

SPI CRC polynomial register (not used in I2S mode), Address offset: 0x10

6.44.2.4 DR

```
__IO uint16_t DR
```

SPI data register, Address offset: 0x0C

6.44.2.5 I2SCFGR

```
__IO uint16_t I2SCFGR
```

SPI_I2S configuration register, Address offset: 0x1C

6.44.2.6 I2SPR

```
__IO uint16_t I2SPR
```

SPI_I2S prescaler register, Address offset: 0x20

6.44.2.7 RESERVED0

```
uint16_t RESERVED0
```

Reserved, 0x02

6.44.2.8 RESERVED1

```
uint16_t RESERVED1
```

Reserved, 0x06

6.44.2.9 RESERVED2

```
uint16_t RESERVED2
```

Reserved, 0x0A

6.44.2.10 RESERVED3

```
uint16_t RESERVED3
```

Reserved, 0x0E

6.44.2.11 RESERVED4

```
uint16_t RESERVED4
```

Reserved, 0x12

6.44.2.12 RESERVED5

```
uint16_t RESERVED5
```

Reserved, 0x16

6.44.2.13 RESERVED6

```
uint16_t RESERVED6
```

Reserved, 0x1A

6.44.2.14 RESERVED7

```
uint16_t RESERVED7
```

Reserved, 0x1E

6.44.2.15 RESERVED8

```
uint16_t RESERVED8
```

Reserved, 0x22

6.44.2.16 RXCRCR

```
__IO uint16_t RXCRCR
```

SPI RX CRC register (not used in I2S mode), Address offset: 0x14

6.44.2.17 SR

```
__IO uint16_t SR
```

SPI status register, Address offset: 0x08

6.44.2.18 TXCRCR

```
__IO uint16_t TXCRCR
```

SPI TX CRC register (not used in I2S mode), Address offset: 0x18

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.45 SYSCFG_TypeDef Struct Reference

System configuration controller.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t MEMRMP
- __IO uint32_t PMC
- __IO uint32_t EXTICR [4]
- uint32_t RESERVED [2]
- __IO uint32_t CMPCR

6.45.1 Detailed Description

System configuration controller.

6.45.2 Field Documentation

6.45.2.1 CMPCR

```
__IO uint32_t CMPCR
```

SYSCFG Compensation cell control register, Address offset: 0x20

6.45.2.2 EXTICR

```
__IO uint32_t EXTICR[4]
```

SYSCFG external interrupt configuration registers, Address offset: 0x08-0x14

6.45.2.3 MEMRMP

```
__IO uint32_t MEMRMP
```

SYSCFG memory remap register, Address offset: 0x00

6.45.2.4 PMC

```
__IO uint32_t PMC
```

SYSCFG peripheral mode configuration register, Address offset: 0x04

6.45.2.5 RESERVED

```
uint32_t RESERVED[2]
```

Reserved, 0x18-0x1C

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.46 SysTick_Type Struct Reference

Structure type to access the System Timer (SysTick).

```
#include <core_cm4.h>
```

Data Fields

- __IO uint32_t CTRL
- __IO uint32_t LOAD
- __IO uint32_t VAL
- __I uint32_t CALIB

6.46.1 Detailed Description

Structure type to access the System Timer (SysTick).

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/core_cm4.h

6.47 TIM_BDTRInitTypeDef Struct Reference

BDTR structure definition.

`#include <stm32f4xx_tim.h>`

Data Fields

- uint16_t TIM_OSSRState
- uint16_t TIM_OSSIState
- uint16_t TIM_LOCKLevel
- uint16_t TIM_DeadTime
- uint16_t TIM_Break
- uint16_t TIM_BreakPolarity
- uint16_t TIM_AutomaticOutput

6.47.1 Detailed Description

BDTR structure definition.

Note

This structure is used only with TIM1 and TIM8.

6.47.2 Field Documentation

6.47.2.1 TIM_AutomaticOutput

`uint16_t TIM_AutomaticOutput`

Specifies whether the TIM Automatic Output feature is enabled or not. This parameter can be a value of TIM_AOE_Bit_Set_Reset

6.47.2.2 TIM_Break

```
uint16_t TIM_Break
```

Specifies whether the TIM Break input is enabled or not. This parameter can be a value of TIM_Break_Input_enable_disable

6.47.2.3 TIM_BreakPolarity

```
uint16_t TIM_BreakPolarity
```

Specifies the TIM Break Input pin polarity. This parameter can be a value of TIM_Break_Polarity

6.47.2.4 TIM_DeadTime

```
uint16_t TIM_DeadTime
```

Specifies the delay time between the switching-off and the switching-on of the outputs. This parameter can be a number between 0x00 and 0xFF

6.47.2.5 TIM_LOCKLevel

```
uint16_t TIM_LOCKLevel
```

Specifies the LOCK level parameters. This parameter can be a value of TIM_Lock_level

6.47.2.6 TIM_OSSIState

```
uint16_t TIM_OSSIState
```

Specifies the Off-State used in Idle state. This parameter can be a value of TIM_OSSI_Off_State_Selection_for_Idle_mode_state

6.47.2.7 TIM_OSSRState

```
uint16_t TIM_OSSRState
```

Specifies the Off-State selection used in Run mode. This parameter can be a value of TIM_OSSR_Off_State_Selection_for_Run_mode

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx_tim.h

6.48 TIM_ICInitTypeDef Struct Reference

TIM Input Capture Init structure definition

```
#include <stm32f4xx_tim.h>
```

Data Fields

- uint16_t TIM_Channel
- uint16_t TIM_ICPolarity
- uint16_t TIM_ICSelection
- uint16_t TIM_ICPrescaler
- uint16_t TIM_ICFilter

6.48.1 Detailed Description

TIM Input Capture Init structure definition

6.48.2 Field Documentation

6.48.2.1 TIM_Channel

```
uint16_t TIM_Channel
```

Specifies the TIM channel. This parameter can be a value of TIM_Channel

6.48.2.2 TIM_ICFilter

```
uint16_t TIM_ICFilter
```

Specifies the input capture filter. This parameter can be a number between 0x0 and 0xF

6.48.2.3 TIM_ICPolarity

```
uint16_t TIM_ICPolarity
```

Specifies the active edge of the input signal. This parameter can be a value of TIM_Input_Capture_Polarity

6.48.2.4 TIM_ICPrescaler

```
uint16_t TIM_ICPrescaler
```

Specifies the Input Capture Prescaler. This parameter can be a value of TIM_Input_Capture_Prescaler

6.48.2.5 TIM_ICSelection

```
uint16_t TIM_ICSelection
```

Specifies the input. This parameter can be a value of TIM_Input_Capture_Selection

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx_tim.h

6.49 TIM_OCInitTypeDef Struct Reference

TIM Output Compare Init structure definition

```
#include <stm32f4xx_tim.h>
```

Data Fields

- uint16_t TIM_OCMode
- uint16_t TIM_OutputState
- uint16_t TIM_OutputNState
- uint32_t TIM_Pulse
- uint16_t TIM_OCPolarity
- uint16_t TIM_OCNPolarity
- uint16_t TIM_OCIdleState
- uint16_t TIM_OCNIdleState

6.49.1 Detailed Description

TIM Output Compare Init structure definition

6.49.2 Field Documentation

6.49.2.1 TIM_OCIdleState

```
uint16_t TIM_OCIdleState
```

Specifies the TIM Output Compare pin state during Idle state. This parameter can be a value of TIM_Output_Compare_Idle_State

Note

This parameter is valid only for TIM1 and TIM8.

6.49.2.2 TIM_OCMode

```
uint16_t TIM_OCMode
```

Specifies the TIM mode. This parameter can be a value of TIM_Output_Compare_and_PWM_modes

6.49.2.3 TIM_OCNIdleState

```
uint16_t TIM_OCNIdleState
```

Specifies the TIM Output Compare pin state during Idle state. This parameter can be a value of TIM_Output_Compare_N_Idle_State

Note

This parameter is valid only for TIM1 and TIM8.

6.49.2.4 TIM_OCNPolarity

```
uint16_t TIM_OCNPolarity
```

Specifies the complementary output polarity. This parameter can be a value of TIM_Output_Compare_N_Polarity

Note

This parameter is valid only for TIM1 and TIM8.

6.49.2.5 TIM_OCPolarity

```
uint16_t TIM_OCPolarity
```

Specifies the output polarity. This parameter can be a value of TIM_Output_Compare_Polarity

6.49.2.6 TIM_OutputNState

```
uint16_t TIM_OutputNState
```

Specifies the TIM complementary Output Compare state. This parameter can be a value of TIM_Output_Compare_N_State

Note

This parameter is valid only for TIM1 and TIM8.

6.49.2.7 TIM_OutputState

```
uint16_t TIM_OutputState
```

Specifies the TIM Output Compare state. This parameter can be a value of TIM_Output_Compare_State

6.49.2.8 TIM_Pulse

```
uint32_t TIM_Pulse
```

Specifies the pulse value to be loaded into the Capture Compare Register. This parameter can be a number between 0x0000 and 0xFFFF

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx_tim.h

6.50 TIM_TimeBaseInitTypeDef Struct Reference

TIM Time Base Init structure definition

```
#include <stm32f4xx_tim.h>
```

Data Fields

- uint16_t TIM_Prescaler
- uint16_t TIM_CounterMode
- uint32_t TIM_Period
- uint16_t TIM_ClockDivision
- uint8_t TIM_RepetitionCounter

6.50.1 Detailed Description

TIM Time Base Init structure definition

Note

This structure is used with all TIMx except for TIM6 and TIM7.

6.50.2 Field Documentation

6.50.2.1 TIM_ClockDivision

```
uint16_t TIM_ClockDivision
```

Specifies the clock division. This parameter can be a value of TIM_Clock_Division_CKD

6.50.2.2 TIM_CounterMode

```
uint16_t TIM_CounterMode
```

Specifies the counter mode. This parameter can be a value of TIM_Counter_Mode

6.50.2.3 TIM_Period

```
uint32_t TIM_Period
```

Specifies the period value to be loaded into the active Auto-Reload Register at the next update event. This parameter must be a number between 0x0000 and 0xFFFF.

6.50.2.4 TIM_Prescaler

```
uint16_t TIM_Prescaler
```

Specifies the prescaler value used to divide the TIM clock. This parameter can be a number between 0x0000 and 0xFFFF

6.50.2.5 TIM_RepetitionCounter

```
uint8_t TIM_RepetitionCounter
```

Specifies the repetition counter value. Each time the RCR downcounter reaches zero, an update event is generated and counting restarts from the RCR value (N). This means in PWM mode that (N+1) corresponds to:

- the number of PWM periods in edge-aligned mode
- the number of half PWM period in center-aligned mode This parameter must be a number between 0x00 and 0xFF.

  Note

  This parameter is valid only for TIM1 and TIM8.

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx_tim.h

6.51 TIM_TypeDef Struct Reference

TIM.

`#include <stm32f4xx.h>`

Data Fields

- __IO uint16_t CR1
- uint16_t RESERVED0
- __IO uint16_t CR2
- uint16_t RESERVED1
- __IO uint16_t SMCR
- uint16_t RESERVED2
- __IO uint16_t DIER
- uint16_t RESERVED3
- __IO uint16_t SR
- uint16_t RESERVED4
- __IO uint16_t EGR
- uint16_t RESERVED5
- __IO uint16_t CCMR1
- uint16_t RESERVED6
- __IO uint16_t CCMR2
- uint16_t RESERVED7
- __IO uint16_t CCER
- uint16_t RESERVED8
- __IO uint32_t CNT
- __IO uint16_t PSC
- uint16_t RESERVED9
- __IO uint32_t ARR
- __IO uint16_t RCR
- uint16_t RESERVED10
- __IO uint32_t CCR1
- __IO uint32_t CCR2
- __IO uint32_t CCR3
- __IO uint32_t CCR4
- __IO uint16_t BDTR
- uint16_t RESERVED11
- __IO uint16_t DCR
- uint16_t RESERVED12
- __IO uint16_t DMAR
- uint16_t RESERVED13
- __IO uint16_t OR
- uint16_t RESERVED14

6.51.1 Detailed Description

TIM.

6.51.2 Field Documentation

6.51.2.1 ARR

```
__IO uint32_t ARR
```

TIM auto-reload register, Address offset: 0x2C

6.51.2.2 BDTR

```
__IO uint16_t BDTR
```

TIM break and dead-time register, Address offset: 0x44

6.51.2.3 CCER

```
__IO uint16_t CCER
```

TIM capture/compare enable register, Address offset: 0x20

6.51.2.4 CCMR1

```
__IO uint16_t CCMR1
```

TIM capture/compare mode register 1, Address offset: 0x18

6.51.2.5 CCMR2

```
__IO uint16_t CCMR2
```

TIM capture/compare mode register 2, Address offset: 0x1C

6.51.2.6 CCR1

```
__IO uint32_t CCR1
```

TIM capture/compare register 1, Address offset: 0x34

6.51.2.7 CCR2

```
__IO uint32_t CCR2
```

TIM capture/compare register 2, Address offset: 0x38

6.51.2.8 CCR3

```
__IO uint32_t CCR3
```

TIM capture/compare register 3, Address offset: 0x3C

6.51.2.9 CCR4

```
__IO uint32_t CCR4
```

TIM capture/compare register 4, Address offset: 0x40

6.51.2.10 CNT

```
__IO uint32_t CNT
```

TIM counter register, Address offset: 0x24

6.51.2.11 CR1

```
__IO uint16_t CR1
```

TIM control register 1, Address offset: 0x00

6.51.2.12 CR2

```
__IO uint16_t CR2
```

TIM control register 2, Address offset: 0x04

6.51.2.13 DCR

```
__IO uint16_t DCR
```

TIM DMA control register, Address offset: 0x48

6.51.2.14 DIER

```
__IO uint16_t DIER
```

TIM DMA/interrupt enable register, Address offset: 0x0C

6.51.2.15 DMAR

```
__IO uint16_t DMAR
```

TIM DMA address for full transfer, Address offset: 0x4C

6.51.2.16 EGR

```
__IO uint16_t EGR
```

TIM event generation register, Address offset: 0x14

6.51.2.17 OR

```
__IO uint16_t OR
```

TIM option register, Address offset: 0x50

6.51.2.18 PSC

```
__IO uint16_t PSC
```

TIM prescaler, Address offset: 0x28

6.51.2.19 RCR

```
__IO uint16_t RCR
```

TIM repetition counter register, Address offset: 0x30

6.51.2.20 RESERVED0

```
uint16_t RESERVED0
```

Reserved, 0x02

6.51.2.21 RESERVED1

```
uint16_t RESERVED1
```

Reserved, 0x06

6.51.2.22 RESERVED10

```
uint16_t RESERVED10
```

Reserved, 0x32

6.51.2.23 RESERVED11

```
uint16_t RESERVED11
```

Reserved, 0x46

6.51.2.24 RESERVED12

```
uint16_t RESERVED12
```

Reserved, 0x4A

6.51.2.25 RESERVED13

```
uint16_t RESERVED13
```

Reserved, 0x4E

6.51.2.26 RESERVED14

```
uint16_t RESERVED14
```

Reserved, 0x52

6.51.2.27 RESERVED2

```
uint16_t RESERVED2
```

Reserved, 0x0A

6.51.2.28 RESERVED3

```
uint16_t RESERVED3
```

Reserved, 0x0E

6.51.2.29 RESERVED4

```
uint16_t RESERVED4
```

Reserved, 0x12

6.51.2.30 RESERVED5

```
uint16_t RESERVED5
```

Reserved, 0x16

6.51.2.31 RESERVED6

```
uint16_t RESERVED6
```

Reserved, 0x1A

6.51.2.32 RESERVED7

```
uint16_t RESERVED7
```

Reserved, 0x1E

6.51.2.33 RESERVED8

```
uint16_t RESERVED8
```

Reserved, 0x22

6.51.2.34 RESERVED9

```
uint16_t RESERVED9
```

Reserved, 0x2A

6.51.2.35 SMCR

```
__IO uint16_t SMCR
```

TIM slave mode control register, Address offset: 0x08

6.51.2.36 SR

```
__IO uint16_t SR
```

TIM status register, Address offset: 0x10

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.52 UART_Communication Struct Reference

Data Fields

- uint8_t **receive** [STORAGE]
- uint8_t **buffer** [STORAGE]
- int **err_code**

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/UART_communication.h

6.53 USART_TypeDef Struct Reference

Universal Synchronous Asynchronous Receiver Transmitter.

`#include <stm32f4xx.h>`

Data Fields

- __IO uint16_t SR
- uint16_t RESERVED0
- __IO uint16_t DR
- uint16_t RESERVED1
- __IO uint16_t BRR
- uint16_t RESERVED2
- __IO uint16_t CR1
- uint16_t RESERVED3
- __IO uint16_t CR2
- uint16_t RESERVED4
- __IO uint16_t CR3
- uint16_t RESERVED5
- __IO uint16_t GTPR
- uint16_t RESERVED6

6.53.1 Detailed Description

Universal Synchronous Asynchronous Receiver Transmitter.

6.53.2 Field Documentation

6.53.2.1 BRR

```
__IO uint16_t BRR
```

USART Baud rate register, Address offset: 0x08

6.53.2.2 CR1

```
__IO uint16_t CR1
```

USART Control register 1, Address offset: 0x0C

6.53.2.3 CR2

```
__IO uint16_t CR2
```

USART Control register 2, Address offset: 0x10

6.53.2.4 CR3

```
__IO uint16_t CR3
```

USART Control register 3, Address offset: 0x14

6.53.2.5 DR

```
__IO uint16_t DR
```

USART Data register, Address offset: 0x04

6.53.2.6 GTPR

```
__IO uint16_t GTPR
```

USART Guard time and prescaler register, Address offset: 0x18

6.53.2.7 RESERVED0

```
uint16_t RESERVED0
```

Reserved, 0x02

6.53.2.8 RESERVED1

```
uint16_t RESERVED1
```

Reserved, 0x06

6.53.2.9 RESERVED2

```
uint16_t RESERVED2
```

Reserved, 0x0A

6.53.2.10 RESERVED3

```
uint16_t RESERVED3
```

Reserved, 0x0E

6.53.2.11 RESERVED4

```
uint16_t RESERVED4
```

Reserved, 0x12

6.53.2.12 RESERVED5

```
uint16_t RESERVED5
```

Reserved, 0x16

6.53.2.13 RESERVED6

```
uint16_t RESERVED6
```

Reserved, 0x1A

6.53.2.14 SR

```
__IO uint16_t SR
```

USART Status register, Address offset: 0x00

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.54 VGA_t Struct Reference

Data Fields

- uint16_t **hsync_cnt**
- uint32_t **start_adr**
- uint32_t **dma2_cr_reg**

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32_ub_vga_screen.h

6.55 WWDG_TypeDef Struct Reference

Window WATCHDOG.

```
#include <stm32f4xx.h>
```

Data Fields

- __IO uint32_t CR
- __IO uint32_t CFR
- __IO uint32_t SR

6.55.1 Detailed Description

Window WATCHDOG.

6.55.2 Field Documentation

6.55.2.1 CFR

```
__IO uint32_t CFR
```

WWDG Configuration register, Address offset: 0x04

6.55.2.2 CR

```
__IO uint32_t CR
```

WWDG Control register, Address offset: 0x00

6.55.2.3 SR

```
__IO uint32_t SR
```

WWDG Status register, Address offset: 0x08

The documentation for this struct was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/stm32f4xx.h

6.56 xPSR_Type Union Reference

Union type to access the Special-Purpose Program Status Registers (xPSR).

```
#include <core_cm4.h>
```

Data Fields

- struct {
  uint32_t ISR:9
  uint32_t _reserved0:7
  uint32_t GE:4
  uint32_t _reserved1:4
  uint32_t T:1
  uint32_t IT:2
  uint32_t Q:1
  uint32_t V:1
  uint32_t C:1
  uint32_t Z:1
  uint32_t N:1
  } b
- uint32_t w

6.56.1 Detailed Description

Union type to access the Special-Purpose Program Status Registers (xPSR).

The documentation for this union was generated from the following file:

- CUBE_IDE/VGA/Core/Inc/core_cm4.h

Chapter 7

File Documentation

7.1 CUBE_IDE/VGA/Core/Inc/API_functions.h File Reference

Header file for API functions.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "stm32_ub_vga_screen.h"
#include "logic_layer.h"
```

Macros

- #define **sgn**(x) ((x<0)?-1:((x>0)?1:0))
- #define **BITMAP_HEIGHT** 77
- #define **BITMAP_WIDTH** 120
- #define **BITMAP_SIZE** 9240

Functions

- int API_draw_line (int x_1, int y_1, int x_2, int y_2, int color, int weight)

  This function gives the user the ability to draw lines on a VGA screen.
- int API_clearscreen (int color)

  This function gives the user the ability to clear the screen to a certain color.
- int API_draw_rectangle (int x, int y, int width, int height, int color, int filled)

  This function gives the user the ability to draw a rectangle on a VGA screen.
- int API_draw_bitmap (int bm_nr, int x_lup, int y_lup)

  This function gives the user the ability to put a bitmap on the VGA screen.

7.1.1 Detailed Description

Header file for API functions.

Authors

Naomi Born

Date

17-05-2023

Version

1.0

7.1.2 Function Documentation

7.1.2.1 API_clearscreen()

```
int API_clearscreen (
            int color )
```

This function gives the user the ability to clear the screen to a certain color.

Parameters

color	Color of the screen

Returns

Error code if error occurs

7.1.2.2 API_draw_bitmap()

```
int API_draw_bitmap (
            int bm_nr,
            int x_lup,
            int y_lup )
```

This function gives the user the ability to put a bitmap on the VGA screen.

Parameters

x_lup	X coordinate of x left up
y_lup	Y coordinate of y left up
bm↩ _nr	Number of the bitmap from 1 to 10

Returns

Error code if error or no error occurs

7.1.2.3 API_draw_line()

```
int API_draw_line (
            int x_1,
            int y_1,
            int x_2,
            int y_2,
            int color,
            int weight )
```

This function gives the user the ability to draw lines on a VGA screen.

Parameters

x_1	Starting point coördinate of x
x_2	Ending point coördinate of x
y_1	Starting point coördinate of y
y_2	Ending point coördinate of y
color	Color of the line
weight	Width of the line

Returns

Error code if error or no error occurs

7.1.2.4 API_draw_rectangle()

```
int API_draw_rectangle (
            int x_1,
            int y_1,
            int width,
            int height,
            int color,
            int filled )
```

This function gives the user the ability to draw a rectangle on a VGA screen.

Parameters

x_1	Starting point coördinate of x (upper left)
y_1	Starting point coördinate of y (upper left)
width	Width of the rectangle max. 320
height	Height of the rectangle max. 240
color	Color of the rectangle/borders
filled	1 is filled rectangle 0 is just borders

Returns

Error code if error occurs

7.2 API_functions.h

Go to the documentation of this file.

```
00001 /*****************************************************/
00010 #ifndef INC_API_FUNCTIONS_H_
00011 #define INC_API_FUNCTIONS_H_
00012
00013 //#include <library-header>
00014 #include <stdio.h>
00015 #include <stdlib.h>
00016 #include <string.h>
00017
00018 //#include other "user-header"
00019 #include "stm32_ub_vga_screen.h"
00020 #include "logic_layer.h"
00021
00022 //#define-statements
00023 #define sgn(x) ((x<0)?-1:((x>0)?1:0)) // macro to return the sign of a number
00024 #define BITMAP_HEIGHT 77
00025 #define BITMAP_WIDTH 120
00026 #define BITMAP_SIZE 9240
00027
00028 // prototype user functions
00029 int API_draw_line(int x_1, int y_1, int x_2, int y_2, int color, int weight);
00030 int API_clearscreen(int color);
00031 int API_draw_rectangle(int x, int y, int width, int height, int color, int filled);
00032 int API_draw_bitmap(int bm_nr, int x_lup, int y_lup);
00033
00034 #endif /* INC_API_FUNCTIONS_H_ */
```

7.3 CUBE_IDE/VGA/Core/Inc/bitmap_arrows.h File Reference

Header file for bitmap arrays.

Variables

- const int **arrow_up** [ ]
- const int **arrow_right** [ ]
- const int **arrow_left** [ ]
- const int **arrow_down** [ ]

7.3.1 Detailed Description

Header file for bitmap arrays.

Authors

Naomi Born

Date

31-05-2023

Version

1.0

7.4 bitmap_arrows.h

Go to the documentation of this file.

```
/*********************************************************/
#ifndef INC_BITMAP_ARROWS_H_
#define INC_BITMAP_ARROWS_H_

const int arrow_up[]  = {
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00022   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00023   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00024   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00025   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00026   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00027   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00028   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00029   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00030   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00031   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00032   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00033   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00034   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00035   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00036   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00037   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00038   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00039   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00040   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00041   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00042   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00043   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00044   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00045   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00046   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00047   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255,
00048   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255,
00049   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255,
00050   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255,
00051   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255,
00052   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255,
00053   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00054   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00055   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00056   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00057   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00058   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00059   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00060   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00061   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00062   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00063   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00064   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00065   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00066   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00067   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00068   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00069   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00070   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00071   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00072   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00073   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00074   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00075   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00076   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00077   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00078   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00079   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255
};

const int arrow_right[]  = {
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
```

```
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255
};

const int arrow_left[]  = {
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
```

```
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
        255, 255, 255, 255, 255, 255, 255,
00206     255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
        255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
        255, 255, 255, 255, 255, 255,
00207     255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
        255, 255, 255, 255,
00208     255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
        255, 255, 255, 255,
00209     255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
        255, 255, 255,
00210     255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
        255, 255,
00211     255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
        255,
00212     255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
        255,
00213     255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
        255,
00214     255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
        255, 255,
00215     255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
        255, 255, 255,
00216     255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
        255, 255, 255, 255,
00217     255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
        255, 255, 255, 255,
00218     255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
        255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
        255, 255, 255, 255, 255, 255,
00219     255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
        255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
        255, 255, 255, 255, 255, 255, 255,
00220     255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
        255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
        255, 255, 255, 255, 255, 255, 255,
00221     255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
        255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
        255, 255, 255, 255, 255, 255, 255, 255,
00222     255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
        255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
        255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255
};

const int arrow_down[]  = {
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00270   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00271   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00272   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00273   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00274   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00275   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00276   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00277   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00278   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00279   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00280   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00281   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00282   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00283   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00284   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00285   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00286   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00287   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00288   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00289   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00290   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00291   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00292   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255,
00293   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255,
00294   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255,
00295   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255,
00296   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255,
00297   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255,
00298   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255,
00299   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255,
00300   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255,
00301   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255,
00302   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255,
00303   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255,
00304   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00305   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00306   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00307   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00308   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00309   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00310   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00311   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00312   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255
};
#endif /* INC_BITMAP_ARROWS_H_ */
```

7.5 CUBE_IDE/VGA/Core/Inc/bitmap_smileys.h File Reference

Header file for bitmap arrays.

Variables

- const int **smiley_happy_col** [ ]
- const int **smiley_angry_col** [ ]
- const int **smiley_sad_col** [ ]
- const int **smiley_happy** [ ]
- const int **smiley_sad** [ ]
- const int **smiley_angry** [ ]

7.5.1 Detailed Description

Header file for bitmap arrays.

Authors

Naomi Born

Date

31-05-2023

Version

1.0

7.6 bitmap_smileys.h

Go to the documentation of this file.

```
/*******************************************************/
#ifndef INC_BITMAP_SMILEYS_H_
#define INC_BITMAP_SMILEYS_H_

const int smiley_happy_col[]  = {
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0,
0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
00028   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00029   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00030   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00031   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00032   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00033   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00034   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00035   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00036   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00037   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00038   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00039   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00040   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00041   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00042   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
```

```
252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0,
0, 0, 252, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0,
0, 0, 0, 252, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0,
0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0,
252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0,
0, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0,
252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0,
0, 0, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252,
252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0,
252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252, 252,
252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252,
252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252, 252,
252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252,
252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252,
252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 0,
0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 252, 252, 252, 252, 252,
252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 247, 247, 247, 247, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 247, 247,
247, 247, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 247, 247, 247, 247, 247, 247, 247,
247, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 247, 247, 247, 247,
247, 247, 247, 247, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 247, 247, 247, 247, 247, 247, 247,
247, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 247, 247, 247, 247,
247, 247, 247, 247, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
00057   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 247, 247, 247, 247, 247, 247, 247, 247,
      247, 247, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 247, 247, 247, 247,
      247, 247, 247, 247, 247, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00058   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 247, 247, 247, 247, 247, 247, 247, 247,
      247, 247, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 247, 247, 247, 247,
      247, 247, 247, 247, 247, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00059   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 247, 247, 247, 247, 247, 247, 247, 247,
      247, 247, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 247, 247, 247, 247,
      247, 247, 247, 247, 247, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00060   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 247, 247, 247, 247, 247, 247, 247, 247,
      247, 247, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 247, 247, 247, 247,
      247, 247, 247, 247, 247, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00061   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 247, 247, 247, 247, 247, 247, 247, 247,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 247, 247, 247, 247,
      247, 247, 247, 247, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00062   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 247, 247, 247, 247, 247, 247, 247,
      247, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      252, 252, 252, 252, 252, 247, 247, 247, 247, 247, 247, 247, 247, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255,
00063   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 247, 247, 247, 247, 252,
      252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      252, 252, 252, 252, 252, 247, 247, 247, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255,
00064   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255,
00065   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255,
00066   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255,
00067   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00068   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00069   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00070   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0,
      0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00071   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0,
```

```
      0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00072   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00073   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00074   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00075   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00076   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00077   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00078   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0,
      0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00079   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00080   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00081   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00082   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00083   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00084   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00085   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
```

```
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255
};

const int smiley_angry_col[]  = {
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
00103   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00104   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00105   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
      0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00106   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0,
      0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00107   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00108   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00109   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00110   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00111   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00112   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00113   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 0,
      0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224,
      224, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00114   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0,
      0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00115   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0,
      0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00116   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0,
      0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00117   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
```

```
      224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00118   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0,
      0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255,
00119   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255,
00120   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0,
      0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255,
00121   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00122   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 0, 0, 0, 0,
      0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00123   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0,
      0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00124   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 0, 0, 0, 0, 224, 224, 224, 224, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 0, 0,
      0, 224, 224, 224, 224, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00125   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00126   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00127   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00128   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00129   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00130   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00131   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
00132   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00133   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00134   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00135   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00136   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00137   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00138   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00139   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00140   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00141   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00142   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00143   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00144   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00145   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00146   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 224, 224, 0, 0, 0, 0, 0, 0, 224, 224,
```

```
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00147   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00148   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0,
      0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0,
      0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00149   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0,
      0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00150   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0,
      0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00151   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00152   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00153   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00154   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00155   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00156   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00157   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00158   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00159   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00160   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
      224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
224, 224, 224, 224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
224, 224, 224, 224, 224, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255
};

const int smiley_sad_col[]  = {
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
00178   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00179   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00180   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00181   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00182   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00183   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00184   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00185   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
      0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00186   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00187   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00188   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00189   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00190   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00191   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00192   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
```

```
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00193   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00194   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00195   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00196   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00197   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00198   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00199   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00200   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00201   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00202   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00203   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00204   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 252, 252, 252, 252, 252, 252, 0,
      0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00205   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 252, 252, 252, 252, 0, 0, 0, 0, 0,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00206   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255,
```

```
00207   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252,
      252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252, 252,
      252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255,
00208   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0,
      252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252,
      252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255,
00209   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0,
      0, 0, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0,
      252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255,
00210   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255,
00211   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255,
00212   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 23, 252, 252, 252, 252, 252, 252, 252,
      252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00213   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 23, 23, 23, 252, 252, 252,
      252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00214   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 252, 252, 252, 252, 252, 23, 252, 252, 252, 252, 23, 252,
      252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00215   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 23, 23, 23, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00216   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 23, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 23, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00217   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 23, 23, 23, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 23,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00218   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 23, 252, 252, 252, 252,
      252, 23, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 23, 23,
      23, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00219   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      23, 23, 23, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 23, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00220   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 23, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00221   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252,
```

```
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00222   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00223   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00224   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 23, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 23, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00225   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 23, 23, 23, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 23, 23, 23, 252, 252, 252, 252, 0, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00226   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 23, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 23, 252, 252, 252, 252, 252, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00227   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00228   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0,
      0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00229   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 23, 252, 252, 252, 252, 252, 252, 252, 252, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00230   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 23, 252, 252,
      252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 23, 23, 23, 252, 252, 252, 252, 0, 0, 0, 0, 0,
      0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00231   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 23, 23, 23, 252,
      252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 23, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00232   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 23, 252, 252,
      252, 252, 252, 252, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00233   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00234   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00235   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
      252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0,
0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
252, 252, 252, 252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
252, 252, 252, 252, 252, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252, 252,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255
};

const int smiley_happy[]  = {
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00268   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00269   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00270   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00271   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00272   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00273   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00274   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00275   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00276   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00277   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00278   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00279   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00280   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00281   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
00282   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00283   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255,
00284   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255,
00285   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
      0, 0, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0,
      0, 0, 0, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255,
00286   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0,
      0, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255,
00287   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00288   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255,
00289   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255,
00290   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255,
00291   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00292   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00293   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00294   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00295   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00296   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00297   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00298   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00299   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00300   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00301   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00302   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255,
00303   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255,
00304   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255,
00305   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255,
00306   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255,
00307   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00308   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00309   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00310   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
      0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
00311   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00312   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00313   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00314   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00315   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00316   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00317   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00318   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
      0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00319   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00320   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00321   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00322   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00323   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00324   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00325   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
```

```
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255
};

const int smiley_sad[]  = {
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00343   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00344   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00345   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
      0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00346   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
      0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00347   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00348   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00349   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00350   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00351   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00352   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00353   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00354   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00355   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00356   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
00357   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00358   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00359   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00360   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00361   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00362   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00363   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00364   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00365   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00366   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255,
00367   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255,
00368   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255,
00369   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
      0, 0, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255,
00370   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255,
00371   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0,
```

```
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255,
00372   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255,
00373   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00374   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 255, 255, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00375   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00376   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00377   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00378   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 255, 255, 255, 255,
      0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00379   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00380   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00381   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00382   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00383   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00384   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00385   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 255, 255, 255, 255, 255, 0, 0, 0,
      0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255,
```

```
00386   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00387   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00388   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00389   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 0, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00390   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 0, 0, 0, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00391   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 0, 0, 0, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 0, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00392   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 0, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00393   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00394   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00395   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00396   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00397   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00398   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
      0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00399   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00400   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255
};

const int smiley_angry[]  = {
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
```

```
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00418   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00419   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00420   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00421   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00422   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00423   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00424   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00425   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
      0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00426   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00427   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00428   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00429   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00430   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00431   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
00432   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00433   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00434   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00435   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
      0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00436   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0,
      0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00437   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
      0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00438   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
      0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00439   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00440   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255,
00441   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00442   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 0, 0, 0, 0,
      0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00443   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0,
      0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00444   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 255, 255, 255, 255, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 255, 255, 255, 255, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00445   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00446   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
      255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00447   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00448   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00449   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00450   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255,
00451   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00452   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00453   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00454   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00455   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00456   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00457   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00458   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00459   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00460   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
00461   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00462   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00463   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00464   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00465   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00466   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00467   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00468   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
      0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0,
      0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255,
00469   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00470   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00471   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00472   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
      0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00473   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
      0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00474   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00475   255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255,
      255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00476 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00477 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00478 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0,
0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00479 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00480 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0,
0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00481 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00482 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00483 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255,
00484 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255,
00485 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255,
00486 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255,
00487 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00488 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
00489 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
```

```
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
  255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255
};

#endif /* INC_BITMAP_SMILEYS_H_ */
```

7.7 CUBE_IDE/VGA/Core/Inc/core_cm4.h File Reference

CMSIS Cortex-M4 Core Peripheral Access Layer Header File.

```
#include <stdint.h>
#include <core_cmInstr.h>
#include <core_cmFunc.h>
#include <core_cm4_simd.h>
```

Data Structures

- union APSR_Type

  Union type to access the Application Program Status Register (APSR).
- union IPSR_Type

  Union type to access the Interrupt Program Status Register (IPSR).
- union xPSR_Type

  Union type to access the Special-Purpose Program Status Registers (xPSR).
- union CONTROL_Type

  Union type to access the Control Registers (CONTROL).
- struct NVIC_Type

  Structure type to access the Nested Vectored Interrupt Controller (NVIC).
- struct SCB_Type

  Structure type to access the System Control Block (SCB).
- struct SCnSCB_Type

  Structure type to access the System Control and ID Register not in the SCB.
- struct SysTick_Type

  Structure type to access the System Timer (SysTick).
- struct ITM_Type

  Structure type to access the Instrumentation Trace Macrocell Register (ITM).
- struct CoreDebug_Type

  Structure type to access the Core Debug Register (CoreDebug).

Macros

- #define **__CORE_CM4_H_GENERIC**
- #define __CM4_CMSIS_VERSION_MAIN (0x02)
- #define __CM4_CMSIS_VERSION_SUB (0x10)
- #define __CM4_CMSIS_VERSION ((__CM4_CMSIS_VERSION_MAIN << 16) | __CM4_CMSIS_VERSION_SUB)
- #define __CORTEX_M (0x04)
- #define **__CORE_CM4_H_DEPENDANT**
- #define __I volatile const
- #define __O volatile
- #define __IO volatile
- #define NVIC_STIR_INTID_Pos 0
- #define NVIC_STIR_INTID_Msk (0x1FFUL << NVIC_STIR_INTID_Pos)
- #define SCB_CPUID_IMPLEMENTER_Pos 24
- #define SCB_CPUID_IMPLEMENTER_Msk (0xFFUL << SCB_CPUID_IMPLEMENTER_Pos)
- #define SCB_CPUID_VARIANT_Pos 20
- #define SCB_CPUID_VARIANT_Msk (0xFUL << SCB_CPUID_VARIANT_Pos)
- #define SCB_CPUID_ARCHITECTURE_Pos 16
- #define SCB_CPUID_ARCHITECTURE_Msk (0xFUL << SCB_CPUID_ARCHITECTURE_Pos)
- #define SCB_CPUID_PARTNO_Pos 4
- #define SCB_CPUID_PARTNO_Msk (0xFFFUL << SCB_CPUID_PARTNO_Pos)
- #define SCB_CPUID_REVISION_Pos 0
- #define SCB_CPUID_REVISION_Msk (0xFUL << SCB_CPUID_REVISION_Pos)
- #define SCB_ICSR_NMIPENDSET_Pos 31
- #define SCB_ICSR_NMIPENDSET_Msk (1UL << SCB_ICSR_NMIPENDSET_Pos)
- #define SCB_ICSR_PENDSVSET_Pos 28
- #define SCB_ICSR_PENDSVSET_Msk (1UL << SCB_ICSR_PENDSVSET_Pos)
- #define SCB_ICSR_PENDSVCLR_Pos 27
- #define SCB_ICSR_PENDSVCLR_Msk (1UL << SCB_ICSR_PENDSVCLR_Pos)
- #define SCB_ICSR_PENDSTSET_Pos 26
- #define SCB_ICSR_PENDSTSET_Msk (1UL << SCB_ICSR_PENDSTSET_Pos)
- #define SCB_ICSR_PENDSTCLR_Pos 25
- #define SCB_ICSR_PENDSTCLR_Msk (1UL << SCB_ICSR_PENDSTCLR_Pos)
- #define SCB_ICSR_ISRPREEMPT_Pos 23
- #define SCB_ICSR_ISRPREEMPT_Msk (1UL << SCB_ICSR_ISRPREEMPT_Pos)
- #define SCB_ICSR_ISRPENDING_Pos 22
- #define SCB_ICSR_ISRPENDING_Msk (1UL << SCB_ICSR_ISRPENDING_Pos)
- #define SCB_ICSR_VECTPENDING_Pos 12
- #define SCB_ICSR_VECTPENDING_Msk (0x1FFUL << SCB_ICSR_VECTPENDING_Pos)
- #define SCB_ICSR_RETTOBASE_Pos 11
- #define SCB_ICSR_RETTOBASE_Msk (1UL << SCB_ICSR_RETTOBASE_Pos)
- #define SCB_ICSR_VECTACTIVE_Pos 0
- #define SCB_ICSR_VECTACTIVE_Msk (0x1FFUL << SCB_ICSR_VECTACTIVE_Pos)
- #define SCB_VTOR_TBLOFF_Pos 7
- #define SCB_VTOR_TBLOFF_Msk (0x1FFFFFFUL << SCB_VTOR_TBLOFF_Pos)
- #define SCB_AIRCR_VECTKEY_Pos 16
- #define SCB_AIRCR_VECTKEY_Msk (0xFFFFUL << SCB_AIRCR_VECTKEY_Pos)
- #define SCB_AIRCR_VECTKEYSTAT_Pos 16
- #define SCB_AIRCR_VECTKEYSTAT_Msk (0xFFFFUL << SCB_AIRCR_VECTKEYSTAT_Pos)
- #define SCB_AIRCR_ENDIANESS_Pos 15
- #define SCB_AIRCR_ENDIANESS_Msk (1UL << SCB_AIRCR_ENDIANESS_Pos)
- #define SCB_AIRCR_PRIGROUP_Pos 8
- #define SCB_AIRCR_PRIGROUP_Msk (7UL << SCB_AIRCR_PRIGROUP_Pos)
- #define SCB_AIRCR_SYSRESETREQ_Pos 2
- #define SCB_AIRCR_SYSRESETREQ_Msk (1UL << SCB_AIRCR_SYSRESETREQ_Pos)

- #define SCB_AIRCR_VECTCLRACTIVE_Pos 1
- #define SCB_AIRCR_VECTCLRACTIVE_Msk (1UL << SCB_AIRCR_VECTCLRACTIVE_Pos)
- #define SCB_AIRCR_VECTRESET_Pos 0
- #define SCB_AIRCR_VECTRESET_Msk (1UL << SCB_AIRCR_VECTRESET_Pos)
- #define SCB_SCR_SEVONPEND_Pos 4
- #define SCB_SCR_SEVONPEND_Msk (1UL << SCB_SCR_SEVONPEND_Pos)
- #define SCB_SCR_SLEEPDEEP_Pos 2
- #define SCB_SCR_SLEEPDEEP_Msk (1UL << SCB_SCR_SLEEPDEEP_Pos)
- #define SCB_SCR_SLEEPONEXIT_Pos 1
- #define SCB_SCR_SLEEPONEXIT_Msk (1UL << SCB_SCR_SLEEPONEXIT_Pos)
- #define SCB_CCR_STKALIGN_Pos 9
- #define SCB_CCR_STKALIGN_Msk (1UL << SCB_CCR_STKALIGN_Pos)
- #define SCB_CCR_BFHFNMIGN_Pos 8
- #define SCB_CCR_BFHFNMIGN_Msk (1UL << SCB_CCR_BFHFNMIGN_Pos)
- #define SCB_CCR_DIV_0_TRP_Pos 4
- #define SCB_CCR_DIV_0_TRP_Msk (1UL << SCB_CCR_DIV_0_TRP_Pos)
- #define SCB_CCR_UNALIGN_TRP_Pos 3
- #define SCB_CCR_UNALIGN_TRP_Msk (1UL << SCB_CCR_UNALIGN_TRP_Pos)
- #define SCB_CCR_USERSETMPEND_Pos 1
- #define SCB_CCR_USERSETMPEND_Msk (1UL << SCB_CCR_USERSETMPEND_Pos)
- #define SCB_CCR_NONBASETHRDENA_Pos 0
- #define SCB_CCR_NONBASETHRDENA_Msk (1UL << SCB_CCR_NONBASETHRDENA_Pos)
- #define SCB_SHCSR_USGFAULTENA_Pos 18
- #define SCB_SHCSR_USGFAULTENA_Msk (1UL << SCB_SHCSR_USGFAULTENA_Pos)
- #define SCB_SHCSR_BUSFAULTENA_Pos 17
- #define SCB_SHCSR_BUSFAULTENA_Msk (1UL << SCB_SHCSR_BUSFAULTENA_Pos)
- #define SCB_SHCSR_MEMFAULTENA_Pos 16
- #define SCB_SHCSR_MEMFAULTENA_Msk (1UL << SCB_SHCSR_MEMFAULTENA_Pos)
- #define SCB_SHCSR_SVCALLPENDED_Pos 15
- #define SCB_SHCSR_SVCALLPENDED_Msk (1UL << SCB_SHCSR_SVCALLPENDED_Pos)
- #define SCB_SHCSR_BUSFAULTPENDED_Pos 14
- #define SCB_SHCSR_BUSFAULTPENDED_Msk (1UL << SCB_SHCSR_BUSFAULTPENDED_Pos)
- #define SCB_SHCSR_MEMFAULTPENDED_Pos 13
- #define SCB_SHCSR_MEMFAULTPENDED_Msk (1UL << SCB_SHCSR_MEMFAULTPENDED_Pos)
- #define SCB_SHCSR_USGFAULTPENDED_Pos 12
- #define SCB_SHCSR_USGFAULTPENDED_Msk (1UL << SCB_SHCSR_USGFAULTPENDED_Pos)
- #define SCB_SHCSR_SYSTICKACT_Pos 11
- #define SCB_SHCSR_SYSTICKACT_Msk (1UL << SCB_SHCSR_SYSTICKACT_Pos)
- #define SCB_SHCSR_PENDSVACT_Pos 10
- #define SCB_SHCSR_PENDSVACT_Msk (1UL << SCB_SHCSR_PENDSVACT_Pos)
- #define SCB_SHCSR_MONITORACT_Pos 8
- #define SCB_SHCSR_MONITORACT_Msk (1UL << SCB_SHCSR_MONITORACT_Pos)
- #define SCB_SHCSR_SVCALLACT_Pos 7
- #define SCB_SHCSR_SVCALLACT_Msk (1UL << SCB_SHCSR_SVCALLACT_Pos)
- #define SCB_SHCSR_USGFAULTACT_Pos 3
- #define SCB_SHCSR_USGFAULTACT_Msk (1UL << SCB_SHCSR_USGFAULTACT_Pos)
- #define SCB_SHCSR_BUSFAULTACT_Pos 1
- #define SCB_SHCSR_BUSFAULTACT_Msk (1UL << SCB_SHCSR_BUSFAULTACT_Pos)
- #define SCB_SHCSR_MEMFAULTACT_Pos 0
- #define SCB_SHCSR_MEMFAULTACT_Msk (1UL << SCB_SHCSR_MEMFAULTACT_Pos)
- #define SCB_CFSR_USGFAULTSR_Pos 16
- #define SCB_CFSR_USGFAULTSR_Msk (0xFFFFUL << SCB_CFSR_USGFAULTSR_Pos)
- #define SCB_CFSR_BUSFAULTSR_Pos 8
- #define SCB_CFSR_BUSFAULTSR_Msk (0xFFUL << SCB_CFSR_BUSFAULTSR_Pos)
- #define SCB_CFSR_MEMFAULTSR_Pos 0

- #define SCB_CFSR_MEMFAULTSR_Msk (0xFFUL << SCB_CFSR_MEMFAULTSR_Pos)
- #define SCB_HFSR_DEBUGEVT_Pos 31
- #define SCB_HFSR_DEBUGEVT_Msk (1UL << SCB_HFSR_DEBUGEVT_Pos)
- #define SCB_HFSR_FORCED_Pos 30
- #define SCB_HFSR_FORCED_Msk (1UL << SCB_HFSR_FORCED_Pos)
- #define SCB_HFSR_VECTTBL_Pos 1
- #define SCB_HFSR_VECTTBL_Msk (1UL << SCB_HFSR_VECTTBL_Pos)
- #define SCB_DFSR_EXTERNAL_Pos 4
- #define SCB_DFSR_EXTERNAL_Msk (1UL << SCB_DFSR_EXTERNAL_Pos)
- #define SCB_DFSR_VCATCH_Pos 3
- #define SCB_DFSR_VCATCH_Msk (1UL << SCB_DFSR_VCATCH_Pos)
- #define SCB_DFSR_DWTTRAP_Pos 2
- #define SCB_DFSR_DWTTRAP_Msk (1UL << SCB_DFSR_DWTTRAP_Pos)
- #define SCB_DFSR_BKPT_Pos 1
- #define SCB_DFSR_BKPT_Msk (1UL << SCB_DFSR_BKPT_Pos)
- #define SCB_DFSR_HALTED_Pos 0
- #define SCB_DFSR_HALTED_Msk (1UL << SCB_DFSR_HALTED_Pos)
- #define SCnSCB_ICTR_INTLINESNUM_Pos 0
- #define SCnSCB_ICTR_INTLINESNUM_Msk (0xFUL << SCnSCB_ICTR_INTLINESNUM_Pos)
- #define SCnSCB_ACTLR_DISOOFP_Pos 9
- #define SCnSCB_ACTLR_DISOOFP_Msk (1UL << SCnSCB_ACTLR_DISOOFP_Pos)
- #define SCnSCB_ACTLR_DISFPCA_Pos 8
- #define SCnSCB_ACTLR_DISFPCA_Msk (1UL << SCnSCB_ACTLR_DISFPCA_Pos)
- #define SCnSCB_ACTLR_DISFOLD_Pos 2
- #define SCnSCB_ACTLR_DISFOLD_Msk (1UL << SCnSCB_ACTLR_DISFOLD_Pos)
- #define SCnSCB_ACTLR_DISDEFWBUF_Pos 1
- #define SCnSCB_ACTLR_DISDEFWBUF_Msk (1UL << SCnSCB_ACTLR_DISDEFWBUF_Pos)
- #define SCnSCB_ACTLR_DISMCYCINT_Pos 0
- #define SCnSCB_ACTLR_DISMCYCINT_Msk (1UL << SCnSCB_ACTLR_DISMCYCINT_Pos)
- #define SysTick_CTRL_COUNTFLAG_Pos 16
- #define SysTick_CTRL_COUNTFLAG_Msk (1UL << SysTick_CTRL_COUNTFLAG_Pos)
- #define SysTick_CTRL_CLKSOURCE_Pos 2
- #define SysTick_CTRL_CLKSOURCE_Msk (1UL << SysTick_CTRL_CLKSOURCE_Pos)
- #define SysTick_CTRL_TICKINT_Pos 1
- #define SysTick_CTRL_TICKINT_Msk (1UL << SysTick_CTRL_TICKINT_Pos)
- #define SysTick_CTRL_ENABLE_Pos 0
- #define SysTick_CTRL_ENABLE_Msk (1UL << SysTick_CTRL_ENABLE_Pos)
- #define SysTick_LOAD_RELOAD_Pos 0
- #define SysTick_LOAD_RELOAD_Msk (0xFFFFFFUL << SysTick_LOAD_RELOAD_Pos)
- #define SysTick_VAL_CURRENT_Pos 0
- #define SysTick_VAL_CURRENT_Msk (0xFFFFFFUL << SysTick_VAL_CURRENT_Pos)
- #define SysTick_CALIB_NOREF_Pos 31
- #define SysTick_CALIB_NOREF_Msk (1UL << SysTick_CALIB_NOREF_Pos)
- #define SysTick_CALIB_SKEW_Pos 30
- #define SysTick_CALIB_SKEW_Msk (1UL << SysTick_CALIB_SKEW_Pos)
- #define SysTick_CALIB_TENMS_Pos 0
- #define SysTick_CALIB_TENMS_Msk (0xFFFFFFUL << SysTick_VAL_CURRENT_Pos)
- #define ITM_TPR_PRIVMASK_Pos 0
- #define ITM_TPR_PRIVMASK_Msk (0xFUL << ITM_TPR_PRIVMASK_Pos)
- #define ITM_TCR_BUSY_Pos 23
- #define ITM_TCR_BUSY_Msk (1UL << ITM_TCR_BUSY_Pos)
- #define ITM_TCR_TraceBusID_Pos 16
- #define ITM_TCR_TraceBusID_Msk (0x7FUL << ITM_TCR_TraceBusID_Pos)
- #define ITM_TCR_GTSFREQ_Pos 10
- #define ITM_TCR_GTSFREQ_Msk (3UL << ITM_TCR_GTSFREQ_Pos)

- #define ITM_TCR_TSPrescale_Pos 8
- #define ITM_TCR_TSPrescale_Msk (3UL << ITM_TCR_TSPrescale_Pos)
- #define ITM_TCR_SWOENA_Pos 4
- #define ITM_TCR_SWOENA_Msk (1UL << ITM_TCR_SWOENA_Pos)
- #define ITM_TCR_TXENA_Pos 3
- #define ITM_TCR_TXENA_Msk (1UL << ITM_TCR_TXENA_Pos)
- #define ITM_TCR_SYNCENA_Pos 2
- #define ITM_TCR_SYNCENA_Msk (1UL << ITM_TCR_SYNCENA_Pos)
- #define ITM_TCR_TSENA_Pos 1
- #define ITM_TCR_TSENA_Msk (1UL << ITM_TCR_TSENA_Pos)
- #define ITM_TCR_ITMENA_Pos 0
- #define ITM_TCR_ITMENA_Msk (1UL << ITM_TCR_ITMENA_Pos)
- #define CoreDebug_DHCSR_DBGKEY_Pos 16
- #define CoreDebug_DHCSR_DBGKEY_Msk (0xFFFFUL << CoreDebug_DHCSR_DBGKEY_Pos)
- #define CoreDebug_DHCSR_S_RESET_ST_Pos 25
- #define CoreDebug_DHCSR_S_RESET_ST_Msk (1UL << CoreDebug_DHCSR_S_RESET_ST_Pos)
- #define CoreDebug_DHCSR_S_RETIRE_ST_Pos 24
- #define CoreDebug_DHCSR_S_RETIRE_ST_Msk (1UL << CoreDebug_DHCSR_S_RETIRE_ST_Pos)
- #define CoreDebug_DHCSR_S_LOCKUP_Pos 19
- #define CoreDebug_DHCSR_S_LOCKUP_Msk (1UL << CoreDebug_DHCSR_S_LOCKUP_Pos)
- #define CoreDebug_DHCSR_S_SLEEP_Pos 18
- #define CoreDebug_DHCSR_S_SLEEP_Msk (1UL << CoreDebug_DHCSR_S_SLEEP_Pos)
- #define CoreDebug_DHCSR_S_HALT_Pos 17
- #define CoreDebug_DHCSR_S_HALT_Msk (1UL << CoreDebug_DHCSR_S_HALT_Pos)
- #define CoreDebug_DHCSR_S_REGRDY_Pos 16
- #define CoreDebug_DHCSR_S_REGRDY_Msk (1UL << CoreDebug_DHCSR_S_REGRDY_Pos)
- #define CoreDebug_DHCSR_C_SNAPSTALL_Pos 5
- #define CoreDebug_DHCSR_C_SNAPSTALL_Msk (1UL << CoreDebug_DHCSR_C_SNAPSTALL_Pos)
- #define CoreDebug_DHCSR_C_MASKINTS_Pos 3
- #define CoreDebug_DHCSR_C_MASKINTS_Msk (1UL << CoreDebug_DHCSR_C_MASKINTS_Pos)
- #define CoreDebug_DHCSR_C_STEP_Pos 2
- #define CoreDebug_DHCSR_C_STEP_Msk (1UL << CoreDebug_DHCSR_C_STEP_Pos)
- #define CoreDebug_DHCSR_C_HALT_Pos 1
- #define CoreDebug_DHCSR_C_HALT_Msk (1UL << CoreDebug_DHCSR_C_HALT_Pos)
- #define CoreDebug_DHCSR_C_DEBUGEN_Pos 0
- #define CoreDebug_DHCSR_C_DEBUGEN_Msk (1UL << CoreDebug_DHCSR_C_DEBUGEN_Pos)
- #define CoreDebug_DCRSR_REGWnR_Pos 16
- #define CoreDebug_DCRSR_REGWnR_Msk (1UL << CoreDebug_DCRSR_REGWnR_Pos)
- #define CoreDebug_DCRSR_REGSEL_Pos 0
- #define CoreDebug_DCRSR_REGSEL_Msk (0x1FUL << CoreDebug_DCRSR_REGSEL_Pos)
- #define CoreDebug_DEMCR_TRCENA_Pos 24
- #define CoreDebug_DEMCR_TRCENA_Msk (1UL << CoreDebug_DEMCR_TRCENA_Pos)
- #define CoreDebug_DEMCR_MON_REQ_Pos 19
- #define CoreDebug_DEMCR_MON_REQ_Msk (1UL << CoreDebug_DEMCR_MON_REQ_Pos)
- #define CoreDebug_DEMCR_MON_STEP_Pos 18
- #define CoreDebug_DEMCR_MON_STEP_Msk (1UL << CoreDebug_DEMCR_MON_STEP_Pos)
- #define CoreDebug_DEMCR_MON_PEND_Pos 17
- #define CoreDebug_DEMCR_MON_PEND_Msk (1UL << CoreDebug_DEMCR_MON_PEND_Pos)
- #define CoreDebug_DEMCR_MON_EN_Pos 16
- #define CoreDebug_DEMCR_MON_EN_Msk (1UL << CoreDebug_DEMCR_MON_EN_Pos)
- #define CoreDebug_DEMCR_VC_HARDERR_Pos 10
- #define CoreDebug_DEMCR_VC_HARDERR_Msk (1UL << CoreDebug_DEMCR_VC_HARDERR_Pos)
- #define CoreDebug_DEMCR_VC_INTERR_Pos 9
- #define CoreDebug_DEMCR_VC_INTERR_Msk (1UL << CoreDebug_DEMCR_VC_INTERR_Pos)
- #define CoreDebug_DEMCR_VC_BUSERR_Pos 8

- #define CoreDebug_DEMCR_VC_BUSERR_Msk (1UL << CoreDebug_DEMCR_VC_BUSERR_Pos)
- #define CoreDebug_DEMCR_VC_STATERR_Pos 7
- #define CoreDebug_DEMCR_VC_STATERR_Msk (1UL << CoreDebug_DEMCR_VC_STATERR_Pos)
- #define CoreDebug_DEMCR_VC_CHKERR_Pos 6
- #define CoreDebug_DEMCR_VC_CHKERR_Msk (1UL << CoreDebug_DEMCR_VC_CHKERR_Pos)
- #define CoreDebug_DEMCR_VC_NOCPERR_Pos 5
- #define CoreDebug_DEMCR_VC_NOCPERR_Msk (1UL << CoreDebug_DEMCR_VC_NOCPERR_Pos)
- #define CoreDebug_DEMCR_VC_MMERR_Pos 4
- #define CoreDebug_DEMCR_VC_MMERR_Msk (1UL << CoreDebug_DEMCR_VC_MMERR_Pos)
- #define CoreDebug_DEMCR_VC_CORERESET_Pos 0
- #define CoreDebug_DEMCR_VC_CORERESET_Msk (1UL << CoreDebug_DEMCR_VC_CORERESET_Pos)

- #define SCS_BASE (0xE000E000UL)
- #define ITM_BASE (0xE0000000UL)
- #define CoreDebug_BASE (0xE000EDF0UL)
- #define SysTick_BASE (SCS_BASE + 0x0010UL)
- #define NVIC_BASE (SCS_BASE + 0x0100UL)
- #define SCB_BASE (SCS_BASE + 0x0D00UL)
- #define SCnSCB ((SCnSCB_Type *) SCS_BASE)
- #define SCB ((SCB_Type *) SCB_BASE)
- #define SysTick ((SysTick_Type *) SysTick_BASE)
- #define NVIC ((NVIC_Type *) NVIC_BASE)
- #define ITM ((ITM_Type *) ITM_BASE)
- #define CoreDebug ((CoreDebug_Type *) CoreDebug_BASE)
- #define ITM_RXBUFFER_EMPTY 0x5AA55AA5
- volatile int32_t ITM_RxBuffer

7.7.1 Detailed Description

CMSIS Cortex-M4 Core Peripheral Access Layer Header File.

Version

V2.10

Date

19. July 2011

Note

ARM Limited (ARM) is supplying this software for use with Cortex-M processor based microcontrollers. This file can be freely distributed within development tools that are supporting such ARM based processors.

7.8 core_cm4.h

Go to the documentation of this file.

```
00001 /**************************************************************************/
00023 #if defined ( __ICCARM__ )
00024  #pragma system_include  /* treat file as system include file for MISRA check */
00025 #endif
00026
00027 #ifdef __cplusplus
00028  extern "C" {
00029 #endif
00030
00031 #ifndef __CORE_CM4_H_GENERIC
00032 #define __CORE_CM4_H_GENERIC
00033
00034
00064 /*******************************************************************************
00065  *                 CMSIS definitions
00066  ******************************************************************************/
00075 /*  CMSIS CM4 definitions */
00076 #define __CM4_CMSIS_VERSION_MAIN  (0x02)
00077 #define __CM4_CMSIS_VERSION_SUB   (0x10)
00078 #define __CM4_CMSIS_VERSION       ((__CM4_CMSIS_VERSION_MAIN << 16) | __CM4_CMSIS_VERSION_SUB)
00080 #define __CORTEX_M                (0x04)
00083 #if   defined ( __CC_ARM )
00084   #define __ASM            __asm
00085   #define __INLINE         __inline
00087 #elif defined ( __ICCARM__ )
00088   #define __ASM           __asm
00089   #define __INLINE        inline
00091 #elif defined ( __GNUC__ )
00092   #define __ASM            __asm
00093   #define __INLINE         inline
00095 #elif defined ( __TASKING__ )
00096   #define __ASM            __asm
00097   #define __INLINE         inline
00099 #endif
00100
00102 #if defined ( __CC_ARM )
00103   #if defined __TARGET_FPU_VFP
00104     #if (__FPU_PRESENT == 1)
00105       #define __FPU_USED       1
00106     #else
00107       #warning "Compiler generates FPU instructions for a device without an FPU (check __FPU_PRESENT)"
00108       #define __FPU_USED       0
00109     #endif
00110   #else
00111     #define __FPU_USED         0
00112   #endif
00113
00114 #elif defined ( __ICCARM__ )
00115   #if defined __ARMVFP__
00116     #if (__FPU_PRESENT == 1)
00117       #define __FPU_USED       1
00118     #else
00119       #warning "Compiler generates FPU instructions for a device without an FPU (check __FPU_PRESENT)"
00120       #define __FPU_USED       0
00121     #endif
00122   #else
00123     #define __FPU_USED         0
00124   #endif
00125
00126 #elif defined ( __GNUC__ )
00127   #if defined (__VFP_FP__) && !defined(__SOFTFP__)
00128     #if (__FPU_PRESENT == 1)
00129       #define __FPU_USED       1
00130     #else
00131       #warning "Compiler generates FPU instructions for a device without an FPU (check __FPU_PRESENT)"
00132       #define __FPU_USED       0
00133     #endif
00134   #else
00135     #define __FPU_USED         0
00136   #endif
00137
00138 #elif defined ( __TASKING__ )
00139     /* add preprocessor checks to define __FPU_USED */
00140     #define __FPU_USED         0
00141 #endif
00142
00143 #include <stdint.h>
00144 #include <core_cmInstr.h>
00145 #include <core_cmFunc.h>
00146 #include <core_cm4_simd.h>
00148 #endif /* __CORE_CM4_H_GENERIC */
00149
```

```
00150 #ifndef __CMSIS_GENERIC
00151
00152 #ifndef __CORE_CM4_H_DEPENDANT
00153 #define __CORE_CM4_H_DEPENDANT
00154
00155 /* check device defines and use defaults */
00156 #if defined __CHECK_DEVICE_DEFINES
00157   #ifndef __CM4_REV
00158     #define __CM4_REV               0x0000
00159     #warning "__CM4_REV not defined in device header file; using default!"
00160   #endif
00161
00162   #ifndef __FPU_PRESENT
00163     #define __FPU_PRESENT             0
00164     #warning "__FPU_PRESENT not defined in device header file; using default!"
00165   #endif
00166
00167   #ifndef __MPU_PRESENT
00168     #define __MPU_PRESENT             0
00169     #warning "__MPU_PRESENT not defined in device header file; using default!"
00170   #endif
00171
00172   #ifndef __NVIC_PRIO_BITS
00173     #define __NVIC_PRIO_BITS          4
00174     #warning "__NVIC_PRIO_BITS not defined in device header file; using default!"
00175   #endif
00176
00177   #ifndef __Vendor_SysTickConfig
00178     #define __Vendor_SysTickConfig    0
00179     #warning "__Vendor_SysTickConfig not defined in device header file; using default!"
00180   #endif
00181 #endif
00182
00183 /* IO definitions (access restrictions to peripheral registers) */
00184 #ifdef __cplusplus
00185   #define   __I     volatile
00186 #else
00187   #define   __I     volatile const
00188 #endif
00189 #define     __O     volatile
00190 #define     __IO    volatile
00196 /*******************************************************************************
00197  *                 Register Abstraction
00198  ******************************************************************************/
00218 typedef union
00219 {
00220   struct
00221   {
00222 #if (__CORTEX_M != 0x04)
00223     uint32_t _reserved0:27;
00224 #else
00225     uint32_t _reserved0:16;
00226     uint32_t GE:4;
00227     uint32_t _reserved1:7;
00228 #endif
00229     uint32_t Q:1;
00230     uint32_t V:1;
00231     uint32_t C:1;
00232     uint32_t Z:1;
00233     uint32_t N:1;
00234   } b;
00235   uint32_t w;
00236 } APSR_Type;
00237
00238
00241 typedef union
00242 {
00243   struct
00244   {
00245     uint32_t ISR:9;
00246     uint32_t _reserved0:23;
00247   } b;
00248   uint32_t w;
00249 } IPSR_Type;
00250
00251
00254 typedef union
00255 {
00256   struct
00257   {
00258     uint32_t ISR:9;
00259 #if (__CORTEX_M != 0x04)
00260     uint32_t _reserved0:15;
00261 #else
00262     uint32_t _reserved0:7;
00263     uint32_t GE:4;
00264     uint32_t _reserved1:4;
```

```
00265 #endif
00266     uint32_t T:1;
00267     uint32_t IT:2;
00268     uint32_t Q:1;
00269     uint32_t V:1;
00270     uint32_t C:1;
00271     uint32_t Z:1;
00272     uint32_t N:1;
00273   } b;
00274   uint32_t w;
00275 } xPSR_Type;
00276
00277
00280 typedef union
00281 {
00282   struct
00283   {
00284     uint32_t nPRIV:1;
00285     uint32_t SPSEL:1;
00286     uint32_t FPCA:1;
00287     uint32_t _reserved0:29;
00288   } b;
00289   uint32_t w;
00290 } CONTROL_Type;
00291
00303 typedef struct
00304 {
00305   __IO uint32_t ISER[8];
00306        uint32_t RESERVED0[24];
00307   __IO uint32_t ICER[8];
00308        uint32_t RSERVED1[24];
00309   __IO uint32_t ISPR[8];
00310        uint32_t RESERVED2[24];
00311   __IO uint32_t ICPR[8];
00312        uint32_t RESERVED3[24];
00313   __IO uint32_t IABR[8];
00314        uint32_t RESERVED4[56];
00315   __IO uint8_t  IP[240];
00316        uint32_t RESERVED5[644];
00317   __O  uint32_t STIR;
00318 }  NVIC_Type;
00319
00320 /* Software Triggered Interrupt Register Definitions */
00321 #define NVIC_STIR_INTID_Pos                 0
00322 #define NVIC_STIR_INTID_Msk                (0x1FFUL << NVIC_STIR_INTID_Pos)
00335 typedef struct
00336 {
00337   __I  uint32_t CPUID;
00338   __IO uint32_t ICSR;
00339   __IO uint32_t VTOR;
00340   __IO uint32_t AIRCR;
00341   __IO uint32_t SCR;
00342   __IO uint32_t CCR;
00343   __IO uint8_t  SHP[12];
00344   __IO uint32_t SHCSR;
00345   __IO uint32_t CFSR;
00346   __IO uint32_t HFSR;
00347   __IO uint32_t DFSR;
00348   __IO uint32_t MMFAR;
00349   __IO uint32_t BFAR;
00350   __IO uint32_t AFSR;
00351   __I  uint32_t PFR[2];
00352   __I  uint32_t DFR;
00353   __I  uint32_t ADR;
00354   __I  uint32_t MMFR[4];
00355   __I  uint32_t ISAR[5];
00356        uint32_t RESERVED0[5];
00357   __IO uint32_t CPACR;
00358 } SCB_Type;
00359
00360 /* SCB CPUID Register Definitions */
00361 #define SCB_CPUID_IMPLEMENTER_Pos          24
00362 #define SCB_CPUID_IMPLEMENTER_Msk          (0xFFUL << SCB_CPUID_IMPLEMENTER_Pos)
00364 #define SCB_CPUID_VARIANT_Pos              20
00365 #define SCB_CPUID_VARIANT_Msk              (0xFUL << SCB_CPUID_VARIANT_Pos)
00367 #define SCB_CPUID_ARCHITECTURE_Pos         16
00368 #define SCB_CPUID_ARCHITECTURE_Msk         (0xFUL << SCB_CPUID_ARCHITECTURE_Pos)
00370 #define SCB_CPUID_PARTNO_Pos                4
00371 #define SCB_CPUID_PARTNO_Msk               (0xFFFUL << SCB_CPUID_PARTNO_Pos)
00373 #define SCB_CPUID_REVISION_Pos              0
00374 #define SCB_CPUID_REVISION_Msk             (0xFUL << SCB_CPUID_REVISION_Pos)
00376 /* SCB Interrupt Control State Register Definitions */
00377 #define SCB_ICSR_NMIPENDSET_Pos            31
00378 #define SCB_ICSR_NMIPENDSET_Msk            (1UL << SCB_ICSR_NMIPENDSET_Pos)
00380 #define SCB_ICSR_PENDSVSET_Pos             28
00381 #define SCB_ICSR_PENDSVSET_Msk             (1UL << SCB_ICSR_PENDSVSET_Pos)
00383 #define SCB_ICSR_PENDSVCLR_Pos             27
```

```
00384 #define SCB_ICSR_PENDSVCLR_Msk             (1UL « SCB_ICSR_PENDSVCLR_Pos)
00386 #define SCB_ICSR_PENDSTSET_Pos             26
00387 #define SCB_ICSR_PENDSTSET_Msk             (1UL « SCB_ICSR_PENDSTSET_Pos)
00389 #define SCB_ICSR_PENDSTCLR_Pos             25
00390 #define SCB_ICSR_PENDSTCLR_Msk             (1UL « SCB_ICSR_PENDSTCLR_Pos)
00392 #define SCB_ICSR_ISRPREEMPT_Pos            23
00393 #define SCB_ICSR_ISRPREEMPT_Msk            (1UL « SCB_ICSR_ISRPREEMPT_Pos)
00395 #define SCB_ICSR_ISRPENDING_Pos            22
00396 #define SCB_ICSR_ISRPENDING_Msk            (1UL « SCB_ICSR_ISRPENDING_Pos)
00398 #define SCB_ICSR_VECTPENDING_Pos           12
00399 #define SCB_ICSR_VECTPENDING_Msk           (0x1FFUL « SCB_ICSR_VECTPENDING_Pos)
00401 #define SCB_ICSR_RETTOBASE_Pos             11
00402 #define SCB_ICSR_RETTOBASE_Msk             (1UL « SCB_ICSR_RETTOBASE_Pos)
00404 #define SCB_ICSR_VECTACTIVE_Pos             0
00405 #define SCB_ICSR_VECTACTIVE_Msk            (0x1FFUL « SCB_ICSR_VECTACTIVE_Pos)
00407 /* SCB Vector Table Offset Register Definitions */
00408 #define SCB_VTOR_TBLOFF_Pos                 7
00409 #define SCB_VTOR_TBLOFF_Msk                (0x1FFFFFFUL « SCB_VTOR_TBLOFF_Pos)
00411 /* SCB Application Interrupt and Reset Control Register Definitions */
00412 #define SCB_AIRCR_VECTKEY_Pos              16
00413 #define SCB_AIRCR_VECTKEY_Msk              (0xFFFFUL « SCB_AIRCR_VECTKEY_Pos)
00415 #define SCB_AIRCR_VECTKEYSTAT_Pos          16
00416 #define SCB_AIRCR_VECTKEYSTAT_Msk          (0xFFFFUL « SCB_AIRCR_VECTKEYSTAT_Pos)
00418 #define SCB_AIRCR_ENDIANESS_Pos            15
00419 #define SCB_AIRCR_ENDIANESS_Msk            (1UL « SCB_AIRCR_ENDIANESS_Pos)
00421 #define SCB_AIRCR_PRIGROUP_Pos              8
00422 #define SCB_AIRCR_PRIGROUP_Msk             (7UL « SCB_AIRCR_PRIGROUP_Pos)
00424 #define SCB_AIRCR_SYSRESETREQ_Pos           2
00425 #define SCB_AIRCR_SYSRESETREQ_Msk          (1UL « SCB_AIRCR_SYSRESETREQ_Pos)
00427 #define SCB_AIRCR_VECTCLRACTIVE_Pos         1
00428 #define SCB_AIRCR_VECTCLRACTIVE_Msk        (1UL « SCB_AIRCR_VECTCLRACTIVE_Pos)
00430 #define SCB_AIRCR_VECTRESET_Pos             0
00431 #define SCB_AIRCR_VECTRESET_Msk            (1UL « SCB_AIRCR_VECTRESET_Pos)
00433 /* SCB System Control Register Definitions */
00434 #define SCB_SCR_SEVONPEND_Pos               4
00435 #define SCB_SCR_SEVONPEND_Msk              (1UL « SCB_SCR_SEVONPEND_Pos)
00437 #define SCB_SCR_SLEEPDEEP_Pos               2
00438 #define SCB_SCR_SLEEPDEEP_Msk              (1UL « SCB_SCR_SLEEPDEEP_Pos)
00440 #define SCB_SCR_SLEEPONEXIT_Pos             1
00441 #define SCB_SCR_SLEEPONEXIT_Msk            (1UL « SCB_SCR_SLEEPONEXIT_Pos)
00443 /* SCB Configuration Control Register Definitions */
00444 #define SCB_CCR_STKALIGN_Pos                9
00445 #define SCB_CCR_STKALIGN_Msk               (1UL « SCB_CCR_STKALIGN_Pos)
00447 #define SCB_CCR_BFHFNMIGN_Pos               8
00448 #define SCB_CCR_BFHFNMIGN_Msk              (1UL « SCB_CCR_BFHFNMIGN_Pos)
00450 #define SCB_CCR_DIV_0_TRP_Pos               4
00451 #define SCB_CCR_DIV_0_TRP_Msk              (1UL « SCB_CCR_DIV_0_TRP_Pos)
00453 #define SCB_CCR_UNALIGN_TRP_Pos             3
00454 #define SCB_CCR_UNALIGN_TRP_Msk            (1UL « SCB_CCR_UNALIGN_TRP_Pos)
00456 #define SCB_CCR_USERSETMPEND_Pos            1
00457 #define SCB_CCR_USERSETMPEND_Msk           (1UL « SCB_CCR_USERSETMPEND_Pos)
00459 #define SCB_CCR_NONBASETHRDENA_Pos          0
00460 #define SCB_CCR_NONBASETHRDENA_Msk         (1UL « SCB_CCR_NONBASETHRDENA_Pos)
00462 /* SCB System Handler Control and State Register Definitions */
00463 #define SCB_SHCSR_USGFAULTENA_Pos          18
00464 #define SCB_SHCSR_USGFAULTENA_Msk          (1UL « SCB_SHCSR_USGFAULTENA_Pos)
00466 #define SCB_SHCSR_BUSFAULTENA_Pos          17
00467 #define SCB_SHCSR_BUSFAULTENA_Msk          (1UL « SCB_SHCSR_BUSFAULTENA_Pos)
00469 #define SCB_SHCSR_MEMFAULTENA_Pos          16
00470 #define SCB_SHCSR_MEMFAULTENA_Msk          (1UL « SCB_SHCSR_MEMFAULTENA_Pos)
00472 #define SCB_SHCSR_SVCALLPENDED_Pos         15
00473 #define SCB_SHCSR_SVCALLPENDED_Msk         (1UL « SCB_SHCSR_SVCALLPENDED_Pos)
00475 #define SCB_SHCSR_BUSFAULTPENDED_Pos       14
00476 #define SCB_SHCSR_BUSFAULTPENDED_Msk       (1UL « SCB_SHCSR_BUSFAULTPENDED_Pos)
00478 #define SCB_SHCSR_MEMFAULTPENDED_Pos       13
00479 #define SCB_SHCSR_MEMFAULTPENDED_Msk       (1UL « SCB_SHCSR_MEMFAULTPENDED_Pos)
00481 #define SCB_SHCSR_USGFAULTPENDED_Pos       12
00482 #define SCB_SHCSR_USGFAULTPENDED_Msk       (1UL « SCB_SHCSR_USGFAULTPENDED_Pos)
00484 #define SCB_SHCSR_SYSTICKACT_Pos           11
00485 #define SCB_SHCSR_SYSTICKACT_Msk           (1UL « SCB_SHCSR_SYSTICKACT_Pos)
00487 #define SCB_SHCSR_PENDSVACT_Pos            10
00488 #define SCB_SHCSR_PENDSVACT_Msk            (1UL « SCB_SHCSR_PENDSVACT_Pos)
00490 #define SCB_SHCSR_MONITORACT_Pos            8
00491 #define SCB_SHCSR_MONITORACT_Msk           (1UL « SCB_SHCSR_MONITORACT_Pos)
00493 #define SCB_SHCSR_SVCALLACT_Pos             7
00494 #define SCB_SHCSR_SVCALLACT_Msk            (1UL « SCB_SHCSR_SVCALLACT_Pos)
00496 #define SCB_SHCSR_USGFAULTACT_Pos           3
00497 #define SCB_SHCSR_USGFAULTACT_Msk          (1UL « SCB_SHCSR_USGFAULTACT_Pos)
00499 #define SCB_SHCSR_BUSFAULTACT_Pos           1
00500 #define SCB_SHCSR_BUSFAULTACT_Msk          (1UL « SCB_SHCSR_BUSFAULTACT_Pos)
00502 #define SCB_SHCSR_MEMFAULTACT_Pos           0
00503 #define SCB_SHCSR_MEMFAULTACT_Msk          (1UL « SCB_SHCSR_MEMFAULTACT_Pos)
00505 /* SCB Configurable Fault Status Registers Definitions */
00506 #define SCB_CFSR_USGFAULTSR_Pos            16
00507 #define SCB_CFSR_USGFAULTSR_Msk            (0xFFFFUL « SCB_CFSR_USGFAULTSR_Pos)
00509 #define SCB_CFSR_BUSFAULTSR_Pos             8
00510 #define SCB_CFSR_BUSFAULTSR_Msk            (0xFFUL « SCB_CFSR_BUSFAULTSR_Pos)
```

```
00512 #define SCB_CFSR_MEMFAULTSR_Pos             0
00513 #define SCB_CFSR_MEMFAULTSR_Msk            (0xFFUL « SCB_CFSR_MEMFAULTSR_Pos)
00515 /* SCB Hard Fault Status Registers Definitions */
00516 #define SCB_HFSR_DEBUGEVT_Pos              31
00517 #define SCB_HFSR_DEBUGEVT_Msk              (1UL « SCB_HFSR_DEBUGEVT_Pos)
00519 #define SCB_HFSR_FORCED_Pos                30
00520 #define SCB_HFSR_FORCED_Msk                (1UL « SCB_HFSR_FORCED_Pos)
00522 #define SCB_HFSR_VECTTBL_Pos                1
00523 #define SCB_HFSR_VECTTBL_Msk               (1UL « SCB_HFSR_VECTTBL_Pos)
00525 /* SCB Debug Fault Status Register Definitions */
00526 #define SCB_DFSR_EXTERNAL_Pos               4
00527 #define SCB_DFSR_EXTERNAL_Msk              (1UL « SCB_DFSR_EXTERNAL_Pos)
00529 #define SCB_DFSR_VCATCH_Pos                 3
00530 #define SCB_DFSR_VCATCH_Msk                (1UL « SCB_DFSR_VCATCH_Pos)
00532 #define SCB_DFSR_DWTTRAP_Pos                2
00533 #define SCB_DFSR_DWTTRAP_Msk               (1UL « SCB_DFSR_DWTTRAP_Pos)
00535 #define SCB_DFSR_BKPT_Pos                   1
00536 #define SCB_DFSR_BKPT_Msk                  (1UL « SCB_DFSR_BKPT_Pos)
00538 #define SCB_DFSR_HALTED_Pos                 0
00539 #define SCB_DFSR_HALTED_Msk                (1UL « SCB_DFSR_HALTED_Pos)
00552 typedef struct
00553 {
00554       uint32_t RESERVED0[1];
00555   __I  uint32_t ICTR;
00556   __IO uint32_t ACTLR;
00557 } SCnSCB_Type;
00558
00559 /* Interrupt Controller Type Register Definitions */
00560 #define SCnSCB_ICTR_INTLINESNUM_Pos         0
00561 #define SCnSCB_ICTR_INTLINESNUM_Msk        (0xFUL « SCnSCB_ICTR_INTLINESNUM_Pos)
00563 /* Auxiliary Control Register Definitions */
00564 #define SCnSCB_ACTLR_DISOOFP_Pos            9
00565 #define SCnSCB_ACTLR_DISOOFP_Msk           (1UL « SCnSCB_ACTLR_DISOOFP_Pos)
00567 #define SCnSCB_ACTLR_DISFPCA_Pos            8
00568 #define SCnSCB_ACTLR_DISFPCA_Msk           (1UL « SCnSCB_ACTLR_DISFPCA_Pos)
00570 #define SCnSCB_ACTLR_DISFOLD_Pos            2
00571 #define SCnSCB_ACTLR_DISFOLD_Msk           (1UL « SCnSCB_ACTLR_DISFOLD_Pos)
00573 #define SCnSCB_ACTLR_DISDEFWBUF_Pos         1
00574 #define SCnSCB_ACTLR_DISDEFWBUF_Msk        (1UL « SCnSCB_ACTLR_DISDEFWBUF_Pos)
00576 #define SCnSCB_ACTLR_DISMCYCINT_Pos         0
00577 #define SCnSCB_ACTLR_DISMCYCINT_Msk        (1UL « SCnSCB_ACTLR_DISMCYCINT_Pos)
00590 typedef struct
00591 {
00592   __IO uint32_t CTRL;
00593   __IO uint32_t LOAD;
00594   __IO uint32_t VAL;
00595   __I  uint32_t CALIB;
00596 } SysTick_Type;
00597
00598 /* SysTick Control / Status Register Definitions */
00599 #define SysTick_CTRL_COUNTFLAG_Pos         16
00600 #define SysTick_CTRL_COUNTFLAG_Msk         (1UL « SysTick_CTRL_COUNTFLAG_Pos)
00602 #define SysTick_CTRL_CLKSOURCE_Pos          2
00603 #define SysTick_CTRL_CLKSOURCE_Msk         (1UL « SysTick_CTRL_CLKSOURCE_Pos)
00605 #define SysTick_CTRL_TICKINT_Pos            1
00606 #define SysTick_CTRL_TICKINT_Msk           (1UL « SysTick_CTRL_TICKINT_Pos)
00608 #define SysTick_CTRL_ENABLE_Pos             0
00609 #define SysTick_CTRL_ENABLE_Msk            (1UL « SysTick_CTRL_ENABLE_Pos)
00611 /* SysTick Reload Register Definitions */
00612 #define SysTick_LOAD_RELOAD_Pos             0
00613 #define SysTick_LOAD_RELOAD_Msk            (0xFFFFFFUL « SysTick_LOAD_RELOAD_Pos)
00615 /* SysTick Current Register Definitions */
00616 #define SysTick_VAL_CURRENT_Pos             0
00617 #define SysTick_VAL_CURRENT_Msk            (0xFFFFFFUL « SysTick_VAL_CURRENT_Pos)
00619 /* SysTick Calibration Register Definitions */
00620 #define SysTick_CALIB_NOREF_Pos            31
00621 #define SysTick_CALIB_NOREF_Msk            (1UL « SysTick_CALIB_NOREF_Pos)
00623 #define SysTick_CALIB_SKEW_Pos             30
00624 #define SysTick_CALIB_SKEW_Msk             (1UL « SysTick_CALIB_SKEW_Pos)
00626 #define SysTick_CALIB_TENMS_Pos             0
00627 #define SysTick_CALIB_TENMS_Msk            (0xFFFFFFUL « SysTick_VAL_CURRENT_Pos)
00640 typedef struct
00641 {
00642   __O  union
00643   {
00644     __O  uint8_t    u8;
00645     __O  uint16_t   u16;
00646     __O  uint32_t   u32;
00647   }  PORT [32];
00648       uint32_t RESERVED0[864];
00649   __IO uint32_t TER;
00650       uint32_t RESERVED1[15];
00651   __IO uint32_t TPR;
00652       uint32_t RESERVED2[15];
00653   __IO uint32_t TCR;
00654 } ITM_Type;
00655
```

```
00656 /* ITM Trace Privilege Register Definitions */
00657 #define ITM_TPR_PRIVMASK_Pos                0
00658 #define ITM_TPR_PRIVMASK_Msk               (0xFUL << ITM_TPR_PRIVMASK_Pos)
00660 /* ITM Trace Control Register Definitions */
00661 #define ITM_TCR_BUSY_Pos                   23
00662 #define ITM_TCR_BUSY_Msk                   (1UL << ITM_TCR_BUSY_Pos)
00664 #define ITM_TCR_TraceBusID_Pos             16
00665 #define ITM_TCR_TraceBusID_Msk             (0x7FUL << ITM_TCR_TraceBusID_Pos)
00667 #define ITM_TCR_GTSFREQ_Pos                10
00668 #define ITM_TCR_GTSFREQ_Msk                (3UL << ITM_TCR_GTSFREQ_Pos)
00670 #define ITM_TCR_TSPrescale_Pos              8
00671 #define ITM_TCR_TSPrescale_Msk             (3UL << ITM_TCR_TSPrescale_Pos)
00673 #define ITM_TCR_SWOENA_Pos                  4
00674 #define ITM_TCR_SWOENA_Msk                 (1UL << ITM_TCR_SWOENA_Pos)
00676 #define ITM_TCR_TXENA_Pos                   3
00677 #define ITM_TCR_TXENA_Msk                  (1UL << ITM_TCR_TXENA_Pos)
00679 #define ITM_TCR_SYNCENA_Pos                 2
00680 #define ITM_TCR_SYNCENA_Msk                (1UL << ITM_TCR_SYNCENA_Pos)
00682 #define ITM_TCR_TSENA_Pos                   1
00683 #define ITM_TCR_TSENA_Msk                  (1UL << ITM_TCR_TSENA_Pos)
00685 #define ITM_TCR_ITMENA_Pos                  0
00686 #define ITM_TCR_ITMENA_Msk                 (1UL << ITM_TCR_ITMENA_Pos)                  /* end of group
      CMSIS_ITM */
00689
00690
00691 #if (__MPU_PRESENT == 1)
00700 typedef struct
00701 {
00702   __I  uint32_t TYPE;
00703   __IO uint32_t CTRL;
00704   __IO uint32_t RNR;
00705   __IO uint32_t RBAR;
00706   __IO uint32_t RASR;
00707   __IO uint32_t RBAR_A1;
00708   __IO uint32_t RASR_A1;
00709   __IO uint32_t RBAR_A2;
00710   __IO uint32_t RASR_A2;
00711   __IO uint32_t RBAR_A3;
00712   __IO uint32_t RASR_A3;
00713 } MPU_Type;
00714
00715 /* MPU Type Register */
00716 #define MPU_TYPE_IREGION_Pos               16
00717 #define MPU_TYPE_IREGION_Msk               (0xFFUL << MPU_TYPE_IREGION_Pos)
00719 #define MPU_TYPE_DREGION_Pos                8
00720 #define MPU_TYPE_DREGION_Msk               (0xFFUL << MPU_TYPE_DREGION_Pos)
00722 #define MPU_TYPE_SEPARATE_Pos               0
00723 #define MPU_TYPE_SEPARATE_Msk              (1UL << MPU_TYPE_SEPARATE_Pos)
00725 /* MPU Control Register */
00726 #define MPU_CTRL_PRIVDEFENA_Pos             2
00727 #define MPU_CTRL_PRIVDEFENA_Msk            (1UL << MPU_CTRL_PRIVDEFENA_Pos)
00729 #define MPU_CTRL_HFNMIENA_Pos               1
00730 #define MPU_CTRL_HFNMIENA_Msk              (1UL << MPU_CTRL_HFNMIENA_Pos)
00732 #define MPU_CTRL_ENABLE_Pos                 0
00733 #define MPU_CTRL_ENABLE_Msk                (1UL << MPU_CTRL_ENABLE_Pos)
00735 /* MPU Region Number Register */
00736 #define MPU_RNR_REGION_Pos                  0
00737 #define MPU_RNR_REGION_Msk                 (0xFFUL << MPU_RNR_REGION_Pos)
00739 /* MPU Region Base Address Register */
00740 #define MPU_RBAR_ADDR_Pos                   5
00741 #define MPU_RBAR_ADDR_Msk                  (0x7FFFFFFUL << MPU_RBAR_ADDR_Pos)
00743 #define MPU_RBAR_VALID_Pos                  4
00744 #define MPU_RBAR_VALID_Msk                 (1UL << MPU_RBAR_VALID_Pos)
00746 #define MPU_RBAR_REGION_Pos                 0
00747 #define MPU_RBAR_REGION_Msk                (0xFUL << MPU_RBAR_REGION_Pos)
00749 /* MPU Region Attribute and Size Register */
00750 #define MPU_RASR_ATTRS_Pos                 16
00751 #define MPU_RASR_ATTRS_Msk                 (0xFFFFUL << MPU_RASR_ATTRS_Pos)
00753 #define MPU_RASR_SRD_Pos                    8
00754 #define MPU_RASR_SRD_Msk                   (0xFFUL << MPU_RASR_SRD_Pos)
00756 #define MPU_RASR_SIZE_Pos                   1
00757 #define MPU_RASR_SIZE_Msk                  (0x1FUL << MPU_RASR_SIZE_Pos)
00759 #define MPU_RASR_ENABLE_Pos                 0
00760 #define MPU_RASR_ENABLE_Msk                (1UL << MPU_RASR_ENABLE_Pos)
00763 #endif
00764
00765
00766 #if (__FPU_PRESENT == 1)
00775 typedef struct
00776 {
00777       uint32_t RESERVED0[1];
00778   __IO uint32_t FPCCR;
00779   __IO uint32_t FPCAR;
00780   __IO uint32_t FPDSCR;
00781   __I  uint32_t MVFR0;
00782   __I  uint32_t MVFR1;
00783 } FPU_Type;
```

```
00784
00785 /* Floating-Point Context Control Register */
00786 #define FPU_FPCCR_ASPEN_Pos                 31
00787 #define FPU_FPCCR_ASPEN_Msk                 (1UL << FPU_FPCCR_ASPEN_Pos)
00789 #define FPU_FPCCR_LSPEN_Pos                 30
00790 #define FPU_FPCCR_LSPEN_Msk                 (1UL << FPU_FPCCR_LSPEN_Pos)
00792 #define FPU_FPCCR_MONRDY_Pos                 8
00793 #define FPU_FPCCR_MONRDY_Msk                (1UL << FPU_FPCCR_MONRDY_Pos)
00795 #define FPU_FPCCR_BFRDY_Pos                  6
00796 #define FPU_FPCCR_BFRDY_Msk                 (1UL << FPU_FPCCR_BFRDY_Pos)
00798 #define FPU_FPCCR_MMRDY_Pos                  5
00799 #define FPU_FPCCR_MMRDY_Msk                 (1UL << FPU_FPCCR_MMRDY_Pos)
00801 #define FPU_FPCCR_HFRDY_Pos                  4
00802 #define FPU_FPCCR_HFRDY_Msk                 (1UL << FPU_FPCCR_HFRDY_Pos)
00804 #define FPU_FPCCR_THREAD_Pos                 3
00805 #define FPU_FPCCR_THREAD_Msk                (1UL << FPU_FPCCR_THREAD_Pos)
00807 #define FPU_FPCCR_USER_Pos                   1
00808 #define FPU_FPCCR_USER_Msk                  (1UL << FPU_FPCCR_USER_Pos)
00810 #define FPU_FPCCR_LSPACT_Pos                 0
00811 #define FPU_FPCCR_LSPACT_Msk                (1UL << FPU_FPCCR_LSPACT_Pos)
00813 /* Floating-Point Context Address Register */
00814 #define FPU_FPCAR_ADDRESS_Pos                3
00815 #define FPU_FPCAR_ADDRESS_Msk               (0x1FFFFFFFUL << FPU_FPCAR_ADDRESS_Pos)
00817 /* Floating-Point Default Status Control Register */
00818 #define FPU_FPDSCR_AHP_Pos                  26
00819 #define FPU_FPDSCR_AHP_Msk                  (1UL << FPU_FPDSCR_AHP_Pos)
00821 #define FPU_FPDSCR_DN_Pos                   25
00822 #define FPU_FPDSCR_DN_Msk                   (1UL << FPU_FPDSCR_DN_Pos)
00824 #define FPU_FPDSCR_FZ_Pos                   24
00825 #define FPU_FPDSCR_FZ_Msk                   (1UL << FPU_FPDSCR_FZ_Pos)
00827 #define FPU_FPDSCR_RMode_Pos                22
00828 #define FPU_FPDSCR_RMode_Msk                (3UL << FPU_FPDSCR_RMode_Pos)
00830 /* Media and FP Feature Register 0 */
00831 #define FPU_MVFR0_FP_rounding_modes_Pos     28
00832 #define FPU_MVFR0_FP_rounding_modes_Msk     (0xFUL << FPU_MVFR0_FP_rounding_modes_Pos)
00834 #define FPU_MVFR0_Short_vectors_Pos         24
00835 #define FPU_MVFR0_Short_vectors_Msk         (0xFUL << FPU_MVFR0_Short_vectors_Pos)
00837 #define FPU_MVFR0_Square_root_Pos           20
00838 #define FPU_MVFR0_Square_root_Msk           (0xFUL << FPU_MVFR0_Square_root_Pos)
00840 #define FPU_MVFR0_Divide_Pos                16
00841 #define FPU_MVFR0_Divide_Msk                (0xFUL << FPU_MVFR0_Divide_Pos)
00843 #define FPU_MVFR0_FP_excep_trapping_Pos     12
00844 #define FPU_MVFR0_FP_excep_trapping_Msk     (0xFUL << FPU_MVFR0_FP_excep_trapping_Pos)
00846 #define FPU_MVFR0_Double_precision_Pos       8
00847 #define FPU_MVFR0_Double_precision_Msk      (0xFUL << FPU_MVFR0_Double_precision_Pos)
00849 #define FPU_MVFR0_Single_precision_Pos       4
00850 #define FPU_MVFR0_Single_precision_Msk      (0xFUL << FPU_MVFR0_Single_precision_Pos)
00852 #define FPU_MVFR0_A_SIMD_registers_Pos       0
00853 #define FPU_MVFR0_A_SIMD_registers_Msk      (0xFUL << FPU_MVFR0_A_SIMD_registers_Pos)
00855 /* Media and FP Feature Register 1 */
00856 #define FPU_MVFR1_FP_fused_MAC_Pos          28
00857 #define FPU_MVFR1_FP_fused_MAC_Msk          (0xFUL << FPU_MVFR1_FP_fused_MAC_Pos)
00859 #define FPU_MVFR1_FP_HPFP_Pos               24
00860 #define FPU_MVFR1_FP_HPFP_Msk               (0xFUL << FPU_MVFR1_FP_HPFP_Pos)
00862 #define FPU_MVFR1_D_NaN_mode_Pos             4
00863 #define FPU_MVFR1_D_NaN_mode_Msk            (0xFUL << FPU_MVFR1_D_NaN_mode_Pos)
00865 #define FPU_MVFR1_FtZ_mode_Pos               0
00866 #define FPU_MVFR1_FtZ_mode_Msk              (0xFUL << FPU_MVFR1_FtZ_mode_Pos)
00869 #endif
00870
00871
00880 typedef struct
00881 {
00882   __IO uint32_t DHCSR;
00883   __O  uint32_t DCRSR;
00884   __IO uint32_t DCRDR;
00885   __IO uint32_t DEMCR;
00886 } CoreDebug_Type;
00887
00888 /* Debug Halting Control and Status Register */
00889 #define CoreDebug_DHCSR_DBGKEY_Pos          16
00890 #define CoreDebug_DHCSR_DBGKEY_Msk          (0xFFFFUL << CoreDebug_DHCSR_DBGKEY_Pos)
00892 #define CoreDebug_DHCSR_S_RESET_ST_Pos      25
00893 #define CoreDebug_DHCSR_S_RESET_ST_Msk      (1UL << CoreDebug_DHCSR_S_RESET_ST_Pos)
00895 #define CoreDebug_DHCSR_S_RETIRE_ST_Pos     24
00896 #define CoreDebug_DHCSR_S_RETIRE_ST_Msk     (1UL << CoreDebug_DHCSR_S_RETIRE_ST_Pos)
00898 #define CoreDebug_DHCSR_S_LOCKUP_Pos        19
00899 #define CoreDebug_DHCSR_S_LOCKUP_Msk        (1UL << CoreDebug_DHCSR_S_LOCKUP_Pos)
00901 #define CoreDebug_DHCSR_S_SLEEP_Pos         18
00902 #define CoreDebug_DHCSR_S_SLEEP_Msk         (1UL << CoreDebug_DHCSR_S_SLEEP_Pos)
00904 #define CoreDebug_DHCSR_S_HALT_Pos          17
00905 #define CoreDebug_DHCSR_S_HALT_Msk          (1UL << CoreDebug_DHCSR_S_HALT_Pos)
00907 #define CoreDebug_DHCSR_S_REGRDY_Pos        16
00908 #define CoreDebug_DHCSR_S_REGRDY_Msk        (1UL << CoreDebug_DHCSR_S_REGRDY_Pos)
00910 #define CoreDebug_DHCSR_C_SNAPSTALL_Pos      5
00911 #define CoreDebug_DHCSR_C_SNAPSTALL_Msk     (1UL << CoreDebug_DHCSR_C_SNAPSTALL_Pos)
00913 #define CoreDebug_DHCSR_C_MASKINTS_Pos       3
```

```
00914 #define CoreDebug_DHCSR_C_MASKINTS_Msk     (1UL « CoreDebug_DHCSR_C_MASKINTS_Pos)
00916 #define CoreDebug_DHCSR_C_STEP_Pos          2
00917 #define CoreDebug_DHCSR_C_STEP_Msk         (1UL « CoreDebug_DHCSR_C_STEP_Pos)
00919 #define CoreDebug_DHCSR_C_HALT_Pos          1
00920 #define CoreDebug_DHCSR_C_HALT_Msk         (1UL « CoreDebug_DHCSR_C_HALT_Pos)
00922 #define CoreDebug_DHCSR_C_DEBUGEN_Pos       0
00923 #define CoreDebug_DHCSR_C_DEBUGEN_Msk      (1UL « CoreDebug_DHCSR_C_DEBUGEN_Pos)
00925 /* Debug Core Register Selector Register */
00926 #define CoreDebug_DCRSR_REGWnR_Pos         16
00927 #define CoreDebug_DCRSR_REGWnR_Msk         (1UL « CoreDebug_DCRSR_REGWnR_Pos)
00929 #define CoreDebug_DCRSR_REGSEL_Pos          0
00930 #define CoreDebug_DCRSR_REGSEL_Msk         (0x1FUL « CoreDebug_DCRSR_REGSEL_Pos)
00932 /* Debug Exception and Monitor Control Register */
00933 #define CoreDebug_DEMCR_TRCENA_Pos         24
00934 #define CoreDebug_DEMCR_TRCENA_Msk         (1UL « CoreDebug_DEMCR_TRCENA_Pos)
00936 #define CoreDebug_DEMCR_MON_REQ_Pos        19
00937 #define CoreDebug_DEMCR_MON_REQ_Msk        (1UL « CoreDebug_DEMCR_MON_REQ_Pos)
00939 #define CoreDebug_DEMCR_MON_STEP_Pos       18
00940 #define CoreDebug_DEMCR_MON_STEP_Msk       (1UL « CoreDebug_DEMCR_MON_STEP_Pos)
00942 #define CoreDebug_DEMCR_MON_PEND_Pos       17
00943 #define CoreDebug_DEMCR_MON_PEND_Msk       (1UL « CoreDebug_DEMCR_MON_PEND_Pos)
00945 #define CoreDebug_DEMCR_MON_EN_Pos         16
00946 #define CoreDebug_DEMCR_MON_EN_Msk         (1UL « CoreDebug_DEMCR_MON_EN_Pos)
00948 #define CoreDebug_DEMCR_VC_HARDERR_Pos     10
00949 #define CoreDebug_DEMCR_VC_HARDERR_Msk     (1UL « CoreDebug_DEMCR_VC_HARDERR_Pos)
00951 #define CoreDebug_DEMCR_VC_INTERR_Pos       9
00952 #define CoreDebug_DEMCR_VC_INTERR_Msk      (1UL « CoreDebug_DEMCR_VC_INTERR_Pos)
00954 #define CoreDebug_DEMCR_VC_BUSERR_Pos       8
00955 #define CoreDebug_DEMCR_VC_BUSERR_Msk      (1UL « CoreDebug_DEMCR_VC_BUSERR_Pos)
00957 #define CoreDebug_DEMCR_VC_STATERR_Pos      7
00958 #define CoreDebug_DEMCR_VC_STATERR_Msk     (1UL « CoreDebug_DEMCR_VC_STATERR_Pos)
00960 #define CoreDebug_DEMCR_VC_CHKERR_Pos       6
00961 #define CoreDebug_DEMCR_VC_CHKERR_Msk      (1UL « CoreDebug_DEMCR_VC_CHKERR_Pos)
00963 #define CoreDebug_DEMCR_VC_NOCPERR_Pos      5
00964 #define CoreDebug_DEMCR_VC_NOCPERR_Msk     (1UL « CoreDebug_DEMCR_VC_NOCPERR_Pos)
00966 #define CoreDebug_DEMCR_VC_MMERR_Pos        4
00967 #define CoreDebug_DEMCR_VC_MMERR_Msk       (1UL « CoreDebug_DEMCR_VC_MMERR_Pos)
00969 #define CoreDebug_DEMCR_VC_CORERESET_Pos    0
00970 #define CoreDebug_DEMCR_VC_CORERESET_Msk   (1UL « CoreDebug_DEMCR_VC_CORERESET_Pos)
00979 /* Memory mapping of Cortex-M4 Hardware */
00980 #define SCS_BASE            (0xE000E000UL)
00981 #define ITM_BASE            (0xE0000000UL)
00982 #define CoreDebug_BASE      (0xE000EDF0UL)
00983 #define SysTick_BASE        (SCS_BASE +  0x0010UL)
00984 #define NVIC_BASE           (SCS_BASE +  0x0100UL)
00985 #define SCB_BASE            (SCS_BASE +  0x0D00UL)
00987 #define SCnSCB              ((SCnSCB_Type    *)     SCS_BASE      )
00988 #define SCB                 ((SCB_Type       *)     SCB_BASE      )
00989 #define SysTick             ((SysTick_Type   *)     SysTick_BASE  )
00990 #define NVIC                ((NVIC_Type      *)     NVIC_BASE     )
00991 #define ITM                 ((ITM_Type       *)     ITM_BASE      )
00992 #define CoreDebug           ((CoreDebug_Type *)     CoreDebug_BASE)
00994 #if (__MPU_PRESENT == 1)
00995   #define MPU_BASE          (SCS_BASE +  0x0D90UL)
00996   #define MPU               ((MPU_Type       *)     MPU_BASE      )
00997 #endif
00998
00999 #if (__FPU_PRESENT == 1)
01000   #define FPU_BASE          (SCS_BASE +  0x0F30UL)
01001   #define FPU               ((FPU_Type       *)     FPU_BASE      )
01002 #endif
01003
01008 /*******************************************************************************
01009  *                Hardware Abstraction Layer
01010  ******************************************************************************/
01021 /* ##########################   NVIC functions  #################################### */
01037 static __INLINE void NVIC_SetPriorityGrouping(uint32_t PriorityGroup)
01038 {
01039   uint32_t reg_value;
01040   uint32_t PriorityGroupTmp = (PriorityGroup & (uint32_t)0x07);               /* only values 0..7 are
      used          */
01041
01042   reg_value  =  SCB->AIRCR;                                                   /* read old register
      configuration    */
01043   reg_value &= ~(SCB_AIRCR_VECTKEY_Msk | SCB_AIRCR_PRIGROUP_Msk);            /* clear bits to change
      */
01044   reg_value  =  (reg_value                                  |
01045                 ((uint32_t)0x5FA « SCB_AIRCR_VECTKEY_Pos) |
01046                 (PriorityGroupTmp « 8));                                     /* Insert write key and
      priorty group */
01047   SCB->AIRCR =  reg_value;
01048 }
01049
01050
01058 static __INLINE uint32_t NVIC_GetPriorityGrouping(void)
01059 {
01060   return ((SCB->AIRCR & SCB_AIRCR_PRIGROUP_Msk) » SCB_AIRCR_PRIGROUP_Pos);   /* read priority grouping
```

```
      field */
01061 }
01062
01063
01071 static __INLINE void NVIC_EnableIRQ(IRQn_Type IRQn)
01072 {
01073 /*  NVIC->ISER[((uint32_t)(IRQn) >> 5)] = (1 << ((uint32_t)(IRQn) & 0x1F));  enable interrupt */
01074   NVIC->ISER[(uint32_t)((int32_t)IRQn) >> 5] = (uint32_t)(1 << ((uint32_t)((int32_t)IRQn) &
      (uint32_t)0x1F)); /* enable interrupt */
01075 }
01076
01077
01085 static __INLINE void NVIC_DisableIRQ(IRQn_Type IRQn)
01086 {
01087   NVIC->ICER[((uint32_t)(IRQn) >> 5)] = (1 << ((uint32_t)(IRQn) & 0x1F)); /* disable interrupt */
01088 }
01089
01090
01100 static __INLINE uint32_t NVIC_GetPendingIRQ(IRQn_Type IRQn)
01101 {
01102   return((uint32_t) ((NVIC->ISPR[(uint32_t)(IRQn) >> 5] & (1 << ((uint32_t)(IRQn) & 0x1F)))?1:0)); /*
      Return 1 if pending else 0 */
01103 }
01104
01105
01113 static __INLINE void NVIC_SetPendingIRQ(IRQn_Type IRQn)
01114 {
01115   NVIC->ISPR[((uint32_t)(IRQn) >> 5)] = (1 << ((uint32_t)(IRQn) & 0x1F)); /* set interrupt pending */
01116 }
01117
01118
01126 static __INLINE void NVIC_ClearPendingIRQ(IRQn_Type IRQn)
01127 {
01128   NVIC->ICPR[((uint32_t)(IRQn) >> 5)] = (1 << ((uint32_t)(IRQn) & 0x1F)); /* Clear pending interrupt */
01129 }
01130
01131
01139 static __INLINE uint32_t NVIC_GetActive(IRQn_Type IRQn)
01140 {
01141   return((uint32_t)((NVIC->IABR[(uint32_t)(IRQn) >> 5] & (1 << ((uint32_t)(IRQn) & 0x1F)))?1:0)); /*
      Return 1 if active else 0 */
01142 }
01143
01144
01156 static __INLINE void NVIC_SetPriority(IRQn_Type IRQn, uint32_t priority)
01157 {
01158   if(IRQn < 0) {
01159     SCB->SHP[((uint32_t)(IRQn) & 0xF)-4] = ((priority << (8 - __NVIC_PRIO_BITS)) & 0xff); } /* set
      Priority for Cortex-M  System Interrupts */
01160   else {
01161     NVIC->IP[(uint32_t)(IRQn)] = ((priority << (8 - __NVIC_PRIO_BITS)) & 0xff);    }        /* set
      Priority for device specific Interrupts  */
01162 }
01163
01164
01177 static __INLINE uint32_t NVIC_GetPriority(IRQn_Type IRQn)
01178 {
01179
01180   if(IRQn < 0) {
01181     return((uint32_t)(SCB->SHP[((uint32_t)(IRQn) & 0xF)-4] >> (8 - __NVIC_PRIO_BITS)));  } /* get
      priority for Cortex-M  system interrupts */
01182   else {
01183     return((uint32_t)(NVIC->IP[(uint32_t)(IRQn)]           >> (8 - __NVIC_PRIO_BITS)));  } /* get
      priority for device specific interrupts  */
01184 }
01185
01186
01201 static __INLINE uint32_t NVIC_EncodePriority (uint32_t PriorityGroup, uint32_t PreemptPriority,
      uint32_t SubPriority)
01202 {
01203   uint32_t PriorityGroupTmp = (PriorityGroup & 0x07);          /* only values 0..7 are used
      */
01204   uint32_t PreemptPriorityBits;
01205   uint32_t SubPriorityBits;
01206
01207   PreemptPriorityBits = ((7 - PriorityGroupTmp) > __NVIC_PRIO_BITS) ? __NVIC_PRIO_BITS : 7 -
      PriorityGroupTmp;
01208   SubPriorityBits     = ((PriorityGroupTmp + __NVIC_PRIO_BITS) < 7) ? 0 : PriorityGroupTmp - 7 +
      __NVIC_PRIO_BITS;
01209
01210   return (
01211            ((PreemptPriority & ((1 << (PreemptPriorityBits)) - 1)) << SubPriorityBits) |
01212            ((SubPriority     & ((1 << (SubPriorityBits    )) - 1)))
01213          );
01214 }
01215
01216
```

```
01231 static __INLINE void NVIC_DecodePriority (uint32_t Priority, uint32_t PriorityGroup, uint32_t*
      pPreemptPriority, uint32_t* pSubPriority)
01232 {
01233   uint32_t PriorityGroupTmp = (PriorityGroup & 0x07);          /* only values 0..7 are used
      */
01234   uint32_t PreemptPriorityBits;
01235   uint32_t SubPriorityBits;
01236
01237   PreemptPriorityBits = ((7 - PriorityGroupTmp) > __NVIC_PRIO_BITS) ? __NVIC_PRIO_BITS : 7 -
      PriorityGroupTmp;
01238   SubPriorityBits     = ((PriorityGroupTmp + __NVIC_PRIO_BITS) < 7) ? 0 : PriorityGroupTmp - 7 +
      __NVIC_PRIO_BITS;
01239
01240   *pPreemptPriority = (Priority » SubPriorityBits) & ((1 « (PreemptPriorityBits)) - 1);
01241   *pSubPriority     = (Priority                   ) & ((1 « (SubPriorityBits    )) - 1);
01242 }
01243
01244
01249 static __INLINE void NVIC_SystemReset(void)
01250 {
01251   __DSB();                                                     /* Ensure all outstanding memory
      accesses included
01252                                                                   buffered write are completed before
      reset */
01253   SCB->AIRCR  = ((0x5FA « SCB_AIRCR_VECTKEY_Pos)      |
01254                  (SCB->AIRCR & SCB_AIRCR_PRIGROUP_Msk) |
01255                  SCB_AIRCR_SYSRESETREQ_Msk);                   /* Keep priority group unchanged */
01256   __DSB();                                                     /* Ensure completion of memory access
      */
01257   while(1);                                                    /* wait until reset */
01258 }
01259
01264 /* ##################################    SysTick function
      ############################################ */
01270 #if (__Vendor_SysTickConfig == 0)
01271
01281 static __INLINE uint32_t SysTick_Config(uint32_t ticks)
01282 {
01283   if (ticks > SysTick_LOAD_RELOAD_Msk)  return (1);            /* Reload value impossible */
01284
01285   SysTick->LOAD  = (ticks & SysTick_LOAD_RELOAD_Msk) - 1;      /* set reload register */
01286   NVIC_SetPriority (SysTick_IRQn, (1«__NVIC_PRIO_BITS) - 1);  /* set Priority for Cortex-M0 System
      Interrupts */
01287   SysTick->VAL   = 0;                                          /* Load the SysTick Counter Value */
01288   SysTick->CTRL  = SysTick_CTRL_CLKSOURCE_Msk |
01289                    SysTick_CTRL_TICKINT_Msk   |
01290                    SysTick_CTRL_ENABLE_Msk;                   /* Enable SysTick IRQ and SysTick Timer
      */
01291   return (0);                                                  /* Function successful */
01292 }
01293
01294 #endif
01295
01300 /* ##################################### Debug In/Output function
      ########################################### */
01306 extern volatile int32_t ITM_RxBuffer;
01307 #define               ITM_RXBUFFER_EMPTY    0x5AA55AA5
01319 static __INLINE uint32_t ITM_SendChar (uint32_t ch)
01320 {
01321   if ((CoreDebug->DEMCR & CoreDebug_DEMCR_TRCENA_Msk)  &&      /* Trace enabled */
01322       (ITM->TCR & ITM_TCR_ITMENA_Msk)                  &&      /* ITM enabled */
01323       (ITM->TER & (1UL « 0)        )                    )     /* ITM Port #0 enabled */
01324   {
01325     while (ITM->PORT[0].u32 == 0);
01326     ITM->PORT[0].u8 = (uint8_t) ch;
01327   }
01328   return (ch);
01329 }
01330
01331
01341 static __INLINE int32_t ITM_ReceiveChar (void) {
01342   int32_t ch = -1;                           /* no character available */
01343
01344   if (ITM_RxBuffer != ITM_RXBUFFER_EMPTY) {
01345     ch = ITM_RxBuffer;
01346     ITM_RxBuffer = ITM_RXBUFFER_EMPTY;       /* ready for next character */
01347   }
01348
01349   return (ch);
01350 }
01351
01352
01361 static __INLINE int32_t ITM_CheckChar (void) {
01362
01363   if (ITM_RxBuffer == ITM_RXBUFFER_EMPTY) {
01364     return (0);                                /* no character available */
01365   } else {
```

```
01366     return (1);                                   /*    character available */
01367   }
01368 }
01369
01372 #endif /* __CORE_CM4_H_DEPENDANT */
01373
01374 #endif /* __CMSIS_GENERIC */
01375
01376 #ifdef __cplusplus
01377 }
01378 #endif
```

7.9 CUBE_IDE/VGA/Core/Inc/core_cm4_simd.h File Reference

CMSIS Cortex-M4 SIMD Header File.

7.9.1 Detailed Description

CMSIS Cortex-M4 SIMD Header File.

Version

V2.10

Date

19. July 2011

Note

7.10 core_cm4_simd.h

Go to the documentation of this file.

```
00001 /**************************************************************************/
00024 #ifdef __cplusplus
00025  extern "C" {
00026 #endif
00027
00028 #ifndef __CORE_CM4_SIMD_H
00029 #define __CORE_CM4_SIMD_H
00030
00031
00032 /*******************************************************************************
00033  *                Hardware Abstraction Layer
00034  ******************************************************************************/
00035
00036
00037 /* ###################  Compiler specific Intrinsics  ########################### */
00043 #if   defined ( __CC_ARM ) /*------------------RealView Compiler -----------------*/
00044 /* ARM armcc specific functions */
00045
00046 /*------ CM4 SOMD Intrinsics -----------------------------------------------------*/
00047 #define __SADD8                           __sadd8
00048 #define __QADD8                           __qadd8
00049 #define __SHADD8                          __shadd8
00050 #define __UADD8                           __uadd8
00051 #define __UQADD8                          __uqadd8
00052 #define __UHADD8                          __uhadd8
00053 #define __SSUB8                           __ssub8
00054 #define __QSUB8                           __qsub8
00055 #define __SHSUB8                          __shsub8
00056 #define __USUB8                           __usub8
00057 #define __UQSUB8                          __uqsub8
00058 #define __UHSUB8                          __uhsub8
00059 #define __SADD16                          __sadd16
00060 #define __QADD16                          __qadd16
00061 #define __SHADD16                         __shadd16
00062 #define __UADD16                          __uadd16
00063 #define __UQADD16                         __uqadd16
00064 #define __UHADD16                         __uhadd16
00065 #define __SSUB16                          __ssub16
00066 #define __QSUB16                          __qsub16
00067 #define __SHSUB16                         __shsub16
00068 #define __USUB16                          __usub16
00069 #define __UQSUB16                         __uqsub16
00070 #define __UHSUB16                         __uhsub16
00071 #define __SASX                            __sasx
00072 #define __QASX                            __qasx
00073 #define __SHASX                           __shasx
00074 #define __UASX                            __uasx
00075 #define __UQASX                           __uqasx
00076 #define __UHASX                           __uhasx
00077 #define __SSAX                            __ssax
00078 #define __QSAX                            __qsax
00079 #define __SHSAX                           __shsax
00080 #define __USAX                            __usax
00081 #define __UQSAX                           __uqsax
00082 #define __UHSAX                           __uhsax
00083 #define __USAD8                           __usad8
00084 #define __USADA8                          __usada8
00085 #define __SSAT16                          __ssat16
00086 #define __USAT16                          __usat16
00087 #define __UXTB16                          __uxtb16
00088 #define __UXTAB16                         __uxtab16
00089 #define __SXTB16                          __sxtb16
00090 #define __SXTAB16                         __sxtab16
00091 #define __SMUAD                           __smuad
00092 #define __SMUADX                          __smuadx
00093 #define __SMLAD                           __smlad
00094 #define __SMLADX                          __smladx
00095 #define __SMLALD                          __smlald
00096 #define __SMLALDX                         __smlaldx
00097 #define __SMUSD                           __smusd
00098 #define __SMUSDX                          __smusdx
00099 #define __SMLSD                           __smlsd
00100 #define __SMLSDX                          __smlsdx
00101 #define __SMLSLD                          __smlsld
00102 #define __SMLSLDX                         __smlsldx
00103 #define __SEL                             __sel
00104 #define __QADD                            __qadd
00105 #define __QSUB                            __qsub
00106
00107 #define __PKHBT(ARG1,ARG2,ARG3)          ( ((((uint32_t)(ARG1))          ) & 0x0000FFFFUL) |  \
00108                                            ((((uint32_t)(ARG2)) << (ARG3)) & 0xFFFF0000UL)  )
00109
```

```
00110 #define __PKHTB(ARG1,ARG2,ARG3)          ( ((((uint32_t)(ARG1))          ) & 0xFFFF0000UL) |  \
00111                                            ((((uint32_t)(ARG2)) >> (ARG3)) & 0x0000FFFFUL)  )
00112
00113
00114 /*-- End CM4 SIMD Intrinsics -----------------------------------------------------*/
00115
00116
00117
00118 #elif defined ( __ICCARM__ ) /*------------------ ICC Compiler -------------------*/
00119 /* IAR iccarm specific functions */
00120
00121 #include <cmsis_iar.h>
00122
00123 /*------ CM4 SIMDDSP Intrinsics -----------------------------------------------------*/
00124 /* intrinsic __SADD8      see intrinsics.h */
00125 /* intrinsic __QADD8      see intrinsics.h */
00126 /* intrinsic __SHADD8     see intrinsics.h */
00127 /* intrinsic __UADD8      see intrinsics.h */
00128 /* intrinsic __UQADD8     see intrinsics.h */
00129 /* intrinsic __UHADD8     see intrinsics.h */
00130 /* intrinsic __SSUB8      see intrinsics.h */
00131 /* intrinsic __QSUB8      see intrinsics.h */
00132 /* intrinsic __SHSUB8     see intrinsics.h */
00133 /* intrinsic __USUB8      see intrinsics.h */
00134 /* intrinsic __UQSUB8     see intrinsics.h */
00135 /* intrinsic __UHSUB8     see intrinsics.h */
00136 /* intrinsic __SADD16     see intrinsics.h */
00137 /* intrinsic __QADD16     see intrinsics.h */
00138 /* intrinsic __SHADD16    see intrinsics.h */
00139 /* intrinsic __UADD16     see intrinsics.h */
00140 /* intrinsic __UQADD16    see intrinsics.h */
00141 /* intrinsic __UHADD16    see intrinsics.h */
00142 /* intrinsic __SSUB16     see intrinsics.h */
00143 /* intrinsic __QSUB16     see intrinsics.h */
00144 /* intrinsic __SHSUB16    see intrinsics.h */
00145 /* intrinsic __USUB16     see intrinsics.h */
00146 /* intrinsic __UQSUB16    see intrinsics.h */
00147 /* intrinsic __UHSUB16    see intrinsics.h */
00148 /* intrinsic __SASX       see intrinsics.h */
00149 /* intrinsic __QASX       see intrinsics.h */
00150 /* intrinsic __SHASX      see intrinsics.h */
00151 /* intrinsic __UASX       see intrinsics.h */
00152 /* intrinsic __UQASX      see intrinsics.h */
00153 /* intrinsic __UHASX      see intrinsics.h */
00154 /* intrinsic __SSAX       see intrinsics.h */
00155 /* intrinsic __QSAX       see intrinsics.h */
00156 /* intrinsic __SHSAX      see intrinsics.h */
00157 /* intrinsic __USAX       see intrinsics.h */
00158 /* intrinsic __UQSAX      see intrinsics.h */
00159 /* intrinsic __UHSAX      see intrinsics.h */
00160 /* intrinsic __USAD8      see intrinsics.h */
00161 /* intrinsic __USADA8     see intrinsics.h */
00162 /* intrinsic __SSAT16     see intrinsics.h */
00163 /* intrinsic __USAT16     see intrinsics.h */
00164 /* intrinsic __UXTB16     see intrinsics.h */
00165 /* intrinsic __SXTB16     see intrinsics.h */
00166 /* intrinsic __UXTAB16    see intrinsics.h */
00167 /* intrinsic __SXTAB16    see intrinsics.h */
00168 /* intrinsic __SMUAD      see intrinsics.h */
00169 /* intrinsic __SMUADX     see intrinsics.h */
00170 /* intrinsic __SMLAD      see intrinsics.h */
00171 /* intrinsic __SMLADX     see intrinsics.h */
00172 /* intrinsic __SMLALD     see intrinsics.h */
00173 /* intrinsic __SMLALDX    see intrinsics.h */
00174 /* intrinsic __SMUSD      see intrinsics.h */
00175 /* intrinsic __SMUSDX     see intrinsics.h */
00176 /* intrinsic __SMLSD      see intrinsics.h */
00177 /* intrinsic __SMLSDX     see intrinsics.h */
00178 /* intrinsic __SMLSLD     see intrinsics.h */
00179 /* intrinsic __SMLSLDX    see intrinsics.h */
00180 /* intrinsic __SEL        see intrinsics.h */
00181 /* intrinsic __QADD       see intrinsics.h */
00182 /* intrinsic __QSUB       see intrinsics.h */
00183 /* intrinsic __PKHBT      see intrinsics.h */
00184 /* intrinsic __PKHTB      see intrinsics.h */
00185
00186 /*-- End CM4 SIMD Intrinsics -----------------------------------------------------*/
00187
00188
00189
00190 #elif defined ( __GNUC__ ) /*------------------ GNU Compiler ---------------------*/
00191 /* GNU gcc specific functions */
00192
00193 /*------ CM4 SIMD Intrinsics -----------------------------------------------------*/
00194 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SADD8(uint32_t op1, uint32_t op2)
00195 {
00196   uint32_t result;
```

```
00197
00198   __ASM volatile ("sadd8 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00199   return(result);
00200 }
00201
00202 __attribute__( ( always_inline ) ) static __INLINE uint32_t __QADD8(uint32_t op1, uint32_t op2)
00203 {
00204   uint32_t result;
00205
00206   __ASM volatile ("qadd8 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00207   return(result);
00208 }
00209
00210 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SHADD8(uint32_t op1, uint32_t op2)
00211 {
00212   uint32_t result;
00213
00214   __ASM volatile ("shadd8 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00215   return(result);
00216 }
00217
00218 __attribute__( ( always_inline ) ) static __INLINE uint32_t __UADD8(uint32_t op1, uint32_t op2)
00219 {
00220   uint32_t result;
00221
00222   __ASM volatile ("uadd8 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00223   return(result);
00224 }
00225
00226 __attribute__( ( always_inline ) ) static __INLINE uint32_t __UQADD8(uint32_t op1, uint32_t op2)
00227 {
00228   uint32_t result;
00229
00230   __ASM volatile ("uqadd8 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00231   return(result);
00232 }
00233
00234 __attribute__( ( always_inline ) ) static __INLINE uint32_t __UHADD8(uint32_t op1, uint32_t op2)
00235 {
00236   uint32_t result;
00237
00238   __ASM volatile ("uhadd8 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00239   return(result);
00240 }
00241
00242
00243 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SSUB8(uint32_t op1, uint32_t op2)
00244 {
00245   uint32_t result;
00246
00247   __ASM volatile ("ssub8 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00248   return(result);
00249 }
00250
00251 __attribute__( ( always_inline ) ) static __INLINE uint32_t __QSUB8(uint32_t op1, uint32_t op2)
00252 {
00253   uint32_t result;
00254
00255   __ASM volatile ("qsub8 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00256   return(result);
00257 }
00258
00259 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SHSUB8(uint32_t op1, uint32_t op2)
00260 {
00261   uint32_t result;
00262
00263   __ASM volatile ("shsub8 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00264   return(result);
00265 }
00266
00267 __attribute__( ( always_inline ) ) static __INLINE uint32_t __USUB8(uint32_t op1, uint32_t op2)
00268 {
00269   uint32_t result;
00270
00271   __ASM volatile ("usub8 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00272   return(result);
00273 }
00274
00275 __attribute__( ( always_inline ) ) static __INLINE uint32_t __UQSUB8(uint32_t op1, uint32_t op2)
00276 {
00277   uint32_t result;
00278
00279   __ASM volatile ("uqsub8 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00280   return(result);
00281 }
00282
00283 __attribute__( ( always_inline ) ) static __INLINE uint32_t __UHSUB8(uint32_t op1, uint32_t op2)
```

```
00284 {
00285   uint32_t result;
00286
00287   __ASM volatile ("uhsub8 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00288   return(result);
00289 }
00290
00291
00292 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SADD16(uint32_t op1, uint32_t op2)
00293 {
00294   uint32_t result;
00295
00296   __ASM volatile ("sadd16 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00297   return(result);
00298 }
00299
00300 __attribute__( ( always_inline ) ) static __INLINE uint32_t __QADD16(uint32_t op1, uint32_t op2)
00301 {
00302   uint32_t result;
00303
00304   __ASM volatile ("qadd16 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00305   return(result);
00306 }
00307
00308 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SHADD16(uint32_t op1, uint32_t op2)
00309 {
00310   uint32_t result;
00311
00312   __ASM volatile ("shadd16 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00313   return(result);
00314 }
00315
00316 __attribute__( ( always_inline ) ) static __INLINE uint32_t __UADD16(uint32_t op1, uint32_t op2)
00317 {
00318   uint32_t result;
00319
00320   __ASM volatile ("uadd16 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00321   return(result);
00322 }
00323
00324 __attribute__( ( always_inline ) ) static __INLINE uint32_t __UQADD16(uint32_t op1, uint32_t op2)
00325 {
00326   uint32_t result;
00327
00328   __ASM volatile ("uqadd16 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00329   return(result);
00330 }
00331
00332 __attribute__( ( always_inline ) ) static __INLINE uint32_t __UHADD16(uint32_t op1, uint32_t op2)
00333 {
00334   uint32_t result;
00335
00336   __ASM volatile ("uhadd16 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00337   return(result);
00338 }
00339
00340 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SSUB16(uint32_t op1, uint32_t op2)
00341 {
00342   uint32_t result;
00343
00344   __ASM volatile ("ssub16 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00345   return(result);
00346 }
00347
00348 __attribute__( ( always_inline ) ) static __INLINE uint32_t __QSUB16(uint32_t op1, uint32_t op2)
00349 {
00350   uint32_t result;
00351
00352   __ASM volatile ("qsub16 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00353   return(result);
00354 }
00355
00356 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SHSUB16(uint32_t op1, uint32_t op2)
00357 {
00358   uint32_t result;
00359
00360   __ASM volatile ("shsub16 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00361   return(result);
00362 }
00363
00364 __attribute__( ( always_inline ) ) static __INLINE uint32_t __USUB16(uint32_t op1, uint32_t op2)
00365 {
00366   uint32_t result;
00367
00368   __ASM volatile ("usub16 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00369   return(result);
00370 }
```

```
00371
00372 __attribute__( ( always_inline ) ) static __INLINE uint32_t __UQSUB16(uint32_t op1, uint32_t op2)
00373 {
00374   uint32_t result;
00375
00376   __ASM volatile ("uqsub16 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00377   return(result);
00378 }
00379
00380 __attribute__( ( always_inline ) ) static __INLINE uint32_t __UHSUB16(uint32_t op1, uint32_t op2)
00381 {
00382   uint32_t result;
00383
00384   __ASM volatile ("uhsub16 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00385   return(result);
00386 }
00387
00388 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SASX(uint32_t op1, uint32_t op2)
00389 {
00390   uint32_t result;
00391
00392   __ASM volatile ("sasx %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00393   return(result);
00394 }
00395
00396 __attribute__( ( always_inline ) ) static __INLINE uint32_t __QASX(uint32_t op1, uint32_t op2)
00397 {
00398   uint32_t result;
00399
00400   __ASM volatile ("qasx %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00401   return(result);
00402 }
00403
00404 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SHASX(uint32_t op1, uint32_t op2)
00405 {
00406   uint32_t result;
00407
00408   __ASM volatile ("shasx %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00409   return(result);
00410 }
00411
00412 __attribute__( ( always_inline ) ) static __INLINE uint32_t __UASX(uint32_t op1, uint32_t op2)
00413 {
00414   uint32_t result;
00415
00416   __ASM volatile ("uasx %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00417   return(result);
00418 }
00419
00420 __attribute__( ( always_inline ) ) static __INLINE uint32_t __UQASX(uint32_t op1, uint32_t op2)
00421 {
00422   uint32_t result;
00423
00424   __ASM volatile ("uqasx %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00425   return(result);
00426 }
00427
00428 __attribute__( ( always_inline ) ) static __INLINE uint32_t __UHASX(uint32_t op1, uint32_t op2)
00429 {
00430   uint32_t result;
00431
00432   __ASM volatile ("uhasx %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00433   return(result);
00434 }
00435
00436 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SSAX(uint32_t op1, uint32_t op2)
00437 {
00438   uint32_t result;
00439
00440   __ASM volatile ("ssax %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00441   return(result);
00442 }
00443
00444 __attribute__( ( always_inline ) ) static __INLINE uint32_t __QSAX(uint32_t op1, uint32_t op2)
00445 {
00446   uint32_t result;
00447
00448   __ASM volatile ("qsax %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00449   return(result);
00450 }
00451
00452 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SHSAX(uint32_t op1, uint32_t op2)
00453 {
00454   uint32_t result;
00455
00456   __ASM volatile ("shsax %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00457   return(result);
```

```
00458 }
00459
00460 __attribute__( ( always_inline ) ) static __INLINE uint32_t __USAX(uint32_t op1, uint32_t op2)
00461 {
00462   uint32_t result;
00463
00464   __ASM volatile ("usax %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00465   return(result);
00466 }
00467
00468 __attribute__( ( always_inline ) ) static __INLINE uint32_t __UQSAX(uint32_t op1, uint32_t op2)
00469 {
00470   uint32_t result;
00471
00472   __ASM volatile ("uqsax %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00473   return(result);
00474 }
00475
00476 __attribute__( ( always_inline ) ) static __INLINE uint32_t __UHSAX(uint32_t op1, uint32_t op2)
00477 {
00478   uint32_t result;
00479
00480   __ASM volatile ("uhsax %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00481   return(result);
00482 }
00483
00484 __attribute__( ( always_inline ) ) static __INLINE uint32_t __USAD8(uint32_t op1, uint32_t op2)
00485 {
00486   uint32_t result;
00487
00488   __ASM volatile ("usad8 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00489   return(result);
00490 }
00491
00492 __attribute__( ( always_inline ) ) static __INLINE uint32_t __USADA8(uint32_t op1, uint32_t op2,
      uint32_t op3)
00493 {
00494   uint32_t result;
00495
00496   __ASM volatile ("usada8 %0, %1, %2, %3" : "=r" (result) : "r" (op1), "r" (op2), "r" (op3) );
00497   return(result);
00498 }
00499
00500 #define __SSAT16(ARG1,ARG2) \
00501 ({                          \
00502   uint32_t __RES, __ARG1 = (ARG1); \
00503   __ASM ("ssat16 %0, %1, %2" : "=r" (__RES) :  "I" (ARG2), "r" (__ARG1) ); \
00504   __RES; \
00505  })
00506
00507 #define __USAT16(ARG1,ARG2) \
00508 ({                          \
00509   uint32_t __RES, __ARG1 = (ARG1); \
00510   __ASM ("usat16 %0, %1, %2" : "=r" (__RES) :  "I" (ARG2), "r" (__ARG1) ); \
00511   __RES; \
00512  })
00513
00514 __attribute__( ( always_inline ) ) static __INLINE uint32_t __UXTB16(uint32_t op1)
00515 {
00516   uint32_t result;
00517
00518   __ASM volatile ("uxtb16 %0, %1" : "=r" (result) : "r" (op1));
00519   return(result);
00520 }
00521
00522 __attribute__( ( always_inline ) ) static __INLINE uint32_t __UXTAB16(uint32_t op1, uint32_t op2)
00523 {
00524   uint32_t result;
00525
00526   __ASM volatile ("uxtab16 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00527   return(result);
00528 }
00529
00530 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SXTB16(uint32_t op1)
00531 {
00532   uint32_t result;
00533
00534   __ASM volatile ("sxtb16 %0, %1" : "=r" (result) : "r" (op1));
00535   return(result);
00536 }
00537
00538 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SXTAB16(uint32_t op1, uint32_t op2)
00539 {
00540   uint32_t result;
00541
00542   __ASM volatile ("sxtab16 %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00543   return(result);
```

```
00544 }
00545
00546 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SMUAD  (uint32_t op1, uint32_t op2)
00547 {
00548   uint32_t result;
00549
00550   __ASM volatile ("smuad %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00551   return(result);
00552 }
00553
00554 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SMUADX (uint32_t op1, uint32_t op2)
00555 {
00556   uint32_t result;
00557
00558   __ASM volatile ("smuadx %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00559   return(result);
00560 }
00561
00562 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SMLAD (uint32_t op1, uint32_t op2,
      uint32_t op3)
00563 {
00564   uint32_t result;
00565
00566   __ASM volatile ("smlad %0, %1, %2, %3" : "=r" (result) : "r" (op1), "r" (op2), "r" (op3) );
00567   return(result);
00568 }
00569
00570 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SMLADX (uint32_t op1, uint32_t op2,
      uint32_t op3)
00571 {
00572   uint32_t result;
00573
00574   __ASM volatile ("smladx %0, %1, %2, %3" : "=r" (result) : "r" (op1), "r" (op2), "r" (op3) );
00575   return(result);
00576 }
00577
00578 #define __SMLALD(ARG1,ARG2,ARG3) \
00579 ({ \
00580   uint32_t __ARG1 = (ARG1), __ARG2 = (ARG2), __ARG3_H = (uint32_t)((uint64_t)(ARG3) » 32), __ARG3_L =
      (uint32_t)((uint64_t)(ARG3) & 0xFFFFFFFFUL); \
00581   __ASM volatile ("smlald %0, %1, %2, %3" : "=r" (__ARG3_L), "=r" (__ARG3_H) : "r" (__ARG1), "r"
      (__ARG2), "0" (__ARG3_L), "1" (__ARG3_H) ); \
00582   (uint64_t)(((uint64_t)__ARG3_H « 32) | __ARG3_L); \
00583  })
00584
00585 #define __SMLALDX(ARG1,ARG2,ARG3) \
00586 ({ \
00587   uint32_t __ARG1 = (ARG1), __ARG2 = (ARG2), __ARG3_H = (uint32_t)((uint64_t)(ARG3) » 32), __ARG3_L =
      (uint32_t)((uint64_t)(ARG3) & 0xFFFFFFFFUL); \
00588   __ASM volatile ("smlaldx %0, %1, %2, %3" : "=r" (__ARG3_L), "=r" (__ARG3_H) : "r" (__ARG1), "r"
      (__ARG2), "0" (__ARG3_L), "1" (__ARG3_H) ); \
00589   (uint64_t)(((uint64_t)__ARG3_H « 32) | __ARG3_L); \
00590  })
00591
00592 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SMUSD  (uint32_t op1, uint32_t op2)
00593 {
00594   uint32_t result;
00595
00596   __ASM volatile ("smusd %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00597   return(result);
00598 }
00599
00600 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SMUSDX (uint32_t op1, uint32_t op2)
00601 {
00602   uint32_t result;
00603
00604   __ASM volatile ("smusdx %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00605   return(result);
00606 }
00607
00608 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SMLSD (uint32_t op1, uint32_t op2,
      uint32_t op3)
00609 {
00610   uint32_t result;
00611
00612   __ASM volatile ("smlsd %0, %1, %2, %3" : "=r" (result) : "r" (op1), "r" (op2), "r" (op3) );
00613   return(result);
00614 }
00615
00616 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SMLSDX (uint32_t op1, uint32_t op2,
      uint32_t op3)
00617 {
00618   uint32_t result;
00619
00620   __ASM volatile ("smlsdx %0, %1, %2, %3" : "=r" (result) : "r" (op1), "r" (op2), "r" (op3) );
00621   return(result);
00622 }
```

```
00623
00624 #define __SMLSLD(ARG1,ARG2,ARG3) \
00625 ({ \
00626   uint32_t __ARG1 = (ARG1), __ARG2 = (ARG2), __ARG3_H = (uint32_t)((ARG3) » 32), __ARG3_L =
      (uint32_t)((ARG3) & 0xFFFFFFFFUL); \
00627   __ASM volatile ("smlsld %0, %1, %2, %3" : "=r" (__ARG3_L), "=r" (__ARG3_H) : "r" (__ARG1), "r"
      (__ARG2), "0" (__ARG3_L), "1" (__ARG3_H) ); \
00628   (uint64_t)(((uint64_t)__ARG3_H « 32) | __ARG3_L); \
00629  })
00630
00631 #define __SMLSLDX(ARG1,ARG2,ARG3) \
00632 ({ \
00633   uint32_t __ARG1 = (ARG1), __ARG2 = (ARG2), __ARG3_H = (uint32_t)((ARG3) » 32), __ARG3_L =
      (uint32_t)((ARG3) & 0xFFFFFFFFUL); \
00634   __ASM volatile ("smlsldx %0, %1, %2, %3" : "=r" (__ARG3_L), "=r" (__ARG3_H) : "r" (__ARG1), "r"
      (__ARG2), "0" (__ARG3_L), "1" (__ARG3_H) ); \
00635   (uint64_t)(((uint64_t)__ARG3_H « 32) | __ARG3_L); \
00636  })
00637
00638 __attribute__( ( always_inline ) ) static __INLINE uint32_t __SEL  (uint32_t op1, uint32_t op2)
00639 {
00640   uint32_t result;
00641
00642   __ASM volatile ("sel %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00643   return(result);
00644 }
00645
00646 __attribute__( ( always_inline ) ) static __INLINE uint32_t __QADD(uint32_t op1, uint32_t op2)
00647 {
00648   uint32_t result;
00649
00650   __ASM volatile ("qadd %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00651   return(result);
00652 }
00653
00654 __attribute__( ( always_inline ) ) static __INLINE uint32_t __QSUB(uint32_t op1, uint32_t op2)
00655 {
00656   uint32_t result;
00657
00658   __ASM volatile ("qsub %0, %1, %2" : "=r" (result) : "r" (op1), "r" (op2) );
00659   return(result);
00660 }
00661
00662 #define __PKHBT(ARG1,ARG2,ARG3) \
00663 ({                          \
00664   uint32_t __RES, __ARG1 = (ARG1), __ARG2 = (ARG2); \
00665   __ASM ("pkhbt %0, %1, %2, lsl %3" : "=r" (__RES) :  "r" (__ARG1), "r" (__ARG2), "I" (ARG3)  ); \
00666   __RES; \
00667  })
00668
00669 #define __PKHTB(ARG1,ARG2,ARG3) \
00670 ({                          \
00671   uint32_t __RES, __ARG1 = (ARG1), __ARG2 = (ARG2); \
00672   if (ARG3 == 0) \
00673     __ASM ("pkhtb %0, %1, %2" : "=r" (__RES) :  "r" (__ARG1), "r" (__ARG2)  ); \
00674   else  \
00675     __ASM ("pkhtb %0, %1, %2, asr %3" : "=r" (__RES) :  "r" (__ARG1), "r" (__ARG2), "I" (ARG3)  ); \
00676   __RES; \
00677  })
00678
00679 /*-- End CM4 SIMD Intrinsics -----------------------------------------------------*/
00680
00681
00682
00683 #elif defined ( __TASKING__ ) /*------------------ TASKING Compiler --------------*/
00684 /* TASKING carm specific functions */
00685
00686
00687 /*------ CM4 SIMD Intrinsics -----------------------------------------------------*/
00688 /* not yet supported */
00689 /*-- End CM4 SIMD Intrinsics -----------------------------------------------------*/
00690
00691
00692 #endif
00693
00697 #endif /* __CORE_CM4_SIMD_H */
00698
00699 #ifdef __cplusplus
00700 }
00701 #endif
```

7.11 CUBE_IDE/VGA/Core/Inc/core_cmFunc.h File Reference

CMSIS Cortex-M Core Function Access Header File.

7.11.1 Detailed Description

CMSIS Cortex-M Core Function Access Header File.

Version

V2.10

Date

26. July 2011

Note

ARM Limited (ARM) is supplying this software for use with Cortex-M processor based microcontrollers. This file can be freely distributed within development tools that are supporting such ARM based processors.

7.12 core_cmFunc.h

Go to the documentation of this file.

```
00001 /**************************************************************************/
00024 #ifndef __CORE_CMFUNC_H
00025 #define __CORE_CMFUNC_H
00026
00027
00028 /* ###########################  Core Function Access  ########################### */
00034 #if   defined ( __CC_ARM ) /*------------------RealView Compiler -----------------*/
00035 /* ARM armcc specific functions */
00036
00037 #if (__ARMCC_VERSION < 400677)
00038   #error "Please use ARM Compiler Toolchain V4.0.677 or later!"
00039 #endif
00040
00041 /* intrinsic void __enable_irq();     */
00042 /* intrinsic void __disable_irq();    */
00043
00050 static __INLINE uint32_t __get_CONTROL(void)
00051 {
00052   register uint32_t __regControl         __ASM("control");
00053   return(__regControl);
00054 }
00055
00056
```

```
00063 static __INLINE void __set_CONTROL(uint32_t control)
00064 {
00065   register uint32_t __regControl         __ASM("control");
00066   __regControl = control;
00067 }
00068
00069
00076 static __INLINE uint32_t __get_IPSR(void)
00077 {
00078   register uint32_t __regIPSR          __ASM("ipsr");
00079   return(__regIPSR);
00080 }
00081
00082
00089 static __INLINE uint32_t __get_APSR(void)
00090 {
00091   register uint32_t __regAPSR          __ASM("apsr");
00092   return(__regAPSR);
00093 }
00094
00095
00102 static __INLINE uint32_t __get_xPSR(void)
00103 {
00104   register uint32_t __regXPSR          __ASM("xpsr");
00105   return(__regXPSR);
00106 }
00107
00108
00115 static __INLINE uint32_t __get_PSP(void)
00116 {
00117   register uint32_t __regProcessStackPointer  __ASM("psp");
00118   return(__regProcessStackPointer);
00119 }
00120
00121
00128 static __INLINE void __set_PSP(uint32_t topOfProcStack)
00129 {
00130   register uint32_t __regProcessStackPointer  __ASM("psp");
00131   __regProcessStackPointer = topOfProcStack;
00132 }
00133
00134
00141 static __INLINE uint32_t __get_MSP(void)
00142 {
00143   register uint32_t __regMainStackPointer     __ASM("msp");
00144   return(__regMainStackPointer);
00145 }
00146
00147
00154 static __INLINE void __set_MSP(uint32_t topOfMainStack)
00155 {
00156   register uint32_t __regMainStackPointer     __ASM("msp");
00157   __regMainStackPointer = topOfMainStack;
00158 }
00159
00160
00167 static __INLINE uint32_t __get_PRIMASK(void)
00168 {
00169   register uint32_t __regPriMask         __ASM("primask");
00170   return(__regPriMask);
00171 }
00172
00173
00180 static __INLINE void __set_PRIMASK(uint32_t priMask)
00181 {
00182   register uint32_t __regPriMask         __ASM("primask");
00183   __regPriMask = (priMask);
00184 }
00185
00186
00187 #if       (__CORTEX_M >= 0x03)
00188
00194 #define __enable_fault_irq                __enable_fiq
00195
00196
00202 #define __disable_fault_irq               __disable_fiq
00203
00204
00211 static __INLINE uint32_t  __get_BASEPRI(void)
00212 {
00213   register uint32_t __regBasePri         __ASM("basepri");
00214   return(__regBasePri);
00215 }
00216
00217
00224 static __INLINE void __set_BASEPRI(uint32_t basePri)
00225 {
```

```
00226   register uint32_t __regBasePri         __ASM("basepri");
00227   __regBasePri = (basePri & 0xff);
00228 }
00229
00230
00237 static __INLINE uint32_t __get_FAULTMASK(void)
00238 {
00239   register uint32_t __regFaultMask       __ASM("faultmask");
00240   return(__regFaultMask);
00241 }
00242
00243
00250 static __INLINE void __set_FAULTMASK(uint32_t faultMask)
00251 {
00252   register uint32_t __regFaultMask       __ASM("faultmask");
00253   __regFaultMask = (faultMask & (uint32_t)1);
00254 }
00255
00256 #endif /* (__CORTEX_M >= 0x03) */
00257
00258
00259 #if       (__CORTEX_M == 0x04)
00260
00267 static __INLINE uint32_t __get_FPSCR(void)
00268 {
00269 #if (__FPU_PRESENT == 1) && (__FPU_USED == 1)
00270   register uint32_t __regfpscr         __ASM("fpscr");
00271   return(__regfpscr);
00272 #else
00273    return(0);
00274 #endif
00275 }
00276
00277
00284 static __INLINE void __set_FPSCR(uint32_t fpscr)
00285 {
00286 #if (__FPU_PRESENT == 1) && (__FPU_USED == 1)
00287   register uint32_t __regfpscr         __ASM("fpscr");
00288   __regfpscr = (fpscr);
00289 #endif
00290 }
00291
00292 #endif /* (__CORTEX_M == 0x04) */
00293
00294
00295 #elif defined ( __ICCARM__ ) /*------------------ ICC Compiler -------------------*/
00296 /* IAR iccarm specific functions */
00297
00298 #include <cmsis_iar.h>
00299
00300 #elif defined ( __GNUC__ ) /*------------------ GNU Compiler ---------------------*/
00301 /* GNU gcc specific functions */
00302
00308 __attribute__( ( always_inline ) ) static __INLINE void __enable_irq(void)
00309 {
00310   __ASM volatile ("cpsie i");
00311 }
00312
00313
00319 __attribute__( ( always_inline ) ) static __INLINE void __disable_irq(void)
00320 {
00321   __ASM volatile ("cpsid i");
00322 }
00323
00324
00331 __attribute__( ( always_inline ) ) static __INLINE uint32_t __get_CONTROL(void)
00332 {
00333   uint32_t result;
00334
00335   __ASM volatile ("MRS %0, control" : "=r" (result) );
00336   return(result);
00337 }
00338
00339
00346 __attribute__( ( always_inline ) ) static __INLINE void __set_CONTROL(uint32_t control)
00347 {
00348   __ASM volatile ("MSR control, %0" : : "r" (control) );
00349 }
00350
00351
00358 __attribute__( ( always_inline ) ) static __INLINE uint32_t __get_IPSR(void)
00359 {
00360   uint32_t result;
00361
00362   __ASM volatile ("MRS %0, ipsr" : "=r" (result) );
00363   return(result);
00364 }
```

```
00365
00366
00373 __attribute__( ( always_inline ) ) static __INLINE uint32_t __get_APSR(void)
00374 {
00375   uint32_t result;
00376
00377   __ASM volatile ("MRS %0, apsr" : "=r" (result) );
00378   return(result);
00379 }
00380
00381
00388 __attribute__( ( always_inline ) ) static __INLINE uint32_t __get_xPSR(void)
00389 {
00390   uint32_t result;
00391
00392   __ASM volatile ("MRS %0, xpsr" : "=r" (result) );
00393   return(result);
00394 }
00395
00396
00403 __attribute__( ( always_inline ) ) static __INLINE uint32_t __get_PSP(void)
00404 {
00405   register uint32_t result;
00406
00407   __ASM volatile ("MRS %0, psp\n"  : "=r" (result) );
00408   return(result);
00409 }
00410
00411
00418 __attribute__( ( always_inline ) ) static __INLINE void __set_PSP(uint32_t topOfProcStack)
00419 {
00420   __ASM volatile ("MSR psp, %0\n" : : "r" (topOfProcStack) );
00421 }
00422
00423
00430 __attribute__( ( always_inline ) ) static __INLINE uint32_t __get_MSP(void)
00431 {
00432   register uint32_t result;
00433
00434   __ASM volatile ("MRS %0, msp\n" : "=r" (result) );
00435   return(result);
00436 }
00437
00438
00445 __attribute__( ( always_inline ) ) static __INLINE void __set_MSP(uint32_t topOfMainStack)
00446 {
00447   __ASM volatile ("MSR msp, %0\n" : : "r" (topOfMainStack) );
00448 }
00449
00450
00457 __attribute__( ( always_inline ) ) static __INLINE uint32_t __get_PRIMASK(void)
00458 {
00459   uint32_t result;
00460
00461   __ASM volatile ("MRS %0, primask" : "=r" (result) );
00462   return(result);
00463 }
00464
00465
00472 __attribute__( ( always_inline ) ) static __INLINE void __set_PRIMASK(uint32_t priMask)
00473 {
00474   __ASM volatile ("MSR primask, %0" : : "r" (priMask) );
00475 }
00476
00477
00478 #if       (__CORTEX_M >= 0x03)
00479
00485 __attribute__( ( always_inline ) ) static __INLINE void __enable_fault_irq(void)
00486 {
00487   __ASM volatile ("cpsie f");
00488 }
00489
00490
00496 __attribute__( ( always_inline ) ) static __INLINE void __disable_fault_irq(void)
00497 {
00498   __ASM volatile ("cpsid f");
00499 }
00500
00501
00508 __attribute__( ( always_inline ) ) static __INLINE uint32_t __get_BASEPRI(void)
00509 {
00510   uint32_t result;
00511
00512   __ASM volatile ("MRS %0, basepri_max" : "=r" (result) );
00513   return(result);
00514 }
00515
```

```
00516
00523 __attribute__( ( always_inline ) ) static __INLINE void __set_BASEPRI(uint32_t value)
00524 {
00525   __ASM volatile ("MSR basepri, %0" : : "r" (value) );
00526 }
00527
00528
00535 __attribute__( ( always_inline ) ) static __INLINE uint32_t __get_FAULTMASK(void)
00536 {
00537   uint32_t result;
00538
00539   __ASM volatile ("MRS %0, faultmask" : "=r" (result) );
00540   return(result);
00541 }
00542
00543
00550 __attribute__( ( always_inline ) ) static __INLINE void __set_FAULTMASK(uint32_t faultMask)
00551 {
00552   __ASM volatile ("MSR faultmask, %0" : : "r" (faultMask) );
00553 }
00554
00555 #endif /* (__CORTEX_M >= 0x03) */
00556
00557
00558 #if       (__CORTEX_M == 0x04)
00559
00566 __attribute__( ( always_inline ) ) static __INLINE uint32_t __get_FPSCR(void)
00567 {
00568 #if (__FPU_PRESENT == 1) && (__FPU_USED == 1)
00569   uint32_t result;
00570
00571   __ASM volatile ("VMRS %0, fpscr" : "=r" (result) );
00572   return(result);
00573 #else
00574    return(0);
00575 #endif
00576 }
00577
00578
00585 __attribute__( ( always_inline ) ) static __INLINE void __set_FPSCR(uint32_t fpscr)
00586 {
00587 #if (__FPU_PRESENT == 1) && (__FPU_USED == 1)
00588   __ASM volatile ("VMSR fpscr, %0" : : "r" (fpscr) );
00589 #endif
00590 }
00591
00592 #endif /* (__CORTEX_M == 0x04) */
00593
00594
00595 #elif defined ( __TASKING__ ) /*------------------ TASKING Compiler --------------*/
00596 /* TASKING carm specific functions */
00597
00598 /*
00599  * The CMSIS functions have been implemented as intrinsics in the compiler.
00600  * Please use "carm -?i" to get an up to date list of all instrinsics,
00601  * Including the CMSIS ones.
00602  */
00603
00604 #endif
00605
00609 #endif /* __CORE_CMFUNC_H */
```

7.13 CUBE_IDE/VGA/Core/Inc/core_cmInstr.h File Reference

CMSIS Cortex-M Core Instruction Access Header File.

7.13.1 Detailed Description

CMSIS Cortex-M Core Instruction Access Header File.

Version

V2.10

Date

19. July 2011

Note

7.14 core_cmInstr.h

Go to the documentation of this file.

```
00001 /**************************************************************************/
00024 #ifndef __CORE_CMINSTR_H
00025 #define __CORE_CMINSTR_H
00026
00027
00028 /* ##########################  Core Instruction Access  ######################### */
00034 #if   defined ( __CC_ARM ) /*------------------RealView Compiler -----------------*/
00035 /* ARM armcc specific functions */
00036
00037 #if (__ARMCC_VERSION < 400677)
00038   #error "Please use ARM Compiler Toolchain V4.0.677 or later!"
00039 #endif
00040
00041
00046 #define __NOP                             __nop
00047
00048
00054 #define __WFI                             __wfi
00055
00056
00062 #define __WFE                             __wfe
00063
00064
00069 #define __SEV                             __sev
00070
00071
00078 #define __ISB()                           __isb(0xF)
00079
00080
00086 #define __DSB()                           __dsb(0xF)
00087
00088
00094 #define __DMB()                           __dmb(0xF)
00095
00096
00104 #define __REV                             __rev
00105
00106
00114 static __INLINE __ASM uint32_t __REV16(uint32_t value)
00115 {
00116   rev16 r0, r0
00117   bx lr
00118 }
00119
00120
00128 static __INLINE __ASM int32_t __REVSH(int32_t value)
00129 {
```

```
00130   revsh r0, r0
00131   bx lr
00132 }
00133
00134
00135 #if       (__CORTEX_M >= 0x03)
00136
00144 #define __RBIT                            __rbit
00145
00146
00154 #define __LDREXB(ptr)                     ((uint8_t ) __ldrex(ptr))
00155
00156
00164 #define __LDREXH(ptr)                     ((uint16_t) __ldrex(ptr))
00165
00166
00174 #define __LDREXW(ptr)                     ((uint32_t ) __ldrex(ptr))
00175
00176
00186 #define __STREXB(value, ptr)              __strex(value, ptr)
00187
00188
00198 #define __STREXH(value, ptr)              __strex(value, ptr)
00199
00200
00210 #define __STREXW(value, ptr)              __strex(value, ptr)
00211
00212
00218 #define __CLREX                           __clrex
00219
00220
00229 #define __SSAT                            __ssat
00230
00231
00240 #define __USAT                            __usat
00241
00242
00250 #define __CLZ                             __clz
00251
00252 #endif /* (__CORTEX_M >= 0x03) */
00253
00254
00255
00256 #elif defined ( __ICCARM__ ) /*------------------ ICC Compiler --------------------*/
00257 /* IAR iccarm specific functions */
00258
00259 #include <cmsis_iar.h>
00260
00261
00262 #elif defined ( __GNUC__ ) /*------------------ GNU Compiler ----------------------*/
00263 /* GNU gcc specific functions */
00264
00269 __attribute__( ( always_inline ) ) static __INLINE void __NOP(void)
00270 {
00271   __ASM volatile ("nop");
00272 }
00273
00274
00280 __attribute__( ( always_inline ) ) static __INLINE void __WFI(void)
00281 {
00282   __ASM volatile ("wfi");
00283 }
00284
00285
00291 __attribute__( ( always_inline ) ) static __INLINE void __WFE(void)
00292 {
00293   __ASM volatile ("wfe");
00294 }
00295
00296
00301 __attribute__( ( always_inline ) ) static __INLINE void __SEV(void)
00302 {
00303   __ASM volatile ("sev");
00304 }
00305
00306
00313 __attribute__( ( always_inline ) ) static __INLINE void __ISB(void)
00314 {
00315   __ASM volatile ("isb");
00316 }
00317
00318
00324 __attribute__( ( always_inline ) ) static __INLINE void __DSB(void)
00325 {
00326   __ASM volatile ("dsb");
00327 }
00328
```

```
00329
00335 __attribute__( ( always_inline ) ) static __INLINE void __DMB(void)
00336 {
00337   __ASM volatile ("dmb");
00338 }
00339
00340
00348 __attribute__( ( always_inline ) ) static __INLINE uint32_t __REV(uint32_t value)
00349 {
00350   uint32_t result;
00351
00352   __ASM volatile ("rev %0, %1" : "=r" (result) : "r" (value) );
00353   return(result);
00354 }
00355
00356
00364 __attribute__( ( always_inline ) ) static __INLINE uint32_t __REV16(uint32_t value)
00365 {
00366   uint32_t result;
00367
00368   __ASM volatile ("rev16 %0, %1" : "=r" (result) : "r" (value) );
00369   return(result);
00370 }
00371
00372
00380 __attribute__( ( always_inline ) ) static __INLINE int32_t __REVSH(int32_t value)
00381 {
00382   uint32_t result;
00383
00384   __ASM volatile ("revsh %0, %1" : "=r" (result) : "r" (value) );
00385   return(result);
00386 }
00387
00388
00389 #if       (__CORTEX_M >= 0x03)
00390
00398 __attribute__( ( always_inline ) ) static __INLINE uint32_t __RBIT(uint32_t value)
00399 {
00400   uint32_t result;
00401
00402    __ASM volatile ("rbit %0, %1" : "=r" (result) : "r" (value) );
00403    return(result);
00404 }
00405
00406
00414 __attribute__( ( always_inline ) ) static __INLINE uint8_t __LDREXB(volatile uint8_t *addr)
00415 {
00416     uint8_t result;
00417
00418    __ASM volatile ("ldrexb %0, [%1]" : "=r" (result) : "r" (addr) );
00419    return(result);
00420 }
00421
00422
00430 __attribute__( ( always_inline ) ) static __INLINE uint16_t __LDREXH(volatile uint16_t *addr)
00431 {
00432     uint16_t result;
00433
00434    __ASM volatile ("ldrexh %0, [%1]" : "=r" (result) : "r" (addr) );
00435    return(result);
00436 }
00437
00438
00446 __attribute__( ( always_inline ) ) static __INLINE uint32_t __LDREXW(volatile uint32_t *addr)
00447 {
00448     uint32_t result;
00449
00450    __ASM volatile ("ldrex %0, [%1]" : "=r" (result) : "r" (addr) );
00451    return(result);
00452 }
00453
00454
00464 __attribute__( ( always_inline ) ) static __INLINE uint32_t __STREXB(uint8_t value, volatile uint8_t
      *addr)
00465 {
00466    uint32_t result;
00467
00468    __ASM volatile ("strexb %0, %2, [%1]" : "=r" (result) : "r" (addr), "r" (value) );
00469    return(result);
00470 }
00471
00472
00482 __attribute__( ( always_inline ) ) static __INLINE uint32_t __STREXH(uint16_t value, volatile uint16_t
      *addr)
00483 {
00484    uint32_t result;
00485
```

```
00486    __ASM volatile ("strexh %0, %2, [%1]" : "=r" (result) : "r" (addr), "r" (value) );
00487    return(result);
00488 }
00489
00490
00500 __attribute__( ( always_inline ) ) static __INLINE uint32_t __STREXW(uint32_t value, volatile uint32_t
      *addr)
00501 {
00502    uint32_t result;
00503
00504    __ASM volatile ("strex %0, %2, [%1]" : "=r" (result) : "r" (addr), "r" (value) );
00505    return(result);
00506 }
00507
00508
00514 __attribute__( ( always_inline ) ) static __INLINE void __CLREX(void)
00515 {
00516   __ASM volatile ("clrex");
00517 }
00518
00519
00528 #define __SSAT(ARG1,ARG2) \
00529 ({                          \
00530   uint32_t __RES, __ARG1 = (ARG1); \
00531   __ASM ("ssat %0, %1, %2" : "=r" (__RES) :  "I" (ARG2), "r" (__ARG1) ); \
00532   __RES; \
00533  })
00534
00535
00544 #define __USAT(ARG1,ARG2) \
00545 ({                          \
00546   uint32_t __RES, __ARG1 = (ARG1); \
00547   __ASM ("usat %0, %1, %2" : "=r" (__RES) :  "I" (ARG2), "r" (__ARG1) ); \
00548   __RES; \
00549  })
00550
00551
00559 __attribute__( ( always_inline ) ) static __INLINE uint8_t __CLZ(uint32_t value)
00560 {
00561   uint8_t result;
00562
00563   __ASM volatile ("clz %0, %1" : "=r" (result) : "r" (value) );
00564   return(result);
00565 }
00566
00567 #endif /* (__CORTEX_M >= 0x03) */
00568
00569
00570
00571
00572 #elif defined ( __TASKING__ ) /*------------------ TASKING Compiler --------------*/
00573 /* TASKING carm specific functions */
00574
00575 /*
00576  * The CMSIS functions have been implemented as intrinsics in the compiler.
00577  * Please use "carm -?i" to get an up to date list of all intrinsics,
00578  * Including the CMSIS ones.
00579  */
00580
00581 #endif
00582  /* end of group CMSIS_Core_InstructionInterface */
00584
00585 #endif /* __CORE_CMINSTR_H */
```

7.15 CUBE_IDE/VGA/Core/Inc/error.h File Reference

This file contains the error ENUMs for the global error handling.

Enumerations

- enum **ERROR** {
  NO_ERR = 0 , **COMMAND_ERR** , **DATA_ERR** , **OOB_ERR** ,
  COLOR_ERR }

7.15.1 Detailed Description

This file contains the error ENUMs for the global error handling.

Authors

Skip Wijtman

Date

6-6-2023

Version

1.0

7.16 error.h

Go to the documentation of this file.

```
00001 /*****************************************************/
00012 #ifndef INC_ERROR_H_
00013 #define INC_ERROR_H_
00014
00015 enum ERROR
00016 {
00017     NO_ERR = 0,      // All errors listed
00018     COMMAND_ERR,     // Command errors such as 'lijn' and 'rechthoek'
00019     DATA_ERR,        // Data error means something in the data is wrong, such as characters which are
      not usable: '/', '[', '(' or '@' are some examples
00020     OOB_ERR,         // OOB, aka 'out of bounds', is an error which is given when pixels are outside
      the maximum ranges
00021     COLOR_ERR        // Color error means a color which is not usable, was submitted
00022 };
00023
00024 #endif /* INC_ERROR_H_ */
```

7.17 CUBE_IDE/VGA/Core/Inc/logic_layer.h File Reference

This file contains the functions which are used in the source file 'logic_layer.c'.

```
#include "UART_communication.h"
#include "stm32_ub_vga_screen.h"
#include "API_functions.h"
#include "error.h"
```

Data Structures

- struct PARSE_STORAGE

Macros

- #define **MAX_CMD_LEN** 12
- #define **MAX_COL_LEN** 13
- #define **LINE_LEN** 5
- #define **RECTANGLE_LEN** 10
- #define **CLEAR_LEN** 12
- #define **TEXT_LEN** 6
- #define **BITMAP_LEN** 7
- #define **LB_ASCII_NUMBERS** 48
- #define **UB_ASCII_NUMBERS** 57
- #define **LB_ASCII_LETTERS** 97
- #define **UB_ASCII_LETTERS** 122
- #define **TRUE** 1
- #define **FALSE** 0
- #define **ERROR_OFF** 5

Typedefs

- typedef struct PARSE_STORAGE **PARSE**
- typedef struct PARSE_STORAGE * **PPARSE**

Functions

- int parse_cmd (UART data)

  Function that parses the command of the received script.
- int draw_options (char cmd, UART data)

  Function that determines with the command which function should be used, Here the script also get parsed further and decodes ASCII to useful data.
- int number_converter (char ASCII)

  Function that converts ASCII numbers to decimals.
- PARSE color_assign (UART data, int i, PARSE parsing)

  Function that reads received script text and when possible converts this to the correspondiong color.
- PARSE parse_data (PARSE parsing, UART data, int LEN, int var_counter, int num_checker, int let_checker, int num_counter)

  This function is used for parsing data and converts ASCII to decimals.

7.17.1 Detailed Description

This file contains the functions which are used in the source file 'logic_layer.c'.

Authors

Skip Wijtman

Date

24-5-2023

Version

1.2

7.17.2 Function Documentation

7.17.2.1 color_assign()

```
PARSE color_assign (
            UART data,
            int i,
            PARSE parsing )
```

Function that reads received script text and when possible converts this to the correspondiong color.

Parameters

data	is a struct variable with the received script
i	is the variable for the loop iterator
parsing	is a struct variable that stores all needed data for functions

Returns

parsing is a struct variable with data info which are retrieved from the script

7.17.2.2 draw_options()

```
int draw_options (
            char cmd,
            UART data )
```

Function that determines with the command which function should be used, Here the script also get parsed further and decodes ASCII to useful data.

Here a switch-case is used to determine the received command, Cases 0 to 4 are numbered as followed:

1. lijn
2. rechthoek
3. clearscherm
4. tekst
5. bitmap

Parameters

cmd	is a interger of the found command
data	is a struct variable with the received script

Returns

Error code

7.17.2.3 number_converter()

```
int number_converter (
            char ASCII )
```

Function that converts ASCII numbers to decimals.

Parameters

ASCII	is character of a ASCII number

Returns

decimal number

7.17.2.4 parse_cmd()

```
int parse_cmd (
            UART data )
```

Function that parses the command of the received script.

Parameters

data	is a struct variable with the received script

Returns

Error code or the current index of found

7.17.2.5 parse_data()

```
PARSE parse_data (
            PARSE parsing,
            UART data,
            int LEN,
            int var_counter,
            int num_checker,
```

```
            int let_checker,
            int num_counter )
```

This function is used for parsing data and converts ASCII to decimals.

Parameters

parsing	is a struct variable that stores all needed data for functions
data	is a struct variable with the received script
LEN	is the length of the command + the comma to skip these is the parser
var_counter	is an index of an array to store the data in from the script
num_checker	is a variable to confirm a ASCII number is found and ensures that no junk color values are stored in the data array
let_checker	is a variable to confirm a letter is found and ensures that no junk number values are stored in the data array
num_counter	is a variable which counts the amount of ASCII numbers between two commas

Returns

Error code

7.18 logic_layer.h

Go to the documentation of this file.

```
00001 /*****************************************************/
00011 #ifndef INC_LOGIC_LAYER_H_
00012 #define INC_LOGIC_LAYER_H_
00013
00014 //#include <library-header>
00015
00016
00017 //#include other "user-header"
00018 #include "UART_communication.h"
00019 #include "stm32_ub_vga_screen.h"
00020 #include "API_functions.h"
00021 #include "error.h"
00022
00023
00024 //struct definition
00025 typedef struct PARSE_STORAGE
00026 {
00027     int number_store[3];
00028     int color;
00029     int loop_iterator;
00030     int var_store[6];
00031     int err_code;
00032 }PARSE, *PPARSE;
00033
00034 //#define-statements
00035 #define MAX_CMD_LEN         12      // Defines for loops
00036 #define MAX_COL_LEN         13
00037
00038 #define LINE_LEN            5       // Defines to skip the command in the received array
00039 #define RECTANGLE_LEN       10
00040 #define CLEAR_LEN           12
00041 #define TEXT_LEN            6
00042 #define BITMAP_LEN          7
00043
00044 #define LB_ASCII_NUMBERS    48      // Defines to mark the lower and upper bounds of the ASCII numbers
00045 #define UB_ASCII_NUMBERS    57
00046
00047 #define LB_ASCII_LETTERS    97      // Defines to mark the lower and upper bounds of the ASCII
      lowercase letters
00048 #define UB_ASCII_LETTERS    122
00049
00050 #define TRUE                1
00051 #define FALSE               0
00052
00053 #define ERROR_OFF           5
00054
00055 //external vars
00056
00057 //prototype user functions
00058 int parse_cmd(UART data);
00059 int draw_options(char cmd, UART data);
00060 int number_converter(char ASCII);
```

```
00061 PARSE color_assign(UART data, int i, PARSE parsing);
00062 PARSE parse_data(PARSE parsing, UART data, int LEN , int var_counter, int num_checker, int
      let_checker, int num_counter);
00063
00064 #endif /* INC_LOGIC_LAYER_H_ */
```

7.19 main.h

```
00001 //-------------------------------------------------------------
00002 // File     : main.h
00003 //-------------------------------------------------------------
00004
00005 //-------------------------------------------------------------
00006 #ifndef __STM32F4_UB_MAIN_H
00007 #define __STM32F4_UB_MAIN_H
00008
00009
00010 //-------------------------------------------------------------
00011 // Includes
00012 //-------------------------------------------------------------
00013 #include "stm32f4xx.h"
00014
00015
00016 //-------------------------------------------------------------
00017 #endif // __STM32F4_UB_MAIN_H
```

7.20 CUBE_IDE/VGA/Core/Inc/misc.h File Reference

This file contains all the functions prototypes for the miscellaneous firmware library functions (add-on to CMSIS functions).

```
#include "stm32f4xx.h"
```

Data Structures

- struct NVIC_InitTypeDef

  NVIC Init Structure definition

Macros

- #define **NVIC_VectTab_RAM** ((uint32_t)0x20000000)
- #define **NVIC_VectTab_FLASH** ((uint32_t)0x08000000)
- #define IS_NVIC_VECTTAB(VECTTAB)
- #define **NVIC_LP_SEVONPEND** ((uint8_t)0x10)
- #define **NVIC_LP_SLEEPDEEP** ((uint8_t)0x04)
- #define **NVIC_LP_SLEEPONEXIT** ((uint8_t)0x02)
- #define IS_NVIC_LP(LP)
- #define NVIC_PriorityGroup_0 ((uint32_t)0x700)
- #define NVIC_PriorityGroup_1 ((uint32_t)0x600)
- #define NVIC_PriorityGroup_2 ((uint32_t)0x500)
- #define NVIC_PriorityGroup_3 ((uint32_t)0x400)
- #define NVIC_PriorityGroup_4 ((uint32_t)0x300)
- #define IS_NVIC_PRIORITY_GROUP(GROUP)
- #define **IS_NVIC_PREEMPTION_PRIORITY**(PRIORITY) ((PRIORITY) $<$ 0x10)
- #define **IS_NVIC_SUB_PRIORITY**(PRIORITY) ((PRIORITY) $<$ 0x10)
- #define **IS_NVIC_OFFSET**(OFFSET) ((OFFSET) $<$ 0x000FFFFF)
- #define **SysTick_CLKSource_HCLK_Div8** ((uint32_t)0xFFFFFFFB)
- #define **SysTick_CLKSource_HCLK** ((uint32_t)0x00000004)
- #define IS_SYSTICK_CLK_SOURCE(SOURCE)

Functions

- void NVIC_PriorityGroupConfig (uint32_t NVIC_PriorityGroup)

  Configures the priority grouping: pre-emption priority and subpriority.
- void NVIC_Init (NVIC_InitTypeDef *NVIC_InitStruct)

  Initializes the NVIC peripheral according to the specified parameters in the NVIC_InitStruct.
- void NVIC_SetVectorTable (uint32_t NVIC_VectTab, uint32_t Offset)

  Sets the vector table location and Offset.
- void NVIC_SystemLPConfig (uint8_t LowPowerMode, FunctionalState NewState)

  Selects the condition for the system to enter low power mode.
- void SysTick_CLKSourceConfig (uint32_t SysTick_CLKSource)

  Configures the SysTick clock source.

7.20.1 Detailed Description

This file contains all the functions prototypes for the miscellaneous firmware library functions (add-on to CMSIS functions).

Author

MCD Application Team

Version

V1.0.0

Date

30-September-2011

Attention

7.21 misc.h

Go to the documentation of this file.

```
00001
00023 /* Define to prevent recursive inclusion -------------------------------------*/
00024 #ifndef __MISC_H
00025 #define __MISC_H
00026
00027 #ifdef __cplusplus
00028  extern "C" {
00029 #endif
00030
00031 /* Includes ------------------------------------------------------------------*/
00032 #include "stm32f4xx.h"
00033
00042 /* Exported types ------------------------------------------------------------*/
00043
00048 typedef struct
00049 {
00050   uint8_t NVIC_IRQChannel;
00055   uint8_t NVIC_IRQChannelPreemptionPriority;
00060   uint8_t NVIC_IRQChannelSubPriority;
00065   FunctionalState NVIC_IRQChannelCmd;
00068 } NVIC_InitTypeDef;
00069
00070 /* Exported constants --------------------------------------------------------*/
00071
00080 #define NVIC_VectTab_RAM             ((uint32_t)0x20000000)
00081 #define NVIC_VectTab_FLASH           ((uint32_t)0x08000000)
00082 #define IS_NVIC_VECTTAB(VECTTAB) (((VECTTAB) == NVIC_VectTab_RAM) || \
00083                                   ((VECTTAB) == NVIC_VectTab_FLASH))
00092 #define NVIC_LP_SEVONPEND            ((uint8_t)0x10)
00093 #define NVIC_LP_SLEEPDEEP            ((uint8_t)0x04)
00094 #define NVIC_LP_SLEEPONEXIT          ((uint8_t)0x02)
00095 #define IS_NVIC_LP(LP) (((LP) == NVIC_LP_SEVONPEND) || \
00096                         ((LP) == NVIC_LP_SLEEPDEEP) || \
00097                         ((LP) == NVIC_LP_SLEEPONEXIT))
00106 #define NVIC_PriorityGroup_0         ((uint32_t)0x700)
00108 #define NVIC_PriorityGroup_1         ((uint32_t)0x600)
00110 #define NVIC_PriorityGroup_2         ((uint32_t)0x500)
00112 #define NVIC_PriorityGroup_3         ((uint32_t)0x400)
00114 #define NVIC_PriorityGroup_4         ((uint32_t)0x300)
00117 #define IS_NVIC_PRIORITY_GROUP(GROUP) (((GROUP) == NVIC_PriorityGroup_0) || \
00118                                        ((GROUP) == NVIC_PriorityGroup_1) || \
00119                                        ((GROUP) == NVIC_PriorityGroup_2) || \
00120                                        ((GROUP) == NVIC_PriorityGroup_3) || \
00121                                        ((GROUP) == NVIC_PriorityGroup_4))
00122
00123 #define IS_NVIC_PREEMPTION_PRIORITY(PRIORITY)  ((PRIORITY) < 0x10)
00124
00125 #define IS_NVIC_SUB_PRIORITY(PRIORITY)  ((PRIORITY) < 0x10)
00126
00127 #define IS_NVIC_OFFSET(OFFSET)  ((OFFSET) < 0x000FFFFF)
00128
00137 #define SysTick_CLKSource_HCLK_Div8    ((uint32_t)0xFFFFFFFB)
00138 #define SysTick_CLKSource_HCLK         ((uint32_t)0x00000004)
00139 #define IS_SYSTICK_CLK_SOURCE(SOURCE) (((SOURCE) == SysTick_CLKSource_HCLK) || \
00140                                        ((SOURCE) == SysTick_CLKSource_HCLK_Div8))
00149 /* Exported macro ------------------------------------------------------------*/
00150 /* Exported functions --------------------------------------------------------*/
00151
00152 void NVIC_PriorityGroupConfig(uint32_t NVIC_PriorityGroup);
00153 void NVIC_Init(NVIC_InitTypeDef* NVIC_InitStruct);
00154 void NVIC_SetVectorTable(uint32_t NVIC_VectTab, uint32_t Offset);
00155 void NVIC_SystemLPConfig(uint8_t LowPowerMode, FunctionalState NewState);
00156 void SysTick_CLKSourceConfig(uint32_t SysTick_CLKSource);
00157
00158 #ifdef __cplusplus
00159 }
00160 #endif
00161
00162 #endif /* __MISC_H */
00163
00172 /******************* (C) COPYRIGHT 2011 STMicroelectronics *****END OF FILE****/
```

7.22 stm32_ub_vga_screen.h

```
00001 //--------------------------------------------------------------
00002 // File     : stm32_ub_vga_screen.h
00003 //--------------------------------------------------------------
00004
00005 //--------------------------------------------------------------
00006 #ifndef __STM32F4_UB_VGA_SCREEN_H
00007 #define __STM32F4_UB_VGA_SCREEN_H
00008
00009
00010 //--------------------------------------------------------------
00011 // Includes
00012 //--------------------------------------------------------------
00013 #include "stm32f4xx.h"
00014 #include "stm32f4xx_gpio.h"
00015 #include "stm32f4xx_rcc.h"
00016 #include "stm32f4xx_tim.h"
00017 #include "misc.h"
00018 #include "stm32f4xx_dma.h"
00019
00020
00021
00022 //--------------------------------------------------------------
00023 // color designation
00024 // 8bit color (R3G3B2)
00025 // Red   (3bit) -> Bit7-Bit5
00026 // Green (3bit) -> Bit4-Bit2
00027 // Blue  (2bit) -> Bit1-Bit0
00028 //--------------------------------------------------------------
00029 #define  VGA_COL_BLACK          0x00
00030 #define  VGA_COL_BLUE           0x2B
00031 #define  VGA_COL_GREEN          0x1C
00032 #define  VGA_COL_RED            0xE0
00033 #define  VGA_COL_WHITE          0xFF
00034
00035 #define  VGA_COL_CYAN           0x1F
00036 #define  VGA_COL_MAGENTA        0xE3
00037 #define  VGA_COL_YELLOW         0xFC
00038
00039 #define VGA_COL_PINK            0xEE
00040 #define VGA_COL_LIGHTBLUE       0x53            // Extra colors (still needs a correct val
00041 #define VGA_COL_LIGHTGREEN      0x7D
00042 #define VGA_COL_LIGHTCYAN       0x9F
00043 #define VGA_COL_LIGHTRED        0xED
00044 #define VGA_COL_LIGHTMAGENTA    0xEF
00045 #define VGA_COL_BROWN           0x44
00046 #define VGA_COL_GRAY            0x6D
00047
00048 //--------------------------------------------------------------
00049 // define the VGA_display
00050 //--------------------------------------------------------------
00051 #define VGA_DISPLAY_X   320
00052 #define VGA_DISPLAY_Y   240
00053
00054
00055
00056 //--------------------------------------------------------------
00057 // VGA Structure
00058 //--------------------------------------------------------------
00059 typedef struct {
00060   uint16_t hsync_cnt;   // counter
00061   uint32_t start_adr;   // start_adres
00062   uint32_t dma2_cr_reg; // Register constant CR-Register
00063 }VGA_t;
00064
00065
00066
00067
00068
00069
00070
00071 //--------------------------------------------------------------
00072 // Timer-1
00073 // Function  = Pixelclock (Speed for  DMA Transfer)
00074 //
00075 // basefreq = 2*APB2 (APB2=84MHz) => TIM_CLK=168MHz
00076 // Frq       = 168MHz/1/12 = 14MHz
00077 //
00078 //--------------------------------------------------------------
00079 #define VGA_TIM1_PERIODE      11
00080 #define VGA_TIM1_PRESCALE      0
00081
00082
00083
00084 //--------------------------------------------------------------
00085 // Timer-2
```

```
00086 // Function  = CH4 : HSync-Signal on PB11
00087 //             CH3 : Trigger point for DMA start
00088 //
00089 // basefreq = 2*APB1 (APB1=48MHz) => TIM_CLK=84MHz
00090 // Frq       = 84MHz/1/2668 = 31,48kHz => T = 31,76us
00091 // 1TIC      = 11,90ns
00092 //
00093 //--------------------------------------------------------------
00094 #define  VGA_TIM2_HSYNC_PERIODE   2667
00095 #define  VGA_TIM2_HSYNC_PRESCALE     0
00096
00097 #define  VGA_TIM2_HSYNC_IMP       320  // HSync-length (3,81us)
00098 #define  VGA_TIM2_HTRIGGER_START  480  // HSync+BackPorch (5,71us)
00099 #define  VGA_TIM2_DMA_DELAY        60  // ease the delay when DMA START (Optimization = none)
00100 //#define  VGA_TIM2_DMA_DELAY        30  // ease the delay when DMA START (Optimization = -O1)
00101
00102
00103 //--------------------------------------------------------------
00104 // VSync-Signal
00105 // Trigger   = Timer2 Update (f=31,48kHz => T = 31,76us)
00106 // 1TIC      = 31,76us
00107 //--------------------------------------------------------------
00108 #define  VGA_VSYNC_PERIODE        525
00109 #define  VGA_VSYNC_IMP  2
00110 #define  VGA_VSYNC_BILD_START      36
00111 #define  VGA_VSYNC_BILD_STOP      514   // (16,38ms)
00112 #define RAM_SIZE        (VGA_DISPLAY_X+1)*VGA_DISPLAY_Y
00113
00114
00115 //--------------------------------------------------------------
00116 // Adress from PORTE (Reg ODR) callback DMA
00117 // (see Page 53+204 of the Manual)
00118 //
00119 // Data-Bit0 => PE8
00120 // Data-Bit7 => PE15
00121 //--------------------------------------------------------------
00122 #define VGA_GPIOE_BASE_ADR     ((uint32_t)0x40021000) // ADR from Port-E
00123 #define VGA_GPIO_ODR_OFFSET    ((uint32_t)0x00000014) // ADR from Register ODR
00124 #define VGA_GPIO_BYTE_OFFSET   ((uint32_t)0x00000001) // Data for 8bit
00125 #define VGA_GPIOE_ODR_ADDRESS   (VGA_GPIOE_BASE_ADR | VGA_GPIO_ODR_OFFSET | VGA_GPIO_BYTE_OFFSET)
00126
00127 //--------------------------------------------------------------
00128 // Define for black on PE8 - PE15
00129 //--------------------------------------------------------------
00130 #define VGA_GPIO_HINIBBLE  ((uint16_t)0xFF00) // GPIO_Pin_8 to GPIO_Pin_15
00131
00132 //--------------------------------------------------------------
00133 // Global Function call
00134 //--------------------------------------------------------------
00135 void UB_VGA_Screen_Init(void);
00136 void UB_VGA_FillScreen(uint8_t color);
00137 void UB_VGA_SetPixel(uint16_t xp, uint16_t yp, uint8_t color);
00138
00139 //--------------------------------------------------------------
00140 #endif // __STM32F4_UB_VGA_SCREEN_H
```

7.23 CUBE_IDE/VGA/Core/Inc/stm32f4xx.h File Reference

CMSIS Cortex-M4 Device Peripheral Access Layer Header File. This file contains all the peripheral register's definitions, bits definitions and memory mapping for STM32F4xx devices.

```
#include "core_cm4.h"
#include "system_stm32f4xx.h"
#include <stdint.h>
```

Data Structures

- struct ADC_TypeDef

  Analog to Digital Converter

- struct ADC_Common_TypeDef

- struct CAN_TxMailBox_TypeDef
  Controller Area Network TxMailBox.
- struct CAN_FIFOMailBox_TypeDef
  Controller Area Network FIFOMailBox.
- struct CAN_FilterRegister_TypeDef
  Controller Area Network FilterRegister.
- struct CAN_TypeDef
  Controller Area Network.
- struct CRC_TypeDef
  CRC calculation unit.
- struct DAC_TypeDef
  Digital to Analog Converter.
- struct DBGMCU_TypeDef
  Debug MCU.
- struct DCMI_TypeDef
  DCMI.
- struct DMA_Stream_TypeDef
  DMA Controller.
- struct DMA_TypeDef
- struct ETH_TypeDef
  Ethernet MAC.
- struct EXTI_TypeDef
  External Interrupt/Event Controller.
- struct FLASH_TypeDef
  FLASH Registers.
- struct FSMC_Bank1_TypeDef
  Flexible Static Memory Controller.
- struct FSMC_Bank1E_TypeDef
  Flexible Static Memory Controller Bank1E.
- struct FSMC_Bank2_TypeDef
  Flexible Static Memory Controller Bank2.
- struct FSMC_Bank3_TypeDef
  Flexible Static Memory Controller Bank3.
- struct FSMC_Bank4_TypeDef
  Flexible Static Memory Controller Bank4.
- struct GPIO_TypeDef
  General Purpose I/O.
- struct SYSCFG_TypeDef
  System configuration controller.
- struct I2C_TypeDef
  Inter-integrated Circuit Interface.
- struct IWDG_TypeDef
  Independent WATCHDOG.
- struct PWR_TypeDef
  Power Control.
- struct RCC_TypeDef
  Reset and Clock Control.
- struct RTC_TypeDef
  Real-Time Clock.
- struct SDIO_TypeDef
  SD host Interface.

- struct SPI_TypeDef

  Serial Peripheral Interface.
- struct TIM_TypeDef

  TIM.
- struct USART_TypeDef

  Universal Synchronous Asynchronous Receiver Transmitter.
- struct WWDG_TypeDef

  Window WATCHDOG.
- struct CRYP_TypeDef

  Crypto Processor.
- struct HASH_TypeDef

  HASH.
- struct RNG_TypeDef

  HASH.

Macros

- #define **STM32F4XX**
- #define HSE_VALUE ((uint32_t)8000000)

  Comment the line below if you will not use the peripherals drivers. In this case, these drivers will not be included and the application code will be based on direct access to peripherals registers.
- #define HSE_STARTUP_TIMEOUT ((uint16_t)0x0500)

  In the following line adjust the External High Speed oscillator (HSE) Startup Timeout value.
- #define HSI_VALUE ((uint32_t)16000000)
- #define __STM32F4XX_STDPERIPH_VERSION_MAIN (0x01)

  STM32F4XX Standard Peripherals Library version number V1.0.0.
- #define __STM32F4XX_STDPERIPH_VERSION_SUB1 (0x00)
- #define __STM32F4XX_STDPERIPH_VERSION_SUB2 (0x00)
- #define __STM32F4XX_STDPERIPH_VERSION_RC (0x00)
- #define __STM32F4XX_STDPERIPH_VERSION
- #define __CM4_REV 0x0001

  Configuration of the Cortex-M4 Processor and Core Peripherals.
- #define __MPU_PRESENT 1
- #define __NVIC_PRIO_BITS 4
- #define __Vendor_SysTickConfig 0
- #define __FPU_PRESENT 1
- #define **IS_FUNCTIONAL_STATE**(STATE) (((STATE) == DISABLE) || ((STATE) == ENABLE))
- #define FLASH_BASE ((uint32_t)0x08000000)
- #define CCMDATARAM_BASE ((uint32_t)0x10000000)
- #define SRAM1_BASE ((uint32_t)0x20000000)
- #define SRAM2_BASE ((uint32_t)0x2001C000)
- #define PERIPH_BASE ((uint32_t)0x40000000)
- #define BKPSRAM_BASE ((uint32_t)0x40024000)
- #define FSMC_R_BASE ((uint32_t)0xA0000000)
- #define CCMDATARAM_BB_BASE ((uint32_t)0x12000000)
- #define SRAM1_BB_BASE ((uint32_t)0x22000000)
- #define SRAM2_BB_BASE ((uint32_t)0x2201C000)
- #define PERIPH_BB_BASE ((uint32_t)0x42000000)
- #define BKPSRAM_BB_BASE ((uint32_t)0x42024000)
- #define **SRAM_BASE** SRAM1_BASE
- #define SRAM_BB_BASE SRAM1_BB_BASE
- #define **APB1PERIPH_BASE** PERIPH_BASE

- #define **APB2PERIPH_BASE** (PERIPH_BASE + 0x00010000)
- #define **AHB1PERIPH_BASE** (PERIPH_BASE + 0x00020000)
- #define AHB2PERIPH_BASE (PERIPH_BASE + 0x10000000)
- #define **TIM2_BASE** (APB1PERIPH_BASE + 0x0000)
- #define **TIM3_BASE** (APB1PERIPH_BASE + 0x0400)
- #define **TIM4_BASE** (APB1PERIPH_BASE + 0x0800)
- #define **TIM5_BASE** (APB1PERIPH_BASE + 0x0C00)
- #define **TIM6_BASE** (APB1PERIPH_BASE + 0x1000)
- #define **TIM7_BASE** (APB1PERIPH_BASE + 0x1400)
- #define **TIM12_BASE** (APB1PERIPH_BASE + 0x1800)
- #define **TIM13_BASE** (APB1PERIPH_BASE + 0x1C00)
- #define **TIM14_BASE** (APB1PERIPH_BASE + 0x2000)
- #define **RTC_BASE** (APB1PERIPH_BASE + 0x2800)
- #define **WWDG_BASE** (APB1PERIPH_BASE + 0x2C00)
- #define **IWDG_BASE** (APB1PERIPH_BASE + 0x3000)
- #define **I2S2ext_BASE** (APB1PERIPH_BASE + 0x3400)
- #define **SPI2_BASE** (APB1PERIPH_BASE + 0x3800)
- #define **SPI3_BASE** (APB1PERIPH_BASE + 0x3C00)
- #define **I2S3ext_BASE** (APB1PERIPH_BASE + 0x4000)
- #define **USART2_BASE** (APB1PERIPH_BASE + 0x4400)
- #define **USART3_BASE** (APB1PERIPH_BASE + 0x4800)
- #define **UART4_BASE** (APB1PERIPH_BASE + 0x4C00)
- #define **UART5_BASE** (APB1PERIPH_BASE + 0x5000)
- #define **I2C1_BASE** (APB1PERIPH_BASE + 0x5400)
- #define **I2C2_BASE** (APB1PERIPH_BASE + 0x5800)
- #define **I2C3_BASE** (APB1PERIPH_BASE + 0x5C00)
- #define **CAN1_BASE** (APB1PERIPH_BASE + 0x6400)
- #define **CAN2_BASE** (APB1PERIPH_BASE + 0x6800)
- #define **PWR_BASE** (APB1PERIPH_BASE + 0x7000)
- #define DAC_BASE (APB1PERIPH_BASE + 0x7400)
- #define **TIM1_BASE** (APB2PERIPH_BASE + 0x0000)
- #define **TIM8_BASE** (APB2PERIPH_BASE + 0x0400)
- #define **USART1_BASE** (APB2PERIPH_BASE + 0x1000)
- #define **USART6_BASE** (APB2PERIPH_BASE + 0x1400)
- #define **ADC1_BASE** (APB2PERIPH_BASE + 0x2000)
- #define **ADC2_BASE** (APB2PERIPH_BASE + 0x2100)
- #define **ADC3_BASE** (APB2PERIPH_BASE + 0x2200)
- #define **ADC_BASE** (APB2PERIPH_BASE + 0x2300)
- #define **SDIO_BASE** (APB2PERIPH_BASE + 0x2C00)
- #define **SPI1_BASE** (APB2PERIPH_BASE + 0x3000)
- #define **SYSCFG_BASE** (APB2PERIPH_BASE + 0x3800)
- #define **EXTI_BASE** (APB2PERIPH_BASE + 0x3C00)
- #define **TIM9_BASE** (APB2PERIPH_BASE + 0x4000)
- #define **TIM10_BASE** (APB2PERIPH_BASE + 0x4400)
- #define TIM11_BASE (APB2PERIPH_BASE + 0x4800)
- #define **GPIOA_BASE** (AHB1PERIPH_BASE + 0x0000)
- #define **GPIOB_BASE** (AHB1PERIPH_BASE + 0x0400)
- #define **GPIOC_BASE** (AHB1PERIPH_BASE + 0x0800)
- #define **GPIOD_BASE** (AHB1PERIPH_BASE + 0x0C00)
- #define **GPIOE_BASE** (AHB1PERIPH_BASE + 0x1000)
- #define **GPIOF_BASE** (AHB1PERIPH_BASE + 0x1400)
- #define **GPIOG_BASE** (AHB1PERIPH_BASE + 0x1800)
- #define **GPIOH_BASE** (AHB1PERIPH_BASE + 0x1C00)
- #define **GPIOI_BASE** (AHB1PERIPH_BASE + 0x2000)
- #define **CRC_BASE** (AHB1PERIPH_BASE + 0x3000)

- #define **RCC_BASE** (AHB1PERIPH_BASE + 0x3800)
- #define **FLASH_R_BASE** (AHB1PERIPH_BASE + 0x3C00)
- #define **DMA1_BASE** (AHB1PERIPH_BASE + 0x6000)
- #define **DMA1_Stream0_BASE** (DMA1_BASE + 0x010)
- #define **DMA1_Stream1_BASE** (DMA1_BASE + 0x028)
- #define **DMA1_Stream2_BASE** (DMA1_BASE + 0x040)
- #define **DMA1_Stream3_BASE** (DMA1_BASE + 0x058)
- #define **DMA1_Stream4_BASE** (DMA1_BASE + 0x070)
- #define **DMA1_Stream5_BASE** (DMA1_BASE + 0x088)
- #define **DMA1_Stream6_BASE** (DMA1_BASE + 0x0A0)
- #define **DMA1_Stream7_BASE** (DMA1_BASE + 0x0B8)
- #define **DMA2_BASE** (AHB1PERIPH_BASE + 0x6400)
- #define **DMA2_Stream0_BASE** (DMA2_BASE + 0x010)
- #define **DMA2_Stream1_BASE** (DMA2_BASE + 0x028)
- #define **DMA2_Stream2_BASE** (DMA2_BASE + 0x040)
- #define **DMA2_Stream3_BASE** (DMA2_BASE + 0x058)
- #define **DMA2_Stream4_BASE** (DMA2_BASE + 0x070)
- #define **DMA2_Stream5_BASE** (DMA2_BASE + 0x088)
- #define **DMA2_Stream6_BASE** (DMA2_BASE + 0x0A0)
- #define **DMA2_Stream7_BASE** (DMA2_BASE + 0x0B8)
- #define **ETH_BASE** (AHB1PERIPH_BASE + 0x8000)
- #define **ETH_MAC_BASE** (ETH_BASE)
- #define **ETH_MMC_BASE** (ETH_BASE + 0x0100)
- #define **ETH_PTP_BASE** (ETH_BASE + 0x0700)
- #define ETH_DMA_BASE (ETH_BASE + 0x1000)
- #define **DCMI_BASE** (AHB2PERIPH_BASE + 0x50000)
- #define **CRYP_BASE** (AHB2PERIPH_BASE + 0x60000)
- #define **HASH_BASE** (AHB2PERIPH_BASE + 0x60400)
- #define RNG_BASE (AHB2PERIPH_BASE + 0x60800)
- #define **FSMC_Bank1_R_BASE** (FSMC_R_BASE + 0x0000)
- #define **FSMC_Bank1E_R_BASE** (FSMC_R_BASE + 0x0104)
- #define **FSMC_Bank2_R_BASE** (FSMC_R_BASE + 0x0060)
- #define **FSMC_Bank3_R_BASE** (FSMC_R_BASE + 0x0080)
- #define **FSMC_Bank4_R_BASE** (FSMC_R_BASE + 0x00A0)
- #define **DBGMCU_BASE** ((uint32_t)0xE0042000)
- #define **TIM2** ((TIM_TypeDef ∗) TIM2_BASE)
- #define **TIM3** ((TIM_TypeDef ∗) TIM3_BASE)
- #define **TIM4** ((TIM_TypeDef ∗) TIM4_BASE)
- #define **TIM5** ((TIM_TypeDef ∗) TIM5_BASE)
- #define **TIM6** ((TIM_TypeDef ∗) TIM6_BASE)
- #define **TIM7** ((TIM_TypeDef ∗) TIM7_BASE)
- #define **TIM12** ((TIM_TypeDef ∗) TIM12_BASE)
- #define **TIM13** ((TIM_TypeDef ∗) TIM13_BASE)
- #define **TIM14** ((TIM_TypeDef ∗) TIM14_BASE)
- #define **RTC** ((RTC_TypeDef ∗) RTC_BASE)
- #define **WWDG** ((WWDG_TypeDef ∗) WWDG_BASE)
- #define **IWDG** ((IWDG_TypeDef ∗) IWDG_BASE)
- #define **I2S2ext** ((SPI_TypeDef ∗) I2S2ext_BASE)
- #define **SPI2** ((SPI_TypeDef ∗) SPI2_BASE)
- #define **SPI3** ((SPI_TypeDef ∗) SPI3_BASE)
- #define **I2S3ext** ((SPI_TypeDef ∗) I2S3ext_BASE)
- #define **USART2** ((USART_TypeDef ∗) USART2_BASE)
- #define **USART3** ((USART_TypeDef ∗) USART3_BASE)
- #define **UART4** ((USART_TypeDef ∗) UART4_BASE)
- #define **UART5** ((USART_TypeDef ∗) UART5_BASE)

- #define **I2C1** ((I2C_TypeDef *) I2C1_BASE)
- #define **I2C2** ((I2C_TypeDef *) I2C2_BASE)
- #define **I2C3** ((I2C_TypeDef *) I2C3_BASE)
- #define **CAN1** ((CAN_TypeDef *) CAN1_BASE)
- #define **CAN2** ((CAN_TypeDef *) CAN2_BASE)
- #define **PWR** ((PWR_TypeDef *) PWR_BASE)
- #define **DAC** ((DAC_TypeDef *) DAC_BASE)
- #define **TIM1** ((TIM_TypeDef *) TIM1_BASE)
- #define **TIM8** ((TIM_TypeDef *) TIM8_BASE)
- #define **USART1** ((USART_TypeDef *) USART1_BASE)
- #define **USART6** ((USART_TypeDef *) USART6_BASE)
- #define **ADC** ((ADC_Common_TypeDef *) ADC_BASE)
- #define **ADC1** ((ADC_TypeDef *) ADC1_BASE)
- #define **ADC2** ((ADC_TypeDef *) ADC2_BASE)
- #define **ADC3** ((ADC_TypeDef *) ADC3_BASE)
- #define **SDIO** ((SDIO_TypeDef *) SDIO_BASE)
- #define **SPI1** ((SPI_TypeDef *) SPI1_BASE)
- #define **SYSCFG** ((SYSCFG_TypeDef *) SYSCFG_BASE)
- #define **EXTI** ((EXTI_TypeDef *) EXTI_BASE)
- #define **TIM9** ((TIM_TypeDef *) TIM9_BASE)
- #define **TIM10** ((TIM_TypeDef *) TIM10_BASE)
- #define **TIM11** ((TIM_TypeDef *) TIM11_BASE)
- #define **GPIOA** ((GPIO_TypeDef *) GPIOA_BASE)
- #define **GPIOB** ((GPIO_TypeDef *) GPIOB_BASE)
- #define **GPIOC** ((GPIO_TypeDef *) GPIOC_BASE)
- #define **GPIOD** ((GPIO_TypeDef *) GPIOD_BASE)
- #define **GPIOE** ((GPIO_TypeDef *) GPIOE_BASE)
- #define **GPIOF** ((GPIO_TypeDef *) GPIOF_BASE)
- #define **GPIOG** ((GPIO_TypeDef *) GPIOG_BASE)
- #define **GPIOH** ((GPIO_TypeDef *) GPIOH_BASE)
- #define **GPIOI** ((GPIO_TypeDef *) GPIOI_BASE)
- #define **CRC** ((CRC_TypeDef *) CRC_BASE)
- #define **RCC** ((RCC_TypeDef *) RCC_BASE)
- #define **FLASH** ((FLASH_TypeDef *) FLASH_R_BASE)
- #define **DMA1** ((DMA_TypeDef *) DMA1_BASE)
- #define **DMA1_Stream0** ((DMA_Stream_TypeDef *) DMA1_Stream0_BASE)
- #define **DMA1_Stream1** ((DMA_Stream_TypeDef *) DMA1_Stream1_BASE)
- #define **DMA1_Stream2** ((DMA_Stream_TypeDef *) DMA1_Stream2_BASE)
- #define **DMA1_Stream3** ((DMA_Stream_TypeDef *) DMA1_Stream3_BASE)
- #define **DMA1_Stream4** ((DMA_Stream_TypeDef *) DMA1_Stream4_BASE)
- #define **DMA1_Stream5** ((DMA_Stream_TypeDef *) DMA1_Stream5_BASE)
- #define **DMA1_Stream6** ((DMA_Stream_TypeDef *) DMA1_Stream6_BASE)
- #define **DMA1_Stream7** ((DMA_Stream_TypeDef *) DMA1_Stream7_BASE)
- #define **DMA2** ((DMA_TypeDef *) DMA2_BASE)
- #define **DMA2_Stream0** ((DMA_Stream_TypeDef *) DMA2_Stream0_BASE)
- #define **DMA2_Stream1** ((DMA_Stream_TypeDef *) DMA2_Stream1_BASE)
- #define **DMA2_Stream2** ((DMA_Stream_TypeDef *) DMA2_Stream2_BASE)
- #define **DMA2_Stream3** ((DMA_Stream_TypeDef *) DMA2_Stream3_BASE)
- #define **DMA2_Stream4** ((DMA_Stream_TypeDef *) DMA2_Stream4_BASE)
- #define **DMA2_Stream5** ((DMA_Stream_TypeDef *) DMA2_Stream5_BASE)
- #define **DMA2_Stream6** ((DMA_Stream_TypeDef *) DMA2_Stream6_BASE)
- #define **DMA2_Stream7** ((DMA_Stream_TypeDef *) DMA2_Stream7_BASE)
- #define **ETH** ((ETH_TypeDef *) ETH_BASE)
- #define **DCMI** ((DCMI_TypeDef *) DCMI_BASE)
- #define **CRYP** ((CRYP_TypeDef *) CRYP_BASE)

- #define **HASH** ((HASH_TypeDef *) HASH_BASE)
- #define **RNG** ((RNG_TypeDef *) RNG_BASE)
- #define **FSMC_Bank1** ((FSMC_Bank1_TypeDef *) FSMC_Bank1_R_BASE)
- #define **FSMC_Bank1E** ((FSMC_Bank1E_TypeDef *) FSMC_Bank1E_R_BASE)
- #define **FSMC_Bank2** ((FSMC_Bank2_TypeDef *) FSMC_Bank2_R_BASE)
- #define **FSMC_Bank3** ((FSMC_Bank3_TypeDef *) FSMC_Bank3_R_BASE)
- #define **FSMC_Bank4** ((FSMC_Bank4_TypeDef *) FSMC_Bank4_R_BASE)
- #define **DBGMCU** ((DBGMCU_TypeDef *) DBGMCU_BASE)
- #define ADC_SR_AWD ((uint8_t)0x01)
- #define ADC_SR_EOC ((uint8_t)0x02)
- #define ADC_SR_JEOC ((uint8_t)0x04)
- #define ADC_SR_JSTRT ((uint8_t)0x08)
- #define ADC_SR_STRT ((uint8_t)0x10)
- #define ADC_SR_OVR ((uint8_t)0x20)
- #define ADC_CR1_AWDCH ((uint32_t)0x0000001F)
- #define ADC_CR1_AWDCH_0 ((uint32_t)0x00000001)
- #define ADC_CR1_AWDCH_1 ((uint32_t)0x00000002)
- #define ADC_CR1_AWDCH_2 ((uint32_t)0x00000004)
- #define ADC_CR1_AWDCH_3 ((uint32_t)0x00000008)
- #define ADC_CR1_AWDCH_4 ((uint32_t)0x00000010)
- #define ADC_CR1_EOCIE ((uint32_t)0x00000020)
- #define ADC_CR1_AWDIE ((uint32_t)0x00000040)
- #define ADC_CR1_JEOCIE ((uint32_t)0x00000080)
- #define ADC_CR1_SCAN ((uint32_t)0x00000100)
- #define ADC_CR1_AWDSGL ((uint32_t)0x00000200)
- #define ADC_CR1_JAUTO ((uint32_t)0x00000400)
- #define ADC_CR1_DISCEN ((uint32_t)0x00000800)
- #define ADC_CR1_JDISCEN ((uint32_t)0x00001000)
- #define ADC_CR1_DISCNUM ((uint32_t)0x0000E000)
- #define ADC_CR1_DISCNUM_0 ((uint32_t)0x00002000)
- #define ADC_CR1_DISCNUM_1 ((uint32_t)0x00004000)
- #define ADC_CR1_DISCNUM_2 ((uint32_t)0x00008000)
- #define ADC_CR1_JAWDEN ((uint32_t)0x00400000)
- #define ADC_CR1_AWDEN ((uint32_t)0x00800000)
- #define ADC_CR1_RES ((uint32_t)0x03000000)
- #define ADC_CR1_RES_0 ((uint32_t)0x01000000)
- #define ADC_CR1_RES_1 ((uint32_t)0x02000000)
- #define ADC_CR1_OVRIE ((uint32_t)0x04000000)
- #define ADC_CR2_ADON ((uint32_t)0x00000001)
- #define ADC_CR2_CONT ((uint32_t)0x00000002)
- #define ADC_CR2_DMA ((uint32_t)0x00000100)
- #define ADC_CR2_DDS ((uint32_t)0x00000200)
- #define ADC_CR2_EOCS ((uint32_t)0x00000400)
- #define ADC_CR2_ALIGN ((uint32_t)0x00000800)
- #define ADC_CR2_JEXTSEL ((uint32_t)0x000F0000)
- #define ADC_CR2_JEXTSEL_0 ((uint32_t)0x00010000)
- #define ADC_CR2_JEXTSEL_1 ((uint32_t)0x00020000)
- #define ADC_CR2_JEXTSEL_2 ((uint32_t)0x00040000)
- #define ADC_CR2_JEXTSEL_3 ((uint32_t)0x00080000)
- #define ADC_CR2_JEXTEN ((uint32_t)0x00300000)
- #define ADC_CR2_JEXTEN_0 ((uint32_t)0x00100000)
- #define ADC_CR2_JEXTEN_1 ((uint32_t)0x00200000)
- #define ADC_CR2_JSWSTART ((uint32_t)0x00400000)
- #define ADC_CR2_EXTSEL ((uint32_t)0x0F000000)
- #define ADC_CR2_EXTSEL_0 ((uint32_t)0x01000000)

- #define ADC_CR2_EXTSEL_1 ((uint32_t)0x02000000)
- #define ADC_CR2_EXTSEL_2 ((uint32_t)0x04000000)
- #define ADC_CR2_EXTSEL_3 ((uint32_t)0x08000000)
- #define ADC_CR2_EXTEN ((uint32_t)0x30000000)
- #define ADC_CR2_EXTEN_0 ((uint32_t)0x10000000)
- #define ADC_CR2_EXTEN_1 ((uint32_t)0x20000000)
- #define ADC_CR2_SWSTART ((uint32_t)0x40000000)
- #define ADC_SMPR1_SMP10 ((uint32_t)0x00000007)
- #define ADC_SMPR1_SMP10_0 ((uint32_t)0x00000001)
- #define ADC_SMPR1_SMP10_1 ((uint32_t)0x00000002)
- #define ADC_SMPR1_SMP10_2 ((uint32_t)0x00000004)
- #define ADC_SMPR1_SMP11 ((uint32_t)0x00000038)
- #define ADC_SMPR1_SMP11_0 ((uint32_t)0x00000008)
- #define ADC_SMPR1_SMP11_1 ((uint32_t)0x00000010)
- #define ADC_SMPR1_SMP11_2 ((uint32_t)0x00000020)
- #define ADC_SMPR1_SMP12 ((uint32_t)0x000001C0)
- #define ADC_SMPR1_SMP12_0 ((uint32_t)0x00000040)
- #define ADC_SMPR1_SMP12_1 ((uint32_t)0x00000080)
- #define ADC_SMPR1_SMP12_2 ((uint32_t)0x00000100)
- #define ADC_SMPR1_SMP13 ((uint32_t)0x00000E00)
- #define ADC_SMPR1_SMP13_0 ((uint32_t)0x00000200)
- #define ADC_SMPR1_SMP13_1 ((uint32_t)0x00000400)
- #define ADC_SMPR1_SMP13_2 ((uint32_t)0x00000800)
- #define ADC_SMPR1_SMP14 ((uint32_t)0x00007000)
- #define ADC_SMPR1_SMP14_0 ((uint32_t)0x00001000)
- #define ADC_SMPR1_SMP14_1 ((uint32_t)0x00002000)
- #define ADC_SMPR1_SMP14_2 ((uint32_t)0x00004000)
- #define ADC_SMPR1_SMP15 ((uint32_t)0x00038000)
- #define ADC_SMPR1_SMP15_0 ((uint32_t)0x00008000)
- #define ADC_SMPR1_SMP15_1 ((uint32_t)0x00010000)
- #define ADC_SMPR1_SMP15_2 ((uint32_t)0x00020000)
- #define ADC_SMPR1_SMP16 ((uint32_t)0x001C0000)
- #define ADC_SMPR1_SMP16_0 ((uint32_t)0x00040000)
- #define ADC_SMPR1_SMP16_1 ((uint32_t)0x00080000)
- #define ADC_SMPR1_SMP16_2 ((uint32_t)0x00100000)
- #define ADC_SMPR1_SMP17 ((uint32_t)0x00E00000)
- #define ADC_SMPR1_SMP17_0 ((uint32_t)0x00200000)
- #define ADC_SMPR1_SMP17_1 ((uint32_t)0x00400000)
- #define ADC_SMPR1_SMP17_2 ((uint32_t)0x00800000)
- #define ADC_SMPR1_SMP18 ((uint32_t)0x07000000)
- #define ADC_SMPR1_SMP18_0 ((uint32_t)0x01000000)
- #define ADC_SMPR1_SMP18_1 ((uint32_t)0x02000000)
- #define ADC_SMPR1_SMP18_2 ((uint32_t)0x04000000)
- #define ADC_SMPR2_SMP0 ((uint32_t)0x00000007)
- #define ADC_SMPR2_SMP0_0 ((uint32_t)0x00000001)
- #define ADC_SMPR2_SMP0_1 ((uint32_t)0x00000002)
- #define ADC_SMPR2_SMP0_2 ((uint32_t)0x00000004)
- #define ADC_SMPR2_SMP1 ((uint32_t)0x00000038)
- #define ADC_SMPR2_SMP1_0 ((uint32_t)0x00000008)
- #define ADC_SMPR2_SMP1_1 ((uint32_t)0x00000010)
- #define ADC_SMPR2_SMP1_2 ((uint32_t)0x00000020)
- #define ADC_SMPR2_SMP2 ((uint32_t)0x000001C0)
- #define ADC_SMPR2_SMP2_0 ((uint32_t)0x00000040)
- #define ADC_SMPR2_SMP2_1 ((uint32_t)0x00000080)
- #define ADC_SMPR2_SMP2_2 ((uint32_t)0x00000100)

- #define ADC_SMPR2_SMP3 ((uint32_t)0x00000E00)
- #define ADC_SMPR2_SMP3_0 ((uint32_t)0x00000200)
- #define ADC_SMPR2_SMP3_1 ((uint32_t)0x00000400)
- #define ADC_SMPR2_SMP3_2 ((uint32_t)0x00000800)
- #define ADC_SMPR2_SMP4 ((uint32_t)0x00007000)
- #define ADC_SMPR2_SMP4_0 ((uint32_t)0x00001000)
- #define ADC_SMPR2_SMP4_1 ((uint32_t)0x00002000)
- #define ADC_SMPR2_SMP4_2 ((uint32_t)0x00004000)
- #define ADC_SMPR2_SMP5 ((uint32_t)0x00038000)
- #define ADC_SMPR2_SMP5_0 ((uint32_t)0x00008000)
- #define ADC_SMPR2_SMP5_1 ((uint32_t)0x00010000)
- #define ADC_SMPR2_SMP5_2 ((uint32_t)0x00020000)
- #define ADC_SMPR2_SMP6 ((uint32_t)0x001C0000)
- #define ADC_SMPR2_SMP6_0 ((uint32_t)0x00040000)
- #define ADC_SMPR2_SMP6_1 ((uint32_t)0x00080000)
- #define ADC_SMPR2_SMP6_2 ((uint32_t)0x00100000)
- #define ADC_SMPR2_SMP7 ((uint32_t)0x00E00000)
- #define ADC_SMPR2_SMP7_0 ((uint32_t)0x00200000)
- #define ADC_SMPR2_SMP7_1 ((uint32_t)0x00400000)
- #define ADC_SMPR2_SMP7_2 ((uint32_t)0x00800000)
- #define ADC_SMPR2_SMP8 ((uint32_t)0x07000000)
- #define ADC_SMPR2_SMP8_0 ((uint32_t)0x01000000)
- #define ADC_SMPR2_SMP8_1 ((uint32_t)0x02000000)
- #define ADC_SMPR2_SMP8_2 ((uint32_t)0x04000000)
- #define ADC_SMPR2_SMP9 ((uint32_t)0x38000000)
- #define ADC_SMPR2_SMP9_0 ((uint32_t)0x08000000)
- #define ADC_SMPR2_SMP9_1 ((uint32_t)0x10000000)
- #define ADC_SMPR2_SMP9_2 ((uint32_t)0x20000000)
- #define ADC_JOFR1_JOFFSET1 ((uint16_t)0x0FFF)
- #define ADC_JOFR2_JOFFSET2 ((uint16_t)0x0FFF)
- #define ADC_JOFR3_JOFFSET3 ((uint16_t)0x0FFF)
- #define ADC_JOFR4_JOFFSET4 ((uint16_t)0x0FFF)
- #define ADC_HTR_HT ((uint16_t)0x0FFF)
- #define ADC_LTR_LT ((uint16_t)0x0FFF)
- #define ADC_SQR1_SQ13 ((uint32_t)0x0000001F)
- #define ADC_SQR1_SQ13_0 ((uint32_t)0x00000001)
- #define ADC_SQR1_SQ13_1 ((uint32_t)0x00000002)
- #define ADC_SQR1_SQ13_2 ((uint32_t)0x00000004)
- #define ADC_SQR1_SQ13_3 ((uint32_t)0x00000008)
- #define ADC_SQR1_SQ13_4 ((uint32_t)0x00000010)
- #define ADC_SQR1_SQ14 ((uint32_t)0x000003E0)
- #define ADC_SQR1_SQ14_0 ((uint32_t)0x00000020)
- #define ADC_SQR1_SQ14_1 ((uint32_t)0x00000040)
- #define ADC_SQR1_SQ14_2 ((uint32_t)0x00000080)
- #define ADC_SQR1_SQ14_3 ((uint32_t)0x00000100)
- #define ADC_SQR1_SQ14_4 ((uint32_t)0x00000200)
- #define ADC_SQR1_SQ15 ((uint32_t)0x00007C00)
- #define ADC_SQR1_SQ15_0 ((uint32_t)0x00000400)
- #define ADC_SQR1_SQ15_1 ((uint32_t)0x00000800)
- #define ADC_SQR1_SQ15_2 ((uint32_t)0x00001000)
- #define ADC_SQR1_SQ15_3 ((uint32_t)0x00002000)
- #define ADC_SQR1_SQ15_4 ((uint32_t)0x00004000)
- #define ADC_SQR1_SQ16 ((uint32_t)0x000F8000)
- #define ADC_SQR1_SQ16_0 ((uint32_t)0x00008000)
- #define ADC_SQR1_SQ16_1 ((uint32_t)0x00010000)

- #define ADC_SQR1_SQ16_2 ((uint32_t)0x00020000)
- #define ADC_SQR1_SQ16_3 ((uint32_t)0x00040000)
- #define ADC_SQR1_SQ16_4 ((uint32_t)0x00080000)
- #define ADC_SQR1_L ((uint32_t)0x00F00000)
- #define ADC_SQR1_L_0 ((uint32_t)0x00100000)
- #define ADC_SQR1_L_1 ((uint32_t)0x00200000)
- #define ADC_SQR1_L_2 ((uint32_t)0x00400000)
- #define ADC_SQR1_L_3 ((uint32_t)0x00800000)
- #define ADC_SQR2_SQ7 ((uint32_t)0x0000001F)
- #define ADC_SQR2_SQ7_0 ((uint32_t)0x00000001)
- #define ADC_SQR2_SQ7_1 ((uint32_t)0x00000002)
- #define ADC_SQR2_SQ7_2 ((uint32_t)0x00000004)
- #define ADC_SQR2_SQ7_3 ((uint32_t)0x00000008)
- #define ADC_SQR2_SQ7_4 ((uint32_t)0x00000010)
- #define ADC_SQR2_SQ8 ((uint32_t)0x000003E0)
- #define ADC_SQR2_SQ8_0 ((uint32_t)0x00000020)
- #define ADC_SQR2_SQ8_1 ((uint32_t)0x00000040)
- #define ADC_SQR2_SQ8_2 ((uint32_t)0x00000080)
- #define ADC_SQR2_SQ8_3 ((uint32_t)0x00000100)
- #define ADC_SQR2_SQ8_4 ((uint32_t)0x00000200)
- #define ADC_SQR2_SQ9 ((uint32_t)0x00007C00)
- #define ADC_SQR2_SQ9_0 ((uint32_t)0x00000400)
- #define ADC_SQR2_SQ9_1 ((uint32_t)0x00000800)
- #define ADC_SQR2_SQ9_2 ((uint32_t)0x00001000)
- #define ADC_SQR2_SQ9_3 ((uint32_t)0x00002000)
- #define ADC_SQR2_SQ9_4 ((uint32_t)0x00004000)
- #define ADC_SQR2_SQ10 ((uint32_t)0x000F8000)
- #define ADC_SQR2_SQ10_0 ((uint32_t)0x00008000)
- #define ADC_SQR2_SQ10_1 ((uint32_t)0x00010000)
- #define ADC_SQR2_SQ10_2 ((uint32_t)0x00020000)
- #define ADC_SQR2_SQ10_3 ((uint32_t)0x00040000)
- #define ADC_SQR2_SQ10_4 ((uint32_t)0x00080000)
- #define ADC_SQR2_SQ11 ((uint32_t)0x01F00000)
- #define ADC_SQR2_SQ11_0 ((uint32_t)0x00100000)
- #define ADC_SQR2_SQ11_1 ((uint32_t)0x00200000)
- #define ADC_SQR2_SQ11_2 ((uint32_t)0x00400000)
- #define ADC_SQR2_SQ11_3 ((uint32_t)0x00800000)
- #define ADC_SQR2_SQ11_4 ((uint32_t)0x01000000)
- #define ADC_SQR2_SQ12 ((uint32_t)0x3E000000)
- #define ADC_SQR2_SQ12_0 ((uint32_t)0x02000000)
- #define ADC_SQR2_SQ12_1 ((uint32_t)0x04000000)
- #define ADC_SQR2_SQ12_2 ((uint32_t)0x08000000)
- #define ADC_SQR2_SQ12_3 ((uint32_t)0x10000000)
- #define ADC_SQR2_SQ12_4 ((uint32_t)0x20000000)
- #define ADC_SQR3_SQ1 ((uint32_t)0x0000001F)
- #define ADC_SQR3_SQ1_0 ((uint32_t)0x00000001)
- #define ADC_SQR3_SQ1_1 ((uint32_t)0x00000002)
- #define ADC_SQR3_SQ1_2 ((uint32_t)0x00000004)
- #define ADC_SQR3_SQ1_3 ((uint32_t)0x00000008)
- #define ADC_SQR3_SQ1_4 ((uint32_t)0x00000010)
- #define ADC_SQR3_SQ2 ((uint32_t)0x000003E0)
- #define ADC_SQR3_SQ2_0 ((uint32_t)0x00000020)
- #define ADC_SQR3_SQ2_1 ((uint32_t)0x00000040)
- #define ADC_SQR3_SQ2_2 ((uint32_t)0x00000080)
- #define ADC_SQR3_SQ2_3 ((uint32_t)0x00000100)

- #define ADC_SQR3_SQ2_4 ((uint32_t)0x00000200)
- #define ADC_SQR3_SQ3 ((uint32_t)0x00007C00)
- #define ADC_SQR3_SQ3_0 ((uint32_t)0x00000400)
- #define ADC_SQR3_SQ3_1 ((uint32_t)0x00000800)
- #define ADC_SQR3_SQ3_2 ((uint32_t)0x00001000)
- #define ADC_SQR3_SQ3_3 ((uint32_t)0x00002000)
- #define ADC_SQR3_SQ3_4 ((uint32_t)0x00004000)
- #define ADC_SQR3_SQ4 ((uint32_t)0x000F8000)
- #define ADC_SQR3_SQ4_0 ((uint32_t)0x00008000)
- #define ADC_SQR3_SQ4_1 ((uint32_t)0x00010000)
- #define ADC_SQR3_SQ4_2 ((uint32_t)0x00020000)
- #define ADC_SQR3_SQ4_3 ((uint32_t)0x00040000)
- #define ADC_SQR3_SQ4_4 ((uint32_t)0x00080000)
- #define ADC_SQR3_SQ5 ((uint32_t)0x01F00000)
- #define ADC_SQR3_SQ5_0 ((uint32_t)0x00100000)
- #define ADC_SQR3_SQ5_1 ((uint32_t)0x00200000)
- #define ADC_SQR3_SQ5_2 ((uint32_t)0x00400000)
- #define ADC_SQR3_SQ5_3 ((uint32_t)0x00800000)
- #define ADC_SQR3_SQ5_4 ((uint32_t)0x01000000)
- #define ADC_SQR3_SQ6 ((uint32_t)0x3E000000)
- #define ADC_SQR3_SQ6_0 ((uint32_t)0x02000000)
- #define ADC_SQR3_SQ6_1 ((uint32_t)0x04000000)
- #define ADC_SQR3_SQ6_2 ((uint32_t)0x08000000)
- #define ADC_SQR3_SQ6_3 ((uint32_t)0x10000000)
- #define ADC_SQR3_SQ6_4 ((uint32_t)0x20000000)
- #define ADC_JSQR_JSQ1 ((uint32_t)0x0000001F)
- #define ADC_JSQR_JSQ1_0 ((uint32_t)0x00000001)
- #define ADC_JSQR_JSQ1_1 ((uint32_t)0x00000002)
- #define ADC_JSQR_JSQ1_2 ((uint32_t)0x00000004)
- #define ADC_JSQR_JSQ1_3 ((uint32_t)0x00000008)
- #define ADC_JSQR_JSQ1_4 ((uint32_t)0x00000010)
- #define ADC_JSQR_JSQ2 ((uint32_t)0x000003E0)
- #define ADC_JSQR_JSQ2_0 ((uint32_t)0x00000020)
- #define ADC_JSQR_JSQ2_1 ((uint32_t)0x00000040)
- #define ADC_JSQR_JSQ2_2 ((uint32_t)0x00000080)
- #define ADC_JSQR_JSQ2_3 ((uint32_t)0x00000100)
- #define ADC_JSQR_JSQ2_4 ((uint32_t)0x00000200)
- #define ADC_JSQR_JSQ3 ((uint32_t)0x00007C00)
- #define ADC_JSQR_JSQ3_0 ((uint32_t)0x00000400)
- #define ADC_JSQR_JSQ3_1 ((uint32_t)0x00000800)
- #define ADC_JSQR_JSQ3_2 ((uint32_t)0x00001000)
- #define ADC_JSQR_JSQ3_3 ((uint32_t)0x00002000)
- #define ADC_JSQR_JSQ3_4 ((uint32_t)0x00004000)
- #define ADC_JSQR_JSQ4 ((uint32_t)0x000F8000)
- #define ADC_JSQR_JSQ4_0 ((uint32_t)0x00008000)
- #define ADC_JSQR_JSQ4_1 ((uint32_t)0x00010000)
- #define ADC_JSQR_JSQ4_2 ((uint32_t)0x00020000)
- #define ADC_JSQR_JSQ4_3 ((uint32_t)0x00040000)
- #define ADC_JSQR_JSQ4_4 ((uint32_t)0x00080000)
- #define ADC_JSQR_JL ((uint32_t)0x00300000)
- #define ADC_JSQR_JL_0 ((uint32_t)0x00100000)
- #define ADC_JSQR_JL_1 ((uint32_t)0x00200000)
- #define ADC_JDR1_JDATA ((uint16_t)0xFFFF)
- #define ADC_JDR2_JDATA ((uint16_t)0xFFFF)
- #define ADC_JDR3_JDATA ((uint16_t)0xFFFF)

- #define ADC_JDR4_JDATA ((uint16_t)0xFFFF)
- #define ADC_DR_DATA ((uint32_t)0x0000FFFF)
- #define ADC_DR_ADC2DATA ((uint32_t)0xFFFF0000)
- #define ADC_CSR_AWD1 ((uint32_t)0x00000001)
- #define ADC_CSR_EOC1 ((uint32_t)0x00000002)
- #define ADC_CSR_JEOC1 ((uint32_t)0x00000004)
- #define ADC_CSR_JSTRT1 ((uint32_t)0x00000008)
- #define ADC_CSR_STRT1 ((uint32_t)0x00000010)
- #define ADC_CSR_DOVR1 ((uint32_t)0x00000020)
- #define ADC_CSR_AWD2 ((uint32_t)0x00000100)
- #define ADC_CSR_EOC2 ((uint32_t)0x00000200)
- #define ADC_CSR_JEOC2 ((uint32_t)0x00000400)
- #define ADC_CSR_JSTRT2 ((uint32_t)0x00000800)
- #define ADC_CSR_STRT2 ((uint32_t)0x00001000)
- #define ADC_CSR_DOVR2 ((uint32_t)0x00002000)
- #define ADC_CSR_AWD3 ((uint32_t)0x00010000)
- #define ADC_CSR_EOC3 ((uint32_t)0x00020000)
- #define ADC_CSR_JEOC3 ((uint32_t)0x00040000)
- #define ADC_CSR_JSTRT3 ((uint32_t)0x00080000)
- #define ADC_CSR_STRT3 ((uint32_t)0x00100000)
- #define ADC_CSR_DOVR3 ((uint32_t)0x00200000)
- #define ADC_CCR_MULTI ((uint32_t)0x0000001F)
- #define ADC_CCR_MULTI_0 ((uint32_t)0x00000001)
- #define ADC_CCR_MULTI_1 ((uint32_t)0x00000002)
- #define ADC_CCR_MULTI_2 ((uint32_t)0x00000004)
- #define ADC_CCR_MULTI_3 ((uint32_t)0x00000008)
- #define ADC_CCR_MULTI_4 ((uint32_t)0x00000010)
- #define ADC_CCR_DELAY ((uint32_t)0x00000F00)
- #define ADC_CCR_DELAY_0 ((uint32_t)0x00000100)
- #define ADC_CCR_DELAY_1 ((uint32_t)0x00000200)
- #define ADC_CCR_DELAY_2 ((uint32_t)0x00000400)
- #define ADC_CCR_DELAY_3 ((uint32_t)0x00000800)
- #define ADC_CCR_DDS ((uint32_t)0x00002000)
- #define ADC_CCR_DMA ((uint32_t)0x0000C000)
- #define ADC_CCR_DMA_0 ((uint32_t)0x00004000)
- #define ADC_CCR_DMA_1 ((uint32_t)0x00008000)
- #define ADC_CCR_ADCPRE ((uint32_t)0x00030000)
- #define ADC_CCR_ADCPRE_0 ((uint32_t)0x00010000)
- #define ADC_CCR_ADCPRE_1 ((uint32_t)0x00020000)
- #define ADC_CCR_VBATE ((uint32_t)0x00400000)
- #define ADC_CCR_TSVREFE ((uint32_t)0x00800000)
- #define ADC_CDR_DATA1 ((uint32_t)0x0000FFFF)
- #define ADC_CDR_DATA2 ((uint32_t)0xFFFF0000)
- #define CAN_MCR_INRQ ((uint16_t)0x0001)
- #define CAN_MCR_SLEEP ((uint16_t)0x0002)
- #define CAN_MCR_TXFP ((uint16_t)0x0004)
- #define CAN_MCR_RFLM ((uint16_t)0x0008)
- #define CAN_MCR_NART ((uint16_t)0x0010)
- #define CAN_MCR_AWUM ((uint16_t)0x0020)
- #define CAN_MCR_ABOM ((uint16_t)0x0040)
- #define CAN_MCR_TTCM ((uint16_t)0x0080)
- #define CAN_MCR_RESET ((uint16_t)0x8000)
- #define CAN_MSR_INAK ((uint16_t)0x0001)
- #define CAN_MSR_SLAK ((uint16_t)0x0002)
- #define CAN_MSR_ERRI ((uint16_t)0x0004)

- #define CAN_MSR_WKUI ((uint16_t)0x0008)
- #define CAN_MSR_SLAKI ((uint16_t)0x0010)
- #define CAN_MSR_TXM ((uint16_t)0x0100)
- #define CAN_MSR_RXM ((uint16_t)0x0200)
- #define CAN_MSR_SAMP ((uint16_t)0x0400)
- #define CAN_MSR_RX ((uint16_t)0x0800)
- #define CAN_TSR_RQCP0 ((uint32_t)0x00000001)
- #define CAN_TSR_TXOK0 ((uint32_t)0x00000002)
- #define CAN_TSR_ALST0 ((uint32_t)0x00000004)
- #define CAN_TSR_TERR0 ((uint32_t)0x00000008)
- #define CAN_TSR_ABRQ0 ((uint32_t)0x00000080)
- #define CAN_TSR_RQCP1 ((uint32_t)0x00000100)
- #define CAN_TSR_TXOK1 ((uint32_t)0x00000200)
- #define CAN_TSR_ALST1 ((uint32_t)0x00000400)
- #define CAN_TSR_TERR1 ((uint32_t)0x00000800)
- #define CAN_TSR_ABRQ1 ((uint32_t)0x00008000)
- #define CAN_TSR_RQCP2 ((uint32_t)0x00010000)
- #define CAN_TSR_TXOK2 ((uint32_t)0x00020000)
- #define CAN_TSR_ALST2 ((uint32_t)0x00040000)
- #define CAN_TSR_TERR2 ((uint32_t)0x00080000)
- #define CAN_TSR_ABRQ2 ((uint32_t)0x00800000)
- #define CAN_TSR_CODE ((uint32_t)0x03000000)
- #define CAN_TSR_TME ((uint32_t)0x1C000000)
- #define CAN_TSR_TME0 ((uint32_t)0x04000000)
- #define CAN_TSR_TME1 ((uint32_t)0x08000000)
- #define CAN_TSR_TME2 ((uint32_t)0x10000000)
- #define CAN_TSR_LOW ((uint32_t)0xE0000000)
- #define CAN_TSR_LOW0 ((uint32_t)0x20000000)
- #define CAN_TSR_LOW1 ((uint32_t)0x40000000)
- #define CAN_TSR_LOW2 ((uint32_t)0x80000000)
- #define CAN_RF0R_FMP0 ((uint8_t)0x03)
- #define CAN_RF0R_FULL0 ((uint8_t)0x08)
- #define CAN_RF0R_FOVR0 ((uint8_t)0x10)
- #define CAN_RF0R_RFOM0 ((uint8_t)0x20)
- #define CAN_RF1R_FMP1 ((uint8_t)0x03)
- #define CAN_RF1R_FULL1 ((uint8_t)0x08)
- #define CAN_RF1R_FOVR1 ((uint8_t)0x10)
- #define CAN_RF1R_RFOM1 ((uint8_t)0x20)
- #define CAN_IER_TMEIE ((uint32_t)0x00000001)
- #define CAN_IER_FMPIE0 ((uint32_t)0x00000002)
- #define CAN_IER_FFIE0 ((uint32_t)0x00000004)
- #define CAN_IER_FOVIE0 ((uint32_t)0x00000008)
- #define CAN_IER_FMPIE1 ((uint32_t)0x00000010)
- #define CAN_IER_FFIE1 ((uint32_t)0x00000020)
- #define CAN_IER_FOVIE1 ((uint32_t)0x00000040)
- #define CAN_IER_EWGIE ((uint32_t)0x00000100)
- #define CAN_IER_EPVIE ((uint32_t)0x00000200)
- #define CAN_IER_BOFIE ((uint32_t)0x00000400)
- #define CAN_IER_LECIE ((uint32_t)0x00000800)
- #define CAN_IER_ERRIE ((uint32_t)0x00008000)
- #define CAN_IER_WKUIE ((uint32_t)0x00010000)
- #define CAN_IER_SLKIE ((uint32_t)0x00020000)
- #define CAN_ESR_EWGF ((uint32_t)0x00000001)
- #define CAN_ESR_EPVF ((uint32_t)0x00000002)
- #define CAN_ESR_BOFF ((uint32_t)0x00000004)

- #define CAN_ESR_LEC ((uint32_t)0x00000070)
- #define CAN_ESR_LEC_0 ((uint32_t)0x00000010)
- #define CAN_ESR_LEC_1 ((uint32_t)0x00000020)
- #define CAN_ESR_LEC_2 ((uint32_t)0x00000040)
- #define CAN_ESR_TEC ((uint32_t)0x00FF0000)
- #define CAN_ESR_REC ((uint32_t)0xFF000000)
- #define CAN_BTR_BRP ((uint32_t)0x000003FF)
- #define CAN_BTR_TS1 ((uint32_t)0x000F0000)
- #define CAN_BTR_TS2 ((uint32_t)0x00700000)
- #define CAN_BTR_SJW ((uint32_t)0x03000000)
- #define CAN_BTR_LBKM ((uint32_t)0x40000000)
- #define CAN_BTR_SILM ((uint32_t)0x80000000)
- #define CAN_TI0R_TXRQ ((uint32_t)0x00000001)
- #define CAN_TI0R_RTR ((uint32_t)0x00000002)
- #define CAN_TI0R_IDE ((uint32_t)0x00000004)
- #define CAN_TI0R_EXID ((uint32_t)0x001FFFF8)
- #define CAN_TI0R_STID ((uint32_t)0xFFE00000)
- #define CAN_TDT0R_DLC ((uint32_t)0x0000000F)
- #define CAN_TDT0R_TGT ((uint32_t)0x00000100)
- #define CAN_TDT0R_TIME ((uint32_t)0xFFFF0000)
- #define CAN_TDL0R_DATA0 ((uint32_t)0x000000FF)
- #define CAN_TDL0R_DATA1 ((uint32_t)0x0000FF00)
- #define CAN_TDL0R_DATA2 ((uint32_t)0x00FF0000)
- #define CAN_TDL0R_DATA3 ((uint32_t)0xFF000000)
- #define CAN_TDH0R_DATA4 ((uint32_t)0x000000FF)
- #define CAN_TDH0R_DATA5 ((uint32_t)0x0000FF00)
- #define CAN_TDH0R_DATA6 ((uint32_t)0x00FF0000)
- #define CAN_TDH0R_DATA7 ((uint32_t)0xFF000000)
- #define CAN_TI1R_TXRQ ((uint32_t)0x00000001)
- #define CAN_TI1R_RTR ((uint32_t)0x00000002)
- #define CAN_TI1R_IDE ((uint32_t)0x00000004)
- #define CAN_TI1R_EXID ((uint32_t)0x001FFFF8)
- #define CAN_TI1R_STID ((uint32_t)0xFFE00000)
- #define CAN_TDT1R_DLC ((uint32_t)0x0000000F)
- #define CAN_TDT1R_TGT ((uint32_t)0x00000100)
- #define CAN_TDT1R_TIME ((uint32_t)0xFFFF0000)
- #define CAN_TDL1R_DATA0 ((uint32_t)0x000000FF)
- #define CAN_TDL1R_DATA1 ((uint32_t)0x0000FF00)
- #define CAN_TDL1R_DATA2 ((uint32_t)0x00FF0000)
- #define CAN_TDL1R_DATA3 ((uint32_t)0xFF000000)
- #define CAN_TDH1R_DATA4 ((uint32_t)0x000000FF)
- #define CAN_TDH1R_DATA5 ((uint32_t)0x0000FF00)
- #define CAN_TDH1R_DATA6 ((uint32_t)0x00FF0000)
- #define CAN_TDH1R_DATA7 ((uint32_t)0xFF000000)
- #define CAN_TI2R_TXRQ ((uint32_t)0x00000001)
- #define CAN_TI2R_RTR ((uint32_t)0x00000002)
- #define CAN_TI2R_IDE ((uint32_t)0x00000004)
- #define CAN_TI2R_EXID ((uint32_t)0x001FFFF8)
- #define CAN_TI2R_STID ((uint32_t)0xFFE00000)
- #define CAN_TDT2R_DLC ((uint32_t)0x0000000F)
- #define CAN_TDT2R_TGT ((uint32_t)0x00000100)
- #define CAN_TDT2R_TIME ((uint32_t)0xFFFF0000)
- #define CAN_TDL2R_DATA0 ((uint32_t)0x000000FF)
- #define CAN_TDL2R_DATA1 ((uint32_t)0x0000FF00)
- #define CAN_TDL2R_DATA2 ((uint32_t)0x00FF0000)

- #define CAN_TDL2R_DATA3 ((uint32_t)0xFF000000)
- #define CAN_TDH2R_DATA4 ((uint32_t)0x000000FF)
- #define CAN_TDH2R_DATA5 ((uint32_t)0x0000FF00)
- #define CAN_TDH2R_DATA6 ((uint32_t)0x00FF0000)
- #define CAN_TDH2R_DATA7 ((uint32_t)0xFF000000)
- #define CAN_RI0R_RTR ((uint32_t)0x00000002)
- #define CAN_RI0R_IDE ((uint32_t)0x00000004)
- #define CAN_RI0R_EXID ((uint32_t)0x001FFFF8)
- #define CAN_RI0R_STID ((uint32_t)0xFFE00000)
- #define CAN_RDT0R_DLC ((uint32_t)0x0000000F)
- #define CAN_RDT0R_FMI ((uint32_t)0x0000FF00)
- #define CAN_RDT0R_TIME ((uint32_t)0xFFFF0000)
- #define CAN_RDL0R_DATA0 ((uint32_t)0x000000FF)
- #define CAN_RDL0R_DATA1 ((uint32_t)0x0000FF00)
- #define CAN_RDL0R_DATA2 ((uint32_t)0x00FF0000)
- #define CAN_RDL0R_DATA3 ((uint32_t)0xFF000000)
- #define CAN_RDH0R_DATA4 ((uint32_t)0x000000FF)
- #define CAN_RDH0R_DATA5 ((uint32_t)0x0000FF00)
- #define CAN_RDH0R_DATA6 ((uint32_t)0x00FF0000)
- #define CAN_RDH0R_DATA7 ((uint32_t)0xFF000000)
- #define CAN_RI1R_RTR ((uint32_t)0x00000002)
- #define CAN_RI1R_IDE ((uint32_t)0x00000004)
- #define CAN_RI1R_EXID ((uint32_t)0x001FFFF8)
- #define CAN_RI1R_STID ((uint32_t)0xFFE00000)
- #define CAN_RDT1R_DLC ((uint32_t)0x0000000F)
- #define CAN_RDT1R_FMI ((uint32_t)0x0000FF00)
- #define CAN_RDT1R_TIME ((uint32_t)0xFFFF0000)
- #define CAN_RDL1R_DATA0 ((uint32_t)0x000000FF)
- #define CAN_RDL1R_DATA1 ((uint32_t)0x0000FF00)
- #define CAN_RDL1R_DATA2 ((uint32_t)0x00FF0000)
- #define CAN_RDL1R_DATA3 ((uint32_t)0xFF000000)
- #define CAN_RDH1R_DATA4 ((uint32_t)0x000000FF)
- #define CAN_RDH1R_DATA5 ((uint32_t)0x0000FF00)
- #define CAN_RDH1R_DATA6 ((uint32_t)0x00FF0000)
- #define CAN_RDH1R_DATA7 ((uint32_t)0xFF000000)
- #define CAN_FMR_FINIT ((uint8_t)0x01)
- #define CAN_FM1R_FBM ((uint16_t)0x3FFF)
- #define CAN_FM1R_FBM0 ((uint16_t)0x0001)
- #define CAN_FM1R_FBM1 ((uint16_t)0x0002)
- #define CAN_FM1R_FBM2 ((uint16_t)0x0004)
- #define CAN_FM1R_FBM3 ((uint16_t)0x0008)
- #define CAN_FM1R_FBM4 ((uint16_t)0x0010)
- #define CAN_FM1R_FBM5 ((uint16_t)0x0020)
- #define CAN_FM1R_FBM6 ((uint16_t)0x0040)
- #define CAN_FM1R_FBM7 ((uint16_t)0x0080)
- #define CAN_FM1R_FBM8 ((uint16_t)0x0100)
- #define CAN_FM1R_FBM9 ((uint16_t)0x0200)
- #define CAN_FM1R_FBM10 ((uint16_t)0x0400)
- #define CAN_FM1R_FBM11 ((uint16_t)0x0800)
- #define CAN_FM1R_FBM12 ((uint16_t)0x1000)
- #define CAN_FM1R_FBM13 ((uint16_t)0x2000)
- #define CAN_FS1R_FSC ((uint16_t)0x3FFF)
- #define CAN_FS1R_FSC0 ((uint16_t)0x0001)
- #define CAN_FS1R_FSC1 ((uint16_t)0x0002)
- #define CAN_FS1R_FSC2 ((uint16_t)0x0004)

- #define CAN_FS1R_FSC3 ((uint16_t)0x0008)
- #define CAN_FS1R_FSC4 ((uint16_t)0x0010)
- #define CAN_FS1R_FSC5 ((uint16_t)0x0020)
- #define CAN_FS1R_FSC6 ((uint16_t)0x0040)
- #define CAN_FS1R_FSC7 ((uint16_t)0x0080)
- #define CAN_FS1R_FSC8 ((uint16_t)0x0100)
- #define CAN_FS1R_FSC9 ((uint16_t)0x0200)
- #define CAN_FS1R_FSC10 ((uint16_t)0x0400)
- #define CAN_FS1R_FSC11 ((uint16_t)0x0800)
- #define CAN_FS1R_FSC12 ((uint16_t)0x1000)
- #define CAN_FS1R_FSC13 ((uint16_t)0x2000)
- #define CAN_FFA1R_FFA ((uint16_t)0x3FFF)
- #define CAN_FFA1R_FFA0 ((uint16_t)0x0001)
- #define CAN_FFA1R_FFA1 ((uint16_t)0x0002)
- #define CAN_FFA1R_FFA2 ((uint16_t)0x0004)
- #define CAN_FFA1R_FFA3 ((uint16_t)0x0008)
- #define CAN_FFA1R_FFA4 ((uint16_t)0x0010)
- #define CAN_FFA1R_FFA5 ((uint16_t)0x0020)
- #define CAN_FFA1R_FFA6 ((uint16_t)0x0040)
- #define CAN_FFA1R_FFA7 ((uint16_t)0x0080)
- #define CAN_FFA1R_FFA8 ((uint16_t)0x0100)
- #define CAN_FFA1R_FFA9 ((uint16_t)0x0200)
- #define CAN_FFA1R_FFA10 ((uint16_t)0x0400)
- #define CAN_FFA1R_FFA11 ((uint16_t)0x0800)
- #define CAN_FFA1R_FFA12 ((uint16_t)0x1000)
- #define CAN_FFA1R_FFA13 ((uint16_t)0x2000)
- #define CAN_FA1R_FACT ((uint16_t)0x3FFF)
- #define CAN_FA1R_FACT0 ((uint16_t)0x0001)
- #define CAN_FA1R_FACT1 ((uint16_t)0x0002)
- #define CAN_FA1R_FACT2 ((uint16_t)0x0004)
- #define CAN_FA1R_FACT3 ((uint16_t)0x0008)
- #define CAN_FA1R_FACT4 ((uint16_t)0x0010)
- #define CAN_FA1R_FACT5 ((uint16_t)0x0020)
- #define CAN_FA1R_FACT6 ((uint16_t)0x0040)
- #define CAN_FA1R_FACT7 ((uint16_t)0x0080)
- #define CAN_FA1R_FACT8 ((uint16_t)0x0100)
- #define CAN_FA1R_FACT9 ((uint16_t)0x0200)
- #define CAN_FA1R_FACT10 ((uint16_t)0x0400)
- #define CAN_FA1R_FACT11 ((uint16_t)0x0800)
- #define CAN_FA1R_FACT12 ((uint16_t)0x1000)
- #define CAN_FA1R_FACT13 ((uint16_t)0x2000)
- #define CAN_F0R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F0R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F0R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F0R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F0R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F0R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F0R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F0R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F0R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F0R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F0R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F0R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F0R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F0R1_FB13 ((uint32_t)0x00002000)

- #define CAN_F0R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F0R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F0R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F0R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F0R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F0R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F0R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F0R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F0R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F0R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F0R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F0R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F0R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F0R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F0R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F0R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F0R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F0R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F1R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F1R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F1R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F1R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F1R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F1R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F1R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F1R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F1R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F1R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F1R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F1R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F1R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F1R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F1R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F1R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F1R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F1R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F1R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F1R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F1R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F1R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F1R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F1R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F1R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F1R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F1R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F1R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F1R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F1R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F1R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F1R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F2R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F2R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F2R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F2R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F2R1_FB4 ((uint32_t)0x00000010)

- #define CAN_F2R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F2R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F2R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F2R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F2R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F2R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F2R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F2R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F2R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F2R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F2R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F2R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F2R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F2R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F2R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F2R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F2R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F2R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F2R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F2R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F2R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F2R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F2R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F2R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F2R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F2R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F2R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F3R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F3R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F3R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F3R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F3R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F3R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F3R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F3R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F3R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F3R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F3R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F3R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F3R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F3R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F3R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F3R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F3R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F3R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F3R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F3R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F3R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F3R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F3R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F3R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F3R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F3R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F3R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F3R1_FB27 ((uint32_t)0x08000000)

- #define CAN_F3R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F3R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F3R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F3R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F4R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F4R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F4R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F4R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F4R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F4R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F4R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F4R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F4R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F4R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F4R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F4R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F4R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F4R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F4R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F4R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F4R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F4R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F4R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F4R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F4R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F4R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F4R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F4R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F4R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F4R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F4R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F4R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F4R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F4R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F4R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F4R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F5R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F5R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F5R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F5R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F5R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F5R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F5R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F5R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F5R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F5R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F5R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F5R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F5R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F5R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F5R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F5R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F5R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F5R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F5R1_FB18 ((uint32_t)0x00040000)

- #define CAN_F5R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F5R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F5R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F5R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F5R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F5R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F5R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F5R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F5R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F5R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F5R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F5R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F5R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F6R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F6R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F6R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F6R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F6R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F6R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F6R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F6R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F6R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F6R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F6R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F6R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F6R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F6R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F6R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F6R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F6R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F6R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F6R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F6R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F6R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F6R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F6R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F6R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F6R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F6R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F6R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F6R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F6R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F6R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F6R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F6R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F7R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F7R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F7R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F7R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F7R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F7R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F7R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F7R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F7R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F7R1_FB9 ((uint32_t)0x00000200)

- #define CAN_F7R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F7R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F7R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F7R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F7R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F7R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F7R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F7R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F7R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F7R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F7R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F7R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F7R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F7R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F7R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F7R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F7R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F7R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F7R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F7R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F7R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F7R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F8R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F8R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F8R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F8R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F8R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F8R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F8R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F8R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F8R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F8R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F8R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F8R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F8R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F8R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F8R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F8R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F8R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F8R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F8R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F8R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F8R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F8R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F8R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F8R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F8R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F8R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F8R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F8R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F8R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F8R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F8R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F8R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F9R1_FB0 ((uint32_t)0x00000001)

- #define CAN_F9R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F9R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F9R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F9R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F9R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F9R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F9R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F9R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F9R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F9R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F9R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F9R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F9R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F9R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F9R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F9R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F9R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F9R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F9R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F9R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F9R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F9R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F9R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F9R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F9R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F9R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F9R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F9R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F9R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F9R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F9R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F10R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F10R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F10R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F10R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F10R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F10R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F10R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F10R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F10R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F10R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F10R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F10R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F10R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F10R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F10R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F10R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F10R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F10R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F10R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F10R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F10R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F10R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F10R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F10R1_FB23 ((uint32_t)0x00800000)

- #define CAN_F10R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F10R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F10R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F10R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F10R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F10R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F10R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F10R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F11R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F11R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F11R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F11R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F11R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F11R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F11R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F11R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F11R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F11R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F11R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F11R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F11R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F11R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F11R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F11R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F11R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F11R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F11R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F11R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F11R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F11R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F11R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F11R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F11R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F11R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F11R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F11R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F11R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F11R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F11R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F11R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F12R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F12R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F12R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F12R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F12R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F12R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F12R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F12R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F12R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F12R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F12R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F12R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F12R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F12R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F12R1_FB14 ((uint32_t)0x00004000)

- #define CAN_F12R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F12R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F12R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F12R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F12R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F12R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F12R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F12R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F12R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F12R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F12R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F12R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F12R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F12R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F12R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F12R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F12R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F13R1_FB0 ((uint32_t)0x00000001)
- #define CAN_F13R1_FB1 ((uint32_t)0x00000002)
- #define CAN_F13R1_FB2 ((uint32_t)0x00000004)
- #define CAN_F13R1_FB3 ((uint32_t)0x00000008)
- #define CAN_F13R1_FB4 ((uint32_t)0x00000010)
- #define CAN_F13R1_FB5 ((uint32_t)0x00000020)
- #define CAN_F13R1_FB6 ((uint32_t)0x00000040)
- #define CAN_F13R1_FB7 ((uint32_t)0x00000080)
- #define CAN_F13R1_FB8 ((uint32_t)0x00000100)
- #define CAN_F13R1_FB9 ((uint32_t)0x00000200)
- #define CAN_F13R1_FB10 ((uint32_t)0x00000400)
- #define CAN_F13R1_FB11 ((uint32_t)0x00000800)
- #define CAN_F13R1_FB12 ((uint32_t)0x00001000)
- #define CAN_F13R1_FB13 ((uint32_t)0x00002000)
- #define CAN_F13R1_FB14 ((uint32_t)0x00004000)
- #define CAN_F13R1_FB15 ((uint32_t)0x00008000)
- #define CAN_F13R1_FB16 ((uint32_t)0x00010000)
- #define CAN_F13R1_FB17 ((uint32_t)0x00020000)
- #define CAN_F13R1_FB18 ((uint32_t)0x00040000)
- #define CAN_F13R1_FB19 ((uint32_t)0x00080000)
- #define CAN_F13R1_FB20 ((uint32_t)0x00100000)
- #define CAN_F13R1_FB21 ((uint32_t)0x00200000)
- #define CAN_F13R1_FB22 ((uint32_t)0x00400000)
- #define CAN_F13R1_FB23 ((uint32_t)0x00800000)
- #define CAN_F13R1_FB24 ((uint32_t)0x01000000)
- #define CAN_F13R1_FB25 ((uint32_t)0x02000000)
- #define CAN_F13R1_FB26 ((uint32_t)0x04000000)
- #define CAN_F13R1_FB27 ((uint32_t)0x08000000)
- #define CAN_F13R1_FB28 ((uint32_t)0x10000000)
- #define CAN_F13R1_FB29 ((uint32_t)0x20000000)
- #define CAN_F13R1_FB30 ((uint32_t)0x40000000)
- #define CAN_F13R1_FB31 ((uint32_t)0x80000000)
- #define CAN_F0R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F0R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F0R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F0R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F0R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F0R2_FB5 ((uint32_t)0x00000020)

- #define CAN_F0R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F0R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F0R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F0R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F0R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F0R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F0R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F0R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F0R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F0R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F0R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F0R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F0R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F0R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F0R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F0R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F0R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F0R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F0R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F0R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F0R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F0R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F0R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F0R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F0R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F0R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F1R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F1R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F1R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F1R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F1R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F1R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F1R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F1R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F1R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F1R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F1R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F1R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F1R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F1R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F1R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F1R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F1R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F1R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F1R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F1R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F1R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F1R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F1R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F1R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F1R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F1R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F1R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F1R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F1R2_FB28 ((uint32_t)0x10000000)

- #define CAN_F1R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F1R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F1R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F2R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F2R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F2R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F2R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F2R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F2R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F2R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F2R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F2R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F2R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F2R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F2R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F2R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F2R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F2R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F2R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F2R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F2R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F2R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F2R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F2R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F2R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F2R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F2R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F2R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F2R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F2R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F2R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F2R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F2R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F2R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F2R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F3R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F3R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F3R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F3R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F3R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F3R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F3R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F3R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F3R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F3R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F3R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F3R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F3R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F3R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F3R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F3R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F3R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F3R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F3R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F3R2_FB19 ((uint32_t)0x00080000)

- #define CAN_F3R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F3R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F3R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F3R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F3R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F3R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F3R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F3R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F3R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F3R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F3R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F3R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F4R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F4R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F4R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F4R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F4R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F4R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F4R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F4R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F4R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F4R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F4R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F4R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F4R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F4R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F4R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F4R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F4R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F4R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F4R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F4R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F4R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F4R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F4R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F4R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F4R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F4R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F4R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F4R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F4R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F4R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F4R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F4R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F5R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F5R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F5R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F5R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F5R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F5R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F5R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F5R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F5R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F5R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F5R2_FB10 ((uint32_t)0x00000400)

- #define CAN_F5R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F5R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F5R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F5R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F5R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F5R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F5R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F5R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F5R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F5R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F5R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F5R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F5R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F5R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F5R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F5R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F5R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F5R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F5R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F5R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F5R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F6R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F6R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F6R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F6R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F6R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F6R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F6R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F6R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F6R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F6R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F6R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F6R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F6R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F6R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F6R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F6R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F6R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F6R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F6R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F6R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F6R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F6R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F6R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F6R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F6R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F6R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F6R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F6R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F6R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F6R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F6R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F6R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F7R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F7R2_FB1 ((uint32_t)0x00000002)

- #define CAN_F7R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F7R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F7R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F7R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F7R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F7R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F7R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F7R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F7R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F7R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F7R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F7R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F7R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F7R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F7R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F7R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F7R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F7R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F7R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F7R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F7R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F7R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F7R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F7R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F7R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F7R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F7R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F7R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F7R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F7R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F8R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F8R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F8R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F8R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F8R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F8R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F8R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F8R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F8R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F8R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F8R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F8R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F8R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F8R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F8R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F8R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F8R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F8R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F8R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F8R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F8R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F8R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F8R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F8R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F8R2_FB24 ((uint32_t)0x01000000)

- #define CAN_F8R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F8R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F8R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F8R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F8R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F8R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F8R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F9R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F9R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F9R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F9R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F9R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F9R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F9R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F9R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F9R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F9R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F9R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F9R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F9R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F9R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F9R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F9R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F9R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F9R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F9R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F9R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F9R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F9R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F9R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F9R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F9R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F9R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F9R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F9R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F9R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F9R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F9R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F9R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F10R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F10R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F10R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F10R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F10R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F10R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F10R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F10R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F10R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F10R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F10R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F10R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F10R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F10R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F10R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F10R2_FB15 ((uint32_t)0x00008000)

- #define CAN_F10R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F10R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F10R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F10R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F10R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F10R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F10R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F10R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F10R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F10R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F10R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F10R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F10R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F10R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F10R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F10R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F11R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F11R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F11R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F11R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F11R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F11R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F11R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F11R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F11R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F11R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F11R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F11R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F11R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F11R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F11R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F11R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F11R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F11R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F11R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F11R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F11R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F11R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F11R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F11R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F11R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F11R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F11R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F11R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F11R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F11R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F11R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F11R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F12R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F12R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F12R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F12R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F12R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F12R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F12R2_FB6 ((uint32_t)0x00000040)

- #define CAN_F12R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F12R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F12R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F12R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F12R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F12R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F12R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F12R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F12R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F12R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F12R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F12R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F12R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F12R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F12R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F12R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F12R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F12R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F12R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F12R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F12R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F12R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F12R2_FB29 ((uint32_t)0x20000000)
- #define CAN_F12R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F12R2_FB31 ((uint32_t)0x80000000)
- #define CAN_F13R2_FB0 ((uint32_t)0x00000001)
- #define CAN_F13R2_FB1 ((uint32_t)0x00000002)
- #define CAN_F13R2_FB2 ((uint32_t)0x00000004)
- #define CAN_F13R2_FB3 ((uint32_t)0x00000008)
- #define CAN_F13R2_FB4 ((uint32_t)0x00000010)
- #define CAN_F13R2_FB5 ((uint32_t)0x00000020)
- #define CAN_F13R2_FB6 ((uint32_t)0x00000040)
- #define CAN_F13R2_FB7 ((uint32_t)0x00000080)
- #define CAN_F13R2_FB8 ((uint32_t)0x00000100)
- #define CAN_F13R2_FB9 ((uint32_t)0x00000200)
- #define CAN_F13R2_FB10 ((uint32_t)0x00000400)
- #define CAN_F13R2_FB11 ((uint32_t)0x00000800)
- #define CAN_F13R2_FB12 ((uint32_t)0x00001000)
- #define CAN_F13R2_FB13 ((uint32_t)0x00002000)
- #define CAN_F13R2_FB14 ((uint32_t)0x00004000)
- #define CAN_F13R2_FB15 ((uint32_t)0x00008000)
- #define CAN_F13R2_FB16 ((uint32_t)0x00010000)
- #define CAN_F13R2_FB17 ((uint32_t)0x00020000)
- #define CAN_F13R2_FB18 ((uint32_t)0x00040000)
- #define CAN_F13R2_FB19 ((uint32_t)0x00080000)
- #define CAN_F13R2_FB20 ((uint32_t)0x00100000)
- #define CAN_F13R2_FB21 ((uint32_t)0x00200000)
- #define CAN_F13R2_FB22 ((uint32_t)0x00400000)
- #define CAN_F13R2_FB23 ((uint32_t)0x00800000)
- #define CAN_F13R2_FB24 ((uint32_t)0x01000000)
- #define CAN_F13R2_FB25 ((uint32_t)0x02000000)
- #define CAN_F13R2_FB26 ((uint32_t)0x04000000)
- #define CAN_F13R2_FB27 ((uint32_t)0x08000000)
- #define CAN_F13R2_FB28 ((uint32_t)0x10000000)
- #define CAN_F13R2_FB29 ((uint32_t)0x20000000)

- #define CAN_F13R2_FB30 ((uint32_t)0x40000000)
- #define CAN_F13R2_FB31 ((uint32_t)0x80000000)
- #define CRC_DR_DR ((uint32_t)0xFFFFFFFF)
- #define CRC_IDR_IDR ((uint8_t)0xFF)
- #define CRC_CR_RESET ((uint8_t)0x01)
- #define **CRYP_CR_ALGODIR** ((uint32_t)0x00000004)
- #define **CRYP_CR_ALGOMODE** ((uint32_t)0x00000038)
- #define **CRYP_CR_ALGOMODE_0** ((uint32_t)0x00000008)
- #define **CRYP_CR_ALGOMODE_1** ((uint32_t)0x00000010)
- #define **CRYP_CR_ALGOMODE_2** ((uint32_t)0x00000020)
- #define **CRYP_CR_ALGOMODE_TDES_ECB** ((uint32_t)0x00000000)
- #define **CRYP_CR_ALGOMODE_TDES_CBC** ((uint32_t)0x00000008)
- #define **CRYP_CR_ALGOMODE_DES_ECB** ((uint32_t)0x00000010)
- #define **CRYP_CR_ALGOMODE_DES_CBC** ((uint32_t)0x00000018)
- #define **CRYP_CR_ALGOMODE_AES_ECB** ((uint32_t)0x00000020)
- #define **CRYP_CR_ALGOMODE_AES_CBC** ((uint32_t)0x00000028)
- #define **CRYP_CR_ALGOMODE_AES_CTR** ((uint32_t)0x00000030)
- #define **CRYP_CR_ALGOMODE_AES_KEY** ((uint32_t)0x00000038)
- #define **CRYP_CR_DATATYPE** ((uint32_t)0x000000C0)
- #define **CRYP_CR_DATATYPE_0** ((uint32_t)0x00000040)
- #define **CRYP_CR_DATATYPE_1** ((uint32_t)0x00000080)
- #define **CRYP_CR_KEYSIZE** ((uint32_t)0x00000300)
- #define **CRYP_CR_KEYSIZE_0** ((uint32_t)0x00000100)
- #define **CRYP_CR_KEYSIZE_1** ((uint32_t)0x00000200)
- #define **CRYP_CR_FFLUSH** ((uint32_t)0x00004000)
- #define **CRYP_CR_CRYPEN** ((uint32_t)0x00008000)
- #define **CRYP_SR_IFEM** ((uint32_t)0x00000001)
- #define **CRYP_SR_IFNF** ((uint32_t)0x00000002)
- #define **CRYP_SR_OFNE** ((uint32_t)0x00000004)
- #define **CRYP_SR_OFFU** ((uint32_t)0x00000008)
- #define **CRYP_SR_BUSY** ((uint32_t)0x00000010)
- #define **CRYP_DMACR_DIEN** ((uint32_t)0x00000001)
- #define **CRYP_DMACR_DOEN** ((uint32_t)0x00000002)
- #define **CRYP_IMSCR_INIM** ((uint32_t)0x00000001)
- #define **CRYP_IMSCR_OUTIM** ((uint32_t)0x00000002)
- #define **CRYP_RISR_OUTRIS** ((uint32_t)0x00000001)
- #define **CRYP_RISR_INRIS** ((uint32_t)0x00000002)
- #define **CRYP_MISR_INMIS** ((uint32_t)0x00000001)
- #define **CRYP_MISR_OUTMIS** ((uint32_t)0x00000002)
- #define DAC_CR_EN1 ((uint32_t)0x00000001)
- #define DAC_CR_BOFF1 ((uint32_t)0x00000002)
- #define DAC_CR_TEN1 ((uint32_t)0x00000004)
- #define DAC_CR_TSEL1 ((uint32_t)0x00000038)
- #define DAC_CR_TSEL1_0 ((uint32_t)0x00000008)
- #define DAC_CR_TSEL1_1 ((uint32_t)0x00000010)
- #define DAC_CR_TSEL1_2 ((uint32_t)0x00000020)
- #define DAC_CR_WAVE1 ((uint32_t)0x000000C0)
- #define DAC_CR_WAVE1_0 ((uint32_t)0x00000040)
- #define DAC_CR_WAVE1_1 ((uint32_t)0x00000080)
- #define DAC_CR_MAMP1 ((uint32_t)0x00000F00)
- #define DAC_CR_MAMP1_0 ((uint32_t)0x00000100)
- #define DAC_CR_MAMP1_1 ((uint32_t)0x00000200)
- #define DAC_CR_MAMP1_2 ((uint32_t)0x00000400)
- #define DAC_CR_MAMP1_3 ((uint32_t)0x00000800)
- #define DAC_CR_DMAEN1 ((uint32_t)0x00001000)

- #define DAC_CR_EN2 ((uint32_t)0x00010000)
- #define DAC_CR_BOFF2 ((uint32_t)0x00020000)
- #define DAC_CR_TEN2 ((uint32_t)0x00040000)
- #define DAC_CR_TSEL2 ((uint32_t)0x00380000)
- #define DAC_CR_TSEL2_0 ((uint32_t)0x00080000)
- #define DAC_CR_TSEL2_1 ((uint32_t)0x00100000)
- #define DAC_CR_TSEL2_2 ((uint32_t)0x00200000)
- #define DAC_CR_WAVE2 ((uint32_t)0x00C00000)
- #define DAC_CR_WAVE2_0 ((uint32_t)0x00400000)
- #define DAC_CR_WAVE2_1 ((uint32_t)0x00800000)
- #define DAC_CR_MAMP2 ((uint32_t)0x0F000000)
- #define DAC_CR_MAMP2_0 ((uint32_t)0x01000000)
- #define DAC_CR_MAMP2_1 ((uint32_t)0x02000000)
- #define DAC_CR_MAMP2_2 ((uint32_t)0x04000000)
- #define DAC_CR_MAMP2_3 ((uint32_t)0x08000000)
- #define DAC_CR_DMAEN2 ((uint32_t)0x10000000)
- #define DAC_SWTRIGR_SWTRIG1 ((uint8_t)0x01)
- #define DAC_SWTRIGR_SWTRIG2 ((uint8_t)0x02)
- #define DAC_DHR12R1_DACC1DHR ((uint16_t)0x0FFF)
- #define DAC_DHR12L1_DACC1DHR ((uint16_t)0xFFF0)
- #define DAC_DHR8R1_DACC1DHR ((uint8_t)0xFF)
- #define DAC_DHR12R2_DACC2DHR ((uint16_t)0x0FFF)
- #define DAC_DHR12L2_DACC2DHR ((uint16_t)0xFFF0)
- #define DAC_DHR8R2_DACC2DHR ((uint8_t)0xFF)
- #define DAC_DHR12RD_DACC1DHR ((uint32_t)0x00000FFF)
- #define DAC_DHR12RD_DACC2DHR ((uint32_t)0x0FFF0000)
- #define DAC_DHR12LD_DACC1DHR ((uint32_t)0x0000FFF0)
- #define DAC_DHR12LD_DACC2DHR ((uint32_t)0xFFF00000)
- #define DAC_DHR8RD_DACC1DHR ((uint16_t)0x00FF)
- #define DAC_DHR8RD_DACC2DHR ((uint16_t)0xFF00)
- #define DAC_DOR1_DACC1DOR ((uint16_t)0x0FFF)
- #define DAC_DOR2_DACC2DOR ((uint16_t)0x0FFF)
- #define DAC_SR_DMAUDR1 ((uint32_t)0x00002000)
- #define DAC_SR_DMAUDR2 ((uint32_t)0x20000000)
- #define **DCMI_CR_CAPTURE** ((uint32_t)0x00000001)
- #define **DCMI_CR_CM** ((uint32_t)0x00000002)
- #define **DCMI_CR_CROP** ((uint32_t)0x00000004)
- #define **DCMI_CR_JPEG** ((uint32_t)0x00000008)
- #define **DCMI_CR_ESS** ((uint32_t)0x00000010)
- #define **DCMI_CR_PCKPOL** ((uint32_t)0x00000020)
- #define **DCMI_CR_HSPOL** ((uint32_t)0x00000040)
- #define **DCMI_CR_VSPOL** ((uint32_t)0x00000080)
- #define **DCMI_CR_FCRC_0** ((uint32_t)0x00000100)
- #define **DCMI_CR_FCRC_1** ((uint32_t)0x00000200)
- #define **DCMI_CR_EDM_0** ((uint32_t)0x00000400)
- #define **DCMI_CR_EDM_1** ((uint32_t)0x00000800)
- #define **DCMI_CR_CRE** ((uint32_t)0x00001000)
- #define **DCMI_CR_ENABLE** ((uint32_t)0x00004000)
- #define **DCMI_SR_HSYNC** ((uint32_t)0x00000001)
- #define **DCMI_SR_VSYNC** ((uint32_t)0x00000002)
- #define **DCMI_SR_FNE** ((uint32_t)0x00000004)
- #define **DCMI_RISR_FRAME_RIS** ((uint32_t)0x00000001)
- #define **DCMI_RISR_OVF_RIS** ((uint32_t)0x00000002)
- #define **DCMI_RISR_ERR_RIS** ((uint32_t)0x00000004)
- #define **DCMI_RISR_VSYNC_RIS** ((uint32_t)0x00000008)

- #define **DCMI_RISR_LINE_RIS** ((uint32_t)0x00000010)
- #define **DCMI_IER_FRAME_IE** ((uint32_t)0x00000001)
- #define **DCMI_IER_OVF_IE** ((uint32_t)0x00000002)
- #define **DCMI_IER_ERR_IE** ((uint32_t)0x00000004)
- #define **DCMI_IER_VSYNC_IE** ((uint32_t)0x00000008)
- #define **DCMI_IER_LINE_IE** ((uint32_t)0x00000010)
- #define **DCMI_MISR_FRAME_MIS** ((uint32_t)0x00000001)
- #define **DCMI_MISR_OVF_MIS** ((uint32_t)0x00000002)
- #define **DCMI_MISR_ERR_MIS** ((uint32_t)0x00000004)
- #define **DCMI_MISR_VSYNC_MIS** ((uint32_t)0x00000008)
- #define **DCMI_MISR_LINE_MIS** ((uint32_t)0x00000010)
- #define **DCMI_ICR_FRAME_ISC** ((uint32_t)0x00000001)
- #define **DCMI_ICR_OVF_ISC** ((uint32_t)0x00000002)
- #define **DCMI_ICR_ERR_ISC** ((uint32_t)0x00000004)
- #define **DCMI_ICR_VSYNC_ISC** ((uint32_t)0x00000008)
- #define **DCMI_ICR_LINE_ISC** ((uint32_t)0x00000010)
- #define **DMA_SxCR_CHSEL** ((uint32_t)0x0E000000)
- #define **DMA_SxCR_CHSEL_0** ((uint32_t)0x02000000)
- #define **DMA_SxCR_CHSEL_1** ((uint32_t)0x04000000)
- #define **DMA_SxCR_CHSEL_2** ((uint32_t)0x08000000)
- #define **DMA_SxCR_MBURST** ((uint32_t)0x01800000)
- #define **DMA_SxCR_MBURST_0** ((uint32_t)0x00800000)
- #define **DMA_SxCR_MBURST_1** ((uint32_t)0x01000000)
- #define **DMA_SxCR_PBURST** ((uint32_t)0x00600000)
- #define **DMA_SxCR_PBURST_0** ((uint32_t)0x00200000)
- #define **DMA_SxCR_PBURST_1** ((uint32_t)0x00400000)
- #define **DMA_SxCR_ACK** ((uint32_t)0x00100000)
- #define **DMA_SxCR_CT** ((uint32_t)0x00080000)
- #define **DMA_SxCR_DBM** ((uint32_t)0x00040000)
- #define **DMA_SxCR_PL** ((uint32_t)0x00030000)
- #define **DMA_SxCR_PL_0** ((uint32_t)0x00010000)
- #define **DMA_SxCR_PL_1** ((uint32_t)0x00020000)
- #define **DMA_SxCR_PINCOS** ((uint32_t)0x00008000)
- #define **DMA_SxCR_MSIZE** ((uint32_t)0x00006000)
- #define **DMA_SxCR_MSIZE_0** ((uint32_t)0x00002000)
- #define **DMA_SxCR_MSIZE_1** ((uint32_t)0x00004000)
- #define **DMA_SxCR_PSIZE** ((uint32_t)0x00001800)
- #define **DMA_SxCR_PSIZE_0** ((uint32_t)0x00000800)
- #define **DMA_SxCR_PSIZE_1** ((uint32_t)0x00001000)
- #define **DMA_SxCR_MINC** ((uint32_t)0x00000400)
- #define **DMA_SxCR_PINC** ((uint32_t)0x00000200)
- #define **DMA_SxCR_CIRC** ((uint32_t)0x00000100)
- #define **DMA_SxCR_DIR** ((uint32_t)0x000000C0)
- #define **DMA_SxCR_DIR_0** ((uint32_t)0x00000040)
- #define **DMA_SxCR_DIR_1** ((uint32_t)0x00000080)
- #define **DMA_SxCR_PFCTRL** ((uint32_t)0x00000020)
- #define **DMA_SxCR_TCIE** ((uint32_t)0x00000010)
- #define **DMA_SxCR_HTIE** ((uint32_t)0x00000008)
- #define **DMA_SxCR_TEIE** ((uint32_t)0x00000004)
- #define **DMA_SxCR_DMEIE** ((uint32_t)0x00000002)
- #define **DMA_SxCR_EN** ((uint32_t)0x00000001)
- #define **DMA_SxNDT** ((uint32_t)0x0000FFFF)
- #define **DMA_SxNDT_0** ((uint32_t)0x00000001)
- #define **DMA_SxNDT_1** ((uint32_t)0x00000002)
- #define **DMA_SxNDT_2** ((uint32_t)0x00000004)

- #define **DMA_SxNDT_3** ((uint32_t)0x00000008)
- #define **DMA_SxNDT_4** ((uint32_t)0x00000010)
- #define **DMA_SxNDT_5** ((uint32_t)0x00000020)
- #define **DMA_SxNDT_6** ((uint32_t)0x00000040)
- #define **DMA_SxNDT_7** ((uint32_t)0x00000080)
- #define **DMA_SxNDT_8** ((uint32_t)0x00000100)
- #define **DMA_SxNDT_9** ((uint32_t)0x00000200)
- #define **DMA_SxNDT_10** ((uint32_t)0x00000400)
- #define **DMA_SxNDT_11** ((uint32_t)0x00000800)
- #define **DMA_SxNDT_12** ((uint32_t)0x00001000)
- #define **DMA_SxNDT_13** ((uint32_t)0x00002000)
- #define **DMA_SxNDT_14** ((uint32_t)0x00004000)
- #define **DMA_SxNDT_15** ((uint32_t)0x00008000)
- #define **DMA_SxFCR_FEIE** ((uint32_t)0x00000080)
- #define **DMA_SxFCR_FS** ((uint32_t)0x00000038)
- #define **DMA_SxFCR_FS_0** ((uint32_t)0x00000008)
- #define **DMA_SxFCR_FS_1** ((uint32_t)0x00000010)
- #define **DMA_SxFCR_FS_2** ((uint32_t)0x00000020)
- #define **DMA_SxFCR_DMDIS** ((uint32_t)0x00000004)
- #define **DMA_SxFCR_FTH** ((uint32_t)0x00000003)
- #define **DMA_SxFCR_FTH_0** ((uint32_t)0x00000001)
- #define **DMA_SxFCR_FTH_1** ((uint32_t)0x00000002)
- #define **DMA_LISR_TCIF3** ((uint32_t)0x08000000)
- #define **DMA_LISR_HTIF3** ((uint32_t)0x04000000)
- #define **DMA_LISR_TEIF3** ((uint32_t)0x02000000)
- #define **DMA_LISR_DMEIF3** ((uint32_t)0x01000000)
- #define **DMA_LISR_FEIF3** ((uint32_t)0x00400000)
- #define **DMA_LISR_TCIF2** ((uint32_t)0x00200000)
- #define **DMA_LISR_HTIF2** ((uint32_t)0x00100000)
- #define **DMA_LISR_TEIF2** ((uint32_t)0x00080000)
- #define **DMA_LISR_DMEIF2** ((uint32_t)0x00040000)
- #define **DMA_LISR_FEIF2** ((uint32_t)0x00010000)
- #define **DMA_LISR_TCIF1** ((uint32_t)0x00000800)
- #define **DMA_LISR_HTIF1** ((uint32_t)0x00000400)
- #define **DMA_LISR_TEIF1** ((uint32_t)0x00000200)
- #define **DMA_LISR_DMEIF1** ((uint32_t)0x00000100)
- #define **DMA_LISR_FEIF1** ((uint32_t)0x00000040)
- #define **DMA_LISR_TCIF0** ((uint32_t)0x00000020)
- #define **DMA_LISR_HTIF0** ((uint32_t)0x00000010)
- #define **DMA_LISR_TEIF0** ((uint32_t)0x00000008)
- #define **DMA_LISR_DMEIF0** ((uint32_t)0x00000004)
- #define **DMA_LISR_FEIF0** ((uint32_t)0x00000001)
- #define **DMA_HISR_TCIF7** ((uint32_t)0x08000000)
- #define **DMA_HISR_HTIF7** ((uint32_t)0x04000000)
- #define **DMA_HISR_TEIF7** ((uint32_t)0x02000000)
- #define **DMA_HISR_DMEIF7** ((uint32_t)0x01000000)
- #define **DMA_HISR_FEIF7** ((uint32_t)0x00400000)
- #define **DMA_HISR_TCIF6** ((uint32_t)0x00200000)
- #define **DMA_HISR_HTIF6** ((uint32_t)0x00100000)
- #define **DMA_HISR_TEIF6** ((uint32_t)0x00080000)
- #define **DMA_HISR_DMEIF6** ((uint32_t)0x00040000)
- #define **DMA_HISR_FEIF6** ((uint32_t)0x00010000)
- #define **DMA_HISR_TCIF5** ((uint32_t)0x00000800)
- #define **DMA_HISR_HTIF5** ((uint32_t)0x00000400)
- #define **DMA_HISR_TEIF5** ((uint32_t)0x00000200)

- #define **DMA_HISR_DMEIF5** ((uint32_t)0x00000100)
- #define **DMA_HISR_FEIF5** ((uint32_t)0x00000040)
- #define **DMA_HISR_TCIF4** ((uint32_t)0x00000020)
- #define **DMA_HISR_HTIF4** ((uint32_t)0x00000010)
- #define **DMA_HISR_TEIF4** ((uint32_t)0x00000008)
- #define **DMA_HISR_DMEIF4** ((uint32_t)0x00000004)
- #define **DMA_HISR_FEIF4** ((uint32_t)0x00000001)
- #define **DMA_LIFCR_CTCIF3** ((uint32_t)0x08000000)
- #define **DMA_LIFCR_CHTIF3** ((uint32_t)0x04000000)
- #define **DMA_LIFCR_CTEIF3** ((uint32_t)0x02000000)
- #define **DMA_LIFCR_CDMEIF3** ((uint32_t)0x01000000)
- #define **DMA_LIFCR_CFEIF3** ((uint32_t)0x00400000)
- #define **DMA_LIFCR_CTCIF2** ((uint32_t)0x00200000)
- #define **DMA_LIFCR_CHTIF2** ((uint32_t)0x00100000)
- #define **DMA_LIFCR_CTEIF2** ((uint32_t)0x00080000)
- #define **DMA_LIFCR_CDMEIF2** ((uint32_t)0x00040000)
- #define **DMA_LIFCR_CFEIF2** ((uint32_t)0x00010000)
- #define **DMA_LIFCR_CTCIF1** ((uint32_t)0x00000800)
- #define **DMA_LIFCR_CHTIF1** ((uint32_t)0x00000400)
- #define **DMA_LIFCR_CTEIF1** ((uint32_t)0x00000200)
- #define **DMA_LIFCR_CDMEIF1** ((uint32_t)0x00000100)
- #define **DMA_LIFCR_CFEIF1** ((uint32_t)0x00000040)
- #define **DMA_LIFCR_CTCIF0** ((uint32_t)0x00000020)
- #define **DMA_LIFCR_CHTIF0** ((uint32_t)0x00000010)
- #define **DMA_LIFCR_CTEIF0** ((uint32_t)0x00000008)
- #define **DMA_LIFCR_CDMEIF0** ((uint32_t)0x00000004)
- #define **DMA_LIFCR_CFEIF0** ((uint32_t)0x00000001)
- #define **DMA_HIFCR_CTCIF7** ((uint32_t)0x08000000)
- #define **DMA_HIFCR_CHTIF7** ((uint32_t)0x04000000)
- #define **DMA_HIFCR_CTEIF7** ((uint32_t)0x02000000)
- #define **DMA_HIFCR_CDMEIF7** ((uint32_t)0x01000000)
- #define **DMA_HIFCR_CFEIF7** ((uint32_t)0x00400000)
- #define **DMA_HIFCR_CTCIF6** ((uint32_t)0x00200000)
- #define **DMA_HIFCR_CHTIF6** ((uint32_t)0x00100000)
- #define **DMA_HIFCR_CTEIF6** ((uint32_t)0x00080000)
- #define **DMA_HIFCR_CDMEIF6** ((uint32_t)0x00040000)
- #define **DMA_HIFCR_CFEIF6** ((uint32_t)0x00010000)
- #define **DMA_HIFCR_CTCIF5** ((uint32_t)0x00000800)
- #define **DMA_HIFCR_CHTIF5** ((uint32_t)0x00000400)
- #define **DMA_HIFCR_CTEIF5** ((uint32_t)0x00000200)
- #define **DMA_HIFCR_CDMEIF5** ((uint32_t)0x00000100)
- #define **DMA_HIFCR_CFEIF5** ((uint32_t)0x00000040)
- #define **DMA_HIFCR_CTCIF4** ((uint32_t)0x00000020)
- #define **DMA_HIFCR_CHTIF4** ((uint32_t)0x00000010)
- #define **DMA_HIFCR_CTEIF4** ((uint32_t)0x00000008)
- #define **DMA_HIFCR_CDMEIF4** ((uint32_t)0x00000004)
- #define **DMA_HIFCR_CFEIF4** ((uint32_t)0x00000001)
- #define EXTI_IMR_MR0 ((uint32_t)0x00000001)
- #define EXTI_IMR_MR1 ((uint32_t)0x00000002)
- #define EXTI_IMR_MR2 ((uint32_t)0x00000004)
- #define EXTI_IMR_MR3 ((uint32_t)0x00000008)
- #define EXTI_IMR_MR4 ((uint32_t)0x00000010)
- #define EXTI_IMR_MR5 ((uint32_t)0x00000020)
- #define EXTI_IMR_MR6 ((uint32_t)0x00000040)
- #define EXTI_IMR_MR7 ((uint32_t)0x00000080)

- #define EXTI_IMR_MR8 ((uint32_t)0x00000100)
- #define EXTI_IMR_MR9 ((uint32_t)0x00000200)
- #define EXTI_IMR_MR10 ((uint32_t)0x00000400)
- #define EXTI_IMR_MR11 ((uint32_t)0x00000800)
- #define EXTI_IMR_MR12 ((uint32_t)0x00001000)
- #define EXTI_IMR_MR13 ((uint32_t)0x00002000)
- #define EXTI_IMR_MR14 ((uint32_t)0x00004000)
- #define EXTI_IMR_MR15 ((uint32_t)0x00008000)
- #define EXTI_IMR_MR16 ((uint32_t)0x00010000)
- #define EXTI_IMR_MR17 ((uint32_t)0x00020000)
- #define EXTI_IMR_MR18 ((uint32_t)0x00040000)
- #define EXTI_IMR_MR19 ((uint32_t)0x00080000)
- #define EXTI_EMR_MR0 ((uint32_t)0x00000001)
- #define EXTI_EMR_MR1 ((uint32_t)0x00000002)
- #define EXTI_EMR_MR2 ((uint32_t)0x00000004)
- #define EXTI_EMR_MR3 ((uint32_t)0x00000008)
- #define EXTI_EMR_MR4 ((uint32_t)0x00000010)
- #define EXTI_EMR_MR5 ((uint32_t)0x00000020)
- #define EXTI_EMR_MR6 ((uint32_t)0x00000040)
- #define EXTI_EMR_MR7 ((uint32_t)0x00000080)
- #define EXTI_EMR_MR8 ((uint32_t)0x00000100)
- #define EXTI_EMR_MR9 ((uint32_t)0x00000200)
- #define EXTI_EMR_MR10 ((uint32_t)0x00000400)
- #define EXTI_EMR_MR11 ((uint32_t)0x00000800)
- #define EXTI_EMR_MR12 ((uint32_t)0x00001000)
- #define EXTI_EMR_MR13 ((uint32_t)0x00002000)
- #define EXTI_EMR_MR14 ((uint32_t)0x00004000)
- #define EXTI_EMR_MR15 ((uint32_t)0x00008000)
- #define EXTI_EMR_MR16 ((uint32_t)0x00010000)
- #define EXTI_EMR_MR17 ((uint32_t)0x00020000)
- #define EXTI_EMR_MR18 ((uint32_t)0x00040000)
- #define EXTI_EMR_MR19 ((uint32_t)0x00080000)
- #define EXTI_RTSR_TR0 ((uint32_t)0x00000001)
- #define EXTI_RTSR_TR1 ((uint32_t)0x00000002)
- #define EXTI_RTSR_TR2 ((uint32_t)0x00000004)
- #define EXTI_RTSR_TR3 ((uint32_t)0x00000008)
- #define EXTI_RTSR_TR4 ((uint32_t)0x00000010)
- #define EXTI_RTSR_TR5 ((uint32_t)0x00000020)
- #define EXTI_RTSR_TR6 ((uint32_t)0x00000040)
- #define EXTI_RTSR_TR7 ((uint32_t)0x00000080)
- #define EXTI_RTSR_TR8 ((uint32_t)0x00000100)
- #define EXTI_RTSR_TR9 ((uint32_t)0x00000200)
- #define EXTI_RTSR_TR10 ((uint32_t)0x00000400)
- #define EXTI_RTSR_TR11 ((uint32_t)0x00000800)
- #define EXTI_RTSR_TR12 ((uint32_t)0x00001000)
- #define EXTI_RTSR_TR13 ((uint32_t)0x00002000)
- #define EXTI_RTSR_TR14 ((uint32_t)0x00004000)
- #define EXTI_RTSR_TR15 ((uint32_t)0x00008000)
- #define EXTI_RTSR_TR16 ((uint32_t)0x00010000)
- #define EXTI_RTSR_TR17 ((uint32_t)0x00020000)
- #define EXTI_RTSR_TR18 ((uint32_t)0x00040000)
- #define EXTI_RTSR_TR19 ((uint32_t)0x00080000)
- #define EXTI_FTSR_TR0 ((uint32_t)0x00000001)
- #define EXTI_FTSR_TR1 ((uint32_t)0x00000002)
- #define EXTI_FTSR_TR2 ((uint32_t)0x00000004)

- #define EXTI_FTSR_TR3 ((uint32_t)0x00000008)
- #define EXTI_FTSR_TR4 ((uint32_t)0x00000010)
- #define EXTI_FTSR_TR5 ((uint32_t)0x00000020)
- #define EXTI_FTSR_TR6 ((uint32_t)0x00000040)
- #define EXTI_FTSR_TR7 ((uint32_t)0x00000080)
- #define EXTI_FTSR_TR8 ((uint32_t)0x00000100)
- #define EXTI_FTSR_TR9 ((uint32_t)0x00000200)
- #define EXTI_FTSR_TR10 ((uint32_t)0x00000400)
- #define EXTI_FTSR_TR11 ((uint32_t)0x00000800)
- #define EXTI_FTSR_TR12 ((uint32_t)0x00001000)
- #define EXTI_FTSR_TR13 ((uint32_t)0x00002000)
- #define EXTI_FTSR_TR14 ((uint32_t)0x00004000)
- #define EXTI_FTSR_TR15 ((uint32_t)0x00008000)
- #define EXTI_FTSR_TR16 ((uint32_t)0x00010000)
- #define EXTI_FTSR_TR17 ((uint32_t)0x00020000)
- #define EXTI_FTSR_TR18 ((uint32_t)0x00040000)
- #define EXTI_FTSR_TR19 ((uint32_t)0x00080000)
- #define EXTI_SWIER_SWIER0 ((uint32_t)0x00000001)
- #define EXTI_SWIER_SWIER1 ((uint32_t)0x00000002)
- #define EXTI_SWIER_SWIER2 ((uint32_t)0x00000004)
- #define EXTI_SWIER_SWIER3 ((uint32_t)0x00000008)
- #define EXTI_SWIER_SWIER4 ((uint32_t)0x00000010)
- #define EXTI_SWIER_SWIER5 ((uint32_t)0x00000020)
- #define EXTI_SWIER_SWIER6 ((uint32_t)0x00000040)
- #define EXTI_SWIER_SWIER7 ((uint32_t)0x00000080)
- #define EXTI_SWIER_SWIER8 ((uint32_t)0x00000100)
- #define EXTI_SWIER_SWIER9 ((uint32_t)0x00000200)
- #define EXTI_SWIER_SWIER10 ((uint32_t)0x00000400)
- #define EXTI_SWIER_SWIER11 ((uint32_t)0x00000800)
- #define EXTI_SWIER_SWIER12 ((uint32_t)0x00001000)
- #define EXTI_SWIER_SWIER13 ((uint32_t)0x00002000)
- #define EXTI_SWIER_SWIER14 ((uint32_t)0x00004000)
- #define EXTI_SWIER_SWIER15 ((uint32_t)0x00008000)
- #define EXTI_SWIER_SWIER16 ((uint32_t)0x00010000)
- #define EXTI_SWIER_SWIER17 ((uint32_t)0x00020000)
- #define EXTI_SWIER_SWIER18 ((uint32_t)0x00040000)
- #define EXTI_SWIER_SWIER19 ((uint32_t)0x00080000)
- #define EXTI_PR_PR0 ((uint32_t)0x00000001)
- #define EXTI_PR_PR1 ((uint32_t)0x00000002)
- #define EXTI_PR_PR2 ((uint32_t)0x00000004)
- #define EXTI_PR_PR3 ((uint32_t)0x00000008)
- #define EXTI_PR_PR4 ((uint32_t)0x00000010)
- #define EXTI_PR_PR5 ((uint32_t)0x00000020)
- #define EXTI_PR_PR6 ((uint32_t)0x00000040)
- #define EXTI_PR_PR7 ((uint32_t)0x00000080)
- #define EXTI_PR_PR8 ((uint32_t)0x00000100)
- #define EXTI_PR_PR9 ((uint32_t)0x00000200)
- #define EXTI_PR_PR10 ((uint32_t)0x00000400)
- #define EXTI_PR_PR11 ((uint32_t)0x00000800)
- #define EXTI_PR_PR12 ((uint32_t)0x00001000)
- #define EXTI_PR_PR13 ((uint32_t)0x00002000)
- #define EXTI_PR_PR14 ((uint32_t)0x00004000)
- #define EXTI_PR_PR15 ((uint32_t)0x00008000)
- #define EXTI_PR_PR16 ((uint32_t)0x00010000)
- #define EXTI_PR_PR17 ((uint32_t)0x00020000)

- #define EXTI_PR_PR18 ((uint32_t)0x00040000)
- #define EXTI_PR_PR19 ((uint32_t)0x00080000)
- #define **FLASH_ACR_LATENCY** ((uint32_t)0x00000007)
- #define **FLASH_ACR_LATENCY_0WS** ((uint32_t)0x00000000)
- #define **FLASH_ACR_LATENCY_1WS** ((uint32_t)0x00000001)
- #define **FLASH_ACR_LATENCY_2WS** ((uint32_t)0x00000002)
- #define **FLASH_ACR_LATENCY_3WS** ((uint32_t)0x00000003)
- #define **FLASH_ACR_LATENCY_4WS** ((uint32_t)0x00000004)
- #define **FLASH_ACR_LATENCY_5WS** ((uint32_t)0x00000005)
- #define **FLASH_ACR_LATENCY_6WS** ((uint32_t)0x00000006)
- #define **FLASH_ACR_LATENCY_7WS** ((uint32_t)0x00000007)
- #define **FLASH_ACR_PRFTEN** ((uint32_t)0x00000100)
- #define **FLASH_ACR_ICEN** ((uint32_t)0x00000200)
- #define **FLASH_ACR_DCEN** ((uint32_t)0x00000400)
- #define **FLASH_ACR_ICRST** ((uint32_t)0x00000800)
- #define **FLASH_ACR_DCRST** ((uint32_t)0x00001000)
- #define **FLASH_ACR_BYTE0_ADDRESS** ((uint32_t)0x40023C00)
- #define **FLASH_ACR_BYTE2_ADDRESS** ((uint32_t)0x40023C03)
- #define **FLASH_SR_EOP** ((uint32_t)0x00000001)
- #define **FLASH_SR_SOP** ((uint32_t)0x00000002)
- #define **FLASH_SR_WRPERR** ((uint32_t)0x00000010)
- #define **FLASH_SR_PGAERR** ((uint32_t)0x00000020)
- #define **FLASH_SR_PGPERR** ((uint32_t)0x00000040)
- #define **FLASH_SR_PGSERR** ((uint32_t)0x00000080)
- #define **FLASH_SR_BSY** ((uint32_t)0x00010000)
- #define **FLASH_CR_PG** ((uint32_t)0x00000001)
- #define **FLASH_CR_SER** ((uint32_t)0x00000002)
- #define **FLASH_CR_MER** ((uint32_t)0x00000004)
- #define **FLASH_CR_SNB_0** ((uint32_t)0x00000008)
- #define **FLASH_CR_SNB_1** ((uint32_t)0x00000010)
- #define **FLASH_CR_SNB_2** ((uint32_t)0x00000020)
- #define **FLASH_CR_SNB_3** ((uint32_t)0x00000040)
- #define **FLASH_CR_PSIZE_0** ((uint32_t)0x00000100)
- #define **FLASH_CR_PSIZE_1** ((uint32_t)0x00000200)
- #define **FLASH_CR_STRT** ((uint32_t)0x00010000)
- #define **FLASH_CR_EOPIE** ((uint32_t)0x01000000)
- #define **FLASH_CR_LOCK** ((uint32_t)0x80000000)
- #define **FLASH_OPTCR_OPTLOCK** ((uint32_t)0x00000001)
- #define **FLASH_OPTCR_OPTSTRT** ((uint32_t)0x00000002)
- #define **FLASH_OPTCR_BOR_LEV_0** ((uint32_t)0x00000004)
- #define **FLASH_OPTCR_BOR_LEV_1** ((uint32_t)0x00000008)
- #define **FLASH_OPTCR_BOR_LEV** ((uint32_t)0x0000000C)
- #define **FLASH_OPTCR_WDG_SW** ((uint32_t)0x00000020)
- #define **FLASH_OPTCR_nRST_STOP** ((uint32_t)0x00000040)
- #define **FLASH_OPTCR_nRST_STDBY** ((uint32_t)0x00000080)
- #define **FLASH_OPTCR_RDP_0** ((uint32_t)0x00000100)
- #define **FLASH_OPTCR_RDP_1** ((uint32_t)0x00000200)
- #define **FLASH_OPTCR_RDP_2** ((uint32_t)0x00000400)
- #define **FLASH_OPTCR_RDP_3** ((uint32_t)0x00000800)
- #define **FLASH_OPTCR_RDP_4** ((uint32_t)0x00001000)
- #define **FLASH_OPTCR_RDP_5** ((uint32_t)0x00002000)
- #define **FLASH_OPTCR_RDP_6** ((uint32_t)0x00004000)
- #define **FLASH_OPTCR_RDP_7** ((uint32_t)0x00008000)
- #define **FLASH_OPTCR_nWRP_0** ((uint32_t)0x00010000)
- #define **FLASH_OPTCR_nWRP_1** ((uint32_t)0x00020000)

- #define **FLASH_OPTCR_nWRP_2** ((uint32_t)0x00040000)
- #define **FLASH_OPTCR_nWRP_3** ((uint32_t)0x00080000)
- #define **FLASH_OPTCR_nWRP_4** ((uint32_t)0x00100000)
- #define **FLASH_OPTCR_nWRP_5** ((uint32_t)0x00200000)
- #define **FLASH_OPTCR_nWRP_6** ((uint32_t)0x00400000)
- #define **FLASH_OPTCR_nWRP_7** ((uint32_t)0x00800000)
- #define **FLASH_OPTCR_nWRP_8** ((uint32_t)0x01000000)
- #define **FLASH_OPTCR_nWRP_9** ((uint32_t)0x02000000)
- #define **FLASH_OPTCR_nWRP_10** ((uint32_t)0x04000000)
- #define **FLASH_OPTCR_nWRP_11** ((uint32_t)0x08000000)
- #define FSMC_BCR1_MBKEN ((uint32_t)0x00000001)
- #define FSMC_BCR1_MUXEN ((uint32_t)0x00000002)
- #define FSMC_BCR1_MTYP ((uint32_t)0x0000000C)
- #define FSMC_BCR1_MTYP_0 ((uint32_t)0x00000004)
- #define FSMC_BCR1_MTYP_1 ((uint32_t)0x00000008)
- #define FSMC_BCR1_MWID ((uint32_t)0x00000030)
- #define FSMC_BCR1_MWID_0 ((uint32_t)0x00000010)
- #define FSMC_BCR1_MWID_1 ((uint32_t)0x00000020)
- #define FSMC_BCR1_FACCEN ((uint32_t)0x00000040)
- #define FSMC_BCR1_BURSTEN ((uint32_t)0x00000100)
- #define FSMC_BCR1_WAITPOL ((uint32_t)0x00000200)
- #define FSMC_BCR1_WRAPMOD ((uint32_t)0x00000400)
- #define FSMC_BCR1_WAITCFG ((uint32_t)0x00000800)
- #define FSMC_BCR1_WREN ((uint32_t)0x00001000)
- #define FSMC_BCR1_WAITEN ((uint32_t)0x00002000)
- #define FSMC_BCR1_EXTMOD ((uint32_t)0x00004000)
- #define FSMC_BCR1_ASYNCWAIT ((uint32_t)0x00008000)
- #define FSMC_BCR1_CBURSTRW ((uint32_t)0x00080000)
- #define FSMC_BCR2_MBKEN ((uint32_t)0x00000001)
- #define FSMC_BCR2_MUXEN ((uint32_t)0x00000002)
- #define FSMC_BCR2_MTYP ((uint32_t)0x0000000C)
- #define FSMC_BCR2_MTYP_0 ((uint32_t)0x00000004)
- #define FSMC_BCR2_MTYP_1 ((uint32_t)0x00000008)
- #define FSMC_BCR2_MWID ((uint32_t)0x00000030)
- #define FSMC_BCR2_MWID_0 ((uint32_t)0x00000010)
- #define FSMC_BCR2_MWID_1 ((uint32_t)0x00000020)
- #define FSMC_BCR2_FACCEN ((uint32_t)0x00000040)
- #define FSMC_BCR2_BURSTEN ((uint32_t)0x00000100)
- #define FSMC_BCR2_WAITPOL ((uint32_t)0x00000200)
- #define FSMC_BCR2_WRAPMOD ((uint32_t)0x00000400)
- #define FSMC_BCR2_WAITCFG ((uint32_t)0x00000800)
- #define FSMC_BCR2_WREN ((uint32_t)0x00001000)
- #define FSMC_BCR2_WAITEN ((uint32_t)0x00002000)
- #define FSMC_BCR2_EXTMOD ((uint32_t)0x00004000)
- #define FSMC_BCR2_ASYNCWAIT ((uint32_t)0x00008000)
- #define FSMC_BCR2_CBURSTRW ((uint32_t)0x00080000)
- #define FSMC_BCR3_MBKEN ((uint32_t)0x00000001)
- #define FSMC_BCR3_MUXEN ((uint32_t)0x00000002)
- #define FSMC_BCR3_MTYP ((uint32_t)0x0000000C)
- #define FSMC_BCR3_MTYP_0 ((uint32_t)0x00000004)
- #define FSMC_BCR3_MTYP_1 ((uint32_t)0x00000008)
- #define FSMC_BCR3_MWID ((uint32_t)0x00000030)
- #define FSMC_BCR3_MWID_0 ((uint32_t)0x00000010)
- #define FSMC_BCR3_MWID_1 ((uint32_t)0x00000020)
- #define FSMC_BCR3_FACCEN ((uint32_t)0x00000040)

- #define FSMC_BCR3_BURSTEN ((uint32_t)0x00000100)
- #define FSMC_BCR3_WAITPOL ((uint32_t)0x00000200)
- #define FSMC_BCR3_WRAPMOD ((uint32_t)0x00000400)
- #define FSMC_BCR3_WAITCFG ((uint32_t)0x00000800)
- #define FSMC_BCR3_WREN ((uint32_t)0x00001000)
- #define FSMC_BCR3_WAITEN ((uint32_t)0x00002000)
- #define FSMC_BCR3_EXTMOD ((uint32_t)0x00004000)
- #define FSMC_BCR3_ASYNCWAIT ((uint32_t)0x00008000)
- #define FSMC_BCR3_CBURSTRW ((uint32_t)0x00080000)
- #define FSMC_BCR4_MBKEN ((uint32_t)0x00000001)
- #define FSMC_BCR4_MUXEN ((uint32_t)0x00000002)
- #define FSMC_BCR4_MTYP ((uint32_t)0x0000000C)
- #define FSMC_BCR4_MTYP_0 ((uint32_t)0x00000004)
- #define FSMC_BCR4_MTYP_1 ((uint32_t)0x00000008)
- #define FSMC_BCR4_MWID ((uint32_t)0x00000030)
- #define FSMC_BCR4_MWID_0 ((uint32_t)0x00000010)
- #define FSMC_BCR4_MWID_1 ((uint32_t)0x00000020)
- #define FSMC_BCR4_FACCEN ((uint32_t)0x00000040)
- #define FSMC_BCR4_BURSTEN ((uint32_t)0x00000100)
- #define FSMC_BCR4_WAITPOL ((uint32_t)0x00000200)
- #define FSMC_BCR4_WRAPMOD ((uint32_t)0x00000400)
- #define FSMC_BCR4_WAITCFG ((uint32_t)0x00000800)
- #define FSMC_BCR4_WREN ((uint32_t)0x00001000)
- #define FSMC_BCR4_WAITEN ((uint32_t)0x00002000)
- #define FSMC_BCR4_EXTMOD ((uint32_t)0x00004000)
- #define FSMC_BCR4_ASYNCWAIT ((uint32_t)0x00008000)
- #define FSMC_BCR4_CBURSTRW ((uint32_t)0x00080000)
- #define FSMC_BTR1_ADDSET ((uint32_t)0x0000000F)
- #define FSMC_BTR1_ADDSET_0 ((uint32_t)0x00000001)
- #define FSMC_BTR1_ADDSET_1 ((uint32_t)0x00000002)
- #define FSMC_BTR1_ADDSET_2 ((uint32_t)0x00000004)
- #define FSMC_BTR1_ADDSET_3 ((uint32_t)0x00000008)
- #define FSMC_BTR1_ADDHLD ((uint32_t)0x000000F0)
- #define FSMC_BTR1_ADDHLD_0 ((uint32_t)0x00000010)
- #define FSMC_BTR1_ADDHLD_1 ((uint32_t)0x00000020)
- #define FSMC_BTR1_ADDHLD_2 ((uint32_t)0x00000040)
- #define FSMC_BTR1_ADDHLD_3 ((uint32_t)0x00000080)
- #define FSMC_BTR1_DATAST ((uint32_t)0x0000FF00)
- #define FSMC_BTR1_DATAST_0 ((uint32_t)0x00000100)
- #define FSMC_BTR1_DATAST_1 ((uint32_t)0x00000200)
- #define FSMC_BTR1_DATAST_2 ((uint32_t)0x00000400)
- #define FSMC_BTR1_DATAST_3 ((uint32_t)0x00000800)
- #define FSMC_BTR1_BUSTURN ((uint32_t)0x000F0000)
- #define FSMC_BTR1_BUSTURN_0 ((uint32_t)0x00010000)
- #define FSMC_BTR1_BUSTURN_1 ((uint32_t)0x00020000)
- #define FSMC_BTR1_BUSTURN_2 ((uint32_t)0x00040000)
- #define FSMC_BTR1_BUSTURN_3 ((uint32_t)0x00080000)
- #define FSMC_BTR1_CLKDIV ((uint32_t)0x00F00000)
- #define FSMC_BTR1_CLKDIV_0 ((uint32_t)0x00100000)
- #define FSMC_BTR1_CLKDIV_1 ((uint32_t)0x00200000)
- #define FSMC_BTR1_CLKDIV_2 ((uint32_t)0x00400000)
- #define FSMC_BTR1_CLKDIV_3 ((uint32_t)0x00800000)
- #define FSMC_BTR1_DATLAT ((uint32_t)0x0F000000)
- #define FSMC_BTR1_DATLAT_0 ((uint32_t)0x01000000)
- #define FSMC_BTR1_DATLAT_1 ((uint32_t)0x02000000)

- #define FSMC_BTR1_DATLAT_2 ((uint32_t)0x04000000)
- #define FSMC_BTR1_DATLAT_3 ((uint32_t)0x08000000)
- #define FSMC_BTR1_ACCMOD ((uint32_t)0x30000000)
- #define FSMC_BTR1_ACCMOD_0 ((uint32_t)0x10000000)
- #define FSMC_BTR1_ACCMOD_1 ((uint32_t)0x20000000)
- #define FSMC_BTR2_ADDSET ((uint32_t)0x0000000F)
- #define FSMC_BTR2_ADDSET_0 ((uint32_t)0x00000001)
- #define FSMC_BTR2_ADDSET_1 ((uint32_t)0x00000002)
- #define FSMC_BTR2_ADDSET_2 ((uint32_t)0x00000004)
- #define FSMC_BTR2_ADDSET_3 ((uint32_t)0x00000008)
- #define FSMC_BTR2_ADDHLD ((uint32_t)0x000000F0)
- #define FSMC_BTR2_ADDHLD_0 ((uint32_t)0x00000010)
- #define FSMC_BTR2_ADDHLD_1 ((uint32_t)0x00000020)
- #define FSMC_BTR2_ADDHLD_2 ((uint32_t)0x00000040)
- #define FSMC_BTR2_ADDHLD_3 ((uint32_t)0x00000080)
- #define FSMC_BTR2_DATAST ((uint32_t)0x0000FF00)
- #define FSMC_BTR2_DATAST_0 ((uint32_t)0x00000100)
- #define FSMC_BTR2_DATAST_1 ((uint32_t)0x00000200)
- #define FSMC_BTR2_DATAST_2 ((uint32_t)0x00000400)
- #define FSMC_BTR2_DATAST_3 ((uint32_t)0x00000800)
- #define FSMC_BTR2_BUSTURN ((uint32_t)0x000F0000)
- #define FSMC_BTR2_BUSTURN_0 ((uint32_t)0x00010000)
- #define FSMC_BTR2_BUSTURN_1 ((uint32_t)0x00020000)
- #define FSMC_BTR2_BUSTURN_2 ((uint32_t)0x00040000)
- #define FSMC_BTR2_BUSTURN_3 ((uint32_t)0x00080000)
- #define FSMC_BTR2_CLKDIV ((uint32_t)0x00F00000)
- #define FSMC_BTR2_CLKDIV_0 ((uint32_t)0x00100000)
- #define FSMC_BTR2_CLKDIV_1 ((uint32_t)0x00200000)
- #define FSMC_BTR2_CLKDIV_2 ((uint32_t)0x00400000)
- #define FSMC_BTR2_CLKDIV_3 ((uint32_t)0x00800000)
- #define FSMC_BTR2_DATLAT ((uint32_t)0x0F000000)
- #define FSMC_BTR2_DATLAT_0 ((uint32_t)0x01000000)
- #define FSMC_BTR2_DATLAT_1 ((uint32_t)0x02000000)
- #define FSMC_BTR2_DATLAT_2 ((uint32_t)0x04000000)
- #define FSMC_BTR2_DATLAT_3 ((uint32_t)0x08000000)
- #define FSMC_BTR2_ACCMOD ((uint32_t)0x30000000)
- #define FSMC_BTR2_ACCMOD_0 ((uint32_t)0x10000000)
- #define FSMC_BTR2_ACCMOD_1 ((uint32_t)0x20000000)
- #define FSMC_BTR3_ADDSET ((uint32_t)0x0000000F)
- #define FSMC_BTR3_ADDSET_0 ((uint32_t)0x00000001)
- #define FSMC_BTR3_ADDSET_1 ((uint32_t)0x00000002)
- #define FSMC_BTR3_ADDSET_2 ((uint32_t)0x00000004)
- #define FSMC_BTR3_ADDSET_3 ((uint32_t)0x00000008)
- #define FSMC_BTR3_ADDHLD ((uint32_t)0x000000F0)
- #define FSMC_BTR3_ADDHLD_0 ((uint32_t)0x00000010)
- #define FSMC_BTR3_ADDHLD_1 ((uint32_t)0x00000020)
- #define FSMC_BTR3_ADDHLD_2 ((uint32_t)0x00000040)
- #define FSMC_BTR3_ADDHLD_3 ((uint32_t)0x00000080)
- #define FSMC_BTR3_DATAST ((uint32_t)0x0000FF00)
- #define FSMC_BTR3_DATAST_0 ((uint32_t)0x00000100)
- #define FSMC_BTR3_DATAST_1 ((uint32_t)0x00000200)
- #define FSMC_BTR3_DATAST_2 ((uint32_t)0x00000400)
- #define FSMC_BTR3_DATAST_3 ((uint32_t)0x00000800)
- #define FSMC_BTR3_BUSTURN ((uint32_t)0x000F0000)
- #define FSMC_BTR3_BUSTURN_0 ((uint32_t)0x00010000)

- #define FSMC_BTR3_BUSTURN_1 ((uint32_t)0x00020000)
- #define FSMC_BTR3_BUSTURN_2 ((uint32_t)0x00040000)
- #define FSMC_BTR3_BUSTURN_3 ((uint32_t)0x00080000)
- #define FSMC_BTR3_CLKDIV ((uint32_t)0x00F00000)
- #define FSMC_BTR3_CLKDIV_0 ((uint32_t)0x00100000)
- #define FSMC_BTR3_CLKDIV_1 ((uint32_t)0x00200000)
- #define FSMC_BTR3_CLKDIV_2 ((uint32_t)0x00400000)
- #define FSMC_BTR3_CLKDIV_3 ((uint32_t)0x00800000)
- #define FSMC_BTR3_DATLAT ((uint32_t)0x0F000000)
- #define FSMC_BTR3_DATLAT_0 ((uint32_t)0x01000000)
- #define FSMC_BTR3_DATLAT_1 ((uint32_t)0x02000000)
- #define FSMC_BTR3_DATLAT_2 ((uint32_t)0x04000000)
- #define FSMC_BTR3_DATLAT_3 ((uint32_t)0x08000000)
- #define FSMC_BTR3_ACCMOD ((uint32_t)0x30000000)
- #define FSMC_BTR3_ACCMOD_0 ((uint32_t)0x10000000)
- #define FSMC_BTR3_ACCMOD_1 ((uint32_t)0x20000000)
- #define FSMC_BTR4_ADDSET ((uint32_t)0x0000000F)
- #define FSMC_BTR4_ADDSET_0 ((uint32_t)0x00000001)
- #define FSMC_BTR4_ADDSET_1 ((uint32_t)0x00000002)
- #define FSMC_BTR4_ADDSET_2 ((uint32_t)0x00000004)
- #define FSMC_BTR4_ADDSET_3 ((uint32_t)0x00000008)
- #define FSMC_BTR4_ADDHLD ((uint32_t)0x000000F0)
- #define FSMC_BTR4_ADDHLD_0 ((uint32_t)0x00000010)
- #define FSMC_BTR4_ADDHLD_1 ((uint32_t)0x00000020)
- #define FSMC_BTR4_ADDHLD_2 ((uint32_t)0x00000040)
- #define FSMC_BTR4_ADDHLD_3 ((uint32_t)0x00000080)
- #define FSMC_BTR4_DATAST ((uint32_t)0x0000FF00)
- #define FSMC_BTR4_DATAST_0 ((uint32_t)0x00000100)
- #define FSMC_BTR4_DATAST_1 ((uint32_t)0x00000200)
- #define FSMC_BTR4_DATAST_2 ((uint32_t)0x00000400)
- #define FSMC_BTR4_DATAST_3 ((uint32_t)0x00000800)
- #define FSMC_BTR4_BUSTURN ((uint32_t)0x000F0000)
- #define FSMC_BTR4_BUSTURN_0 ((uint32_t)0x00010000)
- #define FSMC_BTR4_BUSTURN_1 ((uint32_t)0x00020000)
- #define FSMC_BTR4_BUSTURN_2 ((uint32_t)0x00040000)
- #define FSMC_BTR4_BUSTURN_3 ((uint32_t)0x00080000)
- #define FSMC_BTR4_CLKDIV ((uint32_t)0x00F00000)
- #define FSMC_BTR4_CLKDIV_0 ((uint32_t)0x00100000)
- #define FSMC_BTR4_CLKDIV_1 ((uint32_t)0x00200000)
- #define FSMC_BTR4_CLKDIV_2 ((uint32_t)0x00400000)
- #define FSMC_BTR4_CLKDIV_3 ((uint32_t)0x00800000)
- #define FSMC_BTR4_DATLAT ((uint32_t)0x0F000000)
- #define FSMC_BTR4_DATLAT_0 ((uint32_t)0x01000000)
- #define FSMC_BTR4_DATLAT_1 ((uint32_t)0x02000000)
- #define FSMC_BTR4_DATLAT_2 ((uint32_t)0x04000000)
- #define FSMC_BTR4_DATLAT_3 ((uint32_t)0x08000000)
- #define FSMC_BTR4_ACCMOD ((uint32_t)0x30000000)
- #define FSMC_BTR4_ACCMOD_0 ((uint32_t)0x10000000)
- #define FSMC_BTR4_ACCMOD_1 ((uint32_t)0x20000000)
- #define FSMC_BWTR1_ADDSET ((uint32_t)0x0000000F)
- #define FSMC_BWTR1_ADDSET_0 ((uint32_t)0x00000001)
- #define FSMC_BWTR1_ADDSET_1 ((uint32_t)0x00000002)
- #define FSMC_BWTR1_ADDSET_2 ((uint32_t)0x00000004)
- #define FSMC_BWTR1_ADDSET_3 ((uint32_t)0x00000008)
- #define FSMC_BWTR1_ADDHLD ((uint32_t)0x000000F0)

- #define FSMC_BWTR1_ADDHLD_0 ((uint32_t)0x00000010)
- #define FSMC_BWTR1_ADDHLD_1 ((uint32_t)0x00000020)
- #define FSMC_BWTR1_ADDHLD_2 ((uint32_t)0x00000040)
- #define FSMC_BWTR1_ADDHLD_3 ((uint32_t)0x00000080)
- #define FSMC_BWTR1_DATAST ((uint32_t)0x0000FF00)
- #define FSMC_BWTR1_DATAST_0 ((uint32_t)0x00000100)
- #define FSMC_BWTR1_DATAST_1 ((uint32_t)0x00000200)
- #define FSMC_BWTR1_DATAST_2 ((uint32_t)0x00000400)
- #define FSMC_BWTR1_DATAST_3 ((uint32_t)0x00000800)
- #define FSMC_BWTR1_CLKDIV ((uint32_t)0x00F00000)
- #define FSMC_BWTR1_CLKDIV_0 ((uint32_t)0x00100000)
- #define FSMC_BWTR1_CLKDIV_1 ((uint32_t)0x00200000)
- #define FSMC_BWTR1_CLKDIV_2 ((uint32_t)0x00400000)
- #define FSMC_BWTR1_CLKDIV_3 ((uint32_t)0x00800000)
- #define FSMC_BWTR1_DATLAT ((uint32_t)0x0F000000)
- #define FSMC_BWTR1_DATLAT_0 ((uint32_t)0x01000000)
- #define FSMC_BWTR1_DATLAT_1 ((uint32_t)0x02000000)
- #define FSMC_BWTR1_DATLAT_2 ((uint32_t)0x04000000)
- #define FSMC_BWTR1_DATLAT_3 ((uint32_t)0x08000000)
- #define FSMC_BWTR1_ACCMOD ((uint32_t)0x30000000)
- #define FSMC_BWTR1_ACCMOD_0 ((uint32_t)0x10000000)
- #define FSMC_BWTR1_ACCMOD_1 ((uint32_t)0x20000000)
- #define FSMC_BWTR2_ADDSET ((uint32_t)0x0000000F)
- #define FSMC_BWTR2_ADDSET_0 ((uint32_t)0x00000001)
- #define FSMC_BWTR2_ADDSET_1 ((uint32_t)0x00000002)
- #define FSMC_BWTR2_ADDSET_2 ((uint32_t)0x00000004)
- #define FSMC_BWTR2_ADDSET_3 ((uint32_t)0x00000008)
- #define FSMC_BWTR2_ADDHLD ((uint32_t)0x000000F0)
- #define FSMC_BWTR2_ADDHLD_0 ((uint32_t)0x00000010)
- #define FSMC_BWTR2_ADDHLD_1 ((uint32_t)0x00000020)
- #define FSMC_BWTR2_ADDHLD_2 ((uint32_t)0x00000040)
- #define FSMC_BWTR2_ADDHLD_3 ((uint32_t)0x00000080)
- #define FSMC_BWTR2_DATAST ((uint32_t)0x0000FF00)
- #define FSMC_BWTR2_DATAST_0 ((uint32_t)0x00000100)
- #define FSMC_BWTR2_DATAST_1 ((uint32_t)0x00000200)
- #define FSMC_BWTR2_DATAST_2 ((uint32_t)0x00000400)
- #define FSMC_BWTR2_DATAST_3 ((uint32_t)0x00000800)
- #define FSMC_BWTR2_CLKDIV ((uint32_t)0x00F00000)
- #define FSMC_BWTR2_CLKDIV_0 ((uint32_t)0x00100000)
- #define FSMC_BWTR2_CLKDIV_1 ((uint32_t)0x00200000)
- #define FSMC_BWTR2_CLKDIV_2 ((uint32_t)0x00400000)
- #define FSMC_BWTR2_CLKDIV_3 ((uint32_t)0x00800000)
- #define FSMC_BWTR2_DATLAT ((uint32_t)0x0F000000)
- #define FSMC_BWTR2_DATLAT_0 ((uint32_t)0x01000000)
- #define FSMC_BWTR2_DATLAT_1 ((uint32_t)0x02000000)
- #define FSMC_BWTR2_DATLAT_2 ((uint32_t)0x04000000)
- #define FSMC_BWTR2_DATLAT_3 ((uint32_t)0x08000000)
- #define FSMC_BWTR2_ACCMOD ((uint32_t)0x30000000)
- #define FSMC_BWTR2_ACCMOD_0 ((uint32_t)0x10000000)
- #define FSMC_BWTR2_ACCMOD_1 ((uint32_t)0x20000000)
- #define FSMC_BWTR3_ADDSET ((uint32_t)0x0000000F)
- #define FSMC_BWTR3_ADDSET_0 ((uint32_t)0x00000001)
- #define FSMC_BWTR3_ADDSET_1 ((uint32_t)0x00000002)
- #define FSMC_BWTR3_ADDSET_2 ((uint32_t)0x00000004)
- #define FSMC_BWTR3_ADDSET_3 ((uint32_t)0x00000008)

- #define FSMC_BWTR3_ADDHLD ((uint32_t)0x000000F0)
- #define FSMC_BWTR3_ADDHLD_0 ((uint32_t)0x00000010)
- #define FSMC_BWTR3_ADDHLD_1 ((uint32_t)0x00000020)
- #define FSMC_BWTR3_ADDHLD_2 ((uint32_t)0x00000040)
- #define FSMC_BWTR3_ADDHLD_3 ((uint32_t)0x00000080)
- #define FSMC_BWTR3_DATAST ((uint32_t)0x0000FF00)
- #define FSMC_BWTR3_DATAST_0 ((uint32_t)0x00000100)
- #define FSMC_BWTR3_DATAST_1 ((uint32_t)0x00000200)
- #define FSMC_BWTR3_DATAST_2 ((uint32_t)0x00000400)
- #define FSMC_BWTR3_DATAST_3 ((uint32_t)0x00000800)
- #define FSMC_BWTR3_CLKDIV ((uint32_t)0x00F00000)
- #define FSMC_BWTR3_CLKDIV_0 ((uint32_t)0x00100000)
- #define FSMC_BWTR3_CLKDIV_1 ((uint32_t)0x00200000)
- #define FSMC_BWTR3_CLKDIV_2 ((uint32_t)0x00400000)
- #define FSMC_BWTR3_CLKDIV_3 ((uint32_t)0x00800000)
- #define FSMC_BWTR3_DATLAT ((uint32_t)0x0F000000)
- #define FSMC_BWTR3_DATLAT_0 ((uint32_t)0x01000000)
- #define FSMC_BWTR3_DATLAT_1 ((uint32_t)0x02000000)
- #define FSMC_BWTR3_DATLAT_2 ((uint32_t)0x04000000)
- #define FSMC_BWTR3_DATLAT_3 ((uint32_t)0x08000000)
- #define FSMC_BWTR3_ACCMOD ((uint32_t)0x30000000)
- #define FSMC_BWTR3_ACCMOD_0 ((uint32_t)0x10000000)
- #define FSMC_BWTR3_ACCMOD_1 ((uint32_t)0x20000000)
- #define FSMC_BWTR4_ADDSET ((uint32_t)0x0000000F)
- #define FSMC_BWTR4_ADDSET_0 ((uint32_t)0x00000001)
- #define FSMC_BWTR4_ADDSET_1 ((uint32_t)0x00000002)
- #define FSMC_BWTR4_ADDSET_2 ((uint32_t)0x00000004)
- #define FSMC_BWTR4_ADDSET_3 ((uint32_t)0x00000008)
- #define FSMC_BWTR4_ADDHLD ((uint32_t)0x000000F0)
- #define FSMC_BWTR4_ADDHLD_0 ((uint32_t)0x00000010)
- #define FSMC_BWTR4_ADDHLD_1 ((uint32_t)0x00000020)
- #define FSMC_BWTR4_ADDHLD_2 ((uint32_t)0x00000040)
- #define FSMC_BWTR4_ADDHLD_3 ((uint32_t)0x00000080)
- #define FSMC_BWTR4_DATAST ((uint32_t)0x0000FF00)
- #define FSMC_BWTR4_DATAST_0 ((uint32_t)0x00000100)
- #define FSMC_BWTR4_DATAST_1 ((uint32_t)0x00000200)
- #define FSMC_BWTR4_DATAST_2 ((uint32_t)0x00000400)
- #define FSMC_BWTR4_DATAST_3 ((uint32_t)0x00000800)
- #define FSMC_BWTR4_CLKDIV ((uint32_t)0x00F00000)
- #define FSMC_BWTR4_CLKDIV_0 ((uint32_t)0x00100000)
- #define FSMC_BWTR4_CLKDIV_1 ((uint32_t)0x00200000)
- #define FSMC_BWTR4_CLKDIV_2 ((uint32_t)0x00400000)
- #define FSMC_BWTR4_CLKDIV_3 ((uint32_t)0x00800000)
- #define FSMC_BWTR4_DATLAT ((uint32_t)0x0F000000)
- #define FSMC_BWTR4_DATLAT_0 ((uint32_t)0x01000000)
- #define FSMC_BWTR4_DATLAT_1 ((uint32_t)0x02000000)
- #define FSMC_BWTR4_DATLAT_2 ((uint32_t)0x04000000)
- #define FSMC_BWTR4_DATLAT_3 ((uint32_t)0x08000000)
- #define FSMC_BWTR4_ACCMOD ((uint32_t)0x30000000)
- #define FSMC_BWTR4_ACCMOD_0 ((uint32_t)0x10000000)
- #define FSMC_BWTR4_ACCMOD_1 ((uint32_t)0x20000000)
- #define FSMC_PCR2_PWAITEN ((uint32_t)0x00000002)
- #define FSMC_PCR2_PBKEN ((uint32_t)0x00000004)
- #define FSMC_PCR2_PTYP ((uint32_t)0x00000008)
- #define FSMC_PCR2_PWID ((uint32_t)0x00000030)

- #define FSMC_PCR2_PWID_0 ((uint32_t)0x00000010)
- #define FSMC_PCR2_PWID_1 ((uint32_t)0x00000020)
- #define FSMC_PCR2_ECCEN ((uint32_t)0x00000040)
- #define FSMC_PCR2_TCLR ((uint32_t)0x00001E00)
- #define FSMC_PCR2_TCLR_0 ((uint32_t)0x00000200)
- #define FSMC_PCR2_TCLR_1 ((uint32_t)0x00000400)
- #define FSMC_PCR2_TCLR_2 ((uint32_t)0x00000800)
- #define FSMC_PCR2_TCLR_3 ((uint32_t)0x00001000)
- #define FSMC_PCR2_TAR ((uint32_t)0x0001E000)
- #define FSMC_PCR2_TAR_0 ((uint32_t)0x00002000)
- #define FSMC_PCR2_TAR_1 ((uint32_t)0x00004000)
- #define FSMC_PCR2_TAR_2 ((uint32_t)0x00008000)
- #define FSMC_PCR2_TAR_3 ((uint32_t)0x00010000)
- #define FSMC_PCR2_ECCPS ((uint32_t)0x000E0000)
- #define FSMC_PCR2_ECCPS_0 ((uint32_t)0x00020000)
- #define FSMC_PCR2_ECCPS_1 ((uint32_t)0x00040000)
- #define FSMC_PCR2_ECCPS_2 ((uint32_t)0x00080000)
- #define FSMC_PCR3_PWAITEN ((uint32_t)0x00000002)
- #define FSMC_PCR3_PBKEN ((uint32_t)0x00000004)
- #define FSMC_PCR3_PTYP ((uint32_t)0x00000008)
- #define FSMC_PCR3_PWID ((uint32_t)0x00000030)
- #define FSMC_PCR3_PWID_0 ((uint32_t)0x00000010)
- #define FSMC_PCR3_PWID_1 ((uint32_t)0x00000020)
- #define FSMC_PCR3_ECCEN ((uint32_t)0x00000040)
- #define FSMC_PCR3_TCLR ((uint32_t)0x00001E00)
- #define FSMC_PCR3_TCLR_0 ((uint32_t)0x00000200)
- #define FSMC_PCR3_TCLR_1 ((uint32_t)0x00000400)
- #define FSMC_PCR3_TCLR_2 ((uint32_t)0x00000800)
- #define FSMC_PCR3_TCLR_3 ((uint32_t)0x00001000)
- #define FSMC_PCR3_TAR ((uint32_t)0x0001E000)
- #define FSMC_PCR3_TAR_0 ((uint32_t)0x00002000)
- #define FSMC_PCR3_TAR_1 ((uint32_t)0x00004000)
- #define FSMC_PCR3_TAR_2 ((uint32_t)0x00008000)
- #define FSMC_PCR3_TAR_3 ((uint32_t)0x00010000)
- #define FSMC_PCR3_ECCPS ((uint32_t)0x000E0000)
- #define FSMC_PCR3_ECCPS_0 ((uint32_t)0x00020000)
- #define FSMC_PCR3_ECCPS_1 ((uint32_t)0x00040000)
- #define FSMC_PCR3_ECCPS_2 ((uint32_t)0x00080000)
- #define FSMC_PCR4_PWAITEN ((uint32_t)0x00000002)
- #define FSMC_PCR4_PBKEN ((uint32_t)0x00000004)
- #define FSMC_PCR4_PTYP ((uint32_t)0x00000008)
- #define FSMC_PCR4_PWID ((uint32_t)0x00000030)
- #define FSMC_PCR4_PWID_0 ((uint32_t)0x00000010)
- #define FSMC_PCR4_PWID_1 ((uint32_t)0x00000020)
- #define FSMC_PCR4_ECCEN ((uint32_t)0x00000040)
- #define FSMC_PCR4_TCLR ((uint32_t)0x00001E00)
- #define FSMC_PCR4_TCLR_0 ((uint32_t)0x00000200)
- #define FSMC_PCR4_TCLR_1 ((uint32_t)0x00000400)
- #define FSMC_PCR4_TCLR_2 ((uint32_t)0x00000800)
- #define FSMC_PCR4_TCLR_3 ((uint32_t)0x00001000)
- #define FSMC_PCR4_TAR ((uint32_t)0x0001E000)
- #define FSMC_PCR4_TAR_0 ((uint32_t)0x00002000)
- #define FSMC_PCR4_TAR_1 ((uint32_t)0x00004000)
- #define FSMC_PCR4_TAR_2 ((uint32_t)0x00008000)
- #define FSMC_PCR4_TAR_3 ((uint32_t)0x00010000)

- #define FSMC_PCR4_ECCPS ((uint32_t)0x000E0000)
- #define FSMC_PCR4_ECCPS_0 ((uint32_t)0x00020000)
- #define FSMC_PCR4_ECCPS_1 ((uint32_t)0x00040000)
- #define FSMC_PCR4_ECCPS_2 ((uint32_t)0x00080000)
- #define FSMC_SR2_IRS ((uint8_t)0x01)
- #define FSMC_SR2_ILS ((uint8_t)0x02)
- #define FSMC_SR2_IFS ((uint8_t)0x04)
- #define FSMC_SR2_IREN ((uint8_t)0x08)
- #define FSMC_SR2_ILEN ((uint8_t)0x10)
- #define FSMC_SR2_IFEN ((uint8_t)0x20)
- #define FSMC_SR2_FEMPT ((uint8_t)0x40)
- #define FSMC_SR3_IRS ((uint8_t)0x01)
- #define FSMC_SR3_ILS ((uint8_t)0x02)
- #define FSMC_SR3_IFS ((uint8_t)0x04)
- #define FSMC_SR3_IREN ((uint8_t)0x08)
- #define FSMC_SR3_ILEN ((uint8_t)0x10)
- #define FSMC_SR3_IFEN ((uint8_t)0x20)
- #define FSMC_SR3_FEMPT ((uint8_t)0x40)
- #define FSMC_SR4_IRS ((uint8_t)0x01)
- #define FSMC_SR4_ILS ((uint8_t)0x02)
- #define FSMC_SR4_IFS ((uint8_t)0x04)
- #define FSMC_SR4_IREN ((uint8_t)0x08)
- #define FSMC_SR4_ILEN ((uint8_t)0x10)
- #define FSMC_SR4_IFEN ((uint8_t)0x20)
- #define FSMC_SR4_FEMPT ((uint8_t)0x40)
- #define FSMC_PMEM2_MEMSET2 ((uint32_t)0x000000FF)
- #define FSMC_PMEM2_MEMSET2_0 ((uint32_t)0x00000001)
- #define FSMC_PMEM2_MEMSET2_1 ((uint32_t)0x00000002)
- #define FSMC_PMEM2_MEMSET2_2 ((uint32_t)0x00000004)
- #define FSMC_PMEM2_MEMSET2_3 ((uint32_t)0x00000008)
- #define FSMC_PMEM2_MEMSET2_4 ((uint32_t)0x00000010)
- #define FSMC_PMEM2_MEMSET2_5 ((uint32_t)0x00000020)
- #define FSMC_PMEM2_MEMSET2_6 ((uint32_t)0x00000040)
- #define FSMC_PMEM2_MEMSET2_7 ((uint32_t)0x00000080)
- #define FSMC_PMEM2_MEMWAIT2 ((uint32_t)0x0000FF00)
- #define FSMC_PMEM2_MEMWAIT2_0 ((uint32_t)0x00000100)
- #define FSMC_PMEM2_MEMWAIT2_1 ((uint32_t)0x00000200)
- #define FSMC_PMEM2_MEMWAIT2_2 ((uint32_t)0x00000400)
- #define FSMC_PMEM2_MEMWAIT2_3 ((uint32_t)0x00000800)
- #define FSMC_PMEM2_MEMWAIT2_4 ((uint32_t)0x00001000)
- #define FSMC_PMEM2_MEMWAIT2_5 ((uint32_t)0x00002000)
- #define FSMC_PMEM2_MEMWAIT2_6 ((uint32_t)0x00004000)
- #define FSMC_PMEM2_MEMWAIT2_7 ((uint32_t)0x00008000)
- #define FSMC_PMEM2_MEMHOLD2 ((uint32_t)0x00FF0000)
- #define FSMC_PMEM2_MEMHOLD2_0 ((uint32_t)0x00010000)
- #define FSMC_PMEM2_MEMHOLD2_1 ((uint32_t)0x00020000)
- #define FSMC_PMEM2_MEMHOLD2_2 ((uint32_t)0x00040000)
- #define FSMC_PMEM2_MEMHOLD2_3 ((uint32_t)0x00080000)
- #define FSMC_PMEM2_MEMHOLD2_4 ((uint32_t)0x00100000)
- #define FSMC_PMEM2_MEMHOLD2_5 ((uint32_t)0x00200000)
- #define FSMC_PMEM2_MEMHOLD2_6 ((uint32_t)0x00400000)
- #define FSMC_PMEM2_MEMHOLD2_7 ((uint32_t)0x00800000)
- #define FSMC_PMEM2_MEMHIZ2 ((uint32_t)0xFF000000)
- #define FSMC_PMEM2_MEMHIZ2_0 ((uint32_t)0x01000000)
- #define FSMC_PMEM2_MEMHIZ2_1 ((uint32_t)0x02000000)

- #define FSMC_PMEM2_MEMHIZ2_2 ((uint32_t)0x04000000)
- #define FSMC_PMEM2_MEMHIZ2_3 ((uint32_t)0x08000000)
- #define FSMC_PMEM2_MEMHIZ2_4 ((uint32_t)0x10000000)
- #define FSMC_PMEM2_MEMHIZ2_5 ((uint32_t)0x20000000)
- #define FSMC_PMEM2_MEMHIZ2_6 ((uint32_t)0x40000000)
- #define FSMC_PMEM2_MEMHIZ2_7 ((uint32_t)0x80000000)
- #define FSMC_PMEM3_MEMSET3 ((uint32_t)0x000000FF)
- #define FSMC_PMEM3_MEMSET3_0 ((uint32_t)0x00000001)
- #define FSMC_PMEM3_MEMSET3_1 ((uint32_t)0x00000002)
- #define FSMC_PMEM3_MEMSET3_2 ((uint32_t)0x00000004)
- #define FSMC_PMEM3_MEMSET3_3 ((uint32_t)0x00000008)
- #define FSMC_PMEM3_MEMSET3_4 ((uint32_t)0x00000010)
- #define FSMC_PMEM3_MEMSET3_5 ((uint32_t)0x00000020)
- #define FSMC_PMEM3_MEMSET3_6 ((uint32_t)0x00000040)
- #define FSMC_PMEM3_MEMSET3_7 ((uint32_t)0x00000080)
- #define FSMC_PMEM3_MEMWAIT3 ((uint32_t)0x0000FF00)
- #define FSMC_PMEM3_MEMWAIT3_0 ((uint32_t)0x00000100)
- #define FSMC_PMEM3_MEMWAIT3_1 ((uint32_t)0x00000200)
- #define FSMC_PMEM3_MEMWAIT3_2 ((uint32_t)0x00000400)
- #define FSMC_PMEM3_MEMWAIT3_3 ((uint32_t)0x00000800)
- #define FSMC_PMEM3_MEMWAIT3_4 ((uint32_t)0x00001000)
- #define FSMC_PMEM3_MEMWAIT3_5 ((uint32_t)0x00002000)
- #define FSMC_PMEM3_MEMWAIT3_6 ((uint32_t)0x00004000)
- #define FSMC_PMEM3_MEMWAIT3_7 ((uint32_t)0x00008000)
- #define FSMC_PMEM3_MEMHOLD3 ((uint32_t)0x00FF0000)
- #define FSMC_PMEM3_MEMHOLD3_0 ((uint32_t)0x00010000)
- #define FSMC_PMEM3_MEMHOLD3_1 ((uint32_t)0x00020000)
- #define FSMC_PMEM3_MEMHOLD3_2 ((uint32_t)0x00040000)
- #define FSMC_PMEM3_MEMHOLD3_3 ((uint32_t)0x00080000)
- #define FSMC_PMEM3_MEMHOLD3_4 ((uint32_t)0x00100000)
- #define FSMC_PMEM3_MEMHOLD3_5 ((uint32_t)0x00200000)
- #define FSMC_PMEM3_MEMHOLD3_6 ((uint32_t)0x00400000)
- #define FSMC_PMEM3_MEMHOLD3_7 ((uint32_t)0x00800000)
- #define FSMC_PMEM3_MEMHIZ3 ((uint32_t)0xFF000000)
- #define FSMC_PMEM3_MEMHIZ3_0 ((uint32_t)0x01000000)
- #define FSMC_PMEM3_MEMHIZ3_1 ((uint32_t)0x02000000)
- #define FSMC_PMEM3_MEMHIZ3_2 ((uint32_t)0x04000000)
- #define FSMC_PMEM3_MEMHIZ3_3 ((uint32_t)0x08000000)
- #define FSMC_PMEM3_MEMHIZ3_4 ((uint32_t)0x10000000)
- #define FSMC_PMEM3_MEMHIZ3_5 ((uint32_t)0x20000000)
- #define FSMC_PMEM3_MEMHIZ3_6 ((uint32_t)0x40000000)
- #define FSMC_PMEM3_MEMHIZ3_7 ((uint32_t)0x80000000)
- #define FSMC_PMEM4_MEMSET4 ((uint32_t)0x000000FF)
- #define FSMC_PMEM4_MEMSET4_0 ((uint32_t)0x00000001)
- #define FSMC_PMEM4_MEMSET4_1 ((uint32_t)0x00000002)
- #define FSMC_PMEM4_MEMSET4_2 ((uint32_t)0x00000004)
- #define FSMC_PMEM4_MEMSET4_3 ((uint32_t)0x00000008)
- #define FSMC_PMEM4_MEMSET4_4 ((uint32_t)0x00000010)
- #define FSMC_PMEM4_MEMSET4_5 ((uint32_t)0x00000020)
- #define FSMC_PMEM4_MEMSET4_6 ((uint32_t)0x00000040)
- #define FSMC_PMEM4_MEMSET4_7 ((uint32_t)0x00000080)
- #define FSMC_PMEM4_MEMWAIT4 ((uint32_t)0x0000FF00)
- #define FSMC_PMEM4_MEMWAIT4_0 ((uint32_t)0x00000100)
- #define FSMC_PMEM4_MEMWAIT4_1 ((uint32_t)0x00000200)
- #define FSMC_PMEM4_MEMWAIT4_2 ((uint32_t)0x00000400)

- #define FSMC_PMEM4_MEMWAIT4_3 ((uint32_t)0x00000800)
- #define FSMC_PMEM4_MEMWAIT4_4 ((uint32_t)0x00001000)
- #define FSMC_PMEM4_MEMWAIT4_5 ((uint32_t)0x00002000)
- #define FSMC_PMEM4_MEMWAIT4_6 ((uint32_t)0x00004000)
- #define FSMC_PMEM4_MEMWAIT4_7 ((uint32_t)0x00008000)
- #define FSMC_PMEM4_MEMHOLD4 ((uint32_t)0x00FF0000)
- #define FSMC_PMEM4_MEMHOLD4_0 ((uint32_t)0x00010000)
- #define FSMC_PMEM4_MEMHOLD4_1 ((uint32_t)0x00020000)
- #define FSMC_PMEM4_MEMHOLD4_2 ((uint32_t)0x00040000)
- #define FSMC_PMEM4_MEMHOLD4_3 ((uint32_t)0x00080000)
- #define FSMC_PMEM4_MEMHOLD4_4 ((uint32_t)0x00100000)
- #define FSMC_PMEM4_MEMHOLD4_5 ((uint32_t)0x00200000)
- #define FSMC_PMEM4_MEMHOLD4_6 ((uint32_t)0x00400000)
- #define FSMC_PMEM4_MEMHOLD4_7 ((uint32_t)0x00800000)
- #define FSMC_PMEM4_MEMHIZ4 ((uint32_t)0xFF000000)
- #define FSMC_PMEM4_MEMHIZ4_0 ((uint32_t)0x01000000)
- #define FSMC_PMEM4_MEMHIZ4_1 ((uint32_t)0x02000000)
- #define FSMC_PMEM4_MEMHIZ4_2 ((uint32_t)0x04000000)
- #define FSMC_PMEM4_MEMHIZ4_3 ((uint32_t)0x08000000)
- #define FSMC_PMEM4_MEMHIZ4_4 ((uint32_t)0x10000000)
- #define FSMC_PMEM4_MEMHIZ4_5 ((uint32_t)0x20000000)
- #define FSMC_PMEM4_MEMHIZ4_6 ((uint32_t)0x40000000)
- #define FSMC_PMEM4_MEMHIZ4_7 ((uint32_t)0x80000000)
- #define FSMC_PATT2_ATTSET2 ((uint32_t)0x000000FF)
- #define FSMC_PATT2_ATTSET2_0 ((uint32_t)0x00000001)
- #define FSMC_PATT2_ATTSET2_1 ((uint32_t)0x00000002)
- #define FSMC_PATT2_ATTSET2_2 ((uint32_t)0x00000004)
- #define FSMC_PATT2_ATTSET2_3 ((uint32_t)0x00000008)
- #define FSMC_PATT2_ATTSET2_4 ((uint32_t)0x00000010)
- #define FSMC_PATT2_ATTSET2_5 ((uint32_t)0x00000020)
- #define FSMC_PATT2_ATTSET2_6 ((uint32_t)0x00000040)
- #define FSMC_PATT2_ATTSET2_7 ((uint32_t)0x00000080)
- #define FSMC_PATT2_ATTWAIT2 ((uint32_t)0x0000FF00)
- #define FSMC_PATT2_ATTWAIT2_0 ((uint32_t)0x00000100)
- #define FSMC_PATT2_ATTWAIT2_1 ((uint32_t)0x00000200)
- #define FSMC_PATT2_ATTWAIT2_2 ((uint32_t)0x00000400)
- #define FSMC_PATT2_ATTWAIT2_3 ((uint32_t)0x00000800)
- #define FSMC_PATT2_ATTWAIT2_4 ((uint32_t)0x00001000)
- #define FSMC_PATT2_ATTWAIT2_5 ((uint32_t)0x00002000)
- #define FSMC_PATT2_ATTWAIT2_6 ((uint32_t)0x00004000)
- #define FSMC_PATT2_ATTWAIT2_7 ((uint32_t)0x00008000)
- #define FSMC_PATT2_ATTHOLD2 ((uint32_t)0x00FF0000)
- #define FSMC_PATT2_ATTHOLD2_0 ((uint32_t)0x00010000)
- #define FSMC_PATT2_ATTHOLD2_1 ((uint32_t)0x00020000)
- #define FSMC_PATT2_ATTHOLD2_2 ((uint32_t)0x00040000)
- #define FSMC_PATT2_ATTHOLD2_3 ((uint32_t)0x00080000)
- #define FSMC_PATT2_ATTHOLD2_4 ((uint32_t)0x00100000)
- #define FSMC_PATT2_ATTHOLD2_5 ((uint32_t)0x00200000)
- #define FSMC_PATT2_ATTHOLD2_6 ((uint32_t)0x00400000)
- #define FSMC_PATT2_ATTHOLD2_7 ((uint32_t)0x00800000)
- #define FSMC_PATT2_ATTHIZ2 ((uint32_t)0xFF000000)
- #define FSMC_PATT2_ATTHIZ2_0 ((uint32_t)0x01000000)
- #define FSMC_PATT2_ATTHIZ2_1 ((uint32_t)0x02000000)
- #define FSMC_PATT2_ATTHIZ2_2 ((uint32_t)0x04000000)
- #define FSMC_PATT2_ATTHIZ2_3 ((uint32_t)0x08000000)

- #define FSMC_PATT2_ATTHIZ2_4 ((uint32_t)0x10000000)
- #define FSMC_PATT2_ATTHIZ2_5 ((uint32_t)0x20000000)
- #define FSMC_PATT2_ATTHIZ2_6 ((uint32_t)0x40000000)
- #define FSMC_PATT2_ATTHIZ2_7 ((uint32_t)0x80000000)
- #define FSMC_PATT3_ATTSET3 ((uint32_t)0x000000FF)
- #define FSMC_PATT3_ATTSET3_0 ((uint32_t)0x00000001)
- #define FSMC_PATT3_ATTSET3_1 ((uint32_t)0x00000002)
- #define FSMC_PATT3_ATTSET3_2 ((uint32_t)0x00000004)
- #define FSMC_PATT3_ATTSET3_3 ((uint32_t)0x00000008)
- #define FSMC_PATT3_ATTSET3_4 ((uint32_t)0x00000010)
- #define FSMC_PATT3_ATTSET3_5 ((uint32_t)0x00000020)
- #define FSMC_PATT3_ATTSET3_6 ((uint32_t)0x00000040)
- #define FSMC_PATT3_ATTSET3_7 ((uint32_t)0x00000080)
- #define FSMC_PATT3_ATTWAIT3 ((uint32_t)0x0000FF00)
- #define FSMC_PATT3_ATTWAIT3_0 ((uint32_t)0x00000100)
- #define FSMC_PATT3_ATTWAIT3_1 ((uint32_t)0x00000200)
- #define FSMC_PATT3_ATTWAIT3_2 ((uint32_t)0x00000400)
- #define FSMC_PATT3_ATTWAIT3_3 ((uint32_t)0x00000800)
- #define FSMC_PATT3_ATTWAIT3_4 ((uint32_t)0x00001000)
- #define FSMC_PATT3_ATTWAIT3_5 ((uint32_t)0x00002000)
- #define FSMC_PATT3_ATTWAIT3_6 ((uint32_t)0x00004000)
- #define FSMC_PATT3_ATTWAIT3_7 ((uint32_t)0x00008000)
- #define FSMC_PATT3_ATTHOLD3 ((uint32_t)0x00FF0000)
- #define FSMC_PATT3_ATTHOLD3_0 ((uint32_t)0x00010000)
- #define FSMC_PATT3_ATTHOLD3_1 ((uint32_t)0x00020000)
- #define FSMC_PATT3_ATTHOLD3_2 ((uint32_t)0x00040000)
- #define FSMC_PATT3_ATTHOLD3_3 ((uint32_t)0x00080000)
- #define FSMC_PATT3_ATTHOLD3_4 ((uint32_t)0x00100000)
- #define FSMC_PATT3_ATTHOLD3_5 ((uint32_t)0x00200000)
- #define FSMC_PATT3_ATTHOLD3_6 ((uint32_t)0x00400000)
- #define FSMC_PATT3_ATTHOLD3_7 ((uint32_t)0x00800000)
- #define FSMC_PATT3_ATTHIZ3 ((uint32_t)0xFF000000)
- #define FSMC_PATT3_ATTHIZ3_0 ((uint32_t)0x01000000)
- #define FSMC_PATT3_ATTHIZ3_1 ((uint32_t)0x02000000)
- #define FSMC_PATT3_ATTHIZ3_2 ((uint32_t)0x04000000)
- #define FSMC_PATT3_ATTHIZ3_3 ((uint32_t)0x08000000)
- #define FSMC_PATT3_ATTHIZ3_4 ((uint32_t)0x10000000)
- #define FSMC_PATT3_ATTHIZ3_5 ((uint32_t)0x20000000)
- #define FSMC_PATT3_ATTHIZ3_6 ((uint32_t)0x40000000)
- #define FSMC_PATT3_ATTHIZ3_7 ((uint32_t)0x80000000)
- #define FSMC_PATT4_ATTSET4 ((uint32_t)0x000000FF)
- #define FSMC_PATT4_ATTSET4_0 ((uint32_t)0x00000001)
- #define FSMC_PATT4_ATTSET4_1 ((uint32_t)0x00000002)
- #define FSMC_PATT4_ATTSET4_2 ((uint32_t)0x00000004)
- #define FSMC_PATT4_ATTSET4_3 ((uint32_t)0x00000008)
- #define FSMC_PATT4_ATTSET4_4 ((uint32_t)0x00000010)
- #define FSMC_PATT4_ATTSET4_5 ((uint32_t)0x00000020)
- #define FSMC_PATT4_ATTSET4_6 ((uint32_t)0x00000040)
- #define FSMC_PATT4_ATTSET4_7 ((uint32_t)0x00000080)
- #define FSMC_PATT4_ATTWAIT4 ((uint32_t)0x0000FF00)
- #define FSMC_PATT4_ATTWAIT4_0 ((uint32_t)0x00000100)
- #define FSMC_PATT4_ATTWAIT4_1 ((uint32_t)0x00000200)
- #define FSMC_PATT4_ATTWAIT4_2 ((uint32_t)0x00000400)
- #define FSMC_PATT4_ATTWAIT4_3 ((uint32_t)0x00000800)
- #define FSMC_PATT4_ATTWAIT4_4 ((uint32_t)0x00001000)

- #define FSMC_PATT4_ATTWAIT4_5 ((uint32_t)0x00002000)
- #define FSMC_PATT4_ATTWAIT4_6 ((uint32_t)0x00004000)
- #define FSMC_PATT4_ATTWAIT4_7 ((uint32_t)0x00008000)
- #define FSMC_PATT4_ATTHOLD4 ((uint32_t)0x00FF0000)
- #define FSMC_PATT4_ATTHOLD4_0 ((uint32_t)0x00010000)
- #define FSMC_PATT4_ATTHOLD4_1 ((uint32_t)0x00020000)
- #define FSMC_PATT4_ATTHOLD4_2 ((uint32_t)0x00040000)
- #define FSMC_PATT4_ATTHOLD4_3 ((uint32_t)0x00080000)
- #define FSMC_PATT4_ATTHOLD4_4 ((uint32_t)0x00100000)
- #define FSMC_PATT4_ATTHOLD4_5 ((uint32_t)0x00200000)
- #define FSMC_PATT4_ATTHOLD4_6 ((uint32_t)0x00400000)
- #define FSMC_PATT4_ATTHOLD4_7 ((uint32_t)0x00800000)
- #define FSMC_PATT4_ATTHIZ4 ((uint32_t)0xFF000000)
- #define FSMC_PATT4_ATTHIZ4_0 ((uint32_t)0x01000000)
- #define FSMC_PATT4_ATTHIZ4_1 ((uint32_t)0x02000000)
- #define FSMC_PATT4_ATTHIZ4_2 ((uint32_t)0x04000000)
- #define FSMC_PATT4_ATTHIZ4_3 ((uint32_t)0x08000000)
- #define FSMC_PATT4_ATTHIZ4_4 ((uint32_t)0x10000000)
- #define FSMC_PATT4_ATTHIZ4_5 ((uint32_t)0x20000000)
- #define FSMC_PATT4_ATTHIZ4_6 ((uint32_t)0x40000000)
- #define FSMC_PATT4_ATTHIZ4_7 ((uint32_t)0x80000000)
- #define FSMC_PIO4_IOSET4 ((uint32_t)0x000000FF)
- #define FSMC_PIO4_IOSET4_0 ((uint32_t)0x00000001)
- #define FSMC_PIO4_IOSET4_1 ((uint32_t)0x00000002)
- #define FSMC_PIO4_IOSET4_2 ((uint32_t)0x00000004)
- #define FSMC_PIO4_IOSET4_3 ((uint32_t)0x00000008)
- #define FSMC_PIO4_IOSET4_4 ((uint32_t)0x00000010)
- #define FSMC_PIO4_IOSET4_5 ((uint32_t)0x00000020)
- #define FSMC_PIO4_IOSET4_6 ((uint32_t)0x00000040)
- #define FSMC_PIO4_IOSET4_7 ((uint32_t)0x00000080)
- #define FSMC_PIO4_IOWAIT4 ((uint32_t)0x0000FF00)
- #define FSMC_PIO4_IOWAIT4_0 ((uint32_t)0x00000100)
- #define FSMC_PIO4_IOWAIT4_1 ((uint32_t)0x00000200)
- #define FSMC_PIO4_IOWAIT4_2 ((uint32_t)0x00000400)
- #define FSMC_PIO4_IOWAIT4_3 ((uint32_t)0x00000800)
- #define FSMC_PIO4_IOWAIT4_4 ((uint32_t)0x00001000)
- #define FSMC_PIO4_IOWAIT4_5 ((uint32_t)0x00002000)
- #define FSMC_PIO4_IOWAIT4_6 ((uint32_t)0x00004000)
- #define FSMC_PIO4_IOWAIT4_7 ((uint32_t)0x00008000)
- #define FSMC_PIO4_IOHOLD4 ((uint32_t)0x00FF0000)
- #define FSMC_PIO4_IOHOLD4_0 ((uint32_t)0x00010000)
- #define FSMC_PIO4_IOHOLD4_1 ((uint32_t)0x00020000)
- #define FSMC_PIO4_IOHOLD4_2 ((uint32_t)0x00040000)
- #define FSMC_PIO4_IOHOLD4_3 ((uint32_t)0x00080000)
- #define FSMC_PIO4_IOHOLD4_4 ((uint32_t)0x00100000)
- #define FSMC_PIO4_IOHOLD4_5 ((uint32_t)0x00200000)
- #define FSMC_PIO4_IOHOLD4_6 ((uint32_t)0x00400000)
- #define FSMC_PIO4_IOHOLD4_7 ((uint32_t)0x00800000)
- #define FSMC_PIO4_IOHIZ4 ((uint32_t)0xFF000000)
- #define FSMC_PIO4_IOHIZ4_0 ((uint32_t)0x01000000)
- #define FSMC_PIO4_IOHIZ4_1 ((uint32_t)0x02000000)
- #define FSMC_PIO4_IOHIZ4_2 ((uint32_t)0x04000000)
- #define FSMC_PIO4_IOHIZ4_3 ((uint32_t)0x08000000)
- #define FSMC_PIO4_IOHIZ4_4 ((uint32_t)0x10000000)
- #define FSMC_PIO4_IOHIZ4_5 ((uint32_t)0x20000000)

- #define FSMC_PIO4_IOHIZ4_6 ((uint32_t)0x40000000)
- #define FSMC_PIO4_IOHIZ4_7 ((uint32_t)0x80000000)
- #define FSMC_ECCR2_ECC2 ((uint32_t)0xFFFFFFFF)
- #define FSMC_ECCR3_ECC3 ((uint32_t)0xFFFFFFFF)
- #define **GPIO_MODER_MODER0** ((uint32_t)0x00000003)
- #define **GPIO_MODER_MODER0_0** ((uint32_t)0x00000001)
- #define **GPIO_MODER_MODER0_1** ((uint32_t)0x00000002)
- #define **GPIO_MODER_MODER1** ((uint32_t)0x0000000C)
- #define **GPIO_MODER_MODER1_0** ((uint32_t)0x00000004)
- #define **GPIO_MODER_MODER1_1** ((uint32_t)0x00000008)
- #define **GPIO_MODER_MODER2** ((uint32_t)0x00000030)
- #define **GPIO_MODER_MODER2_0** ((uint32_t)0x00000010)
- #define **GPIO_MODER_MODER2_1** ((uint32_t)0x00000020)
- #define **GPIO_MODER_MODER3** ((uint32_t)0x000000C0)
- #define **GPIO_MODER_MODER3_0** ((uint32_t)0x00000040)
- #define **GPIO_MODER_MODER3_1** ((uint32_t)0x00000080)
- #define **GPIO_MODER_MODER4** ((uint32_t)0x00000300)
- #define **GPIO_MODER_MODER4_0** ((uint32_t)0x00000100)
- #define **GPIO_MODER_MODER4_1** ((uint32_t)0x00000200)
- #define **GPIO_MODER_MODER5** ((uint32_t)0x00000C00)
- #define **GPIO_MODER_MODER5_0** ((uint32_t)0x00000400)
- #define **GPIO_MODER_MODER5_1** ((uint32_t)0x00000800)
- #define **GPIO_MODER_MODER6** ((uint32_t)0x00003000)
- #define **GPIO_MODER_MODER6_0** ((uint32_t)0x00001000)
- #define **GPIO_MODER_MODER6_1** ((uint32_t)0x00002000)
- #define **GPIO_MODER_MODER7** ((uint32_t)0x0000C000)
- #define **GPIO_MODER_MODER7_0** ((uint32_t)0x00004000)
- #define **GPIO_MODER_MODER7_1** ((uint32_t)0x00008000)
- #define **GPIO_MODER_MODER8** ((uint32_t)0x00030000)
- #define **GPIO_MODER_MODER8_0** ((uint32_t)0x00010000)
- #define **GPIO_MODER_MODER8_1** ((uint32_t)0x00020000)
- #define **GPIO_MODER_MODER9** ((uint32_t)0x000C0000)
- #define **GPIO_MODER_MODER9_0** ((uint32_t)0x00040000)
- #define **GPIO_MODER_MODER9_1** ((uint32_t)0x00080000)
- #define **GPIO_MODER_MODER10** ((uint32_t)0x00300000)
- #define **GPIO_MODER_MODER10_0** ((uint32_t)0x00100000)
- #define **GPIO_MODER_MODER10_1** ((uint32_t)0x00200000)
- #define **GPIO_MODER_MODER11** ((uint32_t)0x00C00000)
- #define **GPIO_MODER_MODER11_0** ((uint32_t)0x00400000)
- #define **GPIO_MODER_MODER11_1** ((uint32_t)0x00800000)
- #define **GPIO_MODER_MODER12** ((uint32_t)0x03000000)
- #define **GPIO_MODER_MODER12_0** ((uint32_t)0x01000000)
- #define **GPIO_MODER_MODER12_1** ((uint32_t)0x02000000)
- #define **GPIO_MODER_MODER13** ((uint32_t)0x0C000000)
- #define **GPIO_MODER_MODER13_0** ((uint32_t)0x04000000)
- #define **GPIO_MODER_MODER13_1** ((uint32_t)0x08000000)
- #define **GPIO_MODER_MODER14** ((uint32_t)0x30000000)
- #define **GPIO_MODER_MODER14_0** ((uint32_t)0x10000000)
- #define **GPIO_MODER_MODER14_1** ((uint32_t)0x20000000)
- #define **GPIO_MODER_MODER15** ((uint32_t)0xC0000000)
- #define **GPIO_MODER_MODER15_0** ((uint32_t)0x40000000)
- #define **GPIO_MODER_MODER15_1** ((uint32_t)0x80000000)
- #define **GPIO_OTYPER_OT_0** ((uint32_t)0x00000001)
- #define **GPIO_OTYPER_OT_1** ((uint32_t)0x00000002)
- #define **GPIO_OTYPER_OT_2** ((uint32_t)0x00000004)

- #define **GPIO_OTYPER_OT_3** ((uint32_t)0x00000008)
- #define **GPIO_OTYPER_OT_4** ((uint32_t)0x00000010)
- #define **GPIO_OTYPER_OT_5** ((uint32_t)0x00000020)
- #define **GPIO_OTYPER_OT_6** ((uint32_t)0x00000040)
- #define **GPIO_OTYPER_OT_7** ((uint32_t)0x00000080)
- #define **GPIO_OTYPER_OT_8** ((uint32_t)0x00000100)
- #define **GPIO_OTYPER_OT_9** ((uint32_t)0x00000200)
- #define **GPIO_OTYPER_OT_10** ((uint32_t)0x00000400)
- #define **GPIO_OTYPER_OT_11** ((uint32_t)0x00000800)
- #define **GPIO_OTYPER_OT_12** ((uint32_t)0x00001000)
- #define **GPIO_OTYPER_OT_13** ((uint32_t)0x00002000)
- #define **GPIO_OTYPER_OT_14** ((uint32_t)0x00004000)
- #define **GPIO_OTYPER_OT_15** ((uint32_t)0x00008000)
- #define **GPIO_OSPEEDER_OSPEEDR0** ((uint32_t)0x00000003)
- #define **GPIO_OSPEEDER_OSPEEDR0_0** ((uint32_t)0x00000001)
- #define **GPIO_OSPEEDER_OSPEEDR0_1** ((uint32_t)0x00000002)
- #define **GPIO_OSPEEDER_OSPEEDR1** ((uint32_t)0x0000000C)
- #define **GPIO_OSPEEDER_OSPEEDR1_0** ((uint32_t)0x00000004)
- #define **GPIO_OSPEEDER_OSPEEDR1_1** ((uint32_t)0x00000008)
- #define **GPIO_OSPEEDER_OSPEEDR2** ((uint32_t)0x00000030)
- #define **GPIO_OSPEEDER_OSPEEDR2_0** ((uint32_t)0x00000010)
- #define **GPIO_OSPEEDER_OSPEEDR2_1** ((uint32_t)0x00000020)
- #define **GPIO_OSPEEDER_OSPEEDR3** ((uint32_t)0x000000C0)
- #define **GPIO_OSPEEDER_OSPEEDR3_0** ((uint32_t)0x00000040)
- #define **GPIO_OSPEEDER_OSPEEDR3_1** ((uint32_t)0x00000080)
- #define **GPIO_OSPEEDER_OSPEEDR4** ((uint32_t)0x00000300)
- #define **GPIO_OSPEEDER_OSPEEDR4_0** ((uint32_t)0x00000100)
- #define **GPIO_OSPEEDER_OSPEEDR4_1** ((uint32_t)0x00000200)
- #define **GPIO_OSPEEDER_OSPEEDR5** ((uint32_t)0x00000C00)
- #define **GPIO_OSPEEDER_OSPEEDR5_0** ((uint32_t)0x00000400)
- #define **GPIO_OSPEEDER_OSPEEDR5_1** ((uint32_t)0x00000800)
- #define **GPIO_OSPEEDER_OSPEEDR6** ((uint32_t)0x00003000)
- #define **GPIO_OSPEEDER_OSPEEDR6_0** ((uint32_t)0x00001000)
- #define **GPIO_OSPEEDER_OSPEEDR6_1** ((uint32_t)0x00002000)
- #define **GPIO_OSPEEDER_OSPEEDR7** ((uint32_t)0x0000C000)
- #define **GPIO_OSPEEDER_OSPEEDR7_0** ((uint32_t)0x00004000)
- #define **GPIO_OSPEEDER_OSPEEDR7_1** ((uint32_t)0x00008000)
- #define **GPIO_OSPEEDER_OSPEEDR8** ((uint32_t)0x00030000)
- #define **GPIO_OSPEEDER_OSPEEDR8_0** ((uint32_t)0x00010000)
- #define **GPIO_OSPEEDER_OSPEEDR8_1** ((uint32_t)0x00020000)
- #define **GPIO_OSPEEDER_OSPEEDR9** ((uint32_t)0x000C0000)
- #define **GPIO_OSPEEDER_OSPEEDR9_0** ((uint32_t)0x00040000)
- #define **GPIO_OSPEEDER_OSPEEDR9_1** ((uint32_t)0x00080000)
- #define **GPIO_OSPEEDER_OSPEEDR10** ((uint32_t)0x00300000)
- #define **GPIO_OSPEEDER_OSPEEDR10_0** ((uint32_t)0x00100000)
- #define **GPIO_OSPEEDER_OSPEEDR10_1** ((uint32_t)0x00200000)
- #define **GPIO_OSPEEDER_OSPEEDR11** ((uint32_t)0x00C00000)
- #define **GPIO_OSPEEDER_OSPEEDR11_0** ((uint32_t)0x00400000)
- #define **GPIO_OSPEEDER_OSPEEDR11_1** ((uint32_t)0x00800000)
- #define **GPIO_OSPEEDER_OSPEEDR12** ((uint32_t)0x03000000)
- #define **GPIO_OSPEEDER_OSPEEDR12_0** ((uint32_t)0x01000000)
- #define **GPIO_OSPEEDER_OSPEEDR12_1** ((uint32_t)0x02000000)
- #define **GPIO_OSPEEDER_OSPEEDR13** ((uint32_t)0x0C000000)
- #define **GPIO_OSPEEDER_OSPEEDR13_0** ((uint32_t)0x04000000)
- #define **GPIO_OSPEEDER_OSPEEDR13_1** ((uint32_t)0x08000000)

- #define **GPIO_OSPEEDER_OSPEEDR14** ((uint32_t)0x30000000)
- #define **GPIO_OSPEEDER_OSPEEDR14_0** ((uint32_t)0x10000000)
- #define **GPIO_OSPEEDER_OSPEEDR14_1** ((uint32_t)0x20000000)
- #define **GPIO_OSPEEDER_OSPEEDR15** ((uint32_t)0xC0000000)
- #define **GPIO_OSPEEDER_OSPEEDR15_0** ((uint32_t)0x40000000)
- #define **GPIO_OSPEEDER_OSPEEDR15_1** ((uint32_t)0x80000000)
- #define **GPIO_PUPDR_PUPDR0** ((uint32_t)0x00000003)
- #define **GPIO_PUPDR_PUPDR0_0** ((uint32_t)0x00000001)
- #define **GPIO_PUPDR_PUPDR0_1** ((uint32_t)0x00000002)
- #define **GPIO_PUPDR_PUPDR1** ((uint32_t)0x0000000C)
- #define **GPIO_PUPDR_PUPDR1_0** ((uint32_t)0x00000004)
- #define **GPIO_PUPDR_PUPDR1_1** ((uint32_t)0x00000008)
- #define **GPIO_PUPDR_PUPDR2** ((uint32_t)0x00000030)
- #define **GPIO_PUPDR_PUPDR2_0** ((uint32_t)0x00000010)
- #define **GPIO_PUPDR_PUPDR2_1** ((uint32_t)0x00000020)
- #define **GPIO_PUPDR_PUPDR3** ((uint32_t)0x000000C0)
- #define **GPIO_PUPDR_PUPDR3_0** ((uint32_t)0x00000040)
- #define **GPIO_PUPDR_PUPDR3_1** ((uint32_t)0x00000080)
- #define **GPIO_PUPDR_PUPDR4** ((uint32_t)0x00000300)
- #define **GPIO_PUPDR_PUPDR4_0** ((uint32_t)0x00000100)
- #define **GPIO_PUPDR_PUPDR4_1** ((uint32_t)0x00000200)
- #define **GPIO_PUPDR_PUPDR5** ((uint32_t)0x00000C00)
- #define **GPIO_PUPDR_PUPDR5_0** ((uint32_t)0x00000400)
- #define **GPIO_PUPDR_PUPDR5_1** ((uint32_t)0x00000800)
- #define **GPIO_PUPDR_PUPDR6** ((uint32_t)0x00003000)
- #define **GPIO_PUPDR_PUPDR6_0** ((uint32_t)0x00001000)
- #define **GPIO_PUPDR_PUPDR6_1** ((uint32_t)0x00002000)
- #define **GPIO_PUPDR_PUPDR7** ((uint32_t)0x0000C000)
- #define **GPIO_PUPDR_PUPDR7_0** ((uint32_t)0x00004000)
- #define **GPIO_PUPDR_PUPDR7_1** ((uint32_t)0x00008000)
- #define **GPIO_PUPDR_PUPDR8** ((uint32_t)0x00030000)
- #define **GPIO_PUPDR_PUPDR8_0** ((uint32_t)0x00010000)
- #define **GPIO_PUPDR_PUPDR8_1** ((uint32_t)0x00020000)
- #define **GPIO_PUPDR_PUPDR9** ((uint32_t)0x000C0000)
- #define **GPIO_PUPDR_PUPDR9_0** ((uint32_t)0x00040000)
- #define **GPIO_PUPDR_PUPDR9_1** ((uint32_t)0x00080000)
- #define **GPIO_PUPDR_PUPDR10** ((uint32_t)0x00300000)
- #define **GPIO_PUPDR_PUPDR10_0** ((uint32_t)0x00100000)
- #define **GPIO_PUPDR_PUPDR10_1** ((uint32_t)0x00200000)
- #define **GPIO_PUPDR_PUPDR11** ((uint32_t)0x00C00000)
- #define **GPIO_PUPDR_PUPDR11_0** ((uint32_t)0x00400000)
- #define **GPIO_PUPDR_PUPDR11_1** ((uint32_t)0x00800000)
- #define **GPIO_PUPDR_PUPDR12** ((uint32_t)0x03000000)
- #define **GPIO_PUPDR_PUPDR12_0** ((uint32_t)0x01000000)
- #define **GPIO_PUPDR_PUPDR12_1** ((uint32_t)0x02000000)
- #define **GPIO_PUPDR_PUPDR13** ((uint32_t)0x0C000000)
- #define **GPIO_PUPDR_PUPDR13_0** ((uint32_t)0x04000000)
- #define **GPIO_PUPDR_PUPDR13_1** ((uint32_t)0x08000000)
- #define **GPIO_PUPDR_PUPDR14** ((uint32_t)0x30000000)
- #define **GPIO_PUPDR_PUPDR14_0** ((uint32_t)0x10000000)
- #define **GPIO_PUPDR_PUPDR14_1** ((uint32_t)0x20000000)
- #define **GPIO_PUPDR_PUPDR15** ((uint32_t)0xC0000000)
- #define **GPIO_PUPDR_PUPDR15_0** ((uint32_t)0x40000000)
- #define **GPIO_PUPDR_PUPDR15_1** ((uint32_t)0x80000000)
- #define **GPIO_IDR_IDR_0** ((uint32_t)0x00000001)

- #define **GPIO_IDR_IDR_1** ((uint32_t)0x00000002)
- #define **GPIO_IDR_IDR_2** ((uint32_t)0x00000004)
- #define **GPIO_IDR_IDR_3** ((uint32_t)0x00000008)
- #define **GPIO_IDR_IDR_4** ((uint32_t)0x00000010)
- #define **GPIO_IDR_IDR_5** ((uint32_t)0x00000020)
- #define **GPIO_IDR_IDR_6** ((uint32_t)0x00000040)
- #define **GPIO_IDR_IDR_7** ((uint32_t)0x00000080)
- #define **GPIO_IDR_IDR_8** ((uint32_t)0x00000100)
- #define **GPIO_IDR_IDR_9** ((uint32_t)0x00000200)
- #define **GPIO_IDR_IDR_10** ((uint32_t)0x00000400)
- #define **GPIO_IDR_IDR_11** ((uint32_t)0x00000800)
- #define **GPIO_IDR_IDR_12** ((uint32_t)0x00001000)
- #define **GPIO_IDR_IDR_13** ((uint32_t)0x00002000)
- #define **GPIO_IDR_IDR_14** ((uint32_t)0x00004000)
- #define **GPIO_IDR_IDR_15** ((uint32_t)0x00008000)
- #define **GPIO_OTYPER_IDR_0** GPIO_IDR_IDR_0
- #define **GPIO_OTYPER_IDR_1** GPIO_IDR_IDR_1
- #define **GPIO_OTYPER_IDR_2** GPIO_IDR_IDR_2
- #define **GPIO_OTYPER_IDR_3** GPIO_IDR_IDR_3
- #define **GPIO_OTYPER_IDR_4** GPIO_IDR_IDR_4
- #define **GPIO_OTYPER_IDR_5** GPIO_IDR_IDR_5
- #define **GPIO_OTYPER_IDR_6** GPIO_IDR_IDR_6
- #define **GPIO_OTYPER_IDR_7** GPIO_IDR_IDR_7
- #define **GPIO_OTYPER_IDR_8** GPIO_IDR_IDR_8
- #define **GPIO_OTYPER_IDR_9** GPIO_IDR_IDR_9
- #define **GPIO_OTYPER_IDR_10** GPIO_IDR_IDR_10
- #define **GPIO_OTYPER_IDR_11** GPIO_IDR_IDR_11
- #define **GPIO_OTYPER_IDR_12** GPIO_IDR_IDR_12
- #define **GPIO_OTYPER_IDR_13** GPIO_IDR_IDR_13
- #define **GPIO_OTYPER_IDR_14** GPIO_IDR_IDR_14
- #define **GPIO_OTYPER_IDR_15** GPIO_IDR_IDR_15
- #define **GPIO_ODR_ODR_0** ((uint32_t)0x00000001)
- #define **GPIO_ODR_ODR_1** ((uint32_t)0x00000002)
- #define **GPIO_ODR_ODR_2** ((uint32_t)0x00000004)
- #define **GPIO_ODR_ODR_3** ((uint32_t)0x00000008)
- #define **GPIO_ODR_ODR_4** ((uint32_t)0x00000010)
- #define **GPIO_ODR_ODR_5** ((uint32_t)0x00000020)
- #define **GPIO_ODR_ODR_6** ((uint32_t)0x00000040)
- #define **GPIO_ODR_ODR_7** ((uint32_t)0x00000080)
- #define **GPIO_ODR_ODR_8** ((uint32_t)0x00000100)
- #define **GPIO_ODR_ODR_9** ((uint32_t)0x00000200)
- #define **GPIO_ODR_ODR_10** ((uint32_t)0x00000400)
- #define **GPIO_ODR_ODR_11** ((uint32_t)0x00000800)
- #define **GPIO_ODR_ODR_12** ((uint32_t)0x00001000)
- #define **GPIO_ODR_ODR_13** ((uint32_t)0x00002000)
- #define **GPIO_ODR_ODR_14** ((uint32_t)0x00004000)
- #define **GPIO_ODR_ODR_15** ((uint32_t)0x00008000)
- #define **GPIO_OTYPER_ODR_0** GPIO_ODR_ODR_0
- #define **GPIO_OTYPER_ODR_1** GPIO_ODR_ODR_1
- #define **GPIO_OTYPER_ODR_2** GPIO_ODR_ODR_2
- #define **GPIO_OTYPER_ODR_3** GPIO_ODR_ODR_3
- #define **GPIO_OTYPER_ODR_4** GPIO_ODR_ODR_4
- #define **GPIO_OTYPER_ODR_5** GPIO_ODR_ODR_5
- #define **GPIO_OTYPER_ODR_6** GPIO_ODR_ODR_6
- #define **GPIO_OTYPER_ODR_7** GPIO_ODR_ODR_7

- #define **GPIO_OTYPER_ODR_8** GPIO_ODR_ODR_8
- #define **GPIO_OTYPER_ODR_9** GPIO_ODR_ODR_9
- #define **GPIO_OTYPER_ODR_10** GPIO_ODR_ODR_10
- #define **GPIO_OTYPER_ODR_11** GPIO_ODR_ODR_11
- #define **GPIO_OTYPER_ODR_12** GPIO_ODR_ODR_12
- #define **GPIO_OTYPER_ODR_13** GPIO_ODR_ODR_13
- #define **GPIO_OTYPER_ODR_14** GPIO_ODR_ODR_14
- #define **GPIO_OTYPER_ODR_15** GPIO_ODR_ODR_15
- #define **GPIO_BSRR_BS_0** ((uint32_t)0x00000001)
- #define **GPIO_BSRR_BS_1** ((uint32_t)0x00000002)
- #define **GPIO_BSRR_BS_2** ((uint32_t)0x00000004)
- #define **GPIO_BSRR_BS_3** ((uint32_t)0x00000008)
- #define **GPIO_BSRR_BS_4** ((uint32_t)0x00000010)
- #define **GPIO_BSRR_BS_5** ((uint32_t)0x00000020)
- #define **GPIO_BSRR_BS_6** ((uint32_t)0x00000040)
- #define **GPIO_BSRR_BS_7** ((uint32_t)0x00000080)
- #define **GPIO_BSRR_BS_8** ((uint32_t)0x00000100)
- #define **GPIO_BSRR_BS_9** ((uint32_t)0x00000200)
- #define **GPIO_BSRR_BS_10** ((uint32_t)0x00000400)
- #define **GPIO_BSRR_BS_11** ((uint32_t)0x00000800)
- #define **GPIO_BSRR_BS_12** ((uint32_t)0x00001000)
- #define **GPIO_BSRR_BS_13** ((uint32_t)0x00002000)
- #define **GPIO_BSRR_BS_14** ((uint32_t)0x00004000)
- #define **GPIO_BSRR_BS_15** ((uint32_t)0x00008000)
- #define **GPIO_BSRR_BR_0** ((uint32_t)0x00010000)
- #define **GPIO_BSRR_BR_1** ((uint32_t)0x00020000)
- #define **GPIO_BSRR_BR_2** ((uint32_t)0x00040000)
- #define **GPIO_BSRR_BR_3** ((uint32_t)0x00080000)
- #define **GPIO_BSRR_BR_4** ((uint32_t)0x00100000)
- #define **GPIO_BSRR_BR_5** ((uint32_t)0x00200000)
- #define **GPIO_BSRR_BR_6** ((uint32_t)0x00400000)
- #define **GPIO_BSRR_BR_7** ((uint32_t)0x00800000)
- #define **GPIO_BSRR_BR_8** ((uint32_t)0x01000000)
- #define **GPIO_BSRR_BR_9** ((uint32_t)0x02000000)
- #define **GPIO_BSRR_BR_10** ((uint32_t)0x04000000)
- #define **GPIO_BSRR_BR_11** ((uint32_t)0x08000000)
- #define **GPIO_BSRR_BR_12** ((uint32_t)0x10000000)
- #define **GPIO_BSRR_BR_13** ((uint32_t)0x20000000)
- #define **GPIO_BSRR_BR_14** ((uint32_t)0x40000000)
- #define **GPIO_BSRR_BR_15** ((uint32_t)0x80000000)
- #define **HASH_CR_INIT** ((uint32_t)0x00000004)
- #define **HASH_CR_DMAE** ((uint32_t)0x00000008)
- #define **HASH_CR_DATATYPE** ((uint32_t)0x00000030)
- #define **HASH_CR_DATATYPE_0** ((uint32_t)0x00000010)
- #define **HASH_CR_DATATYPE_1** ((uint32_t)0x00000020)
- #define **HASH_CR_MODE** ((uint32_t)0x00000040)
- #define **HASH_CR_ALGO** ((uint32_t)0x00000080)
- #define **HASH_CR_NBW** ((uint32_t)0x00000F00)
- #define **HASH_CR_NBW_0** ((uint32_t)0x00000100)
- #define **HASH_CR_NBW_1** ((uint32_t)0x00000200)
- #define **HASH_CR_NBW_2** ((uint32_t)0x00000400)
- #define **HASH_CR_NBW_3** ((uint32_t)0x00000800)
- #define **HASH_CR_DINNE** ((uint32_t)0x00001000)
- #define **HASH_CR_LKEY** ((uint32_t)0x00010000)
- #define **HASH_STR_NBW** ((uint32_t)0x0000001F)

- #define **HASH_STR_NBW_0** ((uint32_t)0x00000001)
- #define **HASH_STR_NBW_1** ((uint32_t)0x00000002)
- #define **HASH_STR_NBW_2** ((uint32_t)0x00000004)
- #define **HASH_STR_NBW_3** ((uint32_t)0x00000008)
- #define **HASH_STR_NBW_4** ((uint32_t)0x00000010)
- #define **HASH_STR_DCAL** ((uint32_t)0x00000100)
- #define **HASH_IMR_DINIM** ((uint32_t)0x00000001)
- #define **HASH_IMR_DCIM** ((uint32_t)0x00000002)
- #define **HASH_SR_DINIS** ((uint32_t)0x00000001)
- #define **HASH_SR_DCIS** ((uint32_t)0x00000002)
- #define **HASH_SR_DMAS** ((uint32_t)0x00000004)
- #define **HASH_SR_BUSY** ((uint32_t)0x00000008)
- #define I2C_CR1_PE ((uint16_t)0x0001)
- #define I2C_CR1_SMBUS ((uint16_t)0x0002)
- #define I2C_CR1_SMBTYPE ((uint16_t)0x0008)
- #define I2C_CR1_ENARP ((uint16_t)0x0010)
- #define I2C_CR1_ENPEC ((uint16_t)0x0020)
- #define I2C_CR1_ENGC ((uint16_t)0x0040)
- #define I2C_CR1_NOSTRETCH ((uint16_t)0x0080)
- #define I2C_CR1_START ((uint16_t)0x0100)
- #define I2C_CR1_STOP ((uint16_t)0x0200)
- #define I2C_CR1_ACK ((uint16_t)0x0400)
- #define I2C_CR1_POS ((uint16_t)0x0800)
- #define I2C_CR1_PEC ((uint16_t)0x1000)
- #define I2C_CR1_ALERT ((uint16_t)0x2000)
- #define I2C_CR1_SWRST ((uint16_t)0x8000)
- #define I2C_CR2_FREQ ((uint16_t)0x003F)
- #define I2C_CR2_FREQ_0 ((uint16_t)0x0001)
- #define I2C_CR2_FREQ_1 ((uint16_t)0x0002)
- #define I2C_CR2_FREQ_2 ((uint16_t)0x0004)
- #define I2C_CR2_FREQ_3 ((uint16_t)0x0008)
- #define I2C_CR2_FREQ_4 ((uint16_t)0x0010)
- #define I2C_CR2_FREQ_5 ((uint16_t)0x0020)
- #define I2C_CR2_ITERREN ((uint16_t)0x0100)
- #define I2C_CR2_ITEVTEN ((uint16_t)0x0200)
- #define I2C_CR2_ITBUFEN ((uint16_t)0x0400)
- #define I2C_CR2_DMAEN ((uint16_t)0x0800)
- #define I2C_CR2_LAST ((uint16_t)0x1000)
- #define I2C_OAR1_ADD1_7 ((uint16_t)0x00FE)
- #define I2C_OAR1_ADD8_9 ((uint16_t)0x0300)
- #define I2C_OAR1_ADD0 ((uint16_t)0x0001)
- #define I2C_OAR1_ADD1 ((uint16_t)0x0002)
- #define I2C_OAR1_ADD2 ((uint16_t)0x0004)
- #define I2C_OAR1_ADD3 ((uint16_t)0x0008)
- #define I2C_OAR1_ADD4 ((uint16_t)0x0010)
- #define I2C_OAR1_ADD5 ((uint16_t)0x0020)
- #define I2C_OAR1_ADD6 ((uint16_t)0x0040)
- #define I2C_OAR1_ADD7 ((uint16_t)0x0080)
- #define I2C_OAR1_ADD8 ((uint16_t)0x0100)
- #define I2C_OAR1_ADD9 ((uint16_t)0x0200)
- #define I2C_OAR1_ADDMODE ((uint16_t)0x8000)
- #define I2C_OAR2_ENDUAL ((uint8_t)0x01)
- #define I2C_OAR2_ADD2 ((uint8_t)0xFE)
- #define I2C_DR_DR ((uint8_t)0xFF)
- #define I2C_SR1_SB ((uint16_t)0x0001)

- #define I2C_SR1_ADDR ((uint16_t)0x0002)
- #define I2C_SR1_BTF ((uint16_t)0x0004)
- #define I2C_SR1_ADD10 ((uint16_t)0x0008)
- #define I2C_SR1_STOPF ((uint16_t)0x0010)
- #define I2C_SR1_RXNE ((uint16_t)0x0040)
- #define I2C_SR1_TXE ((uint16_t)0x0080)
- #define I2C_SR1_BERR ((uint16_t)0x0100)
- #define I2C_SR1_ARLO ((uint16_t)0x0200)
- #define I2C_SR1_AF ((uint16_t)0x0400)
- #define I2C_SR1_OVR ((uint16_t)0x0800)
- #define I2C_SR1_PECERR ((uint16_t)0x1000)
- #define I2C_SR1_TIMEOUT ((uint16_t)0x4000)
- #define I2C_SR1_SMBALERT ((uint16_t)0x8000)
- #define I2C_SR2_MSL ((uint16_t)0x0001)
- #define I2C_SR2_BUSY ((uint16_t)0x0002)
- #define I2C_SR2_TRA ((uint16_t)0x0004)
- #define I2C_SR2_GENCALL ((uint16_t)0x0010)
- #define I2C_SR2_SMBDEFAULT ((uint16_t)0x0020)
- #define I2C_SR2_SMBHOST ((uint16_t)0x0040)
- #define I2C_SR2_DUALF ((uint16_t)0x0080)
- #define I2C_SR2_PEC ((uint16_t)0xFF00)
- #define I2C_CCR_CCR ((uint16_t)0x0FFF)
- #define I2C_CCR_DUTY ((uint16_t)0x4000)
- #define I2C_CCR_FS ((uint16_t)0x8000)
- #define I2C_TRISE_TRISE ((uint8_t)0x3F)
- #define IWDG_KR_KEY ((uint16_t)0xFFFF)
- #define IWDG_PR_PR ((uint8_t)0x07)
- #define IWDG_PR_PR_0 ((uint8_t)0x01)
- #define IWDG_PR_PR_1 ((uint8_t)0x02)
- #define IWDG_PR_PR_2 ((uint8_t)0x04)
- #define IWDG_RLR_RL ((uint16_t)0x0FFF)
- #define IWDG_SR_PVU ((uint8_t)0x01)
- #define IWDG_SR_RVU ((uint8_t)0x02)
- #define PWR_CR_LPDS ((uint16_t)0x0001)
- #define PWR_CR_PDDS ((uint16_t)0x0002)
- #define PWR_CR_CWUF ((uint16_t)0x0004)
- #define PWR_CR_CSBF ((uint16_t)0x0008)
- #define PWR_CR_PVDE ((uint16_t)0x0010)
- #define PWR_CR_PLS ((uint16_t)0x00E0)
- #define PWR_CR_PLS_0 ((uint16_t)0x0020)
- #define PWR_CR_PLS_1 ((uint16_t)0x0040)
- #define PWR_CR_PLS_2 ((uint16_t)0x0080)
- #define PWR_CR_PLS_LEV0 ((uint16_t)0x0000)
- #define PWR_CR_PLS_LEV1 ((uint16_t)0x0020)
- #define PWR_CR_PLS_LEV2 ((uint16_t)0x0040)
- #define PWR_CR_PLS_LEV3 ((uint16_t)0x0060)
- #define PWR_CR_PLS_LEV4 ((uint16_t)0x0080)
- #define PWR_CR_PLS_LEV5 ((uint16_t)0x00A0)
- #define PWR_CR_PLS_LEV6 ((uint16_t)0x00C0)
- #define PWR_CR_PLS_LEV7 ((uint16_t)0x00E0)
- #define PWR_CR_DBP ((uint16_t)0x0100)
- #define PWR_CR_FPDS ((uint16_t)0x0200)
- #define PWR_CR_VOS ((uint16_t)0x4000)
- #define **PWR_CR_PMODE** PWR_CR_VOS
- #define PWR_CSR_WUF ((uint16_t)0x0001)

- #define PWR_CSR_SBF ((uint16_t)0x0002)
- #define PWR_CSR_PVDO ((uint16_t)0x0004)
- #define PWR_CSR_BRR ((uint16_t)0x0008)
- #define PWR_CSR_EWUP ((uint16_t)0x0100)
- #define PWR_CSR_BRE ((uint16_t)0x0200)
- #define PWR_CSR_VOSRDY ((uint16_t)0x4000)
- #define **PWR_CSR_REGRDY** PWR_CSR_VOSRDY
- #define **RCC_CR_HSION** ((uint32_t)0x00000001)
- #define **RCC_CR_HSIRDY** ((uint32_t)0x00000002)
- #define **RCC_CR_HSITRIM** ((uint32_t)0x000000F8)
- #define RCC_CR_HSITRIM_0 ((uint32_t)0x00000008)
- #define RCC_CR_HSITRIM_1 ((uint32_t)0x00000010)
- #define RCC_CR_HSITRIM_2 ((uint32_t)0x00000020)
- #define RCC_CR_HSITRIM_3 ((uint32_t)0x00000040)
- #define RCC_CR_HSITRIM_4 ((uint32_t)0x00000080)
- #define **RCC_CR_HSICAL** ((uint32_t)0x0000FF00)
- #define RCC_CR_HSICAL_0 ((uint32_t)0x00000100)
- #define RCC_CR_HSICAL_1 ((uint32_t)0x00000200)
- #define RCC_CR_HSICAL_2 ((uint32_t)0x00000400)
- #define RCC_CR_HSICAL_3 ((uint32_t)0x00000800)
- #define RCC_CR_HSICAL_4 ((uint32_t)0x00001000)
- #define RCC_CR_HSICAL_5 ((uint32_t)0x00002000)
- #define RCC_CR_HSICAL_6 ((uint32_t)0x00004000)
- #define RCC_CR_HSICAL_7 ((uint32_t)0x00008000)
- #define **RCC_CR_HSEON** ((uint32_t)0x00010000)
- #define **RCC_CR_HSERDY** ((uint32_t)0x00020000)
- #define **RCC_CR_HSEBYP** ((uint32_t)0x00040000)
- #define **RCC_CR_CSSON** ((uint32_t)0x00080000)
- #define **RCC_CR_PLLON** ((uint32_t)0x01000000)
- #define **RCC_CR_PLLRDY** ((uint32_t)0x02000000)
- #define **RCC_CR_PLLI2SON** ((uint32_t)0x04000000)
- #define **RCC_CR_PLLI2SRDY** ((uint32_t)0x08000000)
- #define **RCC_PLLCFGR_PLLM** ((uint32_t)0x0000003F)
- #define **RCC_PLLCFGR_PLLM_0** ((uint32_t)0x00000001)
- #define **RCC_PLLCFGR_PLLM_1** ((uint32_t)0x00000002)
- #define **RCC_PLLCFGR_PLLM_2** ((uint32_t)0x00000004)
- #define **RCC_PLLCFGR_PLLM_3** ((uint32_t)0x00000008)
- #define **RCC_PLLCFGR_PLLM_4** ((uint32_t)0x00000010)
- #define **RCC_PLLCFGR_PLLM_5** ((uint32_t)0x00000020)
- #define **RCC_PLLCFGR_PLLN** ((uint32_t)0x00007FC0)
- #define **RCC_PLLCFGR_PLLN_0** ((uint32_t)0x00000040)
- #define **RCC_PLLCFGR_PLLN_1** ((uint32_t)0x00000080)
- #define **RCC_PLLCFGR_PLLN_2** ((uint32_t)0x00000100)
- #define **RCC_PLLCFGR_PLLN_3** ((uint32_t)0x00000200)
- #define **RCC_PLLCFGR_PLLN_4** ((uint32_t)0x00000400)
- #define **RCC_PLLCFGR_PLLN_5** ((uint32_t)0x00000800)
- #define **RCC_PLLCFGR_PLLN_6** ((uint32_t)0x00001000)
- #define **RCC_PLLCFGR_PLLN_7** ((uint32_t)0x00002000)
- #define **RCC_PLLCFGR_PLLN_8** ((uint32_t)0x00004000)
- #define **RCC_PLLCFGR_PLLP** ((uint32_t)0x00030000)
- #define **RCC_PLLCFGR_PLLP_0** ((uint32_t)0x00010000)
- #define **RCC_PLLCFGR_PLLP_1** ((uint32_t)0x00020000)
- #define **RCC_PLLCFGR_PLLSRC** ((uint32_t)0x00400000)
- #define **RCC_PLLCFGR_PLLSRC_HSE** ((uint32_t)0x00400000)
- #define **RCC_PLLCFGR_PLLSRC_HSI** ((uint32_t)0x00000000)

- #define **RCC_PLLCFGR_PLLQ** ((uint32_t)0x0F000000)
- #define **RCC_PLLCFGR_PLLQ_0** ((uint32_t)0x01000000)
- #define **RCC_PLLCFGR_PLLQ_1** ((uint32_t)0x02000000)
- #define **RCC_PLLCFGR_PLLQ_2** ((uint32_t)0x04000000)
- #define **RCC_PLLCFGR_PLLQ_3** ((uint32_t)0x08000000)
- #define RCC_CFGR_SW ((uint32_t)0x00000003)
- #define RCC_CFGR_SW_0 ((uint32_t)0x00000001)
- #define RCC_CFGR_SW_1 ((uint32_t)0x00000002)
- #define RCC_CFGR_SW_HSI ((uint32_t)0x00000000)
- #define RCC_CFGR_SW_HSE ((uint32_t)0x00000001)
- #define RCC_CFGR_SW_PLL ((uint32_t)0x00000002)
- #define RCC_CFGR_SWS ((uint32_t)0x0000000C)
- #define RCC_CFGR_SWS_0 ((uint32_t)0x00000004)
- #define RCC_CFGR_SWS_1 ((uint32_t)0x00000008)
- #define RCC_CFGR_SWS_HSI ((uint32_t)0x00000000)
- #define RCC_CFGR_SWS_HSE ((uint32_t)0x00000004)
- #define RCC_CFGR_SWS_PLL ((uint32_t)0x00000008)
- #define RCC_CFGR_HPRE ((uint32_t)0x000000F0)
- #define RCC_CFGR_HPRE_0 ((uint32_t)0x00000010)
- #define RCC_CFGR_HPRE_1 ((uint32_t)0x00000020)
- #define RCC_CFGR_HPRE_2 ((uint32_t)0x00000040)
- #define RCC_CFGR_HPRE_3 ((uint32_t)0x00000080)
- #define RCC_CFGR_HPRE_DIV1 ((uint32_t)0x00000000)
- #define RCC_CFGR_HPRE_DIV2 ((uint32_t)0x00000080)
- #define RCC_CFGR_HPRE_DIV4 ((uint32_t)0x00000090)
- #define RCC_CFGR_HPRE_DIV8 ((uint32_t)0x000000A0)
- #define RCC_CFGR_HPRE_DIV16 ((uint32_t)0x000000B0)
- #define RCC_CFGR_HPRE_DIV64 ((uint32_t)0x000000C0)
- #define RCC_CFGR_HPRE_DIV128 ((uint32_t)0x000000D0)
- #define RCC_CFGR_HPRE_DIV256 ((uint32_t)0x000000E0)
- #define RCC_CFGR_HPRE_DIV512 ((uint32_t)0x000000F0)
- #define RCC_CFGR_PPRE1 ((uint32_t)0x00001C00)
- #define RCC_CFGR_PPRE1_0 ((uint32_t)0x00000400)
- #define RCC_CFGR_PPRE1_1 ((uint32_t)0x00000800)
- #define RCC_CFGR_PPRE1_2 ((uint32_t)0x00001000)
- #define RCC_CFGR_PPRE1_DIV1 ((uint32_t)0x00000000)
- #define RCC_CFGR_PPRE1_DIV2 ((uint32_t)0x00001000)
- #define RCC_CFGR_PPRE1_DIV4 ((uint32_t)0x00001400)
- #define RCC_CFGR_PPRE1_DIV8 ((uint32_t)0x00001800)
- #define RCC_CFGR_PPRE1_DIV16 ((uint32_t)0x00001C00)
- #define RCC_CFGR_PPRE2 ((uint32_t)0x0000E000)
- #define RCC_CFGR_PPRE2_0 ((uint32_t)0x00002000)
- #define RCC_CFGR_PPRE2_1 ((uint32_t)0x00004000)
- #define RCC_CFGR_PPRE2_2 ((uint32_t)0x00008000)
- #define RCC_CFGR_PPRE2_DIV1 ((uint32_t)0x00000000)
- #define RCC_CFGR_PPRE2_DIV2 ((uint32_t)0x00008000)
- #define RCC_CFGR_PPRE2_DIV4 ((uint32_t)0x0000A000)
- #define RCC_CFGR_PPRE2_DIV8 ((uint32_t)0x0000C000)
- #define RCC_CFGR_PPRE2_DIV16 ((uint32_t)0x0000E000)
- #define **RCC_CFGR_RTCPRE** ((uint32_t)0x001F0000)
- #define **RCC_CFGR_RTCPRE_0** ((uint32_t)0x00010000)
- #define **RCC_CFGR_RTCPRE_1** ((uint32_t)0x00020000)
- #define **RCC_CFGR_RTCPRE_2** ((uint32_t)0x00040000)
- #define **RCC_CFGR_RTCPRE_3** ((uint32_t)0x00080000)
- #define RCC_CFGR_RTCPRE_4 ((uint32_t)0x00100000)

- #define **RCC_CFGR_MCO1** ((uint32_t)0x00600000)
- #define **RCC_CFGR_MCO1_0** ((uint32_t)0x00200000)
- #define **RCC_CFGR_MCO1_1** ((uint32_t)0x00400000)
- #define **RCC_CFGR_I2SSRC** ((uint32_t)0x00800000)
- #define **RCC_CFGR_MCO1PRE** ((uint32_t)0x07000000)
- #define **RCC_CFGR_MCO1PRE_0** ((uint32_t)0x01000000)
- #define **RCC_CFGR_MCO1PRE_1** ((uint32_t)0x02000000)
- #define **RCC_CFGR_MCO1PRE_2** ((uint32_t)0x04000000)
- #define **RCC_CFGR_MCO2PRE** ((uint32_t)0x38000000)
- #define **RCC_CFGR_MCO2PRE_0** ((uint32_t)0x08000000)
- #define **RCC_CFGR_MCO2PRE_1** ((uint32_t)0x10000000)
- #define **RCC_CFGR_MCO2PRE_2** ((uint32_t)0x20000000)
- #define **RCC_CFGR_MCO2** ((uint32_t)0xC0000000)
- #define **RCC_CFGR_MCO2_0** ((uint32_t)0x40000000)
- #define **RCC_CFGR_MCO2_1** ((uint32_t)0x80000000)
- #define **RCC_CIR_LSIRDYF** ((uint32_t)0x00000001)
- #define **RCC_CIR_LSERDYF** ((uint32_t)0x00000002)
- #define **RCC_CIR_HSIRDYF** ((uint32_t)0x00000004)
- #define **RCC_CIR_HSERDYF** ((uint32_t)0x00000008)
- #define **RCC_CIR_PLLRDYF** ((uint32_t)0x00000010)
- #define **RCC_CIR_PLLI2SRDYF** ((uint32_t)0x00000020)
- #define **RCC_CIR_CSSF** ((uint32_t)0x00000080)
- #define **RCC_CIR_LSIRDYIE** ((uint32_t)0x00000100)
- #define **RCC_CIR_LSERDYIE** ((uint32_t)0x00000200)
- #define **RCC_CIR_HSIRDYIE** ((uint32_t)0x00000400)
- #define **RCC_CIR_HSERDYIE** ((uint32_t)0x00000800)
- #define **RCC_CIR_PLLRDYIE** ((uint32_t)0x00001000)
- #define **RCC_CIR_PLLI2SRDYIE** ((uint32_t)0x00002000)
- #define **RCC_CIR_LSIRDYC** ((uint32_t)0x00010000)
- #define **RCC_CIR_LSERDYC** ((uint32_t)0x00020000)
- #define **RCC_CIR_HSIRDYC** ((uint32_t)0x00040000)
- #define **RCC_CIR_HSERDYC** ((uint32_t)0x00080000)
- #define **RCC_CIR_PLLRDYC** ((uint32_t)0x00100000)
- #define **RCC_CIR_PLLI2SRDYC** ((uint32_t)0x00200000)
- #define **RCC_CIR_CSSC** ((uint32_t)0x00800000)
- #define **RCC_AHB1RSTR_GPIOARST** ((uint32_t)0x00000001)
- #define **RCC_AHB1RSTR_GPIOBRST** ((uint32_t)0x00000002)
- #define **RCC_AHB1RSTR_GPIOCRST** ((uint32_t)0x00000004)
- #define **RCC_AHB1RSTR_GPIODRST** ((uint32_t)0x00000008)
- #define **RCC_AHB1RSTR_GPIOERST** ((uint32_t)0x00000010)
- #define **RCC_AHB1RSTR_GPIOFRST** ((uint32_t)0x00000020)
- #define **RCC_AHB1RSTR_GPIOGRST** ((uint32_t)0x00000040)
- #define **RCC_AHB1RSTR_GPIOHRST** ((uint32_t)0x00000080)
- #define **RCC_AHB1RSTR_GPIOIRST** ((uint32_t)0x00000100)
- #define **RCC_AHB1RSTR_CRCRST** ((uint32_t)0x00001000)
- #define **RCC_AHB1RSTR_DMA1RST** ((uint32_t)0x00200000)
- #define **RCC_AHB1RSTR_DMA2RST** ((uint32_t)0x00400000)
- #define **RCC_AHB1RSTR_ETHMACRST** ((uint32_t)0x02000000)
- #define **RCC_AHB1RSTR_OTGHRST** ((uint32_t)0x10000000)
- #define **RCC_AHB2RSTR_DCMIRST** ((uint32_t)0x00000001)
- #define **RCC_AHB2RSTR_CRYPRST** ((uint32_t)0x00000010)
- #define **RCC_AHB2RSTR_HSAHRST** ((uint32_t)0x00000020)
- #define **RCC_AHB2RSTR_RNGRST** ((uint32_t)0x00000040)
- #define **RCC_AHB2RSTR_OTGFSRST** ((uint32_t)0x00000080)
- #define **RCC_AHB3RSTR_FSMCRST** ((uint32_t)0x00000001)

- #define **RCC_APB1RSTR_TIM2RST** ((uint32_t)0x00000001)
- #define **RCC_APB1RSTR_TIM3RST** ((uint32_t)0x00000002)
- #define **RCC_APB1RSTR_TIM4RST** ((uint32_t)0x00000004)
- #define **RCC_APB1RSTR_TIM5RST** ((uint32_t)0x00000008)
- #define **RCC_APB1RSTR_TIM6RST** ((uint32_t)0x00000010)
- #define **RCC_APB1RSTR_TIM7RST** ((uint32_t)0x00000020)
- #define **RCC_APB1RSTR_TIM12RST** ((uint32_t)0x00000040)
- #define **RCC_APB1RSTR_TIM13RST** ((uint32_t)0x00000080)
- #define **RCC_APB1RSTR_TIM14RST** ((uint32_t)0x00000100)
- #define **RCC_APB1RSTR_WWDGEN** ((uint32_t)0x00000800)
- #define **RCC_APB1RSTR_SPI2RST** ((uint32_t)0x00008000)
- #define **RCC_APB1RSTR_SPI3RST** ((uint32_t)0x00010000)
- #define **RCC_APB1RSTR_USART2RST** ((uint32_t)0x00020000)
- #define **RCC_APB1RSTR_USART3RST** ((uint32_t)0x00040000)
- #define **RCC_APB1RSTR_UART4RST** ((uint32_t)0x00080000)
- #define **RCC_APB1RSTR_UART5RST** ((uint32_t)0x00100000)
- #define **RCC_APB1RSTR_I2C1RST** ((uint32_t)0x00200000)
- #define **RCC_APB1RSTR_I2C2RST** ((uint32_t)0x00400000)
- #define **RCC_APB1RSTR_I2C3RST** ((uint32_t)0x00800000)
- #define **RCC_APB1RSTR_CAN1RST** ((uint32_t)0x02000000)
- #define **RCC_APB1RSTR_CAN2RST** ((uint32_t)0x04000000)
- #define **RCC_APB1RSTR_PWRRST** ((uint32_t)0x10000000)
- #define **RCC_APB1RSTR_DACRST** ((uint32_t)0x20000000)
- #define **RCC_APB2RSTR_TIM1RST** ((uint32_t)0x00000001)
- #define **RCC_APB2RSTR_TIM8RST** ((uint32_t)0x00000002)
- #define **RCC_APB2RSTR_USART1RST** ((uint32_t)0x00000010)
- #define **RCC_APB2RSTR_USART6RST** ((uint32_t)0x00000020)
- #define **RCC_APB2RSTR_ADCRST** ((uint32_t)0x00000100)
- #define **RCC_APB2RSTR_SDIORST** ((uint32_t)0x00000800)
- #define **RCC_APB2RSTR_SPI1RST** ((uint32_t)0x00001000)
- #define **RCC_APB2RSTR_SYSCFGRST** ((uint32_t)0x00004000)
- #define **RCC_APB2RSTR_TIM9RST** ((uint32_t)0x00010000)
- #define **RCC_APB2RSTR_TIM10RST** ((uint32_t)0x00020000)
- #define **RCC_APB2RSTR_TIM11RST** ((uint32_t)0x00040000)
- #define **RCC_APB2RSTR_SPI1** RCC_APB2RSTR_SPI1RST
- #define **RCC_AHB1ENR_GPIOAEN** ((uint32_t)0x00000001)
- #define **RCC_AHB1ENR_GPIOBEN** ((uint32_t)0x00000002)
- #define **RCC_AHB1ENR_GPIOCEN** ((uint32_t)0x00000004)
- #define **RCC_AHB1ENR_GPIODEN** ((uint32_t)0x00000008)
- #define **RCC_AHB1ENR_GPIOEEN** ((uint32_t)0x00000010)
- #define **RCC_AHB1ENR_GPIOFEN** ((uint32_t)0x00000020)
- #define **RCC_AHB1ENR_GPIOGEN** ((uint32_t)0x00000040)
- #define **RCC_AHB1ENR_GPIOHEN** ((uint32_t)0x00000080)
- #define **RCC_AHB1ENR_GPIOIEN** ((uint32_t)0x00000100)
- #define **RCC_AHB1ENR_CRCEN** ((uint32_t)0x00001000)
- #define **RCC_AHB1ENR_BKPSRAMEN** ((uint32_t)0x00040000)
- #define **RCC_AHB1ENR_CCMDATARAMEN** ((uint32_t)0x00100000)
- #define **RCC_AHB1ENR_DMA1EN** ((uint32_t)0x00200000)
- #define **RCC_AHB1ENR_DMA2EN** ((uint32_t)0x00400000)
- #define **RCC_AHB1ENR_ETHMACEN** ((uint32_t)0x02000000)
- #define **RCC_AHB1ENR_ETHMACTXEN** ((uint32_t)0x04000000)
- #define **RCC_AHB1ENR_ETHMACRXEN** ((uint32_t)0x08000000)
- #define **RCC_AHB1ENR_ETHMACPTPEN** ((uint32_t)0x10000000)
- #define **RCC_AHB1ENR_OTGHSEN** ((uint32_t)0x20000000)
- #define **RCC_AHB1ENR_OTGHSULPIEN** ((uint32_t)0x40000000)

- #define **RCC_AHB2ENR_DCMIEN** ((uint32_t)0x00000001)
- #define **RCC_AHB2ENR_CRYPEN** ((uint32_t)0x00000010)
- #define **RCC_AHB2ENR_HASHEN** ((uint32_t)0x00000020)
- #define **RCC_AHB2ENR_RNGEN** ((uint32_t)0x00000040)
- #define **RCC_AHB2ENR_OTGFSEN** ((uint32_t)0x00000080)
- #define **RCC_AHB3ENR_FSMCEN** ((uint32_t)0x00000001)
- #define **RCC_APB1ENR_TIM2EN** ((uint32_t)0x00000001)
- #define **RCC_APB1ENR_TIM3EN** ((uint32_t)0x00000002)
- #define **RCC_APB1ENR_TIM4EN** ((uint32_t)0x00000004)
- #define **RCC_APB1ENR_TIM5EN** ((uint32_t)0x00000008)
- #define **RCC_APB1ENR_TIM6EN** ((uint32_t)0x00000010)
- #define **RCC_APB1ENR_TIM7EN** ((uint32_t)0x00000020)
- #define **RCC_APB1ENR_TIM12EN** ((uint32_t)0x00000040)
- #define **RCC_APB1ENR_TIM13EN** ((uint32_t)0x00000080)
- #define **RCC_APB1ENR_TIM14EN** ((uint32_t)0x00000100)
- #define **RCC_APB1ENR_WWDGEN** ((uint32_t)0x00000800)
- #define **RCC_APB1ENR_SPI2EN** ((uint32_t)0x00004000)
- #define **RCC_APB1ENR_SPI3EN** ((uint32_t)0x00008000)
- #define **RCC_APB1ENR_USART2EN** ((uint32_t)0x00020000)
- #define **RCC_APB1ENR_USART3EN** ((uint32_t)0x00040000)
- #define **RCC_APB1ENR_UART4EN** ((uint32_t)0x00080000)
- #define **RCC_APB1ENR_UART5EN** ((uint32_t)0x00100000)
- #define **RCC_APB1ENR_I2C1EN** ((uint32_t)0x00200000)
- #define **RCC_APB1ENR_I2C2EN** ((uint32_t)0x00400000)
- #define **RCC_APB1ENR_I2C3EN** ((uint32_t)0x00800000)
- #define **RCC_APB1ENR_CAN1EN** ((uint32_t)0x02000000)
- #define **RCC_APB1ENR_CAN2EN** ((uint32_t)0x04000000)
- #define **RCC_APB1ENR_PWREN** ((uint32_t)0x10000000)
- #define **RCC_APB1ENR_DACEN** ((uint32_t)0x20000000)
- #define **RCC_APB2ENR_TIM1EN** ((uint32_t)0x00000001)
- #define **RCC_APB2ENR_TIM8EN** ((uint32_t)0x00000002)
- #define **RCC_APB2ENR_USART1EN** ((uint32_t)0x00000010)
- #define **RCC_APB2ENR_USART6EN** ((uint32_t)0x00000020)
- #define **RCC_APB2ENR_ADC1EN** ((uint32_t)0x00000100)
- #define **RCC_APB2ENR_ADC2EN** ((uint32_t)0x00000200)
- #define **RCC_APB2ENR_ADC3EN** ((uint32_t)0x00000400)
- #define **RCC_APB2ENR_SDIOEN** ((uint32_t)0x00000800)
- #define **RCC_APB2ENR_SPI1EN** ((uint32_t)0x00001000)
- #define **RCC_APB2ENR_SYSCFGEN** ((uint32_t)0x00004000)
- #define **RCC_APB2ENR_TIM11EN** ((uint32_t)0x00040000)
- #define **RCC_APB2ENR_TIM10EN** ((uint32_t)0x00020000)
- #define **RCC_APB2ENR_TIM9EN** ((uint32_t)0x00010000)
- #define **RCC_AHB1LPENR_GPIOALPEN** ((uint32_t)0x00000001)
- #define **RCC_AHB1LPENR_GPIOBLPEN** ((uint32_t)0x00000002)
- #define **RCC_AHB1LPENR_GPIOCLPEN** ((uint32_t)0x00000004)
- #define **RCC_AHB1LPENR_GPIODLPEN** ((uint32_t)0x00000008)
- #define **RCC_AHB1LPENR_GPIOELPEN** ((uint32_t)0x00000010)
- #define **RCC_AHB1LPENR_GPIOFLPEN** ((uint32_t)0x00000020)
- #define **RCC_AHB1LPENR_GPIOGLPEN** ((uint32_t)0x00000040)
- #define **RCC_AHB1LPENR_GPIOHLPEN** ((uint32_t)0x00000080)
- #define **RCC_AHB1LPENR_GPIOILPEN** ((uint32_t)0x00000100)
- #define **RCC_AHB1LPENR_CRCLPEN** ((uint32_t)0x00001000)
- #define **RCC_AHB1LPENR_FLITFLPEN** ((uint32_t)0x00008000)
- #define **RCC_AHB1LPENR_SRAM1LPEN** ((uint32_t)0x00010000)
- #define **RCC_AHB1LPENR_SRAM2LPEN** ((uint32_t)0x00020000)

- #define **RCC_AHB1LPENR_BKPSRAMLPEN** ((uint32_t)0x00040000)
- #define **RCC_AHB1LPENR_DMA1LPEN** ((uint32_t)0x00200000)
- #define **RCC_AHB1LPENR_DMA2LPEN** ((uint32_t)0x00400000)
- #define **RCC_AHB1LPENR_ETHMACLPEN** ((uint32_t)0x02000000)
- #define **RCC_AHB1LPENR_ETHMACTXLPEN** ((uint32_t)0x04000000)
- #define **RCC_AHB1LPENR_ETHMACRXLPEN** ((uint32_t)0x08000000)
- #define **RCC_AHB1LPENR_ETHMACPTPLPEN** ((uint32_t)0x10000000)
- #define **RCC_AHB1LPENR_OTGHSLPEN** ((uint32_t)0x20000000)
- #define **RCC_AHB1LPENR_OTGHSULPILPEN** ((uint32_t)0x40000000)
- #define **RCC_AHB2LPENR_DCMILPEN** ((uint32_t)0x00000001)
- #define **RCC_AHB2LPENR_CRYPLPEN** ((uint32_t)0x00000010)
- #define **RCC_AHB2LPENR_HASHLPEN** ((uint32_t)0x00000020)
- #define **RCC_AHB2LPENR_RNGLPEN** ((uint32_t)0x00000040)
- #define **RCC_AHB2LPENR_OTGFSLPEN** ((uint32_t)0x00000080)
- #define **RCC_AHB3LPENR_FSMCLPEN** ((uint32_t)0x00000001)
- #define **RCC_APB1LPENR_TIM2LPEN** ((uint32_t)0x00000001)
- #define **RCC_APB1LPENR_TIM3LPEN** ((uint32_t)0x00000002)
- #define **RCC_APB1LPENR_TIM4LPEN** ((uint32_t)0x00000004)
- #define **RCC_APB1LPENR_TIM5LPEN** ((uint32_t)0x00000008)
- #define **RCC_APB1LPENR_TIM6LPEN** ((uint32_t)0x00000010)
- #define **RCC_APB1LPENR_TIM7LPEN** ((uint32_t)0x00000020)
- #define **RCC_APB1LPENR_TIM12LPEN** ((uint32_t)0x00000040)
- #define **RCC_APB1LPENR_TIM13LPEN** ((uint32_t)0x00000080)
- #define **RCC_APB1LPENR_TIM14LPEN** ((uint32_t)0x00000100)
- #define **RCC_APB1LPENR_WWDGLPEN** ((uint32_t)0x00000800)
- #define **RCC_APB1LPENR_SPI2LPEN** ((uint32_t)0x00004000)
- #define **RCC_APB1LPENR_SPI3LPEN** ((uint32_t)0x00008000)
- #define **RCC_APB1LPENR_USART2LPEN** ((uint32_t)0x00020000)
- #define **RCC_APB1LPENR_USART3LPEN** ((uint32_t)0x00040000)
- #define **RCC_APB1LPENR_UART4LPEN** ((uint32_t)0x00080000)
- #define **RCC_APB1LPENR_UART5LPEN** ((uint32_t)0x00100000)
- #define **RCC_APB1LPENR_I2C1LPEN** ((uint32_t)0x00200000)
- #define **RCC_APB1LPENR_I2C2LPEN** ((uint32_t)0x00400000)
- #define **RCC_APB1LPENR_I2C3LPEN** ((uint32_t)0x00800000)
- #define **RCC_APB1LPENR_CAN1LPEN** ((uint32_t)0x02000000)
- #define **RCC_APB1LPENR_CAN2LPEN** ((uint32_t)0x04000000)
- #define **RCC_APB1LPENR_PWRLPEN** ((uint32_t)0x10000000)
- #define **RCC_APB1LPENR_DACLPEN** ((uint32_t)0x20000000)
- #define **RCC_APB2LPENR_TIM1LPEN** ((uint32_t)0x00000001)
- #define **RCC_APB2LPENR_TIM8LPEN** ((uint32_t)0x00000002)
- #define **RCC_APB2LPENR_USART1LPEN** ((uint32_t)0x00000010)
- #define **RCC_APB2LPENR_USART6LPEN** ((uint32_t)0x00000020)
- #define **RCC_APB2LPENR_ADC1LPEN** ((uint32_t)0x00000100)
- #define **RCC_APB2LPENR_ADC2PEN** ((uint32_t)0x00000200)
- #define **RCC_APB2LPENR_ADC3LPEN** ((uint32_t)0x00000400)
- #define **RCC_APB2LPENR_SDIOLPEN** ((uint32_t)0x00000800)
- #define **RCC_APB2LPENR_SPI1LPEN** ((uint32_t)0x00001000)
- #define **RCC_APB2LPENR_SYSCFGLPEN** ((uint32_t)0x00004000)
- #define **RCC_APB2LPENR_TIM9LPEN** ((uint32_t)0x00010000)
- #define **RCC_APB2LPENR_TIM10LPEN** ((uint32_t)0x00020000)
- #define **RCC_APB2LPENR_TIM11LPEN** ((uint32_t)0x00040000)
- #define **RCC_BDCR_LSEON** ((uint32_t)0x00000001)
- #define **RCC_BDCR_LSERDY** ((uint32_t)0x00000002)
- #define **RCC_BDCR_LSEBYP** ((uint32_t)0x00000004)
- #define **RCC_BDCR_RTCSEL** ((uint32_t)0x00000300)

- #define **RCC_BDCR_RTCSEL_0** ((uint32_t)0x00000100)
- #define **RCC_BDCR_RTCSEL_1** ((uint32_t)0x00000200)
- #define **RCC_BDCR_RTCEN** ((uint32_t)0x00008000)
- #define **RCC_BDCR_BDRST** ((uint32_t)0x00010000)
- #define **RCC_CSR_LSION** ((uint32_t)0x00000001)
- #define **RCC_CSR_LSIRDY** ((uint32_t)0x00000002)
- #define **RCC_CSR_RMVF** ((uint32_t)0x01000000)
- #define **RCC_CSR_BORRSTF** ((uint32_t)0x02000000)
- #define **RCC_CSR_PADRSTF** ((uint32_t)0x04000000)
- #define **RCC_CSR_PORRSTF** ((uint32_t)0x08000000)
- #define **RCC_CSR_SFTRSTF** ((uint32_t)0x10000000)
- #define **RCC_CSR_WDGRSTF** ((uint32_t)0x20000000)
- #define **RCC_CSR_WWDGRSTF** ((uint32_t)0x40000000)
- #define **RCC_CSR_LPWRRSTF** ((uint32_t)0x80000000)
- #define **RCC_SSCGR_MODPER** ((uint32_t)0x00001FFF)
- #define **RCC_SSCGR_INCSTEP** ((uint32_t)0x0FFFE000)
- #define **RCC_SSCGR_SPREADSEL** ((uint32_t)0x40000000)
- #define **RCC_SSCGR_SSCGEN** ((uint32_t)0x80000000)
- #define **RCC_PLLI2SCFGR_PLLI2SN** ((uint32_t)0x00007FC0)
- #define **RCC_PLLI2SCFGR_PLLI2SR** ((uint32_t)0x70000000)
- #define **RNG_CR_RNGEN** ((uint32_t)0x00000004)
- #define **RNG_CR_IE** ((uint32_t)0x00000008)
- #define **RNG_SR_DRDY** ((uint32_t)0x00000001)
- #define **RNG_SR_CECS** ((uint32_t)0x00000002)
- #define **RNG_SR_SECS** ((uint32_t)0x00000004)
- #define **RNG_SR_CEIS** ((uint32_t)0x00000020)
- #define **RNG_SR_SEIS** ((uint32_t)0x00000040)
- #define **RTC_TR_PM** ((uint32_t)0x00400000)
- #define **RTC_TR_HT** ((uint32_t)0x00300000)
- #define **RTC_TR_HT_0** ((uint32_t)0x00100000)
- #define **RTC_TR_HT_1** ((uint32_t)0x00200000)
- #define **RTC_TR_HU** ((uint32_t)0x000F0000)
- #define **RTC_TR_HU_0** ((uint32_t)0x00010000)
- #define **RTC_TR_HU_1** ((uint32_t)0x00020000)
- #define **RTC_TR_HU_2** ((uint32_t)0x00040000)
- #define **RTC_TR_HU_3** ((uint32_t)0x00080000)
- #define **RTC_TR_MNT** ((uint32_t)0x00007000)
- #define **RTC_TR_MNT_0** ((uint32_t)0x00001000)
- #define **RTC_TR_MNT_1** ((uint32_t)0x00002000)
- #define **RTC_TR_MNT_2** ((uint32_t)0x00004000)
- #define **RTC_TR_MNU** ((uint32_t)0x00000F00)
- #define **RTC_TR_MNU_0** ((uint32_t)0x00000100)
- #define **RTC_TR_MNU_1** ((uint32_t)0x00000200)
- #define **RTC_TR_MNU_2** ((uint32_t)0x00000400)
- #define **RTC_TR_MNU_3** ((uint32_t)0x00000800)
- #define **RTC_TR_ST** ((uint32_t)0x00000070)
- #define **RTC_TR_ST_0** ((uint32_t)0x00000010)
- #define **RTC_TR_ST_1** ((uint32_t)0x00000020)
- #define **RTC_TR_ST_2** ((uint32_t)0x00000040)
- #define **RTC_TR_SU** ((uint32_t)0x0000000F)
- #define **RTC_TR_SU_0** ((uint32_t)0x00000001)
- #define **RTC_TR_SU_1** ((uint32_t)0x00000002)
- #define **RTC_TR_SU_2** ((uint32_t)0x00000004)
- #define **RTC_TR_SU_3** ((uint32_t)0x00000008)
- #define **RTC_DR_YT** ((uint32_t)0x00F00000)

- #define **RTC_DR_YT_0** ((uint32_t)0x00100000)
- #define **RTC_DR_YT_1** ((uint32_t)0x00200000)
- #define **RTC_DR_YT_2** ((uint32_t)0x00400000)
- #define **RTC_DR_YT_3** ((uint32_t)0x00800000)
- #define **RTC_DR_YU** ((uint32_t)0x000F0000)
- #define **RTC_DR_YU_0** ((uint32_t)0x00010000)
- #define **RTC_DR_YU_1** ((uint32_t)0x00020000)
- #define **RTC_DR_YU_2** ((uint32_t)0x00040000)
- #define **RTC_DR_YU_3** ((uint32_t)0x00080000)
- #define **RTC_DR_WDU** ((uint32_t)0x0000E000)
- #define **RTC_DR_WDU_0** ((uint32_t)0x00002000)
- #define **RTC_DR_WDU_1** ((uint32_t)0x00004000)
- #define **RTC_DR_WDU_2** ((uint32_t)0x00008000)
- #define **RTC_DR_MT** ((uint32_t)0x00001000)
- #define **RTC_DR_MU** ((uint32_t)0x00000F00)
- #define **RTC_DR_MU_0** ((uint32_t)0x00000100)
- #define **RTC_DR_MU_1** ((uint32_t)0x00000200)
- #define **RTC_DR_MU_2** ((uint32_t)0x00000400)
- #define **RTC_DR_MU_3** ((uint32_t)0x00000800)
- #define **RTC_DR_DT** ((uint32_t)0x00000030)
- #define **RTC_DR_DT_0** ((uint32_t)0x00000010)
- #define **RTC_DR_DT_1** ((uint32_t)0x00000020)
- #define **RTC_DR_DU** ((uint32_t)0x0000000F)
- #define **RTC_DR_DU_0** ((uint32_t)0x00000001)
- #define **RTC_DR_DU_1** ((uint32_t)0x00000002)
- #define **RTC_DR_DU_2** ((uint32_t)0x00000004)
- #define **RTC_DR_DU_3** ((uint32_t)0x00000008)
- #define **RTC_CR_COE** ((uint32_t)0x00800000)
- #define **RTC_CR_OSEL** ((uint32_t)0x00600000)
- #define **RTC_CR_OSEL_0** ((uint32_t)0x00200000)
- #define **RTC_CR_OSEL_1** ((uint32_t)0x00400000)
- #define **RTC_CR_POL** ((uint32_t)0x00100000)
- #define **RTC_CR_COSEL** ((uint32_t)0x00080000)
- #define **RTC_CR_BCK** ((uint32_t)0x00040000)
- #define **RTC_CR_SUB1H** ((uint32_t)0x00020000)
- #define **RTC_CR_ADD1H** ((uint32_t)0x00010000)
- #define **RTC_CR_TSIE** ((uint32_t)0x00008000)
- #define **RTC_CR_WUTIE** ((uint32_t)0x00004000)
- #define **RTC_CR_ALRBIE** ((uint32_t)0x00002000)
- #define **RTC_CR_ALRAIE** ((uint32_t)0x00001000)
- #define **RTC_CR_TSE** ((uint32_t)0x00000800)
- #define **RTC_CR_WUTE** ((uint32_t)0x00000400)
- #define **RTC_CR_ALRBE** ((uint32_t)0x00000200)
- #define **RTC_CR_ALRAE** ((uint32_t)0x00000100)
- #define **RTC_CR_DCE** ((uint32_t)0x00000080)
- #define **RTC_CR_FMT** ((uint32_t)0x00000040)
- #define **RTC_CR_BYPSHAD** ((uint32_t)0x00000020)
- #define **RTC_CR_REFCKON** ((uint32_t)0x00000010)
- #define **RTC_CR_TSEDGE** ((uint32_t)0x00000008)
- #define **RTC_CR_WUCKSEL** ((uint32_t)0x00000007)
- #define **RTC_CR_WUCKSEL_0** ((uint32_t)0x00000001)
- #define **RTC_CR_WUCKSEL_1** ((uint32_t)0x00000002)
- #define **RTC_CR_WUCKSEL_2** ((uint32_t)0x00000004)
- #define **RTC_ISR_RECALPF** ((uint32_t)0x00010000)
- #define **RTC_ISR_TAMP1F** ((uint32_t)0x00002000)

- #define **RTC_ISR_TSOVF** ((uint32_t)0x00001000)
- #define **RTC_ISR_TSF** ((uint32_t)0x00000800)
- #define **RTC_ISR_WUTF** ((uint32_t)0x00000400)
- #define **RTC_ISR_ALRBF** ((uint32_t)0x00000200)
- #define **RTC_ISR_ALRAF** ((uint32_t)0x00000100)
- #define **RTC_ISR_INIT** ((uint32_t)0x00000080)
- #define **RTC_ISR_INITF** ((uint32_t)0x00000040)
- #define **RTC_ISR_RSF** ((uint32_t)0x00000020)
- #define **RTC_ISR_INITS** ((uint32_t)0x00000010)
- #define **RTC_ISR_SHPF** ((uint32_t)0x00000008)
- #define **RTC_ISR_WUTWF** ((uint32_t)0x00000004)
- #define **RTC_ISR_ALRBWF** ((uint32_t)0x00000002)
- #define **RTC_ISR_ALRAWF** ((uint32_t)0x00000001)
- #define **RTC_PRER_PREDIV_A** ((uint32_t)0x007F0000)
- #define **RTC_PRER_PREDIV_S** ((uint32_t)0x00001FFF)
- #define **RTC_WUTR_WUT** ((uint32_t)0x0000FFFF)
- #define **RTC_CALIBR_DCS** ((uint32_t)0x00000080)
- #define **RTC_CALIBR_DC** ((uint32_t)0x0000001F)
- #define **RTC_ALRMAR_MSK4** ((uint32_t)0x80000000)
- #define **RTC_ALRMAR_WDSEL** ((uint32_t)0x40000000)
- #define **RTC_ALRMAR_DT** ((uint32_t)0x30000000)
- #define **RTC_ALRMAR_DT_0** ((uint32_t)0x10000000)
- #define **RTC_ALRMAR_DT_1** ((uint32_t)0x20000000)
- #define **RTC_ALRMAR_DU** ((uint32_t)0x0F000000)
- #define **RTC_ALRMAR_DU_0** ((uint32_t)0x01000000)
- #define **RTC_ALRMAR_DU_1** ((uint32_t)0x02000000)
- #define **RTC_ALRMAR_DU_2** ((uint32_t)0x04000000)
- #define **RTC_ALRMAR_DU_3** ((uint32_t)0x08000000)
- #define **RTC_ALRMAR_MSK3** ((uint32_t)0x00800000)
- #define **RTC_ALRMAR_PM** ((uint32_t)0x00400000)
- #define **RTC_ALRMAR_HT** ((uint32_t)0x00300000)
- #define **RTC_ALRMAR_HT_0** ((uint32_t)0x00100000)
- #define **RTC_ALRMAR_HT_1** ((uint32_t)0x00200000)
- #define **RTC_ALRMAR_HU** ((uint32_t)0x000F0000)
- #define **RTC_ALRMAR_HU_0** ((uint32_t)0x00010000)
- #define **RTC_ALRMAR_HU_1** ((uint32_t)0x00020000)
- #define **RTC_ALRMAR_HU_2** ((uint32_t)0x00040000)
- #define **RTC_ALRMAR_HU_3** ((uint32_t)0x00080000)
- #define **RTC_ALRMAR_MSK2** ((uint32_t)0x00008000)
- #define **RTC_ALRMAR_MNT** ((uint32_t)0x00007000)
- #define **RTC_ALRMAR_MNT_0** ((uint32_t)0x00001000)
- #define **RTC_ALRMAR_MNT_1** ((uint32_t)0x00002000)
- #define **RTC_ALRMAR_MNT_2** ((uint32_t)0x00004000)
- #define **RTC_ALRMAR_MNU** ((uint32_t)0x00000F00)
- #define **RTC_ALRMAR_MNU_0** ((uint32_t)0x00000100)
- #define **RTC_ALRMAR_MNU_1** ((uint32_t)0x00000200)
- #define **RTC_ALRMAR_MNU_2** ((uint32_t)0x00000400)
- #define **RTC_ALRMAR_MNU_3** ((uint32_t)0x00000800)
- #define **RTC_ALRMAR_MSK1** ((uint32_t)0x00000080)
- #define **RTC_ALRMAR_ST** ((uint32_t)0x00000070)
- #define **RTC_ALRMAR_ST_0** ((uint32_t)0x00000010)
- #define **RTC_ALRMAR_ST_1** ((uint32_t)0x00000020)
- #define **RTC_ALRMAR_ST_2** ((uint32_t)0x00000040)
- #define **RTC_ALRMAR_SU** ((uint32_t)0x0000000F)
- #define **RTC_ALRMAR_SU_0** ((uint32_t)0x00000001)

- #define **RTC_ALRMAR_SU_1** ((uint32_t)0x00000002)
- #define **RTC_ALRMAR_SU_2** ((uint32_t)0x00000004)
- #define **RTC_ALRMAR_SU_3** ((uint32_t)0x00000008)
- #define **RTC_ALRMBR_MSK4** ((uint32_t)0x80000000)
- #define **RTC_ALRMBR_WDSEL** ((uint32_t)0x40000000)
- #define **RTC_ALRMBR_DT** ((uint32_t)0x30000000)
- #define **RTC_ALRMBR_DT_0** ((uint32_t)0x10000000)
- #define **RTC_ALRMBR_DT_1** ((uint32_t)0x20000000)
- #define **RTC_ALRMBR_DU** ((uint32_t)0x0F000000)
- #define **RTC_ALRMBR_DU_0** ((uint32_t)0x01000000)
- #define **RTC_ALRMBR_DU_1** ((uint32_t)0x02000000)
- #define **RTC_ALRMBR_DU_2** ((uint32_t)0x04000000)
- #define **RTC_ALRMBR_DU_3** ((uint32_t)0x08000000)
- #define **RTC_ALRMBR_MSK3** ((uint32_t)0x00800000)
- #define **RTC_ALRMBR_PM** ((uint32_t)0x00400000)
- #define **RTC_ALRMBR_HT** ((uint32_t)0x00300000)
- #define **RTC_ALRMBR_HT_0** ((uint32_t)0x00100000)
- #define **RTC_ALRMBR_HT_1** ((uint32_t)0x00200000)
- #define **RTC_ALRMBR_HU** ((uint32_t)0x000F0000)
- #define **RTC_ALRMBR_HU_0** ((uint32_t)0x00010000)
- #define **RTC_ALRMBR_HU_1** ((uint32_t)0x00020000)
- #define **RTC_ALRMBR_HU_2** ((uint32_t)0x00040000)
- #define **RTC_ALRMBR_HU_3** ((uint32_t)0x00080000)
- #define **RTC_ALRMBR_MSK2** ((uint32_t)0x00008000)
- #define **RTC_ALRMBR_MNT** ((uint32_t)0x00007000)
- #define **RTC_ALRMBR_MNT_0** ((uint32_t)0x00001000)
- #define **RTC_ALRMBR_MNT_1** ((uint32_t)0x00002000)
- #define **RTC_ALRMBR_MNT_2** ((uint32_t)0x00004000)
- #define **RTC_ALRMBR_MNU** ((uint32_t)0x00000F00)
- #define **RTC_ALRMBR_MNU_0** ((uint32_t)0x00000100)
- #define **RTC_ALRMBR_MNU_1** ((uint32_t)0x00000200)
- #define **RTC_ALRMBR_MNU_2** ((uint32_t)0x00000400)
- #define **RTC_ALRMBR_MNU_3** ((uint32_t)0x00000800)
- #define **RTC_ALRMBR_MSK1** ((uint32_t)0x00000080)
- #define **RTC_ALRMBR_ST** ((uint32_t)0x00000070)
- #define **RTC_ALRMBR_ST_0** ((uint32_t)0x00000010)
- #define **RTC_ALRMBR_ST_1** ((uint32_t)0x00000020)
- #define **RTC_ALRMBR_ST_2** ((uint32_t)0x00000040)
- #define **RTC_ALRMBR_SU** ((uint32_t)0x0000000F)
- #define **RTC_ALRMBR_SU_0** ((uint32_t)0x00000001)
- #define **RTC_ALRMBR_SU_1** ((uint32_t)0x00000002)
- #define **RTC_ALRMBR_SU_2** ((uint32_t)0x00000004)
- #define **RTC_ALRMBR_SU_3** ((uint32_t)0x00000008)
- #define **RTC_WPR_KEY** ((uint32_t)0x000000FF)
- #define **RTC_SSR_SS** ((uint32_t)0x0000FFFF)
- #define **RTC_SHIFTR_SUBFS** ((uint32_t)0x00007FFF)
- #define **RTC_SHIFTR_ADD1S** ((uint32_t)0x80000000)
- #define **RTC_TSTR_PM** ((uint32_t)0x00400000)
- #define **RTC_TSTR_HT** ((uint32_t)0x00300000)
- #define **RTC_TSTR_HT_0** ((uint32_t)0x00100000)
- #define **RTC_TSTR_HT_1** ((uint32_t)0x00200000)
- #define **RTC_TSTR_HU** ((uint32_t)0x000F0000)
- #define **RTC_TSTR_HU_0** ((uint32_t)0x00010000)
- #define **RTC_TSTR_HU_1** ((uint32_t)0x00020000)
- #define **RTC_TSTR_HU_2** ((uint32_t)0x00040000)

- #define **RTC_TSTR_HU_3** ((uint32_t)0x00080000)
- #define **RTC_TSTR_MNT** ((uint32_t)0x00007000)
- #define **RTC_TSTR_MNT_0** ((uint32_t)0x00001000)
- #define **RTC_TSTR_MNT_1** ((uint32_t)0x00002000)
- #define **RTC_TSTR_MNT_2** ((uint32_t)0x00004000)
- #define **RTC_TSTR_MNU** ((uint32_t)0x00000F00)
- #define **RTC_TSTR_MNU_0** ((uint32_t)0x00000100)
- #define **RTC_TSTR_MNU_1** ((uint32_t)0x00000200)
- #define **RTC_TSTR_MNU_2** ((uint32_t)0x00000400)
- #define **RTC_TSTR_MNU_3** ((uint32_t)0x00000800)
- #define **RTC_TSTR_ST** ((uint32_t)0x00000070)
- #define **RTC_TSTR_ST_0** ((uint32_t)0x00000010)
- #define **RTC_TSTR_ST_1** ((uint32_t)0x00000020)
- #define **RTC_TSTR_ST_2** ((uint32_t)0x00000040)
- #define **RTC_TSTR_SU** ((uint32_t)0x0000000F)
- #define **RTC_TSTR_SU_0** ((uint32_t)0x00000001)
- #define **RTC_TSTR_SU_1** ((uint32_t)0x00000002)
- #define **RTC_TSTR_SU_2** ((uint32_t)0x00000004)
- #define **RTC_TSTR_SU_3** ((uint32_t)0x00000008)
- #define **RTC_TSDR_WDU** ((uint32_t)0x0000E000)
- #define **RTC_TSDR_WDU_0** ((uint32_t)0x00002000)
- #define **RTC_TSDR_WDU_1** ((uint32_t)0x00004000)
- #define **RTC_TSDR_WDU_2** ((uint32_t)0x00008000)
- #define **RTC_TSDR_MT** ((uint32_t)0x00001000)
- #define **RTC_TSDR_MU** ((uint32_t)0x00000F00)
- #define **RTC_TSDR_MU_0** ((uint32_t)0x00000100)
- #define **RTC_TSDR_MU_1** ((uint32_t)0x00000200)
- #define **RTC_TSDR_MU_2** ((uint32_t)0x00000400)
- #define **RTC_TSDR_MU_3** ((uint32_t)0x00000800)
- #define **RTC_TSDR_DT** ((uint32_t)0x00000030)
- #define **RTC_TSDR_DT_0** ((uint32_t)0x00000010)
- #define **RTC_TSDR_DT_1** ((uint32_t)0x00000020)
- #define **RTC_TSDR_DU** ((uint32_t)0x0000000F)
- #define **RTC_TSDR_DU_0** ((uint32_t)0x00000001)
- #define **RTC_TSDR_DU_1** ((uint32_t)0x00000002)
- #define **RTC_TSDR_DU_2** ((uint32_t)0x00000004)
- #define **RTC_TSDR_DU_3** ((uint32_t)0x00000008)
- #define **RTC_TSSSR_SS** ((uint32_t)0x0000FFFF)
- #define **RTC_CALR_CALP** ((uint32_t)0x00008000)
- #define **RTC_CALR_CALW8** ((uint32_t)0x00004000)
- #define **RTC_CALR_CALW16** ((uint32_t)0x00002000)
- #define **RTC_CALR_CALM** ((uint32_t)0x000001FF)
- #define **RTC_CALR_CALM_0** ((uint32_t)0x00000001)
- #define **RTC_CALR_CALM_1** ((uint32_t)0x00000002)
- #define **RTC_CALR_CALM_2** ((uint32_t)0x00000004)
- #define **RTC_CALR_CALM_3** ((uint32_t)0x00000008)
- #define **RTC_CALR_CALM_4** ((uint32_t)0x00000010)
- #define **RTC_CALR_CALM_5** ((uint32_t)0x00000020)
- #define **RTC_CALR_CALM_6** ((uint32_t)0x00000040)
- #define **RTC_CALR_CALM_7** ((uint32_t)0x00000080)
- #define **RTC_CALR_CALM_8** ((uint32_t)0x00000100)
- #define **RTC_TAFCR_ALARMOUTTYPE** ((uint32_t)0x00040000)
- #define **RTC_TAFCR_TSINSEL** ((uint32_t)0x00020000)
- #define **RTC_TAFCR_TAMPINSEL** ((uint32_t)0x00010000)
- #define **RTC_TAFCR_TAMPPUDIS** ((uint32_t)0x00008000)

- #define **RTC_TAFCR_TAMPPRCH** ((uint32_t)0x00006000)
- #define **RTC_TAFCR_TAMPPRCH_0** ((uint32_t)0x00002000)
- #define **RTC_TAFCR_TAMPPRCH_1** ((uint32_t)0x00004000)
- #define **RTC_TAFCR_TAMPFLT** ((uint32_t)0x00001800)
- #define **RTC_TAFCR_TAMPFLT_0** ((uint32_t)0x00000800)
- #define **RTC_TAFCR_TAMPFLT_1** ((uint32_t)0x00001000)
- #define **RTC_TAFCR_TAMPFREQ** ((uint32_t)0x00000700)
- #define **RTC_TAFCR_TAMPFREQ_0** ((uint32_t)0x00000100)
- #define **RTC_TAFCR_TAMPFREQ_1** ((uint32_t)0x00000200)
- #define **RTC_TAFCR_TAMPFREQ_2** ((uint32_t)0x00000400)
- #define **RTC_TAFCR_TAMPTS** ((uint32_t)0x00000080)
- #define **RTC_TAFCR_TAMPIE** ((uint32_t)0x00000004)
- #define **RTC_TAFCR_TAMP1TRG** ((uint32_t)0x00000002)
- #define **RTC_TAFCR_TAMP1E** ((uint32_t)0x00000001)
- #define **RTC_ALRMASSR_MASKSS** ((uint32_t)0x0F000000)
- #define **RTC_ALRMASSR_MASKSS_0** ((uint32_t)0x01000000)
- #define **RTC_ALRMASSR_MASKSS_1** ((uint32_t)0x02000000)
- #define **RTC_ALRMASSR_MASKSS_2** ((uint32_t)0x04000000)
- #define **RTC_ALRMASSR_MASKSS_3** ((uint32_t)0x08000000)
- #define **RTC_ALRMASSR_SS** ((uint32_t)0x00007FFF)
- #define **RTC_ALRMBSSR_MASKSS** ((uint32_t)0x0F000000)
- #define **RTC_ALRMBSSR_MASKSS_0** ((uint32_t)0x01000000)
- #define **RTC_ALRMBSSR_MASKSS_1** ((uint32_t)0x02000000)
- #define **RTC_ALRMBSSR_MASKSS_2** ((uint32_t)0x04000000)
- #define **RTC_ALRMBSSR_MASKSS_3** ((uint32_t)0x08000000)
- #define **RTC_ALRMBSSR_SS** ((uint32_t)0x00007FFF)
- #define **RTC_BKP0R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP1R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP2R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP3R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP4R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP5R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP6R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP7R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP8R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP9R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP10R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP11R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP12R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP13R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP14R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP15R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP16R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP17R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP18R** ((uint32_t)0xFFFFFFFF)
- #define **RTC_BKP19R** ((uint32_t)0xFFFFFFFF)
- #define SDIO_POWER_PWRCTRL ((uint8_t)0x03)
- #define SDIO_POWER_PWRCTRL_0 ((uint8_t)0x01)
- #define SDIO_POWER_PWRCTRL_1 ((uint8_t)0x02)
- #define SDIO_CLKCR_CLKDIV ((uint16_t)0x00FF)
- #define SDIO_CLKCR_CLKEN ((uint16_t)0x0100)
- #define SDIO_CLKCR_PWRSAV ((uint16_t)0x0200)
- #define SDIO_CLKCR_BYPASS ((uint16_t)0x0400)
- #define SDIO_CLKCR_WIDBUS ((uint16_t)0x1800)
- #define SDIO_CLKCR_WIDBUS_0 ((uint16_t)0x0800)

- #define SDIO_CLKCR_WIDBUS_1 ((uint16_t)0x1000)
- #define SDIO_CLKCR_NEGEDGE ((uint16_t)0x2000)
- #define SDIO_CLKCR_HWFC_EN ((uint16_t)0x4000)
- #define SDIO_ARG_CMDARG ((uint32_t)0xFFFFFFFF)
- #define SDIO_CMD_CMDINDEX ((uint16_t)0x003F)
- #define SDIO_CMD_WAITRESP ((uint16_t)0x00C0)
- #define SDIO_CMD_WAITRESP_0 ((uint16_t)0x0040)
- #define SDIO_CMD_WAITRESP_1 ((uint16_t)0x0080)
- #define SDIO_CMD_WAITINT ((uint16_t)0x0100)
- #define SDIO_CMD_WAITPEND ((uint16_t)0x0200)
- #define SDIO_CMD_CPSMEN ((uint16_t)0x0400)
- #define SDIO_CMD_SDIOSUSPEND ((uint16_t)0x0800)
- #define SDIO_CMD_ENCMDCOMPL ((uint16_t)0x1000)
- #define SDIO_CMD_NIEN ((uint16_t)0x2000)
- #define SDIO_CMD_CEATACMD ((uint16_t)0x4000)
- #define SDIO_RESPCMD_RESPCMD ((uint8_t)0x3F)
- #define SDIO_RESP0_CARDSTATUS0 ((uint32_t)0xFFFFFFFF)
- #define SDIO_RESP1_CARDSTATUS1 ((uint32_t)0xFFFFFFFF)
- #define SDIO_RESP2_CARDSTATUS2 ((uint32_t)0xFFFFFFFF)
- #define SDIO_RESP3_CARDSTATUS3 ((uint32_t)0xFFFFFFFF)
- #define SDIO_RESP4_CARDSTATUS4 ((uint32_t)0xFFFFFFFF)
- #define SDIO_DTIMER_DATATIME ((uint32_t)0xFFFFFFFF)
- #define SDIO_DLEN_DATALENGTH ((uint32_t)0x01FFFFFF)
- #define SDIO_DCTRL_DTEN ((uint16_t)0x0001)
- #define SDIO_DCTRL_DTDIR ((uint16_t)0x0002)
- #define SDIO_DCTRL_DTMODE ((uint16_t)0x0004)
- #define SDIO_DCTRL_DMAEN ((uint16_t)0x0008)
- #define SDIO_DCTRL_DBLOCKSIZE ((uint16_t)0x00F0)
- #define SDIO_DCTRL_DBLOCKSIZE_0 ((uint16_t)0x0010)
- #define SDIO_DCTRL_DBLOCKSIZE_1 ((uint16_t)0x0020)
- #define SDIO_DCTRL_DBLOCKSIZE_2 ((uint16_t)0x0040)
- #define SDIO_DCTRL_DBLOCKSIZE_3 ((uint16_t)0x0080)
- #define SDIO_DCTRL_RWSTART ((uint16_t)0x0100)
- #define SDIO_DCTRL_RWSTOP ((uint16_t)0x0200)
- #define SDIO_DCTRL_RWMOD ((uint16_t)0x0400)
- #define SDIO_DCTRL_SDIOEN ((uint16_t)0x0800)
- #define SDIO_DCOUNT_DATACOUNT ((uint32_t)0x01FFFFFF)
- #define SDIO_STA_CCRCFAIL ((uint32_t)0x00000001)
- #define SDIO_STA_DCRCFAIL ((uint32_t)0x00000002)
- #define SDIO_STA_CTIMEOUT ((uint32_t)0x00000004)
- #define SDIO_STA_DTIMEOUT ((uint32_t)0x00000008)
- #define SDIO_STA_TXUNDERR ((uint32_t)0x00000010)
- #define SDIO_STA_RXOVERR ((uint32_t)0x00000020)
- #define SDIO_STA_CMDREND ((uint32_t)0x00000040)
- #define SDIO_STA_CMDSENT ((uint32_t)0x00000080)
- #define SDIO_STA_DATAEND ((uint32_t)0x00000100)
- #define SDIO_STA_STBITERR ((uint32_t)0x00000200)
- #define SDIO_STA_DBCKEND ((uint32_t)0x00000400)
- #define SDIO_STA_CMDACT ((uint32_t)0x00000800)
- #define SDIO_STA_TXACT ((uint32_t)0x00001000)
- #define SDIO_STA_RXACT ((uint32_t)0x00002000)
- #define SDIO_STA_TXFIFOHE ((uint32_t)0x00004000)
- #define SDIO_STA_RXFIFOHF ((uint32_t)0x00008000)
- #define SDIO_STA_TXFIFOF ((uint32_t)0x00010000)
- #define SDIO_STA_RXFIFOF ((uint32_t)0x00020000)

- #define SDIO_STA_TXFIFOE ((uint32_t)0x00040000)
- #define SDIO_STA_RXFIFOE ((uint32_t)0x00080000)
- #define SDIO_STA_TXDAVL ((uint32_t)0x00100000)
- #define SDIO_STA_RXDAVL ((uint32_t)0x00200000)
- #define SDIO_STA_SDIOIT ((uint32_t)0x00400000)
- #define SDIO_STA_CEATAEND ((uint32_t)0x00800000)
- #define SDIO_ICR_CCRCFAILC ((uint32_t)0x00000001)
- #define SDIO_ICR_DCRCFAILC ((uint32_t)0x00000002)
- #define SDIO_ICR_CTIMEOUTC ((uint32_t)0x00000004)
- #define SDIO_ICR_DTIMEOUTC ((uint32_t)0x00000008)
- #define SDIO_ICR_TXUNDERRC ((uint32_t)0x00000010)
- #define SDIO_ICR_RXOVERRC ((uint32_t)0x00000020)
- #define SDIO_ICR_CMDRENDC ((uint32_t)0x00000040)
- #define SDIO_ICR_CMDSENTC ((uint32_t)0x00000080)
- #define SDIO_ICR_DATAENDC ((uint32_t)0x00000100)
- #define SDIO_ICR_STBITERRC ((uint32_t)0x00000200)
- #define SDIO_ICR_DBCKENDC ((uint32_t)0x00000400)
- #define SDIO_ICR_SDIOITC ((uint32_t)0x00400000)
- #define SDIO_ICR_CEATAENDC ((uint32_t)0x00800000)
- #define SDIO_MASK_CCRCFAILIE ((uint32_t)0x00000001)
- #define SDIO_MASK_DCRCFAILIE ((uint32_t)0x00000002)
- #define SDIO_MASK_CTIMEOUTIE ((uint32_t)0x00000004)
- #define SDIO_MASK_DTIMEOUTIE ((uint32_t)0x00000008)
- #define SDIO_MASK_TXUNDERRIE ((uint32_t)0x00000010)
- #define SDIO_MASK_RXOVERRIE ((uint32_t)0x00000020)
- #define SDIO_MASK_CMDRENDIE ((uint32_t)0x00000040)
- #define SDIO_MASK_CMDSENTIE ((uint32_t)0x00000080)
- #define SDIO_MASK_DATAENDIE ((uint32_t)0x00000100)
- #define SDIO_MASK_STBITERRIE ((uint32_t)0x00000200)
- #define SDIO_MASK_DBCKENDIE ((uint32_t)0x00000400)
- #define SDIO_MASK_CMDACTIE ((uint32_t)0x00000800)
- #define SDIO_MASK_TXACTIE ((uint32_t)0x00001000)
- #define SDIO_MASK_RXACTIE ((uint32_t)0x00002000)
- #define SDIO_MASK_TXFIFOHEIE ((uint32_t)0x00004000)
- #define SDIO_MASK_RXFIFOHFIE ((uint32_t)0x00008000)
- #define SDIO_MASK_TXFIFOFIE ((uint32_t)0x00010000)
- #define SDIO_MASK_RXFIFOFIE ((uint32_t)0x00020000)
- #define SDIO_MASK_TXFIFOEIE ((uint32_t)0x00040000)
- #define SDIO_MASK_RXFIFOEIE ((uint32_t)0x00080000)
- #define SDIO_MASK_TXDAVLIE ((uint32_t)0x00100000)
- #define SDIO_MASK_RXDAVLIE ((uint32_t)0x00200000)
- #define SDIO_MASK_SDIOITIE ((uint32_t)0x00400000)
- #define SDIO_MASK_CEATAENDIE ((uint32_t)0x00800000)
- #define SDIO_FIFOCNT_FIFOCOUNT ((uint32_t)0x00FFFFFF)
- #define SDIO_FIFO_FIFODATA ((uint32_t)0xFFFFFFFF)
- #define SPI_CR1_CPHA ((uint16_t)0x0001)
- #define SPI_CR1_CPOL ((uint16_t)0x0002)
- #define SPI_CR1_MSTR ((uint16_t)0x0004)
- #define SPI_CR1_BR ((uint16_t)0x0038)
- #define SPI_CR1_BR_0 ((uint16_t)0x0008)
- #define SPI_CR1_BR_1 ((uint16_t)0x0010)
- #define SPI_CR1_BR_2 ((uint16_t)0x0020)
- #define SPI_CR1_SPE ((uint16_t)0x0040)
- #define SPI_CR1_LSBFIRST ((uint16_t)0x0080)
- #define SPI_CR1_SSI ((uint16_t)0x0100)

- #define SPI_CR1_SSM ((uint16_t)0x0200)
- #define SPI_CR1_RXONLY ((uint16_t)0x0400)
- #define SPI_CR1_DFF ((uint16_t)0x0800)
- #define SPI_CR1_CRCNEXT ((uint16_t)0x1000)
- #define SPI_CR1_CRCEN ((uint16_t)0x2000)
- #define SPI_CR1_BIDIOE ((uint16_t)0x4000)
- #define SPI_CR1_BIDIMODE ((uint16_t)0x8000)
- #define SPI_CR2_RXDMAEN ((uint8_t)0x01)
- #define SPI_CR2_TXDMAEN ((uint8_t)0x02)
- #define SPI_CR2_SSOE ((uint8_t)0x04)
- #define SPI_CR2_ERRIE ((uint8_t)0x20)
- #define SPI_CR2_RXNEIE ((uint8_t)0x40)
- #define SPI_CR2_TXEIE ((uint8_t)0x80)
- #define SPI_SR_RXNE ((uint8_t)0x01)
- #define SPI_SR_TXE ((uint8_t)0x02)
- #define SPI_SR_CHSIDE ((uint8_t)0x04)
- #define SPI_SR_UDR ((uint8_t)0x08)
- #define SPI_SR_CRCERR ((uint8_t)0x10)
- #define SPI_SR_MODF ((uint8_t)0x20)
- #define SPI_SR_OVR ((uint8_t)0x40)
- #define SPI_SR_BSY ((uint8_t)0x80)
- #define SPI_DR_DR ((uint16_t)0xFFFF)
- #define SPI_CRCPR_CRCPOLY ((uint16_t)0xFFFF)
- #define SPI_RXCRCR_RXCRC ((uint16_t)0xFFFF)
- #define SPI_TXCRCR_TXCRC ((uint16_t)0xFFFF)
- #define SPI_I2SCFGR_CHLEN ((uint16_t)0x0001)
- #define SPI_I2SCFGR_DATLEN ((uint16_t)0x0006)
- #define SPI_I2SCFGR_DATLEN_0 ((uint16_t)0x0002)
- #define SPI_I2SCFGR_DATLEN_1 ((uint16_t)0x0004)
- #define SPI_I2SCFGR_CKPOL ((uint16_t)0x0008)
- #define SPI_I2SCFGR_I2SSTD ((uint16_t)0x0030)
- #define SPI_I2SCFGR_I2SSTD_0 ((uint16_t)0x0010)
- #define SPI_I2SCFGR_I2SSTD_1 ((uint16_t)0x0020)
- #define SPI_I2SCFGR_PCMSYNC ((uint16_t)0x0080)
- #define SPI_I2SCFGR_I2SCFG ((uint16_t)0x0300)
- #define SPI_I2SCFGR_I2SCFG_0 ((uint16_t)0x0100)
- #define SPI_I2SCFGR_I2SCFG_1 ((uint16_t)0x0200)
- #define SPI_I2SCFGR_I2SE ((uint16_t)0x0400)
- #define SPI_I2SCFGR_I2SMOD ((uint16_t)0x0800)
- #define SPI_I2SPR_I2SDIV ((uint16_t)0x00FF)
- #define SPI_I2SPR_ODD ((uint16_t)0x0100)
- #define SPI_I2SPR_MCKOE ((uint16_t)0x0200)
- #define SYSCFG_MEMRMP_MEM_MODE ((uint32_t)0x00000003)
- #define **SYSCFG_MEMRMP_MEM_MODE_0** ((uint32_t)0x00000001)
- #define **SYSCFG_MEMRMP_MEM_MODE_1** ((uint32_t)0x00000002)
- #define SYSCFG_PMC_MII_RMII_SEL ((uint32_t)0x00800000)
- #define **SYSCFG_PMC_MII_RMII** SYSCFG_PMC_MII_RMII_SEL
- #define SYSCFG_EXTICR1_EXTI0 ((uint16_t)0x000F)
- #define SYSCFG_EXTICR1_EXTI1 ((uint16_t)0x00F0)
- #define SYSCFG_EXTICR1_EXTI2 ((uint16_t)0x0F00)
- #define SYSCFG_EXTICR1_EXTI3 ((uint16_t)0xF000)
- #define SYSCFG_EXTICR1_EXTI0_PA ((uint16_t)0x0000)

  EXTI0 configuration

- #define SYSCFG_EXTICR1_EXTI0_PB ((uint16_t)0x0001)
- #define SYSCFG_EXTICR1_EXTI0_PC ((uint16_t)0x0002)
- #define SYSCFG_EXTICR1_EXTI0_PD ((uint16_t)0x0003)
- #define SYSCFG_EXTICR1_EXTI0_PE ((uint16_t)0x0004)
- #define SYSCFG_EXTICR1_EXTI0_PF ((uint16_t)0x0005)
- #define SYSCFG_EXTICR1_EXTI0_PG ((uint16_t)0x0006)
- #define SYSCFG_EXTICR1_EXTI0_PH ((uint16_t)0x0007)
- #define SYSCFG_EXTICR1_EXTI0_PI ((uint16_t)0x0008)
- #define SYSCFG_EXTICR1_EXTI1_PA ((uint16_t)0x0000)

  EXTI1 configuration

- #define SYSCFG_EXTICR1_EXTI1_PB ((uint16_t)0x0010)
- #define SYSCFG_EXTICR1_EXTI1_PC ((uint16_t)0x0020)
- #define SYSCFG_EXTICR1_EXTI1_PD ((uint16_t)0x0030)
- #define SYSCFG_EXTICR1_EXTI1_PE ((uint16_t)0x0040)
- #define SYSCFG_EXTICR1_EXTI1_PF ((uint16_t)0x0050)
- #define SYSCFG_EXTICR1_EXTI1_PG ((uint16_t)0x0060)
- #define SYSCFG_EXTICR1_EXTI1_PH ((uint16_t)0x0070)
- #define SYSCFG_EXTICR1_EXTI1_PI ((uint16_t)0x0080)
- #define SYSCFG_EXTICR1_EXTI2_PA ((uint16_t)0x0000)

  EXTI2 configuration

- #define SYSCFG_EXTICR1_EXTI2_PB ((uint16_t)0x0100)
- #define SYSCFG_EXTICR1_EXTI2_PC ((uint16_t)0x0200)
- #define SYSCFG_EXTICR1_EXTI2_PD ((uint16_t)0x0300)
- #define SYSCFG_EXTICR1_EXTI2_PE ((uint16_t)0x0400)
- #define SYSCFG_EXTICR1_EXTI2_PF ((uint16_t)0x0500)
- #define SYSCFG_EXTICR1_EXTI2_PG ((uint16_t)0x0600)
- #define SYSCFG_EXTICR1_EXTI2_PH ((uint16_t)0x0700)
- #define SYSCFG_EXTICR1_EXTI2_PI ((uint16_t)0x0800)
- #define SYSCFG_EXTICR1_EXTI3_PA ((uint16_t)0x0000)

  EXTI3 configuration

- #define SYSCFG_EXTICR1_EXTI3_PB ((uint16_t)0x1000)
- #define SYSCFG_EXTICR1_EXTI3_PC ((uint16_t)0x2000)
- #define SYSCFG_EXTICR1_EXTI3_PD ((uint16_t)0x3000)
- #define SYSCFG_EXTICR1_EXTI3_PE ((uint16_t)0x4000)
- #define SYSCFG_EXTICR1_EXTI3_PF ((uint16_t)0x5000)
- #define SYSCFG_EXTICR1_EXTI3_PG ((uint16_t)0x6000)
- #define SYSCFG_EXTICR1_EXTI3_PH ((uint16_t)0x7000)
- #define SYSCFG_EXTICR1_EXTI3_PI ((uint16_t)0x8000)
- #define SYSCFG_EXTICR2_EXTI4 ((uint16_t)0x000F)
- #define SYSCFG_EXTICR2_EXTI5 ((uint16_t)0x00F0)
- #define SYSCFG_EXTICR2_EXTI6 ((uint16_t)0x0F00)
- #define SYSCFG_EXTICR2_EXTI7 ((uint16_t)0xF000)
- #define SYSCFG_EXTICR2_EXTI4_PA ((uint16_t)0x0000)

  EXTI4 configuration

- #define SYSCFG_EXTICR2_EXTI4_PB ((uint16_t)0x0001)
- #define SYSCFG_EXTICR2_EXTI4_PC ((uint16_t)0x0002)
- #define SYSCFG_EXTICR2_EXTI4_PD ((uint16_t)0x0003)
- #define SYSCFG_EXTICR2_EXTI4_PE ((uint16_t)0x0004)
- #define SYSCFG_EXTICR2_EXTI4_PF ((uint16_t)0x0005)
- #define SYSCFG_EXTICR2_EXTI4_PG ((uint16_t)0x0006)

- #define SYSCFG_EXTICR2_EXTI4_PH ((uint16_t)0x0007)
- #define SYSCFG_EXTICR2_EXTI4_PI ((uint16_t)0x0008)
- #define SYSCFG_EXTICR2_EXTI5_PA ((uint16_t)0x0000)

  EXTI5 configuration

- #define SYSCFG_EXTICR2_EXTI5_PB ((uint16_t)0x0010)
- #define SYSCFG_EXTICR2_EXTI5_PC ((uint16_t)0x0020)
- #define SYSCFG_EXTICR2_EXTI5_PD ((uint16_t)0x0030)
- #define SYSCFG_EXTICR2_EXTI5_PE ((uint16_t)0x0040)
- #define SYSCFG_EXTICR2_EXTI5_PF ((uint16_t)0x0050)
- #define SYSCFG_EXTICR2_EXTI5_PG ((uint16_t)0x0060)
- #define SYSCFG_EXTICR2_EXTI5_PH ((uint16_t)0x0070)
- #define SYSCFG_EXTICR2_EXTI5_PI ((uint16_t)0x0080)
- #define SYSCFG_EXTICR2_EXTI6_PA ((uint16_t)0x0000)

  EXTI6 configuration

- #define SYSCFG_EXTICR2_EXTI6_PB ((uint16_t)0x0100)
- #define SYSCFG_EXTICR2_EXTI6_PC ((uint16_t)0x0200)
- #define SYSCFG_EXTICR2_EXTI6_PD ((uint16_t)0x0300)
- #define SYSCFG_EXTICR2_EXTI6_PE ((uint16_t)0x0400)
- #define SYSCFG_EXTICR2_EXTI6_PF ((uint16_t)0x0500)
- #define SYSCFG_EXTICR2_EXTI6_PG ((uint16_t)0x0600)
- #define SYSCFG_EXTICR2_EXTI6_PH ((uint16_t)0x0700)
- #define SYSCFG_EXTICR2_EXTI6_PI ((uint16_t)0x0800)
- #define SYSCFG_EXTICR2_EXTI7_PA ((uint16_t)0x0000)

  EXTI7 configuration

- #define SYSCFG_EXTICR2_EXTI7_PB ((uint16_t)0x1000)
- #define SYSCFG_EXTICR2_EXTI7_PC ((uint16_t)0x2000)
- #define SYSCFG_EXTICR2_EXTI7_PD ((uint16_t)0x3000)
- #define SYSCFG_EXTICR2_EXTI7_PE ((uint16_t)0x4000)
- #define SYSCFG_EXTICR2_EXTI7_PF ((uint16_t)0x5000)
- #define SYSCFG_EXTICR2_EXTI7_PG ((uint16_t)0x6000)
- #define SYSCFG_EXTICR2_EXTI7_PH ((uint16_t)0x7000)
- #define SYSCFG_EXTICR2_EXTI7_PI ((uint16_t)0x8000)
- #define SYSCFG_EXTICR3_EXTI8 ((uint16_t)0x000F)
- #define SYSCFG_EXTICR3_EXTI9 ((uint16_t)0x00F0)
- #define SYSCFG_EXTICR3_EXTI10 ((uint16_t)0x0F00)
- #define SYSCFG_EXTICR3_EXTI11 ((uint16_t)0xF000)
- #define SYSCFG_EXTICR3_EXTI8_PA ((uint16_t)0x0000)

  EXTI8 configuration

- #define SYSCFG_EXTICR3_EXTI8_PB ((uint16_t)0x0001)
- #define SYSCFG_EXTICR3_EXTI8_PC ((uint16_t)0x0002)
- #define SYSCFG_EXTICR3_EXTI8_PD ((uint16_t)0x0003)
- #define SYSCFG_EXTICR3_EXTI8_PE ((uint16_t)0x0004)
- #define SYSCFG_EXTICR3_EXTI8_PF ((uint16_t)0x0005)
- #define SYSCFG_EXTICR3_EXTI8_PG ((uint16_t)0x0006)
- #define SYSCFG_EXTICR3_EXTI8_PH ((uint16_t)0x0007)
- #define SYSCFG_EXTICR3_EXTI8_PI ((uint16_t)0x0008)
- #define SYSCFG_EXTICR3_EXTI9_PA ((uint16_t)0x0000)

  EXTI9 configuration

- #define SYSCFG_EXTICR3_EXTI9_PB ((uint16_t)0x0010)

- #define SYSCFG_EXTICR3_EXTI9_PC ((uint16_t)0x0020)
- #define SYSCFG_EXTICR3_EXTI9_PD ((uint16_t)0x0030)
- #define SYSCFG_EXTICR3_EXTI9_PE ((uint16_t)0x0040)
- #define SYSCFG_EXTICR3_EXTI9_PF ((uint16_t)0x0050)
- #define SYSCFG_EXTICR3_EXTI9_PG ((uint16_t)0x0060)
- #define SYSCFG_EXTICR3_EXTI9_PH ((uint16_t)0x0070)
- #define SYSCFG_EXTICR3_EXTI9_PI ((uint16_t)0x0080)
- #define SYSCFG_EXTICR3_EXTI10_PA ((uint16_t)0x0000)

  EXTI10 configuration

- #define SYSCFG_EXTICR3_EXTI10_PB ((uint16_t)0x0100)
- #define SYSCFG_EXTICR3_EXTI10_PC ((uint16_t)0x0200)
- #define SYSCFG_EXTICR3_EXTI10_PD ((uint16_t)0x0300)
- #define SYSCFG_EXTICR3_EXTI10_PE ((uint16_t)0x0400)
- #define SYSCFG_EXTICR3_EXTI10_PF ((uint16_t)0x0500)
- #define SYSCFG_EXTICR3_EXTI10_PG ((uint16_t)0x0600)
- #define SYSCFG_EXTICR3_EXTI10_PH ((uint16_t)0x0700)
- #define SYSCFG_EXTICR3_EXTI10_PI ((uint16_t)0x0800)
- #define SYSCFG_EXTICR3_EXTI11_PA ((uint16_t)0x0000)

  EXTI11 configuration

- #define SYSCFG_EXTICR3_EXTI11_PB ((uint16_t)0x1000)
- #define SYSCFG_EXTICR3_EXTI11_PC ((uint16_t)0x2000)
- #define SYSCFG_EXTICR3_EXTI11_PD ((uint16_t)0x3000)
- #define SYSCFG_EXTICR3_EXTI11_PE ((uint16_t)0x4000)
- #define SYSCFG_EXTICR3_EXTI11_PF ((uint16_t)0x5000)
- #define SYSCFG_EXTICR3_EXTI11_PG ((uint16_t)0x6000)
- #define SYSCFG_EXTICR3_EXTI11_PH ((uint16_t)0x7000)
- #define SYSCFG_EXTICR3_EXTI11_PI ((uint16_t)0x8000)
- #define SYSCFG_EXTICR4_EXTI12 ((uint16_t)0x000F)
- #define SYSCFG_EXTICR4_EXTI13 ((uint16_t)0x00F0)
- #define SYSCFG_EXTICR4_EXTI14 ((uint16_t)0x0F00)
- #define SYSCFG_EXTICR4_EXTI15 ((uint16_t)0xF000)
- #define SYSCFG_EXTICR4_EXTI12_PA ((uint16_t)0x0000)

  EXTI12 configuration

- #define SYSCFG_EXTICR4_EXTI12_PB ((uint16_t)0x0001)
- #define SYSCFG_EXTICR4_EXTI12_PC ((uint16_t)0x0002)
- #define SYSCFG_EXTICR4_EXTI12_PD ((uint16_t)0x0003)
- #define SYSCFG_EXTICR4_EXTI12_PE ((uint16_t)0x0004)
- #define SYSCFG_EXTICR4_EXTI12_PF ((uint16_t)0x0005)
- #define SYSCFG_EXTICR4_EXTI12_PG ((uint16_t)0x0006)
- #define SYSCFG_EXTICR3_EXTI12_PH ((uint16_t)0x0007)
- #define SYSCFG_EXTICR4_EXTI13_PA ((uint16_t)0x0000)

  EXTI13 configuration

- #define SYSCFG_EXTICR4_EXTI13_PB ((uint16_t)0x0010)
- #define SYSCFG_EXTICR4_EXTI13_PC ((uint16_t)0x0020)
- #define SYSCFG_EXTICR4_EXTI13_PD ((uint16_t)0x0030)
- #define SYSCFG_EXTICR4_EXTI13_PE ((uint16_t)0x0040)
- #define SYSCFG_EXTICR4_EXTI13_PF ((uint16_t)0x0050)
- #define SYSCFG_EXTICR4_EXTI13_PG ((uint16_t)0x0060)
- #define SYSCFG_EXTICR3_EXTI13_PH ((uint16_t)0x0070)
- #define SYSCFG_EXTICR4_EXTI14_PA ((uint16_t)0x0000)

EXTI14 configuration

- #define SYSCFG_EXTICR4_EXTI14_PB ((uint16_t)0x0100)
- #define SYSCFG_EXTICR4_EXTI14_PC ((uint16_t)0x0200)
- #define SYSCFG_EXTICR4_EXTI14_PD ((uint16_t)0x0300)
- #define SYSCFG_EXTICR4_EXTI14_PE ((uint16_t)0x0400)
- #define SYSCFG_EXTICR4_EXTI14_PF ((uint16_t)0x0500)
- #define SYSCFG_EXTICR4_EXTI14_PG ((uint16_t)0x0600)
- #define SYSCFG_EXTICR3_EXTI14_PH ((uint16_t)0x0700)
- #define SYSCFG_EXTICR4_EXTI15_PA ((uint16_t)0x0000)

EXTI15 configuration

- #define SYSCFG_EXTICR4_EXTI15_PB ((uint16_t)0x1000)
- #define SYSCFG_EXTICR4_EXTI15_PC ((uint16_t)0x2000)
- #define SYSCFG_EXTICR4_EXTI15_PD ((uint16_t)0x3000)
- #define SYSCFG_EXTICR4_EXTI15_PE ((uint16_t)0x4000)
- #define SYSCFG_EXTICR4_EXTI15_PF ((uint16_t)0x5000)
- #define SYSCFG_EXTICR4_EXTI15_PG ((uint16_t)0x6000)
- #define SYSCFG_EXTICR3_EXTI15_PH ((uint16_t)0x7000)
- #define SYSCFG_CMPCR_CMP_PD ((uint32_t)0x00000001)
- #define SYSCFG_CMPCR_READY ((uint32_t)0x00000100)
- #define TIM_CR1_CEN ((uint16_t)0x0001)
- #define TIM_CR1_UDIS ((uint16_t)0x0002)
- #define TIM_CR1_URS ((uint16_t)0x0004)
- #define TIM_CR1_OPM ((uint16_t)0x0008)
- #define TIM_CR1_DIR ((uint16_t)0x0010)
- #define TIM_CR1_CMS ((uint16_t)0x0060)
- #define TIM_CR1_CMS_0 ((uint16_t)0x0020)
- #define TIM_CR1_CMS_1 ((uint16_t)0x0040)
- #define TIM_CR1_ARPE ((uint16_t)0x0080)
- #define TIM_CR1_CKD ((uint16_t)0x0300)
- #define TIM_CR1_CKD_0 ((uint16_t)0x0100)
- #define TIM_CR1_CKD_1 ((uint16_t)0x0200)
- #define TIM_CR2_CCPC ((uint16_t)0x0001)
- #define TIM_CR2_CCUS ((uint16_t)0x0004)
- #define TIM_CR2_CCDS ((uint16_t)0x0008)
- #define TIM_CR2_MMS ((uint16_t)0x0070)
- #define TIM_CR2_MMS_0 ((uint16_t)0x0010)
- #define TIM_CR2_MMS_1 ((uint16_t)0x0020)
- #define TIM_CR2_MMS_2 ((uint16_t)0x0040)
- #define TIM_CR2_TI1S ((uint16_t)0x0080)
- #define TIM_CR2_OIS1 ((uint16_t)0x0100)
- #define TIM_CR2_OIS1N ((uint16_t)0x0200)
- #define TIM_CR2_OIS2 ((uint16_t)0x0400)
- #define TIM_CR2_OIS2N ((uint16_t)0x0800)
- #define TIM_CR2_OIS3 ((uint16_t)0x1000)
- #define TIM_CR2_OIS3N ((uint16_t)0x2000)
- #define TIM_CR2_OIS4 ((uint16_t)0x4000)
- #define TIM_SMCR_SMS ((uint16_t)0x0007)
- #define TIM_SMCR_SMS_0 ((uint16_t)0x0001)
- #define TIM_SMCR_SMS_1 ((uint16_t)0x0002)
- #define TIM_SMCR_SMS_2 ((uint16_t)0x0004)
- #define TIM_SMCR_TS ((uint16_t)0x0070)
- #define TIM_SMCR_TS_0 ((uint16_t)0x0010)
- #define TIM_SMCR_TS_1 ((uint16_t)0x0020)

- #define TIM_SMCR_TS_2 ((uint16_t)0x0040)
- #define TIM_SMCR_MSM ((uint16_t)0x0080)
- #define TIM_SMCR_ETF ((uint16_t)0x0F00)
- #define TIM_SMCR_ETF_0 ((uint16_t)0x0100)
- #define TIM_SMCR_ETF_1 ((uint16_t)0x0200)
- #define TIM_SMCR_ETF_2 ((uint16_t)0x0400)
- #define TIM_SMCR_ETF_3 ((uint16_t)0x0800)
- #define TIM_SMCR_ETPS ((uint16_t)0x3000)
- #define TIM_SMCR_ETPS_0 ((uint16_t)0x1000)
- #define TIM_SMCR_ETPS_1 ((uint16_t)0x2000)
- #define TIM_SMCR_ECE ((uint16_t)0x4000)
- #define TIM_SMCR_ETP ((uint16_t)0x8000)
- #define TIM_DIER_UIE ((uint16_t)0x0001)
- #define TIM_DIER_CC1IE ((uint16_t)0x0002)
- #define TIM_DIER_CC2IE ((uint16_t)0x0004)
- #define TIM_DIER_CC3IE ((uint16_t)0x0008)
- #define TIM_DIER_CC4IE ((uint16_t)0x0010)
- #define TIM_DIER_COMIE ((uint16_t)0x0020)
- #define TIM_DIER_TIE ((uint16_t)0x0040)
- #define TIM_DIER_BIE ((uint16_t)0x0080)
- #define TIM_DIER_UDE ((uint16_t)0x0100)
- #define TIM_DIER_CC1DE ((uint16_t)0x0200)
- #define TIM_DIER_CC2DE ((uint16_t)0x0400)
- #define TIM_DIER_CC3DE ((uint16_t)0x0800)
- #define TIM_DIER_CC4DE ((uint16_t)0x1000)
- #define TIM_DIER_COMDE ((uint16_t)0x2000)
- #define TIM_DIER_TDE ((uint16_t)0x4000)
- #define TIM_SR_UIF ((uint16_t)0x0001)
- #define TIM_SR_CC1IF ((uint16_t)0x0002)
- #define TIM_SR_CC2IF ((uint16_t)0x0004)
- #define TIM_SR_CC3IF ((uint16_t)0x0008)
- #define TIM_SR_CC4IF ((uint16_t)0x0010)
- #define TIM_SR_COMIF ((uint16_t)0x0020)
- #define TIM_SR_TIF ((uint16_t)0x0040)
- #define TIM_SR_BIF ((uint16_t)0x0080)
- #define TIM_SR_CC1OF ((uint16_t)0x0200)
- #define TIM_SR_CC2OF ((uint16_t)0x0400)
- #define TIM_SR_CC3OF ((uint16_t)0x0800)
- #define TIM_SR_CC4OF ((uint16_t)0x1000)
- #define TIM_EGR_UG ((uint8_t)0x01)
- #define TIM_EGR_CC1G ((uint8_t)0x02)
- #define TIM_EGR_CC2G ((uint8_t)0x04)
- #define TIM_EGR_CC3G ((uint8_t)0x08)
- #define TIM_EGR_CC4G ((uint8_t)0x10)
- #define TIM_EGR_COMG ((uint8_t)0x20)
- #define TIM_EGR_TG ((uint8_t)0x40)
- #define TIM_EGR_BG ((uint8_t)0x80)
- #define TIM_CCMR1_CC1S ((uint16_t)0x0003)
- #define TIM_CCMR1_CC1S_0 ((uint16_t)0x0001)
- #define TIM_CCMR1_CC1S_1 ((uint16_t)0x0002)
- #define TIM_CCMR1_OC1FE ((uint16_t)0x0004)
- #define TIM_CCMR1_OC1PE ((uint16_t)0x0008)
- #define TIM_CCMR1_OC1M ((uint16_t)0x0070)
- #define TIM_CCMR1_OC1M_0 ((uint16_t)0x0010)
- #define TIM_CCMR1_OC1M_1 ((uint16_t)0x0020)

- #define TIM_CCMR1_OC1M_2 ((uint16_t)0x0040)
- #define TIM_CCMR1_OC1CE ((uint16_t)0x0080)
- #define TIM_CCMR1_CC2S ((uint16_t)0x0300)
- #define TIM_CCMR1_CC2S_0 ((uint16_t)0x0100)
- #define TIM_CCMR1_CC2S_1 ((uint16_t)0x0200)
- #define TIM_CCMR1_OC2FE ((uint16_t)0x0400)
- #define TIM_CCMR1_OC2PE ((uint16_t)0x0800)
- #define TIM_CCMR1_OC2M ((uint16_t)0x7000)
- #define TIM_CCMR1_OC2M_0 ((uint16_t)0x1000)
- #define TIM_CCMR1_OC2M_1 ((uint16_t)0x2000)
- #define TIM_CCMR1_OC2M_2 ((uint16_t)0x4000)
- #define TIM_CCMR1_OC2CE ((uint16_t)0x8000)
- #define TIM_CCMR1_IC1PSC ((uint16_t)0x000C)
- #define TIM_CCMR1_IC1PSC_0 ((uint16_t)0x0004)
- #define TIM_CCMR1_IC1PSC_1 ((uint16_t)0x0008)
- #define TIM_CCMR1_IC1F ((uint16_t)0x00F0)
- #define TIM_CCMR1_IC1F_0 ((uint16_t)0x0010)
- #define TIM_CCMR1_IC1F_1 ((uint16_t)0x0020)
- #define TIM_CCMR1_IC1F_2 ((uint16_t)0x0040)
- #define TIM_CCMR1_IC1F_3 ((uint16_t)0x0080)
- #define TIM_CCMR1_IC2PSC ((uint16_t)0x0C00)
- #define TIM_CCMR1_IC2PSC_0 ((uint16_t)0x0400)
- #define TIM_CCMR1_IC2PSC_1 ((uint16_t)0x0800)
- #define TIM_CCMR1_IC2F ((uint16_t)0xF000)
- #define TIM_CCMR1_IC2F_0 ((uint16_t)0x1000)
- #define TIM_CCMR1_IC2F_1 ((uint16_t)0x2000)
- #define TIM_CCMR1_IC2F_2 ((uint16_t)0x4000)
- #define TIM_CCMR1_IC2F_3 ((uint16_t)0x8000)
- #define TIM_CCMR2_CC3S ((uint16_t)0x0003)
- #define TIM_CCMR2_CC3S_0 ((uint16_t)0x0001)
- #define TIM_CCMR2_CC3S_1 ((uint16_t)0x0002)
- #define TIM_CCMR2_OC3FE ((uint16_t)0x0004)
- #define TIM_CCMR2_OC3PE ((uint16_t)0x0008)
- #define TIM_CCMR2_OC3M ((uint16_t)0x0070)
- #define TIM_CCMR2_OC3M_0 ((uint16_t)0x0010)
- #define TIM_CCMR2_OC3M_1 ((uint16_t)0x0020)
- #define TIM_CCMR2_OC3M_2 ((uint16_t)0x0040)
- #define TIM_CCMR2_OC3CE ((uint16_t)0x0080)
- #define TIM_CCMR2_CC4S ((uint16_t)0x0300)
- #define TIM_CCMR2_CC4S_0 ((uint16_t)0x0100)
- #define TIM_CCMR2_CC4S_1 ((uint16_t)0x0200)
- #define TIM_CCMR2_OC4FE ((uint16_t)0x0400)
- #define TIM_CCMR2_OC4PE ((uint16_t)0x0800)
- #define TIM_CCMR2_OC4M ((uint16_t)0x7000)
- #define TIM_CCMR2_OC4M_0 ((uint16_t)0x1000)
- #define TIM_CCMR2_OC4M_1 ((uint16_t)0x2000)
- #define TIM_CCMR2_OC4M_2 ((uint16_t)0x4000)
- #define TIM_CCMR2_OC4CE ((uint16_t)0x8000)
- #define TIM_CCMR2_IC3PSC ((uint16_t)0x000C)
- #define TIM_CCMR2_IC3PSC_0 ((uint16_t)0x0004)
- #define TIM_CCMR2_IC3PSC_1 ((uint16_t)0x0008)
- #define TIM_CCMR2_IC3F ((uint16_t)0x00F0)
- #define TIM_CCMR2_IC3F_0 ((uint16_t)0x0010)
- #define TIM_CCMR2_IC3F_1 ((uint16_t)0x0020)
- #define TIM_CCMR2_IC3F_2 ((uint16_t)0x0040)

- #define TIM_CCMR2_IC3F_3 ((uint16_t)0x0080)
- #define TIM_CCMR2_IC4PSC ((uint16_t)0x0C00)
- #define TIM_CCMR2_IC4PSC_0 ((uint16_t)0x0400)
- #define TIM_CCMR2_IC4PSC_1 ((uint16_t)0x0800)
- #define TIM_CCMR2_IC4F ((uint16_t)0xF000)
- #define TIM_CCMR2_IC4F_0 ((uint16_t)0x1000)
- #define TIM_CCMR2_IC4F_1 ((uint16_t)0x2000)
- #define TIM_CCMR2_IC4F_2 ((uint16_t)0x4000)
- #define TIM_CCMR2_IC4F_3 ((uint16_t)0x8000)
- #define TIM_CCER_CC1E ((uint16_t)0x0001)
- #define TIM_CCER_CC1P ((uint16_t)0x0002)
- #define TIM_CCER_CC1NE ((uint16_t)0x0004)
- #define TIM_CCER_CC1NP ((uint16_t)0x0008)
- #define TIM_CCER_CC2E ((uint16_t)0x0010)
- #define TIM_CCER_CC2P ((uint16_t)0x0020)
- #define TIM_CCER_CC2NE ((uint16_t)0x0040)
- #define TIM_CCER_CC2NP ((uint16_t)0x0080)
- #define TIM_CCER_CC3E ((uint16_t)0x0100)
- #define TIM_CCER_CC3P ((uint16_t)0x0200)
- #define TIM_CCER_CC3NE ((uint16_t)0x0400)
- #define TIM_CCER_CC3NP ((uint16_t)0x0800)
- #define TIM_CCER_CC4E ((uint16_t)0x1000)
- #define TIM_CCER_CC4P ((uint16_t)0x2000)
- #define TIM_CCER_CC4NP ((uint16_t)0x8000)
- #define TIM_CNT_CNT ((uint16_t)0xFFFF)
- #define TIM_PSC_PSC ((uint16_t)0xFFFF)
- #define TIM_ARR_ARR ((uint16_t)0xFFFF)
- #define TIM_RCR_REP ((uint8_t)0xFF)
- #define TIM_CCR1_CCR1 ((uint16_t)0xFFFF)
- #define TIM_CCR2_CCR2 ((uint16_t)0xFFFF)
- #define TIM_CCR3_CCR3 ((uint16_t)0xFFFF)
- #define TIM_CCR4_CCR4 ((uint16_t)0xFFFF)
- #define TIM_BDTR_DTG ((uint16_t)0x00FF)
- #define TIM_BDTR_DTG_0 ((uint16_t)0x0001)
- #define TIM_BDTR_DTG_1 ((uint16_t)0x0002)
- #define TIM_BDTR_DTG_2 ((uint16_t)0x0004)
- #define TIM_BDTR_DTG_3 ((uint16_t)0x0008)
- #define TIM_BDTR_DTG_4 ((uint16_t)0x0010)
- #define TIM_BDTR_DTG_5 ((uint16_t)0x0020)
- #define TIM_BDTR_DTG_6 ((uint16_t)0x0040)
- #define TIM_BDTR_DTG_7 ((uint16_t)0x0080)
- #define TIM_BDTR_LOCK ((uint16_t)0x0300)
- #define TIM_BDTR_LOCK_0 ((uint16_t)0x0100)
- #define TIM_BDTR_LOCK_1 ((uint16_t)0x0200)
- #define TIM_BDTR_OSSI ((uint16_t)0x0400)
- #define TIM_BDTR_OSSR ((uint16_t)0x0800)
- #define TIM_BDTR_BKE ((uint16_t)0x1000)
- #define TIM_BDTR_BKP ((uint16_t)0x2000)
- #define TIM_BDTR_AOE ((uint16_t)0x4000)
- #define TIM_BDTR_MOE ((uint16_t)0x8000)
- #define TIM_DCR_DBA ((uint16_t)0x001F)
- #define TIM_DCR_DBA_0 ((uint16_t)0x0001)
- #define TIM_DCR_DBA_1 ((uint16_t)0x0002)
- #define TIM_DCR_DBA_2 ((uint16_t)0x0004)
- #define TIM_DCR_DBA_3 ((uint16_t)0x0008)

- #define TIM_DCR_DBA_4 ((uint16_t)0x0010)
- #define TIM_DCR_DBL ((uint16_t)0x1F00)
- #define TIM_DCR_DBL_0 ((uint16_t)0x0100)
- #define TIM_DCR_DBL_1 ((uint16_t)0x0200)
- #define TIM_DCR_DBL_2 ((uint16_t)0x0400)
- #define TIM_DCR_DBL_3 ((uint16_t)0x0800)
- #define TIM_DCR_DBL_4 ((uint16_t)0x1000)
- #define TIM_DMAR_DMAB ((uint16_t)0xFFFF)
- #define TIM_OR_TI4_RMP ((uint16_t)0x00C0)
- #define TIM_OR_TI4_RMP_0 ((uint16_t)0x0040)
- #define TIM_OR_TI4_RMP_1 ((uint16_t)0x0080)
- #define TIM_OR_ITR1_RMP ((uint16_t)0x0C00)
- #define TIM_OR_ITR1_RMP_0 ((uint16_t)0x0400)
- #define TIM_OR_ITR1_RMP_1 ((uint16_t)0x0800)
- #define USART_SR_PE ((uint16_t)0x0001)
- #define USART_SR_FE ((uint16_t)0x0002)
- #define USART_SR_NE ((uint16_t)0x0004)
- #define USART_SR_ORE ((uint16_t)0x0008)
- #define USART_SR_IDLE ((uint16_t)0x0010)
- #define USART_SR_RXNE ((uint16_t)0x0020)
- #define USART_SR_TC ((uint16_t)0x0040)
- #define USART_SR_TXE ((uint16_t)0x0080)
- #define USART_SR_LBD ((uint16_t)0x0100)
- #define USART_SR_CTS ((uint16_t)0x0200)
- #define USART_DR_DR ((uint16_t)0x01FF)
- #define USART_BRR_DIV_Fraction ((uint16_t)0x000F)
- #define USART_BRR_DIV_Mantissa ((uint16_t)0xFFF0)
- #define USART_CR1_SBK ((uint16_t)0x0001)
- #define USART_CR1_RWU ((uint16_t)0x0002)
- #define USART_CR1_RE ((uint16_t)0x0004)
- #define USART_CR1_TE ((uint16_t)0x0008)
- #define USART_CR1_IDLEIE ((uint16_t)0x0010)
- #define USART_CR1_RXNEIE ((uint16_t)0x0020)
- #define USART_CR1_TCIE ((uint16_t)0x0040)
- #define USART_CR1_TXEIE ((uint16_t)0x0080)
- #define USART_CR1_PEIE ((uint16_t)0x0100)
- #define USART_CR1_PS ((uint16_t)0x0200)
- #define USART_CR1_PCE ((uint16_t)0x0400)
- #define USART_CR1_WAKE ((uint16_t)0x0800)
- #define USART_CR1_M ((uint16_t)0x1000)
- #define USART_CR1_UE ((uint16_t)0x2000)
- #define USART_CR1_OVER8 ((uint16_t)0x8000)
- #define USART_CR2_ADD ((uint16_t)0x000F)
- #define USART_CR2_LBDL ((uint16_t)0x0020)
- #define USART_CR2_LBDIE ((uint16_t)0x0040)
- #define USART_CR2_LBCL ((uint16_t)0x0100)
- #define USART_CR2_CPHA ((uint16_t)0x0200)
- #define USART_CR2_CPOL ((uint16_t)0x0400)
- #define USART_CR2_CLKEN ((uint16_t)0x0800)
- #define USART_CR2_STOP ((uint16_t)0x3000)
- #define USART_CR2_STOP_0 ((uint16_t)0x1000)
- #define USART_CR2_STOP_1 ((uint16_t)0x2000)
- #define USART_CR2_LINEN ((uint16_t)0x4000)
- #define USART_CR3_EIE ((uint16_t)0x0001)
- #define USART_CR3_IREN ((uint16_t)0x0002)

- #define USART_CR3_IRLP ((uint16_t)0x0004)
- #define USART_CR3_HDSEL ((uint16_t)0x0008)
- #define USART_CR3_NACK ((uint16_t)0x0010)
- #define USART_CR3_SCEN ((uint16_t)0x0020)
- #define USART_CR3_DMAR ((uint16_t)0x0040)
- #define USART_CR3_DMAT ((uint16_t)0x0080)
- #define USART_CR3_RTSE ((uint16_t)0x0100)
- #define USART_CR3_CTSE ((uint16_t)0x0200)
- #define USART_CR3_CTSIE ((uint16_t)0x0400)
- #define USART_CR3_ONEBIT ((uint16_t)0x0800)
- #define USART_GTPR_PSC ((uint16_t)0x00FF)
- #define USART_GTPR_PSC_0 ((uint16_t)0x0001)
- #define USART_GTPR_PSC_1 ((uint16_t)0x0002)
- #define USART_GTPR_PSC_2 ((uint16_t)0x0004)
- #define USART_GTPR_PSC_3 ((uint16_t)0x0008)
- #define USART_GTPR_PSC_4 ((uint16_t)0x0010)
- #define USART_GTPR_PSC_5 ((uint16_t)0x0020)
- #define USART_GTPR_PSC_6 ((uint16_t)0x0040)
- #define USART_GTPR_PSC_7 ((uint16_t)0x0080)
- #define USART_GTPR_GT ((uint16_t)0xFF00)
- #define WWDG_CR_T ((uint8_t)0x7F)
- #define WWDG_CR_T0 ((uint8_t)0x01)
- #define WWDG_CR_T1 ((uint8_t)0x02)
- #define WWDG_CR_T2 ((uint8_t)0x04)
- #define WWDG_CR_T3 ((uint8_t)0x08)
- #define WWDG_CR_T4 ((uint8_t)0x10)
- #define WWDG_CR_T5 ((uint8_t)0x20)
- #define WWDG_CR_T6 ((uint8_t)0x40)
- #define WWDG_CR_WDGA ((uint8_t)0x80)
- #define WWDG_CFR_W ((uint16_t)0x007F)
- #define WWDG_CFR_W0 ((uint16_t)0x0001)
- #define WWDG_CFR_W1 ((uint16_t)0x0002)
- #define WWDG_CFR_W2 ((uint16_t)0x0004)
- #define WWDG_CFR_W3 ((uint16_t)0x0008)
- #define WWDG_CFR_W4 ((uint16_t)0x0010)
- #define WWDG_CFR_W5 ((uint16_t)0x0020)
- #define WWDG_CFR_W6 ((uint16_t)0x0040)
- #define WWDG_CFR_WDGTB ((uint16_t)0x0180)
- #define WWDG_CFR_WDGTB0 ((uint16_t)0x0080)
- #define WWDG_CFR_WDGTB1 ((uint16_t)0x0100)
- #define WWDG_CFR_EWI ((uint16_t)0x0200)
- #define WWDG_SR_EWIF ((uint8_t)0x01)
- #define **DBGMCU_IDCODE_DEV_ID** ((uint32_t)0x00000FFF)
- #define **DBGMCU_IDCODE_REV_ID** ((uint32_t)0xFFFF0000)
- #define **DBGMCU_CR_DBG_SLEEP** ((uint32_t)0x00000001)
- #define **DBGMCU_CR_DBG_STOP** ((uint32_t)0x00000002)
- #define **DBGMCU_CR_DBG_STANDBY** ((uint32_t)0x00000004)
- #define **DBGMCU_CR_TRACE_IOEN** ((uint32_t)0x00000020)
- #define **DBGMCU_CR_TRACE_MODE** ((uint32_t)0x000000C0)
- #define DBGMCU_CR_TRACE_MODE_0 ((uint32_t)0x00000040)
- #define DBGMCU_CR_TRACE_MODE_1 ((uint32_t)0x00000080)
- #define **DBGMCU_APB1_FZ_DBG_TIM2_STOP** ((uint32_t)0x00000001)
- #define **DBGMCU_APB1_FZ_DBG_TIM3_STOP** ((uint32_t)0x00000002)
- #define **DBGMCU_APB1_FZ_DBG_TIM4_STOP** ((uint32_t)0x00000004)
- #define **DBGMCU_APB1_FZ_DBG_TIM5_STOP** ((uint32_t)0x00000008)

- #define **DBGMCU_APB1_FZ_DBG_TIM6_STOP** ((uint32_t)0x00000010)
- #define **DBGMCU_APB1_FZ_DBG_TIM7_STOP** ((uint32_t)0x00000020)
- #define **DBGMCU_APB1_FZ_DBG_TIM12_STOP** ((uint32_t)0x00000040)
- #define **DBGMCU_APB1_FZ_DBG_TIM13_STOP** ((uint32_t)0x00000080)
- #define **DBGMCU_APB1_FZ_DBG_TIM14_STOP** ((uint32_t)0x00000100)
- #define **DBGMCU_APB1_FZ_DBG_RTC_STOP** ((uint32_t)0x00000400)
- #define **DBGMCU_APB1_FZ_DBG_WWDG_STOP** ((uint32_t)0x00000800)
- #define **DBGMCU_APB1_FZ_DBG_IWDG_STOP** ((uint32_t)0x00001000)
- #define **DBGMCU_APB1_FZ_DBG_I2C1_SMBUS_TIMEOUT** ((uint32_t)0x00200000)
- #define **DBGMCU_APB1_FZ_DBG_I2C2_SMBUS_TIMEOUT** ((uint32_t)0x00400000)
- #define **DBGMCU_APB1_FZ_DBG_I2C3_SMBUS_TIMEOUT** ((uint32_t)0x00800000)
- #define **DBGMCU_APB1_FZ_DBG_CAN1_STOP** ((uint32_t)0x02000000)
- #define **DBGMCU_APB1_FZ_DBG_CAN2_STOP** ((uint32_t)0x04000000)
- #define **DBGMCU_APB1_FZ_DBG_IWDEG_STOP** DBGMCU_APB1_FZ_DBG_IWDG_STOP
- #define **DBGMCU_APB1_FZ_DBG_TIM1_STOP** ((uint32_t)0x00000001)
- #define **DBGMCU_APB1_FZ_DBG_TIM8_STOP** ((uint32_t)0x00000002)
- #define **DBGMCU_APB1_FZ_DBG_TIM9_STOP** ((uint32_t)0x00010000)
- #define **DBGMCU_APB1_FZ_DBG_TIM10_STOP** ((uint32_t)0x00020000)
- #define **DBGMCU_APB1_FZ_DBG_TIM11_STOP** ((uint32_t)0x00040000)
- #define **ETH_MACCR_WD** ((uint32_t)0x00800000) /* Watchdog disable */
- #define **ETH_MACCR_JD** ((uint32_t)0x00400000) /* Jabber disable */
- #define **ETH_MACCR_IFG** ((uint32_t)0x000E0000) /* Inter-frame gap */
- #define **ETH_MACCR_IFG_96Bit** ((uint32_t)0x00000000) /* Minimum IFG between frames during transmission is 96Bit */
- #define **ETH_MACCR_IFG_88Bit** ((uint32_t)0x00020000) /* Minimum IFG between frames during transmission is 88Bit */
- #define **ETH_MACCR_IFG_80Bit** ((uint32_t)0x00040000) /* Minimum IFG between frames during transmission is 80Bit */
- #define **ETH_MACCR_IFG_72Bit** ((uint32_t)0x00060000) /* Minimum IFG between frames during transmission is 72Bit */
- #define **ETH_MACCR_IFG_64Bit** ((uint32_t)0x00080000) /* Minimum IFG between frames during transmission is 64Bit */
- #define **ETH_MACCR_IFG_56Bit** ((uint32_t)0x000A0000) /* Minimum IFG between frames during transmission is 56Bit */
- #define **ETH_MACCR_IFG_48Bit** ((uint32_t)0x000C0000) /* Minimum IFG between frames during transmission is 48Bit */
- #define **ETH_MACCR_IFG_40Bit** ((uint32_t)0x000E0000) /* Minimum IFG between frames during transmission is 40Bit */
- #define **ETH_MACCR_CSD** ((uint32_t)0x00010000) /* Carrier sense disable (during transmission) */
- #define **ETH_MACCR_FES** ((uint32_t)0x00004000) /* Fast ethernet speed */
- #define **ETH_MACCR_ROD** ((uint32_t)0x00002000) /* Receive own disable */
- #define **ETH_MACCR_LM** ((uint32_t)0x00001000) /* loopback mode */
- #define **ETH_MACCR_DM** ((uint32_t)0x00000800) /* Duplex mode */
- #define **ETH_MACCR_IPCO** ((uint32_t)0x00000400) /* IP Checksum offload */
- #define **ETH_MACCR_RD** ((uint32_t)0x00000200) /* Retry disable */
- #define **ETH_MACCR_APCS** ((uint32_t)0x00000080) /* Automatic Pad/CRC stripping */
- #define ETH_MACCR_BL
- #define **ETH_MACCR_BL_10** ((uint32_t)0x00000000) /* k = min (n, 10) */
- #define **ETH_MACCR_BL_8** ((uint32_t)0x00000020) /* k = min (n, 8) */
- #define **ETH_MACCR_BL_4** ((uint32_t)0x00000040) /* k = min (n, 4) */
- #define **ETH_MACCR_BL_1** ((uint32_t)0x00000060) /* k = min (n, 1) */
- #define **ETH_MACCR_DC** ((uint32_t)0x00000010) /* Defferal check */
- #define **ETH_MACCR_TE** ((uint32_t)0x00000008) /* Transmitter enable */
- #define **ETH_MACCR_RE** ((uint32_t)0x00000004) /* Receiver enable */
- #define **ETH_MACFFR_RA** ((uint32_t)0x80000000) /* Receive all */

- #define **ETH_MACFFR_HPF** ((uint32_t)0x00000400) /* Hash or perfect filter */
- #define **ETH_MACFFR_SAF** ((uint32_t)0x00000200) /* Source address filter enable */
- #define **ETH_MACFFR_SAIF** ((uint32_t)0x00000100) /* SA inverse filtering */
- #define **ETH_MACFFR_PCF** ((uint32_t)0x000000C0) /* Pass control frames: 3 cases */
- #define **ETH_MACFFR_PCF_BlockAll** ((uint32_t)0x00000040) /* MAC filters all control frames from reaching the application */
- #define **ETH_MACFFR_PCF_ForwardAll** ((uint32_t)0x00000080) /* MAC forwards all control frames to application even if they fail the Address Filter */
- #define **ETH_MACFFR_PCF_ForwardPassedAddrFilter** ((uint32_t)0x000000C0) /* MAC forwards control frames that pass the Address Filter. */
- #define **ETH_MACFFR_BFD** ((uint32_t)0x00000020) /* Broadcast frame disable */
- #define **ETH_MACFFR_PAM** ((uint32_t)0x00000010) /* Pass all mutlicast */
- #define **ETH_MACFFR_DAIF** ((uint32_t)0x00000008) /* DA Inverse filtering */
- #define **ETH_MACFFR_HM** ((uint32_t)0x00000004) /* Hash multicast */
- #define **ETH_MACFFR_HU** ((uint32_t)0x00000002) /* Hash unicast */
- #define **ETH_MACFFR_PM** ((uint32_t)0x00000001) /* Promiscuous mode */
- #define **ETH_MACHTHR_HTH** ((uint32_t)0xFFFFFFFF) /* Hash table high */
- #define **ETH_MACHTLR_HTL** ((uint32_t)0xFFFFFFFF) /* Hash table low */
- #define **ETH_MACMIIAR_PA** ((uint32_t)0x0000F800) /* Physical layer address */
- #define **ETH_MACMIIAR_MR** ((uint32_t)0x000007C0) /* MII register in the selected PHY */
- #define **ETH_MACMIIAR_CR** ((uint32_t)0x0000001C) /* CR clock range: 6 cases */
- #define **ETH_MACMIIAR_CR_Div42** ((uint32_t)0x00000000) /* HCLK:60-100 MHz; MDC clock= HCLK/42 */
- #define **ETH_MACMIIAR_CR_Div62** ((uint32_t)0x00000004) /* HCLK:100-150 MHz; MDC clock= HCLK/62 */
- #define **ETH_MACMIIAR_CR_Div16** ((uint32_t)0x00000008) /* HCLK:20-35 MHz; MDC clock= HCLK/16 */
- #define **ETH_MACMIIAR_CR_Div26** ((uint32_t)0x0000000C) /* HCLK:35-60 MHz; MDC clock= HCLK/26 */
- #define **ETH_MACMIIAR_CR_Div102** ((uint32_t)0x00000010) /* HCLK:150-168 MHz; MDC clock= HCLK/102 */
- #define **ETH_MACMIIAR_MW** ((uint32_t)0x00000002) /* MII write */
- #define **ETH_MACMIIAR_MB** ((uint32_t)0x00000001) /* MII busy */
- #define **ETH_MACMIIDR_MD** ((uint32_t)0x0000FFFF) /* MII data: read/write data from/to PHY */
- #define **ETH_MACFCR_PT** ((uint32_t)0xFFFF0000) /* Pause time */
- #define **ETH_MACFCR_ZQPD** ((uint32_t)0x00000080) /* Zero-quanta pause disable */
- #define **ETH_MACFCR_PLT** ((uint32_t)0x00000030) /* Pause low threshold: 4 cases */
- #define **ETH_MACFCR_PLT_Minus4** ((uint32_t)0x00000000) /* Pause time minus 4 slot times */
- #define **ETH_MACFCR_PLT_Minus28** ((uint32_t)0x00000010) /* Pause time minus 28 slot times */
- #define **ETH_MACFCR_PLT_Minus144** ((uint32_t)0x00000020) /* Pause time minus 144 slot times */
- #define **ETH_MACFCR_PLT_Minus256** ((uint32_t)0x00000030) /* Pause time minus 256 slot times */
- #define **ETH_MACFCR_UPFD** ((uint32_t)0x00000008) /* Unicast pause frame detect */
- #define **ETH_MACFCR_RFCE** ((uint32_t)0x00000004) /* Receive flow control enable */
- #define **ETH_MACFCR_TFCE** ((uint32_t)0x00000002) /* Transmit flow control enable */
- #define **ETH_MACFCR_FCBBPA** ((uint32_t)0x00000001) /* Flow control busy/backpressure activate */
- #define **ETH_MACVLANTR_VLANTC** ((uint32_t)0x00010000) /* 12-bit VLAN tag comparison */
- #define **ETH_MACVLANTR_VLANTI** ((uint32_t)0x0000FFFF) /* VLAN tag identifier (for receive frames) */
- #define **ETH_MACRWUFFR_D** ((uint32_t)0xFFFFFFFF) /* Wake-up frame filter register data */
- #define **ETH_MACPMTCSR_WFFRPR** ((uint32_t)0x80000000) /* Wake-Up Frame Filter Register Pointer Reset */
- #define **ETH_MACPMTCSR_GU** ((uint32_t)0x00000200) /* Global Unicast */
- #define **ETH_MACPMTCSR_WFR** ((uint32_t)0x00000040) /* Wake-Up Frame Received */
- #define **ETH_MACPMTCSR_MPR** ((uint32_t)0x00000020) /* Magic Packet Received */
- #define **ETH_MACPMTCSR_WFE** ((uint32_t)0x00000004) /* Wake-Up Frame Enable */
- #define **ETH_MACPMTCSR_MPE** ((uint32_t)0x00000002) /* Magic Packet Enable */
- #define **ETH_MACPMTCSR_PD** ((uint32_t)0x00000001) /* Power Down */
- #define **ETH_MACSR_TSTS** ((uint32_t)0x00000200) /* Time stamp trigger status */

- #define **ETH_MACSR_MMCTS** ((uint32_t)0x00000040) /* MMC transmit status */
- #define **ETH_MACSR_MMMCRS** ((uint32_t)0x00000020) /* MMC receive status */
- #define **ETH_MACSR_MMCS** ((uint32_t)0x00000010) /* MMC status */
- #define **ETH_MACSR_PMTS** ((uint32_t)0x00000008) /* PMT status */
- #define **ETH_MACIMR_TSTIM** ((uint32_t)0x00000200) /* Time stamp trigger interrupt mask */
- #define **ETH_MACIMR_PMTIM** ((uint32_t)0x00000008) /* PMT interrupt mask */
- #define **ETH_MACA0HR_MACA0H** ((uint32_t)0x0000FFFF) /* MAC address0 high */
- #define **ETH_MACA0LR_MACA0L** ((uint32_t)0xFFFFFFFF) /* MAC address0 low */
- #define **ETH_MACA1HR_AE** ((uint32_t)0x80000000) /* Address enable */
- #define **ETH_MACA1HR_SA** ((uint32_t)0x40000000) /* Source address */
- #define **ETH_MACA1HR_MBC** ((uint32_t)0x3F000000) /* Mask byte control: bits to mask for comparison of the MAC Address bytes */
- #define **ETH_MACA1HR_MBC_HBits15_8** ((uint32_t)0x20000000) /* Mask MAC Address high reg bits [15:8] */
- #define **ETH_MACA1HR_MBC_HBits7_0** ((uint32_t)0x10000000) /* Mask MAC Address high reg bits [7:0] */
- #define **ETH_MACA1HR_MBC_LBits31_24** ((uint32_t)0x08000000) /* Mask MAC Address low reg bits [31:24] */
- #define **ETH_MACA1HR_MBC_LBits23_16** ((uint32_t)0x04000000) /* Mask MAC Address low reg bits [23:16] */
- #define **ETH_MACA1HR_MBC_LBits15_8** ((uint32_t)0x02000000) /* Mask MAC Address low reg bits [15:8] */
- #define **ETH_MACA1HR_MBC_LBits7_0** ((uint32_t)0x01000000) /* Mask MAC Address low reg bits [7:0] */
- #define **ETH_MACA1HR_MACA1H** ((uint32_t)0x0000FFFF) /* MAC address1 high */
- #define **ETH_MACA1LR_MACA1L** ((uint32_t)0xFFFFFFFF) /* MAC address1 low */
- #define **ETH_MACA2HR_AE** ((uint32_t)0x80000000) /* Address enable */
- #define **ETH_MACA2HR_SA** ((uint32_t)0x40000000) /* Source address */
- #define **ETH_MACA2HR_MBC** ((uint32_t)0x3F000000) /* Mask byte control */
- #define **ETH_MACA2HR_MBC_HBits15_8** ((uint32_t)0x20000000) /* Mask MAC Address high reg bits [15:8] */
- #define **ETH_MACA2HR_MBC_HBits7_0** ((uint32_t)0x10000000) /* Mask MAC Address high reg bits [7:0] */
- #define **ETH_MACA2HR_MBC_LBits31_24** ((uint32_t)0x08000000) /* Mask MAC Address low reg bits [31:24] */
- #define **ETH_MACA2HR_MBC_LBits23_16** ((uint32_t)0x04000000) /* Mask MAC Address low reg bits [23:16] */
- #define **ETH_MACA2HR_MBC_LBits15_8** ((uint32_t)0x02000000) /* Mask MAC Address low reg bits [15:8] */
- #define **ETH_MACA2HR_MBC_LBits7_0** ((uint32_t)0x01000000) /* Mask MAC Address low reg bits [70] */
- #define **ETH_MACA2HR_MACA2H** ((uint32_t)0x0000FFFF) /* MAC address1 high */
- #define **ETH_MACA2LR_MACA2L** ((uint32_t)0xFFFFFFFF) /* MAC address2 low */
- #define **ETH_MACA3HR_AE** ((uint32_t)0x80000000) /* Address enable */
- #define **ETH_MACA3HR_SA** ((uint32_t)0x40000000) /* Source address */
- #define **ETH_MACA3HR_MBC** ((uint32_t)0x3F000000) /* Mask byte control */
- #define **ETH_MACA3HR_MBC_HBits15_8** ((uint32_t)0x20000000) /* Mask MAC Address high reg bits [15:8] */
- #define **ETH_MACA3HR_MBC_HBits7_0** ((uint32_t)0x10000000) /* Mask MAC Address high reg bits [7:0] */
- #define **ETH_MACA3HR_MBC_LBits31_24** ((uint32_t)0x08000000) /* Mask MAC Address low reg bits [31:24] */
- #define **ETH_MACA3HR_MBC_LBits23_16** ((uint32_t)0x04000000) /* Mask MAC Address low reg bits [23:16] */
- #define **ETH_MACA3HR_MBC_LBits15_8** ((uint32_t)0x02000000) /* Mask MAC Address low reg bits [15:8] */
- #define **ETH_MACA3HR_MBC_LBits7_0** ((uint32_t)0x01000000) /* Mask MAC Address low reg bits [70] */
- #define **ETH_MACA3HR_MACA3H** ((uint32_t)0x0000FFFF) /* MAC address3 high */

- #define **ETH_MACA3LR_MACA3L** ((uint32_t)0xFFFFFFFF) /* MAC address3 low */
- #define **ETH_MMCCR_MCFHP** ((uint32_t)0x00000020) /* MMC counter Full-Half preset */
- #define **ETH_MMCCR_MCP** ((uint32_t)0x00000010) /* MMC counter preset */
- #define **ETH_MMCCR_MCF** ((uint32_t)0x00000008) /* MMC Counter Freeze */
- #define **ETH_MMCCR_ROR** ((uint32_t)0x00000004) /* Reset on Read */
- #define **ETH_MMCCR_CSR** ((uint32_t)0x00000002) /* Counter Stop Rollover */
- #define **ETH_MMCCR_CR** ((uint32_t)0x00000001) /* Counters Reset */
- #define **ETH_MMCRIR_RGUFS** ((uint32_t)0x00020000) /* Set when Rx good unicast frames counter reaches half the maximum value */
- #define **ETH_MMCRIR_RFAES** ((uint32_t)0x00000040) /* Set when Rx alignment error counter reaches half the maximum value */
- #define **ETH_MMCRIR_RFCES** ((uint32_t)0x00000020) /* Set when Rx crc error counter reaches half the maximum value */
- #define **ETH_MMCTIR_TGFS** ((uint32_t)0x00200000) /* Set when Tx good frame count counter reaches half the maximum value */
- #define **ETH_MMCTIR_TGFMSCS** ((uint32_t)0x00008000) /* Set when Tx good multi col counter reaches half the maximum value */
- #define **ETH_MMCTIR_TGFSCS** ((uint32_t)0x00004000) /* Set when Tx good single col counter reaches half the maximum value */
- #define **ETH_MMCRIMR_RGUFM** ((uint32_t)0x00020000) /* Mask the interrupt when Rx good unicast frames counter reaches half the maximum value */
- #define **ETH_MMCRIMR_RFAEM** ((uint32_t)0x00000040) /* Mask the interrupt when when Rx alignment error counter reaches half the maximum value */
- #define **ETH_MMCRIMR_RFCEM** ((uint32_t)0x00000020) /* Mask the interrupt when Rx crc error counter reaches half the maximum value */
- #define **ETH_MMCTIMR_TGFM** ((uint32_t)0x00200000) /* Mask the interrupt when Tx good frame count counter reaches half the maximum value */
- #define **ETH_MMCTIMR_TGFMSCM** ((uint32_t)0x00008000) /* Mask the interrupt when Tx good multi col counter reaches half the maximum value */
- #define **ETH_MMCTIMR_TGFSCM** ((uint32_t)0x00004000) /* Mask the interrupt when Tx good single col counter reaches half the maximum value */
- #define **ETH_MMCTGFSCCR_TGFSCC** ((uint32_t)0xFFFFFFFF) /* Number of successfully transmitted frames after a single collision in Half-duplex mode. */
- #define **ETH_MMCTGFMSCCR_TGFMSCC** ((uint32_t)0xFFFFFFFF) /* Number of successfully transmitted frames after more than a single collision in Half-duplex mode. */
- #define **ETH_MMCTGFCR_TGFC** ((uint32_t)0xFFFFFFFF) /* Number of good frames transmitted. */
- #define **ETH_MMCRFCECR_RFCEC** ((uint32_t)0xFFFFFFFF) /* Number of frames received with CRC error. */
- #define **ETH_MMCRFAECR_RFAEC** ((uint32_t)0xFFFFFFFF) /* Number of frames received with alignment (dribble) error */
- #define **ETH_MMCRGUFCR_RGUFC** ((uint32_t)0xFFFFFFFF) /* Number of good unicast frames received. */
- #define **ETH_PTPTSCR_TSCNT** ((uint32_t)0x00030000) /* Time stamp clock node type */
- #define **ETH_PTPTSSR_TSSMRME** ((uint32_t)0x00008000) /* Time stamp snapshot for message relevant to master enable */
- #define **ETH_PTPTSSR_TSSEME** ((uint32_t)0x00004000) /* Time stamp snapshot for event message enable */
- #define **ETH_PTPTSSR_TSSIPV4FE** ((uint32_t)0x00002000) /* Time stamp snapshot for IPv4 frames enable */
- #define **ETH_PTPTSSR_TSSIPV6FE** ((uint32_t)0x00001000) /* Time stamp snapshot for IPv6 frames enable */
- #define **ETH_PTPTSSR_TSSPTPOEFE** ((uint32_t)0x00000800) /* Time stamp snapshot for PTP over ethernet frames enable */
- #define **ETH_PTPTSSR_TSPTPPSV2E** ((uint32_t)0x00000400) /* Time stamp PTP packet snooping for version2 format enable */
- #define **ETH_PTPTSSR_TSSSR** ((uint32_t)0x00000200) /* Time stamp Sub-seconds rollover */

- #define **ETH_PTPTSSR_TSSARFE** ((uint32_t)0x00000100) /* Time stamp snapshot for all received frames enable */
- #define **ETH_PTPTSCR_TSARU** ((uint32_t)0x00000020) /* Addend register update */
- #define **ETH_PTPTSCR_TSITE** ((uint32_t)0x00000010) /* Time stamp interrupt trigger enable */
- #define **ETH_PTPTSCR_TSSTU** ((uint32_t)0x00000008) /* Time stamp update */
- #define **ETH_PTPTSCR_TSSTI** ((uint32_t)0x00000004) /* Time stamp initialize */
- #define **ETH_PTPTSCR_TSFCU** ((uint32_t)0x00000002) /* Time stamp fine or coarse update */
- #define **ETH_PTPTSCR_TSE** ((uint32_t)0x00000001) /* Time stamp enable */
- #define **ETH_PTPSSIR_STSSI** ((uint32_t)0x000000FF) /* System time Sub-second increment value */
- #define **ETH_PTPTSHR_STS** ((uint32_t)0xFFFFFFFF) /* System Time second */
- #define **ETH_PTPTSLR_STPNS** ((uint32_t)0x80000000) /* System Time Positive or negative time */
- #define **ETH_PTPTSLR_STSS** ((uint32_t)0x7FFFFFFF) /* System Time sub-seconds */
- #define **ETH_PTPTSHUR_TSUS** ((uint32_t)0xFFFFFFFF) /* Time stamp update seconds */
- #define **ETH_PTPTSLUR_TSUPNS** ((uint32_t)0x80000000) /* Time stamp update Positive or negative time */
- #define **ETH_PTPTSLUR_TSUSS** ((uint32_t)0x7FFFFFFF) /* Time stamp update sub-seconds */
- #define **ETH_PTPTSAR_TSA** ((uint32_t)0xFFFFFFFF) /* Time stamp addend */
- #define **ETH_PTPTTHR_TTSH** ((uint32_t)0xFFFFFFFF) /* Target time stamp high */
- #define **ETH_PTPTTLR_TTSL** ((uint32_t)0xFFFFFFFF) /* Target time stamp low */
- #define **ETH_PTPTSSR_TSTTR** ((uint32_t)0x00000020) /* Time stamp target time reached */
- #define **ETH_PTPTSSR_TSSO** ((uint32_t)0x00000010) /* Time stamp seconds overflow */
- #define **ETH_DMABMR_AAB** ((uint32_t)0x02000000) /* Address-Aligned beats */
- #define **ETH_DMABMR_FPM** ((uint32_t)0x01000000) /* 4xPBL mode */
- #define **ETH_DMABMR_USP** ((uint32_t)0x00800000) /* Use separate PBL */
- #define **ETH_DMABMR_RDP** ((uint32_t)0x007E0000) /* RxDMA PBL */
- #define **ETH_DMABMR_RDP_1Beat** ((uint32_t)0x00020000) /* maximum number of beats to be transferred in one RxDMA transaction is 1 */
- #define **ETH_DMABMR_RDP_2Beat** ((uint32_t)0x00040000) /* maximum number of beats to be transferred in one RxDMA transaction is 2 */
- #define **ETH_DMABMR_RDP_4Beat** ((uint32_t)0x00080000) /* maximum number of beats to be transferred in one RxDMA transaction is 4 */
- #define **ETH_DMABMR_RDP_8Beat** ((uint32_t)0x00100000) /* maximum number of beats to be transferred in one RxDMA transaction is 8 */
- #define **ETH_DMABMR_RDP_16Beat** ((uint32_t)0x00200000) /* maximum number of beats to be transferred in one RxDMA transaction is 16 */
- #define **ETH_DMABMR_RDP_32Beat** ((uint32_t)0x00400000) /* maximum number of beats to be transferred in one RxDMA transaction is 32 */
- #define **ETH_DMABMR_RDP_4xPBL_4Beat** ((uint32_t)0x01020000) /* maximum number of beats to be transferred in one RxDMA transaction is 4 */
- #define **ETH_DMABMR_RDP_4xPBL_8Beat** ((uint32_t)0x01040000) /* maximum number of beats to be transferred in one RxDMA transaction is 8 */
- #define **ETH_DMABMR_RDP_4xPBL_16Beat** ((uint32_t)0x01080000) /* maximum number of beats to be transferred in one RxDMA transaction is 16 */
- #define **ETH_DMABMR_RDP_4xPBL_32Beat** ((uint32_t)0x01100000) /* maximum number of beats to be transferred in one RxDMA transaction is 32 */
- #define **ETH_DMABMR_RDP_4xPBL_64Beat** ((uint32_t)0x01200000) /* maximum number of beats to be transferred in one RxDMA transaction is 64 */
- #define **ETH_DMABMR_RDP_4xPBL_128Beat** ((uint32_t)0x01400000) /* maximum number of beats to be transferred in one RxDMA transaction is 128 */
- #define **ETH_DMABMR_FB** ((uint32_t)0x00010000) /* Fixed Burst */
- #define **ETH_DMABMR_RTPR** ((uint32_t)0x0000C000) /* Rx Tx priority ratio */
- #define **ETH_DMABMR_RTPR_1_1** ((uint32_t)0x00000000) /* Rx Tx priority ratio */
- #define **ETH_DMABMR_RTPR_2_1** ((uint32_t)0x00004000) /* Rx Tx priority ratio */
- #define **ETH_DMABMR_RTPR_3_1** ((uint32_t)0x00008000) /* Rx Tx priority ratio */
- #define **ETH_DMABMR_RTPR_4_1** ((uint32_t)0x0000C000) /* Rx Tx priority ratio */

- #define **ETH_DMABMR_PBL** ((uint32_t)0x00003F00) /* Programmable burst length */
- #define **ETH_DMABMR_PBL_1Beat** ((uint32_t)0x00000100) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 1 */
- #define **ETH_DMABMR_PBL_2Beat** ((uint32_t)0x00000200) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 2 */
- #define **ETH_DMABMR_PBL_4Beat** ((uint32_t)0x00000400) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 4 */
- #define **ETH_DMABMR_PBL_8Beat** ((uint32_t)0x00000800) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 8 */
- #define **ETH_DMABMR_PBL_16Beat** ((uint32_t)0x00001000) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 16 */
- #define **ETH_DMABMR_PBL_32Beat** ((uint32_t)0x00002000) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 32 */
- #define **ETH_DMABMR_PBL_4xPBL_4Beat** ((uint32_t)0x01000100) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 4 */
- #define **ETH_DMABMR_PBL_4xPBL_8Beat** ((uint32_t)0x01000200) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 8 */
- #define **ETH_DMABMR_PBL_4xPBL_16Beat** ((uint32_t)0x01000400) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 16 */
- #define **ETH_DMABMR_PBL_4xPBL_32Beat** ((uint32_t)0x01000800) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 32 */
- #define **ETH_DMABMR_PBL_4xPBL_64Beat** ((uint32_t)0x01001000) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 64 */
- #define **ETH_DMABMR_PBL_4xPBL_128Beat** ((uint32_t)0x01002000) /* maximum number of beats to be transferred in one TxDMA (or both) transaction is 128 */
- #define **ETH_DMABMR_EDE** ((uint32_t)0x00000080) /* Enhanced Descriptor Enable */
- #define **ETH_DMABMR_DSL** ((uint32_t)0x0000007C) /* Descriptor Skip Length */
- #define **ETH_DMABMR_DA** ((uint32_t)0x00000002) /* DMA arbitration scheme */
- #define **ETH_DMABMR_SR** ((uint32_t)0x00000001) /* Software reset */
- #define **ETH_DMATPDR_TPD** ((uint32_t)0xFFFFFFFF) /* Transmit poll demand */
- #define **ETH_DMARPDR_RPD** ((uint32_t)0xFFFFFFFF) /* Receive poll demand */
- #define **ETH_DMARDLAR_SRL** ((uint32_t)0xFFFFFFFF) /* Start of receive list */
- #define **ETH_DMATDLAR_STL** ((uint32_t)0xFFFFFFFF) /* Start of transmit list */
- #define **ETH_DMASR_TSTS** ((uint32_t)0x20000000) /* Time-stamp trigger status */
- #define **ETH_DMASR_PMTS** ((uint32_t)0x10000000) /* PMT status */
- #define **ETH_DMASR_MMCS** ((uint32_t)0x08000000) /* MMC status */
- #define **ETH_DMASR_EBS** ((uint32_t)0x03800000) /* Error bits status */
- #define **ETH_DMASR_EBS_DescAccess** ((uint32_t)0x02000000) /* Error bits 0-data buffer, 1-desc. access */
- #define **ETH_DMASR_EBS_ReadTransf** ((uint32_t)0x01000000) /* Error bits 0-write trnsf, 1-read transfr */
- #define **ETH_DMASR_EBS_DataTransfTx** ((uint32_t)0x00800000) /* Error bits 0-Rx DMA, 1-Tx DMA */
- #define **ETH_DMASR_TPS** ((uint32_t)0x00700000) /* Transmit process state */
- #define **ETH_DMASR_TPS_Stopped** ((uint32_t)0x00000000) /* Stopped - Reset or Stop Tx Command issued */
- #define **ETH_DMASR_TPS_Fetching** ((uint32_t)0x00100000) /* Running - fetching the Tx descriptor */
- #define **ETH_DMASR_TPS_Waiting** ((uint32_t)0x00200000) /* Running - waiting for status */
- #define **ETH_DMASR_TPS_Reading** ((uint32_t)0x00300000) /* Running - reading the data from host memory */
- #define **ETH_DMASR_TPS_Suspended** ((uint32_t)0x00600000) /* Suspended - Tx Descriptor unavailabe */
- #define **ETH_DMASR_TPS_Closing** ((uint32_t)0x00700000) /* Running - closing Rx descriptor */
- #define **ETH_DMASR_RPS** ((uint32_t)0x000E0000) /* Receive process state */
- #define **ETH_DMASR_RPS_Stopped** ((uint32_t)0x00000000) /* Stopped - Reset or Stop Rx Command issued */
- #define **ETH_DMASR_RPS_Fetching** ((uint32_t)0x00020000) /* Running - fetching the Rx descriptor */
- #define **ETH_DMASR_RPS_Waiting** ((uint32_t)0x00060000) /* Running - waiting for packet */

- #define **ETH_DMASR_RPS_Suspended** ((uint32_t)0x00080000) /* Suspended - Rx Descriptor unavailable */
- #define **ETH_DMASR_RPS_Closing** ((uint32_t)0x000A0000) /* Running - closing descriptor */
- #define **ETH_DMASR_RPS_Queuing** ((uint32_t)0x000E0000) /* Running - queuing the recieve frame into host memory */
- #define **ETH_DMASR_NIS** ((uint32_t)0x00010000) /* Normal interrupt summary */
- #define **ETH_DMASR_AIS** ((uint32_t)0x00008000) /* Abnormal interrupt summary */
- #define **ETH_DMASR_ERS** ((uint32_t)0x00004000) /* Early receive status */
- #define **ETH_DMASR_FBES** ((uint32_t)0x00002000) /* Fatal bus error status */
- #define **ETH_DMASR_ETS** ((uint32_t)0x00000400) /* Early transmit status */
- #define **ETH_DMASR_RWTS** ((uint32_t)0x00000200) /* Receive watchdog timeout status */
- #define **ETH_DMASR_RPSS** ((uint32_t)0x00000100) /* Receive process stopped status */
- #define **ETH_DMASR_RBUS** ((uint32_t)0x00000080) /* Receive buffer unavailable status */
- #define **ETH_DMASR_RS** ((uint32_t)0x00000040) /* Receive status */
- #define **ETH_DMASR_TUS** ((uint32_t)0x00000020) /* Transmit underflow status */
- #define **ETH_DMASR_ROS** ((uint32_t)0x00000010) /* Receive overflow status */
- #define **ETH_DMASR_TJTS** ((uint32_t)0x00000008) /* Transmit jabber timeout status */
- #define **ETH_DMASR_TBUS** ((uint32_t)0x00000004) /* Transmit buffer unavailable status */
- #define **ETH_DMASR_TPSS** ((uint32_t)0x00000002) /* Transmit process stopped status */
- #define **ETH_DMASR_TS** ((uint32_t)0x00000001) /* Transmit status */
- #define **ETH_DMAOMR_DTCEFD** ((uint32_t)0x04000000) /* Disable Dropping of TCP/IP checksum error frames */
- #define **ETH_DMAOMR_RSF** ((uint32_t)0x02000000) /* Receive store and forward */
- #define **ETH_DMAOMR_DFRF** ((uint32_t)0x01000000) /* Disable flushing of received frames */
- #define **ETH_DMAOMR_TSF** ((uint32_t)0x00200000) /* Transmit store and forward */
- #define **ETH_DMAOMR_FTF** ((uint32_t)0x00100000) /* Flush transmit FIFO */
- #define **ETH_DMAOMR_TTC** ((uint32_t)0x0001C000) /* Transmit threshold control */
- #define **ETH_DMAOMR_TTC_64Bytes** ((uint32_t)0x00000000) /* threshold level of the MTL Transmit FIFO is 64 Bytes */
- #define **ETH_DMAOMR_TTC_128Bytes** ((uint32_t)0x00004000) /* threshold level of the MTL Transmit FIFO is 128 Bytes */
- #define **ETH_DMAOMR_TTC_192Bytes** ((uint32_t)0x00008000) /* threshold level of the MTL Transmit FIFO is 192 Bytes */
- #define **ETH_DMAOMR_TTC_256Bytes** ((uint32_t)0x0000C000) /* threshold level of the MTL Transmit FIFO is 256 Bytes */
- #define **ETH_DMAOMR_TTC_40Bytes** ((uint32_t)0x00010000) /* threshold level of the MTL Transmit FIFO is 40 Bytes */
- #define **ETH_DMAOMR_TTC_32Bytes** ((uint32_t)0x00014000) /* threshold level of the MTL Transmit FIFO is 32 Bytes */
- #define **ETH_DMAOMR_TTC_24Bytes** ((uint32_t)0x00018000) /* threshold level of the MTL Transmit FIFO is 24 Bytes */
- #define **ETH_DMAOMR_TTC_16Bytes** ((uint32_t)0x0001C000) /* threshold level of the MTL Transmit FIFO is 16 Bytes */
- #define **ETH_DMAOMR_ST** ((uint32_t)0x00002000) /* Start/stop transmission command */
- #define **ETH_DMAOMR_FEF** ((uint32_t)0x00000080) /* Forward error frames */
- #define **ETH_DMAOMR_FUGF** ((uint32_t)0x00000040) /* Forward undersized good frames */
- #define **ETH_DMAOMR_RTC** ((uint32_t)0x00000018) /* receive threshold control */
- #define **ETH_DMAOMR_RTC_64Bytes** ((uint32_t)0x00000000) /* threshold level of the MTL Receive FIFO is 64 Bytes */
- #define **ETH_DMAOMR_RTC_32Bytes** ((uint32_t)0x00000008) /* threshold level of the MTL Receive FIFO is 32 Bytes */
- #define **ETH_DMAOMR_RTC_96Bytes** ((uint32_t)0x00000010) /* threshold level of the MTL Receive FIFO is 96 Bytes */
- #define **ETH_DMAOMR_RTC_128Bytes** ((uint32_t)0x00000018) /* threshold level of the MTL Receive FIFO is 128 Bytes */

- #define **ETH_DMAOMR_OSF** ((uint32_t)0x00000004) /* operate on second frame */
- #define **ETH_DMAOMR_SR** ((uint32_t)0x00000002) /* Start/stop receive */
- #define **ETH_DMAIER_NISE** ((uint32_t)0x00010000) /* Normal interrupt summary enable */
- #define **ETH_DMAIER_AISE** ((uint32_t)0x00008000) /* Abnormal interrupt summary enable */
- #define **ETH_DMAIER_ERIE** ((uint32_t)0x00004000) /* Early receive interrupt enable */
- #define **ETH_DMAIER_FBEIE** ((uint32_t)0x00002000) /* Fatal bus error interrupt enable */
- #define **ETH_DMAIER_ETIE** ((uint32_t)0x00000400) /* Early transmit interrupt enable */
- #define **ETH_DMAIER_RWTIE** ((uint32_t)0x00000200) /* Receive watchdog timeout interrupt enable */
- #define **ETH_DMAIER_RPSIE** ((uint32_t)0x00000100) /* Receive process stopped interrupt enable */
- #define **ETH_DMAIER_RBUIE** ((uint32_t)0x00000080) /* Receive buffer unavailable interrupt enable */
- #define **ETH_DMAIER_RIE** ((uint32_t)0x00000040) /* Receive interrupt enable */
- #define **ETH_DMAIER_TUIE** ((uint32_t)0x00000020) /* Transmit Underflow interrupt enable */
- #define **ETH_DMAIER_ROIE** ((uint32_t)0x00000010) /* Receive Overflow interrupt enable */
- #define **ETH_DMAIER_TJTIE** ((uint32_t)0x00000008) /* Transmit jabber timeout interrupt enable */
- #define **ETH_DMAIER_TBUIE** ((uint32_t)0x00000004) /* Transmit buffer unavailable interrupt enable */
- #define **ETH_DMAIER_TPSIE** ((uint32_t)0x00000002) /* Transmit process stopped interrupt enable */
- #define **ETH_DMAIER_TIE** ((uint32_t)0x00000001) /* Transmit interrupt enable */
- #define **ETH_DMAMFBOCR_OFOC** ((uint32_t)0x10000000) /* Overflow bit for FIFO overflow counter */
- #define **ETH_DMAMFBOCR_MFA** ((uint32_t)0x0FFE0000) /* Number of frames missed by the application */
- #define **ETH_DMAMFBOCR_OMFC** ((uint32_t)0x00010000) /* Overflow bit for missed frame counter */
- #define **ETH_DMAMFBOCR_MFC** ((uint32_t)0x0000FFFF) /* Number of frames missed by the controller */
- #define **ETH_DMACHTDR_HTDAP** ((uint32_t)0xFFFFFFFF) /* Host transmit descriptor address pointer */
- #define **ETH_DMACHRDR_HRDAP** ((uint32_t)0xFFFFFFFF) /* Host receive descriptor address pointer */
- #define **ETH_DMACHTBAR_HTBAP** ((uint32_t)0xFFFFFFFF) /* Host transmit buffer address pointer */
- #define **ETH_DMACHRBAR_HRBAP** ((uint32_t)0xFFFFFFFF) /* Host receive buffer address pointer */
- #define **SET_BIT**(REG, BIT) ((REG) |= (BIT))
- #define **CLEAR_BIT**(REG, BIT) ((REG) &= ~(BIT))
- #define **READ_BIT**(REG, BIT) ((REG) & (BIT))
- #define **CLEAR_REG**(REG) ((REG) = (0x0))
- #define **WRITE_REG**(REG, VAL) ((REG) = (VAL))
- #define **READ_REG**(REG) ((REG))
- #define **MODIFY_REG**(REG, CLEARMASK, SETMASK) WRITE_REG((REG), (((READ_REG(REG)) & (~(CLEARMASK))) | (SETMASK)))

Typedefs

- typedef enum IRQn **IRQn_Type**

  STM32F4XX Interrupt Number Definition, according to the selected device in Library_configuration_section.
- typedef int32_t s32
- typedef int16_t **s16**
- typedef int8_t **s8**
- typedef const int32_t sc32
- typedef const int16_t sc16
- typedef const int8_t sc8
- typedef __IO int32_t **vs32**
- typedef __IO int16_t **vs16**
- typedef __IO int8_t **vs8**
- typedef __I int32_t vsc32
- typedef __I int16_t vsc16
- typedef __I int8_t vsc8
- typedef uint32_t **u32**
- typedef uint16_t **u16**

- typedef uint8_t **u8**
- typedef const uint32_t uc32
- typedef const uint16_t uc16
- typedef const uint8_t uc8
- typedef __IO uint32_t **vu32**
- typedef __IO uint16_t **vu16**
- typedef __IO uint8_t **vu8**
- typedef __I uint32_t vuc32
- typedef __I uint16_t vuc16
- typedef __I uint8_t vuc8
- typedef enum FlagStatus **ITStatus**

Enumerations

- enum IRQn {
  NonMaskableInt_IRQn = -14 , MemoryManagement_IRQn = -12 , BusFault_IRQn = -11 , UsageFault_IRQn = -10 ,
  SVCall_IRQn = -5 , DebugMonitor_IRQn = -4 , PendSV_IRQn = -2 , SysTick_IRQn = -1 ,
  WWDG_IRQn = 0 , PVD_IRQn = 1 , TAMP_STAMP_IRQn = 2 , RTC_WKUP_IRQn = 3 ,
  FLASH_IRQn = 4 , RCC_IRQn = 5 , EXTI0_IRQn = 6 , EXTI1_IRQn = 7 ,
  EXTI2_IRQn = 8 , EXTI3_IRQn = 9 , EXTI4_IRQn = 10 , DMA1_Stream0_IRQn = 11 ,
  DMA1_Stream1_IRQn = 12 , DMA1_Stream2_IRQn = 13 , DMA1_Stream3_IRQn = 14 , DMA1_Stream4_IRQn = 15 ,
  DMA1_Stream5_IRQn = 16 , DMA1_Stream6_IRQn = 17 , ADC_IRQn = 18 , CAN1_TX_IRQn = 19 ,
  CAN1_RX0_IRQn = 20 , CAN1_RX1_IRQn = 21 , CAN1_SCE_IRQn = 22 , EXTI9_5_IRQn = 23 ,
  TIM1_BRK_TIM9_IRQn = 24 , TIM1_UP_TIM10_IRQn = 25 , TIM1_TRG_COM_TIM11_IRQn = 26 ,
  TIM1_CC_IRQn = 27 ,
  TIM2_IRQn = 28 , TIM3_IRQn = 29 , TIM4_IRQn = 30 , I2C1_EV_IRQn = 31 ,
  I2C1_ER_IRQn = 32 , I2C2_EV_IRQn = 33 , I2C2_ER_IRQn = 34 , SPI1_IRQn = 35 ,
  SPI2_IRQn = 36 , USART1_IRQn = 37 , USART2_IRQn = 38 , USART3_IRQn = 39 ,
  EXTI15_10_IRQn = 40 , RTC_Alarm_IRQn = 41 , OTG_FS_WKUP_IRQn = 42 , TIM8_BRK_TIM12_IRQn = 43 ,
  TIM8_UP_TIM13_IRQn = 44 , TIM8_TRG_COM_TIM14_IRQn = 45 , TIM8_CC_IRQn = 46 , DMA1_Stream7_IRQn = 47 ,
  FSMC_IRQn = 48 , SDIO_IRQn = 49 , TIM5_IRQn = 50 , SPI3_IRQn = 51 ,
  UART4_IRQn = 52 , UART5_IRQn = 53 , TIM6_DAC_IRQn = 54 , TIM7_IRQn = 55 ,
  DMA2_Stream0_IRQn = 56 , DMA2_Stream1_IRQn = 57 , DMA2_Stream2_IRQn = 58 , DMA2_Stream3_IRQn = 59 ,
  DMA2_Stream4_IRQn = 60 , ETH_IRQn = 61 , ETH_WKUP_IRQn = 62 , CAN2_TX_IRQn = 63 ,
  CAN2_RX0_IRQn = 64 , CAN2_RX1_IRQn = 65 , CAN2_SCE_IRQn = 66 , OTG_FS_IRQn = 67 ,
  DMA2_Stream5_IRQn = 68 , DMA2_Stream6_IRQn = 69 , DMA2_Stream7_IRQn = 70 , USART6_IRQn = 71 ,
  I2C3_EV_IRQn = 72 , I2C3_ER_IRQn = 73 , OTG_HS_EP1_OUT_IRQn = 74 , OTG_HS_EP1_IN_IRQn = 75 ,
  OTG_HS_WKUP_IRQn = 76 , OTG_HS_IRQn = 77 , DCMI_IRQn = 78 , CRYP_IRQn = 79 ,
  HASH_RNG_IRQn = 80 , FPU_IRQn = 81 }

  STM32F4XX Interrupt Number Definition, according to the selected device in Library_configuration_section.
- enum **FlagStatus** { **RESET** = 0 , **SET** = !RESET }
- enum **FunctionalState** { **DISABLE** = 0 , **ENABLE** = !DISABLE }
- enum **ErrorStatus** { **ERROR** = 0 , **SUCCESS** = !ERROR }

7.23.1 Detailed Description

CMSIS Cortex-M4 Device Peripheral Access Layer Header File. This file contains all the peripheral register's definitions, bits definitions and memory mapping for STM32F4xx devices.

Author

MCD Application Team

Version

V1.0.0

Date

30-September-2011

The file is the unique include file that the application programmer is using in the C source code, usually in main.c. This file contains:

- Configuration section that allows to select:
  - The device used in the target application
  - To use or not the peripherals drivers in application code(i.e. code will be based on direct access to peripherals registers rather than drivers API), this option is controlled by "#define USE_STDPERIPH↩ _DRIVER"
  - To change few application-specific parameters such as the HSE crystal frequency
- Data structures and the address mapping for all peripherals
- Peripheral's registers declarations and bits definition
- Macros to access peripherals registers hardware

Attention

7.24 stm32f4xx.h

Go to the documentation of this file.

```
00001
00047 #ifndef __STM32F4xx_H
00048 #define __STM32F4xx_H
00049
00050 #ifdef __cplusplus
00051  extern "C" {
00052 #endif /* __cplusplus */
```

```
00053
00058 /* Uncomment the line below according to the target STM32 device used in your
00059    application
00060   */
00061
00062 #if !defined (STM32F4XX)
00063   #define STM32F4XX
00064 #endif
00065
00066 /*  Tip: To avoid modifying this file each time you need to switch between these
00067         devices, you can define the device in your toolchain compiler preprocessor.
00068   */
00069
00070 #if !defined (STM32F4XX)
00071  #error "Please select first the target STM32F4XX device used in your application (in stm32f4xx.h
      file)"
00072 #endif
00073
00074 #if !defined  (USE_STDPERIPH_DRIVER)
00080   /*#define USE_STDPERIPH_DRIVER*/
00081 #endif /* USE_STDPERIPH_DRIVER */
00082
00091 #if !defined  (HSE_VALUE)
00092   #define HSE_VALUE    ((uint32_t)8000000)
00093 #endif /* HSE_VALUE */
00094
00099 #if !defined  (HSE_STARTUP_TIMEOUT)
00100   #define HSE_STARTUP_TIMEOUT    ((uint16_t)0x0500)
00101 #endif /* HSE_STARTUP_TIMEOUT */
00102
00103 #if !defined  (HSI_VALUE)
00104   #define HSI_VALUE    ((uint32_t)16000000)
00105 #endif /* HSI_VALUE */
00106
00110 #define __STM32F4XX_STDPERIPH_VERSION_MAIN   (0x01)
00111 #define __STM32F4XX_STDPERIPH_VERSION_SUB1   (0x00)
00112 #define __STM32F4XX_STDPERIPH_VERSION_SUB2   (0x00)
00113 #define __STM32F4XX_STDPERIPH_VERSION_RC     (0x00)
00114 #define __STM32F4XX_STDPERIPH_VERSION        ((__STM32F4XX_STDPERIPH_VERSION_MAIN << 24)\
00115                                              |(__STM32F4XX_STDPERIPH_VERSION_SUB1 << 16)\
00116                                              |(__STM32F4XX_STDPERIPH_VERSION_SUB2 << 8)\
00117                                              |(__STM32F4XX_STDPERIPH_VERSION_RC))
00118
00130 #define __CM4_REV                 0x0001
00131 #define __MPU_PRESENT             1
00132 #define __NVIC_PRIO_BITS          4
00133 #define __Vendor_SysTickConfig    0
00134 #define __FPU_PRESENT             1
00140 typedef enum IRQn
00141 {
00142 /******  Cortex-M4 Processor Exceptions Numbers
      ***********************************************************/
00143   NonMaskableInt_IRQn         = -14,
00144   MemoryManagement_IRQn       = -12,
00145   BusFault_IRQn               = -11,
00146   UsageFault_IRQn             = -10,
00147   SVCall_IRQn                 = -5,
00148   DebugMonitor_IRQn           = -4,
00149   PendSV_IRQn                 = -2,
00150   SysTick_IRQn                = -1,
00151 /******  STM32 specific Interrupt Numbers
      *******************************************************************/
00152   WWDG_IRQn                   = 0,
00153   PVD_IRQn                    = 1,
00154   TAMP_STAMP_IRQn             = 2,
00155   RTC_WKUP_IRQn               = 3,
00156   FLASH_IRQn                  = 4,
00157   RCC_IRQn                    = 5,
00158   EXTI0_IRQn                  = 6,
00159   EXTI1_IRQn                  = 7,
00160   EXTI2_IRQn                  = 8,
00161   EXTI3_IRQn                  = 9,
00162   EXTI4_IRQn                  = 10,
00163   DMA1_Stream0_IRQn           = 11,
00164   DMA1_Stream1_IRQn           = 12,
00165   DMA1_Stream2_IRQn           = 13,
00166   DMA1_Stream3_IRQn           = 14,
00167   DMA1_Stream4_IRQn           = 15,
00168   DMA1_Stream5_IRQn           = 16,
00169   DMA1_Stream6_IRQn           = 17,
00170   ADC_IRQn                    = 18,
00171   CAN1_TX_IRQn                = 19,
00172   CAN1_RX0_IRQn               = 20,
00173   CAN1_RX1_IRQn               = 21,
00174   CAN1_SCE_IRQn               = 22,
00175   EXTI9_5_IRQn                = 23,
00176   TIM1_BRK_TIM9_IRQn          = 24,
```

```
00177   TIM1_UP_TIM10_IRQn          = 25,
00178   TIM1_TRG_COM_TIM11_IRQn     = 26,
00179   TIM1_CC_IRQn                = 27,
00180   TIM2_IRQn                   = 28,
00181   TIM3_IRQn                   = 29,
00182   TIM4_IRQn                   = 30,
00183   I2C1_EV_IRQn                = 31,
00184   I2C1_ER_IRQn                = 32,
00185   I2C2_EV_IRQn                = 33,
00186   I2C2_ER_IRQn                = 34,
00187   SPI1_IRQn                   = 35,
00188   SPI2_IRQn                   = 36,
00189   USART1_IRQn                 = 37,
00190   USART2_IRQn                 = 38,
00191   USART3_IRQn                 = 39,
00192   EXTI15_10_IRQn              = 40,
00193   RTC_Alarm_IRQn              = 41,
00194   OTG_FS_WKUP_IRQn            = 42,
00195   TIM8_BRK_TIM12_IRQn         = 43,
00196   TIM8_UP_TIM13_IRQn          = 44,
00197   TIM8_TRG_COM_TIM14_IRQn     = 45,
00198   TIM8_CC_IRQn                = 46,
00199   DMA1_Stream7_IRQn           = 47,
00200   FSMC_IRQn                   = 48,
00201   SDIO_IRQn                   = 49,
00202   TIM5_IRQn                   = 50,
00203   SPI3_IRQn                   = 51,
00204   UART4_IRQn                  = 52,
00205   UART5_IRQn                  = 53,
00206   TIM6_DAC_IRQn               = 54,
00207   TIM7_IRQn                   = 55,
00208   DMA2_Stream0_IRQn           = 56,
00209   DMA2_Stream1_IRQn           = 57,
00210   DMA2_Stream2_IRQn           = 58,
00211   DMA2_Stream3_IRQn           = 59,
00212   DMA2_Stream4_IRQn           = 60,
00213   ETH_IRQn                    = 61,
00214   ETH_WKUP_IRQn               = 62,
00215   CAN2_TX_IRQn                = 63,
00216   CAN2_RX0_IRQn               = 64,
00217   CAN2_RX1_IRQn               = 65,
00218   CAN2_SCE_IRQn               = 66,
00219   OTG_FS_IRQn                 = 67,
00220   DMA2_Stream5_IRQn           = 68,
00221   DMA2_Stream6_IRQn           = 69,
00222   DMA2_Stream7_IRQn           = 70,
00223   USART6_IRQn                 = 71,
00224   I2C3_EV_IRQn                = 72,
00225   I2C3_ER_IRQn                = 73,
00226   OTG_HS_EP1_OUT_IRQn         = 74,
00227   OTG_HS_EP1_IN_IRQn          = 75,
00228   OTG_HS_WKUP_IRQn            = 76,
00229   OTG_HS_IRQn                 = 77,
00230   DCMI_IRQn                   = 78,
00231   CRYP_IRQn                   = 79,
00232   HASH_RNG_IRQn               = 80,
00233   FPU_IRQn                    = 81
00234 } IRQn_Type;
00235
00240 #include "core_cm4.h"             /* Cortex-M4 processor and core peripherals */
00241 #include "system_stm32f4xx.h"
00242 #include <stdint.h>
00243
00248 typedef int32_t  s32;
00249 typedef int16_t s16;
00250 typedef int8_t  s8;
00251
00252 typedef const int32_t sc32;
00253 typedef const int16_t sc16;
00254 typedef const int8_t sc8;
00256 typedef __IO int32_t  vs32;
00257 typedef __IO int16_t  vs16;
00258 typedef __IO int8_t   vs8;
00259
00260 typedef __I int32_t vsc32;
00261 typedef __I int16_t vsc16;
00262 typedef __I int8_t vsc8;
00264 typedef uint32_t  u32;
00265 typedef uint16_t u16;
00266 typedef uint8_t  u8;
00267
00268 typedef const uint32_t uc32;
00269 typedef const uint16_t uc16;
00270 typedef const uint8_t uc8;
00272 typedef __IO uint32_t  vu32;
00273 typedef __IO uint16_t vu16;
00274 typedef __IO uint8_t  vu8;
```

```
00275
00276 typedef __I uint32_t vuc32;
00277 typedef __I uint16_t vuc16;
00278 typedef __I uint8_t vuc8;
00280 typedef enum {RESET = 0, SET = !RESET} FlagStatus, ITStatus;
00281
00282 typedef enum {DISABLE = 0, ENABLE = !DISABLE} FunctionalState;
00283 #define IS_FUNCTIONAL_STATE(STATE) (((STATE) == DISABLE) || ((STATE) == ENABLE))
00284
00285 typedef enum {ERROR = 0, SUCCESS = !ERROR} ErrorStatus;
00286
00299 typedef struct
00300 {
00301   __IO uint32_t SR;
00302   __IO uint32_t CR1;
00303   __IO uint32_t CR2;
00304   __IO uint32_t SMPR1;
00305   __IO uint32_t SMPR2;
00306   __IO uint32_t JOFR1;
00307   __IO uint32_t JOFR2;
00308   __IO uint32_t JOFR3;
00309   __IO uint32_t JOFR4;
00310   __IO uint32_t HTR;
00311   __IO uint32_t LTR;
00312   __IO uint32_t SQR1;
00313   __IO uint32_t SQR2;
00314   __IO uint32_t SQR3;
00315   __IO uint32_t JSQR;
00316   __IO uint32_t JDR1;
00317   __IO uint32_t JDR2;
00318   __IO uint32_t JDR3;
00319   __IO uint32_t JDR4;
00320   __IO uint32_t DR;
00321 } ADC_TypeDef;
00322
00323 typedef struct
00324 {
00325   __IO uint32_t CSR;
00326   __IO uint32_t CCR;
00327   __IO uint32_t CDR;
00329 } ADC_Common_TypeDef;
00330
00331
00336 typedef struct
00337 {
00338   __IO uint32_t TIR;
00339   __IO uint32_t TDTR;
00340   __IO uint32_t TDLR;
00341   __IO uint32_t TDHR;
00342 } CAN_TxMailBox_TypeDef;
00343
00348 typedef struct
00349 {
00350   __IO uint32_t RIR;
00351   __IO uint32_t RDTR;
00352   __IO uint32_t RDLR;
00353   __IO uint32_t RDHR;
00354 } CAN_FIFOMailBox_TypeDef;
00355
00360 typedef struct
00361 {
00362   __IO uint32_t FR1;
00363   __IO uint32_t FR2;
00364 } CAN_FilterRegister_TypeDef;
00365
00370 typedef struct
00371 {
00372   __IO uint32_t                 MCR;
00373   __IO uint32_t                 MSR;
00374   __IO uint32_t                 TSR;
00375   __IO uint32_t                 RF0R;
00376   __IO uint32_t                 RF1R;
00377   __IO uint32_t                 IER;
00378   __IO uint32_t                 ESR;
00379   __IO uint32_t                 BTR;
00380   uint32_t                      RESERVED0[88];
00381   CAN_TxMailBox_TypeDef         sTxMailBox[3];
00382   CAN_FIFOMailBox_TypeDef       sFIFOMailBox[2];
00383   uint32_t                      RESERVED1[12];
00384   __IO uint32_t                 FMR;
00385   __IO uint32_t                 FM1R;
00386   uint32_t                      RESERVED2;
00387   __IO uint32_t                 FS1R;
00388   uint32_t                      RESERVED3;
00389   __IO uint32_t                 FFA1R;
00390   uint32_t                      RESERVED4;
00391   __IO uint32_t                 FA1R;
```

```
00392   uint32_t                   RESERVED5[8];
00393   CAN_FilterRegister_TypeDef sFilterRegister[28];
00394 } CAN_TypeDef;
00395
00400 typedef struct
00401 {
00402   __IO uint32_t DR;
00403   __IO uint8_t  IDR;
00404   uint8_t       RESERVED0;
00405   uint16_t      RESERVED1;
00406   __IO uint32_t CR;
00407 } CRC_TypeDef;
00408
00413 typedef struct
00414 {
00415   __IO uint32_t CR;
00416   __IO uint32_t SWTRIGR;
00417   __IO uint32_t DHR12R1;
00418   __IO uint32_t DHR12L1;
00419   __IO uint32_t DHR8R1;
00420   __IO uint32_t DHR12R2;
00421   __IO uint32_t DHR12L2;
00422   __IO uint32_t DHR8R2;
00423   __IO uint32_t DHR12RD;
00424   __IO uint32_t DHR12LD;
00425   __IO uint32_t DHR8RD;
00426   __IO uint32_t DOR1;
00427   __IO uint32_t DOR2;
00428   __IO uint32_t SR;
00429 } DAC_TypeDef;
00430
00435 typedef struct
00436 {
00437   __IO uint32_t IDCODE;
00438   __IO uint32_t CR;
00439   __IO uint32_t APB1FZ;
00440   __IO uint32_t APB2FZ;
00441 }DBGMCU_TypeDef;
00442
00447 typedef struct
00448 {
00449   __IO uint32_t CR;
00450   __IO uint32_t SR;
00451   __IO uint32_t RISR;
00452   __IO uint32_t IER;
00453   __IO uint32_t MISR;
00454   __IO uint32_t ICR;
00455   __IO uint32_t ESCR;
00456   __IO uint32_t ESUR;
00457   __IO uint32_t CWSTRTR;
00458   __IO uint32_t CWSIZER;
00459   __IO uint32_t DR;
00460 } DCMI_TypeDef;
00461
00466 typedef struct
00467 {
00468   __IO uint32_t CR;
00469   __IO uint32_t NDTR;
00470   __IO uint32_t PAR;
00471   __IO uint32_t M0AR;
00472   __IO uint32_t M1AR;
00473   __IO uint32_t FCR;
00474 } DMA_Stream_TypeDef;
00475
00476 typedef struct
00477 {
00478   __IO uint32_t LISR;
00479   __IO uint32_t HISR;
00480   __IO uint32_t LIFCR;
00481   __IO uint32_t HIFCR;
00482 } DMA_TypeDef;
00483
00488 typedef struct
00489 {
00490   __IO uint32_t MACCR;
00491   __IO uint32_t MACFFR;
00492   __IO uint32_t MACHTHR;
00493   __IO uint32_t MACHTLR;
00494   __IO uint32_t MACMIIAR;
00495   __IO uint32_t MACMIIDR;
00496   __IO uint32_t MACFCR;
00497   __IO uint32_t MACVLANTR;             /*    8 */
00498   uint32_t      RESERVED0[2];
00499   __IO uint32_t MACRWUFFR;             /*   11 */
00500   __IO uint32_t MACPMTCSR;
00501   uint32_t      RESERVED1[2];
00502   __IO uint32_t MACSR;                 /*   15 */
```

```
00503   __IO uint32_t MACIMR;
00504   __IO uint32_t MACA0HR;
00505   __IO uint32_t MACA0LR;
00506   __IO uint32_t MACA1HR;
00507   __IO uint32_t MACA1LR;
00508   __IO uint32_t MACA2HR;
00509   __IO uint32_t MACA2LR;
00510   __IO uint32_t MACA3HR;
00511   __IO uint32_t MACA3LR;               /*   24 */
00512   uint32_t      RESERVED2[40];
00513   __IO uint32_t MMCCR;                 /*   65 */
00514   __IO uint32_t MMCRIR;
00515   __IO uint32_t MMCTIR;
00516   __IO uint32_t MMCRIMR;
00517   __IO uint32_t MMCTIMR;               /*   69 */
00518   uint32_t      RESERVED3[14];
00519   __IO uint32_t MMCTGFSCCR;            /*   84 */
00520   __IO uint32_t MMCTGFMSCCR;
00521   uint32_t      RESERVED4[5];
00522   __IO uint32_t MMCTGFCR;
00523   uint32_t      RESERVED5[10];
00524   __IO uint32_t MMCRFCECR;
00525   __IO uint32_t MMCRFAECR;
00526   uint32_t      RESERVED6[10];
00527   __IO uint32_t MMCRGUFCR;
00528   uint32_t      RESERVED7[334];
00529   __IO uint32_t PTPTSCR;
00530   __IO uint32_t PTPSSIR;
00531   __IO uint32_t PTPTSHR;
00532   __IO uint32_t PTPTSLR;
00533   __IO uint32_t PTPTSHUR;
00534   __IO uint32_t PTPTSLUR;
00535   __IO uint32_t PTPTSAR;
00536   __IO uint32_t PTPTTHR;
00537   __IO uint32_t PTPTTLR;
00538   __IO uint32_t RESERVED8;
00539   __IO uint32_t PTPTSSR;
00540   uint32_t      RESERVED9[565];
00541   __IO uint32_t DMABMR;
00542   __IO uint32_t DMATPDR;
00543   __IO uint32_t DMARPDR;
00544   __IO uint32_t DMARDLAR;
00545   __IO uint32_t DMATDLAR;
00546   __IO uint32_t DMASR;
00547   __IO uint32_t DMAOMR;
00548   __IO uint32_t DMAIER;
00549   __IO uint32_t DMAMFBOCR;
00550   __IO uint32_t DMARSWTR;
00551   uint32_t      RESERVED10[8];
00552   __IO uint32_t DMACHTDR;
00553   __IO uint32_t DMACHRDR;
00554   __IO uint32_t DMACHTBAR;
00555   __IO uint32_t DMACHRBAR;
00556 } ETH_TypeDef;
00557
00562 typedef struct
00563 {
00564   __IO uint32_t IMR;
00565   __IO uint32_t EMR;
00566   __IO uint32_t RTSR;
00567   __IO uint32_t FTSR;
00568   __IO uint32_t SWIER;
00569   __IO uint32_t PR;
00570 } EXTI_TypeDef;
00571
00576 typedef struct
00577 {
00578   __IO uint32_t ACR;
00579   __IO uint32_t KEYR;
00580   __IO uint32_t OPTKEYR;
00581   __IO uint32_t SR;
00582   __IO uint32_t CR;
00583   __IO uint32_t OPTCR;
00584 } FLASH_TypeDef;
00585
00590 typedef struct
00591 {
00592   __IO uint32_t BTCR[8];
00593 } FSMC_Bank1_TypeDef;
00594
00599 typedef struct
00600 {
00601   __IO uint32_t BWTR[7];
00602 } FSMC_Bank1E_TypeDef;
00603
00608 typedef struct
00609 {
```

```
00610   __IO uint32_t PCR2;
00611   __IO uint32_t SR2;
00612   __IO uint32_t PMEM2;
00613   __IO uint32_t PATT2;
00614   uint32_t      RESERVED0;
00615   __IO uint32_t ECCR2;
00616 } FSMC_Bank2_TypeDef;
00617
00622 typedef struct
00623 {
00624   __IO uint32_t PCR3;
00625   __IO uint32_t SR3;
00626   __IO uint32_t PMEM3;
00627   __IO uint32_t PATT3;
00628   uint32_t      RESERVED0;
00629   __IO uint32_t ECCR3;
00630 } FSMC_Bank3_TypeDef;
00631
00636 typedef struct
00637 {
00638   __IO uint32_t PCR4;
00639   __IO uint32_t SR4;
00640   __IO uint32_t PMEM4;
00641   __IO uint32_t PATT4;
00642   __IO uint32_t PIO4;
00643 } FSMC_Bank4_TypeDef;
00644
00649 typedef struct
00650 {
00651   __IO uint32_t MODER;
00652   __IO uint32_t OTYPER;
00653   __IO uint32_t OSPEEDR;
00654   __IO uint32_t PUPDR;
00655   __IO uint32_t IDR;
00656   __IO uint32_t ODR;
00657   __IO uint16_t BSRRL;
00658   __IO uint16_t BSRRH;
00659   __IO uint32_t LCKR;
00660   __IO uint32_t AFR[2];
00661 } GPIO_TypeDef;
00662
00667 typedef struct
00668 {
00669   __IO uint32_t MEMRMP;
00670   __IO uint32_t PMC;
00671   __IO uint32_t EXTICR[4];
00672   uint32_t      RESERVED[2];
00673   __IO uint32_t CMPCR;
00674 } SYSCFG_TypeDef;
00675
00680 typedef struct
00681 {
00682   __IO uint16_t CR1;
00683   uint16_t      RESERVED0;
00684   __IO uint16_t CR2;
00685   uint16_t      RESERVED1;
00686   __IO uint16_t OAR1;
00687   uint16_t      RESERVED2;
00688   __IO uint16_t OAR2;
00689   uint16_t      RESERVED3;
00690   __IO uint16_t DR;
00691   uint16_t      RESERVED4;
00692   __IO uint16_t SR1;
00693   uint16_t      RESERVED5;
00694   __IO uint16_t SR2;
00695   uint16_t      RESERVED6;
00696   __IO uint16_t CCR;
00697   uint16_t      RESERVED7;
00698   __IO uint16_t TRISE;
00699   uint16_t      RESERVED8;
00700 } I2C_TypeDef;
00701
00706 typedef struct
00707 {
00708   __IO uint32_t KR;
00709   __IO uint32_t PR;
00710   __IO uint32_t RLR;
00711   __IO uint32_t SR;
00712 } IWDG_TypeDef;
00713
00718 typedef struct
00719 {
00720   __IO uint32_t CR;
00721   __IO uint32_t CSR;
00722 } PWR_TypeDef;
00723
00728 typedef struct
```

```
00729 {
00730   __IO uint32_t CR;
00731   __IO uint32_t PLLCFGR;
00732   __IO uint32_t CFGR;
00733   __IO uint32_t CIR;
00734   __IO uint32_t AHB1RSTR;
00735   __IO uint32_t AHB2RSTR;
00736   __IO uint32_t AHB3RSTR;
00737   uint32_t      RESERVED0;
00738   __IO uint32_t APB1RSTR;
00739   __IO uint32_t APB2RSTR;
00740   uint32_t      RESERVED1[2];
00741   __IO uint32_t AHB1ENR;
00742   __IO uint32_t AHB2ENR;
00743   __IO uint32_t AHB3ENR;
00744   uint32_t      RESERVED2;
00745   __IO uint32_t APB1ENR;
00746   __IO uint32_t APB2ENR;
00747   uint32_t      RESERVED3[2];
00748   __IO uint32_t AHB1LPENR;
00749   __IO uint32_t AHB2LPENR;
00750   __IO uint32_t AHB3LPENR;
00751   uint32_t      RESERVED4;
00752   __IO uint32_t APB1LPENR;
00753   __IO uint32_t APB2LPENR;
00754   uint32_t      RESERVED5[2];
00755   __IO uint32_t BDCR;
00756   __IO uint32_t CSR;
00757   uint32_t      RESERVED6[2];
00758   __IO uint32_t SSCGR;
00759   __IO uint32_t PLLI2SCFGR;
00760 } RCC_TypeDef;
00761
00766 typedef struct
00767 {
00768   __IO uint32_t TR;
00769   __IO uint32_t DR;
00770   __IO uint32_t CR;
00771   __IO uint32_t ISR;
00772   __IO uint32_t PRER;
00773   __IO uint32_t WUTR;
00774   __IO uint32_t CALIBR;
00775   __IO uint32_t ALRMAR;
00776   __IO uint32_t ALRMBR;
00777   __IO uint32_t WPR;
00778   __IO uint32_t SSR;
00779   __IO uint32_t SHIFTR;
00780   __IO uint32_t TSTR;
00781   __IO uint32_t TSDR;
00782   __IO uint32_t TSSSR;
00783   __IO uint32_t CALR;
00784   __IO uint32_t TAFCR;
00785   __IO uint32_t ALRMASSR;
00786   __IO uint32_t ALRMBSSR;
00787   uint32_t RESERVED7;
00788   __IO uint32_t BKP0R;
00789   __IO uint32_t BKP1R;
00790   __IO uint32_t BKP2R;
00791   __IO uint32_t BKP3R;
00792   __IO uint32_t BKP4R;
00793   __IO uint32_t BKP5R;
00794   __IO uint32_t BKP6R;
00795   __IO uint32_t BKP7R;
00796   __IO uint32_t BKP8R;
00797   __IO uint32_t BKP9R;
00798   __IO uint32_t BKP10R;
00799   __IO uint32_t BKP11R;
00800   __IO uint32_t BKP12R;
00801   __IO uint32_t BKP13R;
00802   __IO uint32_t BKP14R;
00803   __IO uint32_t BKP15R;
00804   __IO uint32_t BKP16R;
00805   __IO uint32_t BKP17R;
00806   __IO uint32_t BKP18R;
00807   __IO uint32_t BKP19R;
00808 } RTC_TypeDef;
00809
00814 typedef struct
00815 {
00816   __IO uint32_t POWER;
00817   __IO uint32_t CLKCR;
00818   __IO uint32_t ARG;
00819   __IO uint32_t CMD;
00820   __I uint32_t  RESPCMD;
00821   __I uint32_t  RESP1;
00822   __I uint32_t  RESP2;
00823   __I uint32_t  RESP3;
```

```
00824   __I uint32_t  RESP4;
00825   __IO uint32_t DTIMER;
00826   __IO uint32_t DLEN;
00827   __IO uint32_t DCTRL;
00828   __I uint32_t  DCOUNT;
00829   __I uint32_t  STA;
00830   __IO uint32_t ICR;
00831   __IO uint32_t MASK;
00832   uint32_t      RESERVED0[2];
00833   __I uint32_t  FIFOCNT;
00834   uint32_t      RESERVED1[13];
00835   __IO uint32_t FIFO;
00836 } SDIO_TypeDef;
00837
00842 typedef struct
00843 {
00844   __IO uint16_t CR1;
00845   uint16_t      RESERVED0;
00846   __IO uint16_t CR2;
00847   uint16_t      RESERVED1;
00848   __IO uint16_t SR;
00849   uint16_t      RESERVED2;
00850   __IO uint16_t DR;
00851   uint16_t      RESERVED3;
00852   __IO uint16_t CRCPR;
00853   uint16_t      RESERVED4;
00854   __IO uint16_t RXCRCR;
00855   uint16_t      RESERVED5;
00856   __IO uint16_t TXCRCR;
00857   uint16_t      RESERVED6;
00858   __IO uint16_t I2SCFGR;
00859   uint16_t      RESERVED7;
00860   __IO uint16_t I2SPR;
00861   uint16_t      RESERVED8;
00862 } SPI_TypeDef;
00863
00868 typedef struct
00869 {
00870   __IO uint16_t CR1;
00871   uint16_t      RESERVED0;
00872   __IO uint16_t CR2;
00873   uint16_t      RESERVED1;
00874   __IO uint16_t SMCR;
00875   uint16_t      RESERVED2;
00876   __IO uint16_t DIER;
00877   uint16_t      RESERVED3;
00878   __IO uint16_t SR;
00879   uint16_t      RESERVED4;
00880   __IO uint16_t EGR;
00881   uint16_t      RESERVED5;
00882   __IO uint16_t CCMR1;
00883   uint16_t      RESERVED6;
00884   __IO uint16_t CCMR2;
00885   uint16_t      RESERVED7;
00886   __IO uint16_t CCER;
00887   uint16_t      RESERVED8;
00888   __IO uint32_t CNT;
00889   __IO uint16_t PSC;
00890   uint16_t      RESERVED9;
00891   __IO uint32_t ARR;
00892   __IO uint16_t RCR;
00893   uint16_t      RESERVED10;
00894   __IO uint32_t CCR1;
00895   __IO uint32_t CCR2;
00896   __IO uint32_t CCR3;
00897   __IO uint32_t CCR4;
00898   __IO uint16_t BDTR;
00899   uint16_t      RESERVED11;
00900   __IO uint16_t DCR;
00901   uint16_t      RESERVED12;
00902   __IO uint16_t DMAR;
00903   uint16_t      RESERVED13;
00904   __IO uint16_t OR;
00905   uint16_t      RESERVED14;
00906 } TIM_TypeDef;
00907
00912 typedef struct
00913 {
00914   __IO uint16_t SR;
00915   uint16_t      RESERVED0;
00916   __IO uint16_t DR;
00917   uint16_t      RESERVED1;
00918   __IO uint16_t BRR;
00919   uint16_t      RESERVED2;
00920   __IO uint16_t CR1;
00921   uint16_t      RESERVED3;
00922   __IO uint16_t CR2;
```

```
00923   uint16_t      RESERVED4;
00924   __IO uint16_t CR3;
00925   uint16_t      RESERVED5;
00926   __IO uint16_t GTPR;
00927   uint16_t      RESERVED6;
00928 } USART_TypeDef;
00929
00934 typedef struct
00935 {
00936   __IO uint32_t CR;
00937   __IO uint32_t CFR;
00938   __IO uint32_t SR;
00939 } WWDG_TypeDef;
00940
00945 typedef struct
00946 {
00947   __IO uint32_t CR;
00948   __IO uint32_t SR;
00949   __IO uint32_t DR;
00950   __IO uint32_t DOUT;
00951   __IO uint32_t DMACR;
00952   __IO uint32_t IMSCR;
00953   __IO uint32_t RISR;
00954   __IO uint32_t MISR;
00955   __IO uint32_t K0LR;
00956   __IO uint32_t K0RR;
00957   __IO uint32_t K1LR;
00958   __IO uint32_t K1RR;
00959   __IO uint32_t K2LR;
00960   __IO uint32_t K2RR;
00961   __IO uint32_t K3LR;
00962   __IO uint32_t K3RR;
00963   __IO uint32_t IV0LR;
00964   __IO uint32_t IV0RR;
00965   __IO uint32_t IV1LR;
00966   __IO uint32_t IV1RR;
00967 } CRYP_TypeDef;
00968
00973 typedef struct
00974 {
00975   __IO uint32_t CR;
00976   __IO uint32_t DIN;
00977   __IO uint32_t STR;
00978   __IO uint32_t HR[5];
00979   __IO uint32_t IMR;
00980   __IO uint32_t SR;
00981   uint32_t  RESERVED[52];
00982   __IO uint32_t CSR[51];
00983 } HASH_TypeDef;
00984
00989 typedef struct
00990 {
00991   __IO uint32_t CR;
00992   __IO uint32_t SR;
00993   __IO uint32_t DR;
00994 } RNG_TypeDef;
00995
01003 #define FLASH_BASE            ((uint32_t)0x08000000)
01004 #define CCMDATARAM_BASE       ((uint32_t)0x10000000)
01005 #define SRAM1_BASE            ((uint32_t)0x20000000)
01006 #define SRAM2_BASE            ((uint32_t)0x2001C000)
01007 #define PERIPH_BASE           ((uint32_t)0x40000000)
01008 #define BKPSRAM_BASE          ((uint32_t)0x40024000)
01009 #define FSMC_R_BASE           ((uint32_t)0xA0000000)
01011 #define CCMDATARAM_BB_BASE    ((uint32_t)0x12000000)
01012 #define SRAM1_BB_BASE         ((uint32_t)0x22000000)
01013 #define SRAM2_BB_BASE         ((uint32_t)0x2201C000)
01014 #define PERIPH_BB_BASE        ((uint32_t)0x42000000)
01015 #define BKPSRAM_BB_BASE       ((uint32_t)0x42024000)
01017 /* Legacy defines */
01018 #define SRAM_BASE             SRAM1_BASE
01019 #define SRAM_BB_BASE          SRAM1_BB_BASE
01020
01021
01023 #define APB1PERIPH_BASE       PERIPH_BASE
01024 #define APB2PERIPH_BASE       (PERIPH_BASE + 0x00010000)
01025 #define AHB1PERIPH_BASE       (PERIPH_BASE + 0x00020000)
01026 #define AHB2PERIPH_BASE       (PERIPH_BASE + 0x10000000)
01027
01029 #define TIM2_BASE             (APB1PERIPH_BASE + 0x0000)
01030 #define TIM3_BASE             (APB1PERIPH_BASE + 0x0400)
01031 #define TIM4_BASE             (APB1PERIPH_BASE + 0x0800)
01032 #define TIM5_BASE             (APB1PERIPH_BASE + 0x0C00)
01033 #define TIM6_BASE             (APB1PERIPH_BASE + 0x1000)
01034 #define TIM7_BASE             (APB1PERIPH_BASE + 0x1400)
01035 #define TIM12_BASE            (APB1PERIPH_BASE + 0x1800)
01036 #define TIM13_BASE            (APB1PERIPH_BASE + 0x1C00)
```

```
01037 #define TIM14_BASE            (APB1PERIPH_BASE + 0x2000)
01038 #define RTC_BASE              (APB1PERIPH_BASE + 0x2800)
01039 #define WWDG_BASE             (APB1PERIPH_BASE + 0x2C00)
01040 #define IWDG_BASE             (APB1PERIPH_BASE + 0x3000)
01041 #define I2S2ext_BASE          (APB1PERIPH_BASE + 0x3400)
01042 #define SPI2_BASE             (APB1PERIPH_BASE + 0x3800)
01043 #define SPI3_BASE             (APB1PERIPH_BASE + 0x3C00)
01044 #define I2S3ext_BASE          (APB1PERIPH_BASE + 0x4000)
01045 #define USART2_BASE           (APB1PERIPH_BASE + 0x4400)
01046 #define USART3_BASE           (APB1PERIPH_BASE + 0x4800)
01047 #define UART4_BASE            (APB1PERIPH_BASE + 0x4C00)
01048 #define UART5_BASE            (APB1PERIPH_BASE + 0x5000)
01049 #define I2C1_BASE             (APB1PERIPH_BASE + 0x5400)
01050 #define I2C2_BASE             (APB1PERIPH_BASE + 0x5800)
01051 #define I2C3_BASE             (APB1PERIPH_BASE + 0x5C00)
01052 #define CAN1_BASE             (APB1PERIPH_BASE + 0x6400)
01053 #define CAN2_BASE             (APB1PERIPH_BASE + 0x6800)
01054 #define PWR_BASE              (APB1PERIPH_BASE + 0x7000)
01055 #define DAC_BASE              (APB1PERIPH_BASE + 0x7400)
01056
01058 #define TIM1_BASE             (APB2PERIPH_BASE + 0x0000)
01059 #define TIM8_BASE             (APB2PERIPH_BASE + 0x0400)
01060 #define USART1_BASE           (APB2PERIPH_BASE + 0x1000)
01061 #define USART6_BASE           (APB2PERIPH_BASE + 0x1400)
01062 #define ADC1_BASE             (APB2PERIPH_BASE + 0x2000)
01063 #define ADC2_BASE             (APB2PERIPH_BASE + 0x2100)
01064 #define ADC3_BASE             (APB2PERIPH_BASE + 0x2200)
01065 #define ADC_BASE              (APB2PERIPH_BASE + 0x2300)
01066 #define SDIO_BASE             (APB2PERIPH_BASE + 0x2C00)
01067 #define SPI1_BASE             (APB2PERIPH_BASE + 0x3000)
01068 #define SYSCFG_BASE           (APB2PERIPH_BASE + 0x3800)
01069 #define EXTI_BASE             (APB2PERIPH_BASE + 0x3C00)
01070 #define TIM9_BASE             (APB2PERIPH_BASE + 0x4000)
01071 #define TIM10_BASE            (APB2PERIPH_BASE + 0x4400)
01072 #define TIM11_BASE            (APB2PERIPH_BASE + 0x4800)
01073
01075 #define GPIOA_BASE            (AHB1PERIPH_BASE + 0x0000)
01076 #define GPIOB_BASE            (AHB1PERIPH_BASE + 0x0400)
01077 #define GPIOC_BASE            (AHB1PERIPH_BASE + 0x0800)
01078 #define GPIOD_BASE            (AHB1PERIPH_BASE + 0x0C00)
01079 #define GPIOE_BASE            (AHB1PERIPH_BASE + 0x1000)
01080 #define GPIOF_BASE            (AHB1PERIPH_BASE + 0x1400)
01081 #define GPIOG_BASE            (AHB1PERIPH_BASE + 0x1800)
01082 #define GPIOH_BASE            (AHB1PERIPH_BASE + 0x1C00)
01083 #define GPIOI_BASE            (AHB1PERIPH_BASE + 0x2000)
01084 #define CRC_BASE              (AHB1PERIPH_BASE + 0x3000)
01085 #define RCC_BASE              (AHB1PERIPH_BASE + 0x3800)
01086 #define FLASH_R_BASE          (AHB1PERIPH_BASE + 0x3C00)
01087 #define DMA1_BASE             (AHB1PERIPH_BASE + 0x6000)
01088 #define DMA1_Stream0_BASE     (DMA1_BASE + 0x010)
01089 #define DMA1_Stream1_BASE     (DMA1_BASE + 0x028)
01090 #define DMA1_Stream2_BASE     (DMA1_BASE + 0x040)
01091 #define DMA1_Stream3_BASE     (DMA1_BASE + 0x058)
01092 #define DMA1_Stream4_BASE     (DMA1_BASE + 0x070)
01093 #define DMA1_Stream5_BASE     (DMA1_BASE + 0x088)
01094 #define DMA1_Stream6_BASE     (DMA1_BASE + 0x0A0)
01095 #define DMA1_Stream7_BASE     (DMA1_BASE + 0x0B8)
01096 #define DMA2_BASE             (AHB1PERIPH_BASE + 0x6400)
01097 #define DMA2_Stream0_BASE     (DMA2_BASE + 0x010)
01098 #define DMA2_Stream1_BASE     (DMA2_BASE + 0x028)
01099 #define DMA2_Stream2_BASE     (DMA2_BASE + 0x040)
01100 #define DMA2_Stream3_BASE     (DMA2_BASE + 0x058)
01101 #define DMA2_Stream4_BASE     (DMA2_BASE + 0x070)
01102 #define DMA2_Stream5_BASE     (DMA2_BASE + 0x088)
01103 #define DMA2_Stream6_BASE     (DMA2_BASE + 0x0A0)
01104 #define DMA2_Stream7_BASE     (DMA2_BASE + 0x0B8)
01105 #define ETH_BASE              (AHB1PERIPH_BASE + 0x8000)
01106 #define ETH_MAC_BASE          (ETH_BASE)
01107 #define ETH_MMC_BASE          (ETH_BASE + 0x0100)
01108 #define ETH_PTP_BASE          (ETH_BASE + 0x0700)
01109 #define ETH_DMA_BASE          (ETH_BASE + 0x1000)
01110
01112 #define DCMI_BASE             (AHB2PERIPH_BASE + 0x50000)
01113 #define CRYP_BASE             (AHB2PERIPH_BASE + 0x60000)
01114 #define HASH_BASE             (AHB2PERIPH_BASE + 0x60400)
01115 #define RNG_BASE              (AHB2PERIPH_BASE + 0x60800)
01116
01118 #define FSMC_Bank1_R_BASE     (FSMC_R_BASE + 0x0000)
01119 #define FSMC_Bank1E_R_BASE    (FSMC_R_BASE + 0x0104)
01120 #define FSMC_Bank2_R_BASE     (FSMC_R_BASE + 0x0060)
01121 #define FSMC_Bank3_R_BASE     (FSMC_R_BASE + 0x0080)
01122 #define FSMC_Bank4_R_BASE     (FSMC_R_BASE + 0x00A0)
01123
01124 /* Debug MCU registers base address */
01125 #define DBGMCU_BASE           ((uint32_t )0xE0042000)
01126
01134 #define TIM2                ((TIM_TypeDef *) TIM2_BASE)
```

```
01135 #define TIM3                ((TIM_TypeDef *) TIM3_BASE)
01136 #define TIM4                ((TIM_TypeDef *) TIM4_BASE)
01137 #define TIM5                ((TIM_TypeDef *) TIM5_BASE)
01138 #define TIM6                ((TIM_TypeDef *) TIM6_BASE)
01139 #define TIM7                ((TIM_TypeDef *) TIM7_BASE)
01140 #define TIM12               ((TIM_TypeDef *) TIM12_BASE)
01141 #define TIM13               ((TIM_TypeDef *) TIM13_BASE)
01142 #define TIM14               ((TIM_TypeDef *) TIM14_BASE)
01143 #define RTC                 ((RTC_TypeDef *) RTC_BASE)
01144 #define WWDG                ((WWDG_TypeDef *) WWDG_BASE)
01145 #define IWDG                ((IWDG_TypeDef *) IWDG_BASE)
01146 #define I2S2ext             ((SPI_TypeDef *) I2S2ext_BASE)
01147 #define SPI2                ((SPI_TypeDef *) SPI2_BASE)
01148 #define SPI3                ((SPI_TypeDef *) SPI3_BASE)
01149 #define I2S3ext             ((SPI_TypeDef *) I2S3ext_BASE)
01150 #define USART2              ((USART_TypeDef *) USART2_BASE)
01151 #define USART3              ((USART_TypeDef *) USART3_BASE)
01152 #define UART4               ((USART_TypeDef *) UART4_BASE)
01153 #define UART5               ((USART_TypeDef *) UART5_BASE)
01154 #define I2C1                ((I2C_TypeDef *) I2C1_BASE)
01155 #define I2C2                ((I2C_TypeDef *) I2C2_BASE)
01156 #define I2C3                ((I2C_TypeDef *) I2C3_BASE)
01157 #define CAN1                ((CAN_TypeDef *) CAN1_BASE)
01158 #define CAN2                ((CAN_TypeDef *) CAN2_BASE)
01159 #define PWR                 ((PWR_TypeDef *) PWR_BASE)
01160 #define DAC                 ((DAC_TypeDef *) DAC_BASE)
01161 #define TIM1                ((TIM_TypeDef *) TIM1_BASE)
01162 #define TIM8                ((TIM_TypeDef *) TIM8_BASE)
01163 #define USART1              ((USART_TypeDef *) USART1_BASE)
01164 #define USART6              ((USART_TypeDef *) USART6_BASE)
01165 #define ADC                 ((ADC_Common_TypeDef *) ADC_BASE)
01166 #define ADC1                ((ADC_TypeDef *) ADC1_BASE)
01167 #define ADC2                ((ADC_TypeDef *) ADC2_BASE)
01168 #define ADC3                ((ADC_TypeDef *) ADC3_BASE)
01169 #define SDIO                ((SDIO_TypeDef *) SDIO_BASE)
01170 #define SPI1                ((SPI_TypeDef *) SPI1_BASE)
01171 #define SYSCFG              ((SYSCFG_TypeDef *) SYSCFG_BASE)
01172 #define EXTI                ((EXTI_TypeDef *) EXTI_BASE)
01173 #define TIM9                ((TIM_TypeDef *) TIM9_BASE)
01174 #define TIM10               ((TIM_TypeDef *) TIM10_BASE)
01175 #define TIM11               ((TIM_TypeDef *) TIM11_BASE)
01176 #define GPIOA               ((GPIO_TypeDef *) GPIOA_BASE)
01177 #define GPIOB               ((GPIO_TypeDef *) GPIOB_BASE)
01178 #define GPIOC               ((GPIO_TypeDef *) GPIOC_BASE)
01179 #define GPIOD               ((GPIO_TypeDef *) GPIOD_BASE)
01180 #define GPIOE               ((GPIO_TypeDef *) GPIOE_BASE)
01181 #define GPIOF               ((GPIO_TypeDef *) GPIOF_BASE)
01182 #define GPIOG               ((GPIO_TypeDef *) GPIOG_BASE)
01183 #define GPIOH               ((GPIO_TypeDef *) GPIOH_BASE)
01184 #define GPIOI               ((GPIO_TypeDef *) GPIOI_BASE)
01185 #define CRC                 ((CRC_TypeDef *) CRC_BASE)
01186 #define RCC                 ((RCC_TypeDef *) RCC_BASE)
01187 #define FLASH               ((FLASH_TypeDef *) FLASH_R_BASE)
01188 #define DMA1                ((DMA_TypeDef *) DMA1_BASE)
01189 #define DMA1_Stream0        ((DMA_Stream_TypeDef *) DMA1_Stream0_BASE)
01190 #define DMA1_Stream1        ((DMA_Stream_TypeDef *) DMA1_Stream1_BASE)
01191 #define DMA1_Stream2        ((DMA_Stream_TypeDef *) DMA1_Stream2_BASE)
01192 #define DMA1_Stream3        ((DMA_Stream_TypeDef *) DMA1_Stream3_BASE)
01193 #define DMA1_Stream4        ((DMA_Stream_TypeDef *) DMA1_Stream4_BASE)
01194 #define DMA1_Stream5        ((DMA_Stream_TypeDef *) DMA1_Stream5_BASE)
01195 #define DMA1_Stream6        ((DMA_Stream_TypeDef *) DMA1_Stream6_BASE)
01196 #define DMA1_Stream7        ((DMA_Stream_TypeDef *) DMA1_Stream7_BASE)
01197 #define DMA2                ((DMA_TypeDef *) DMA2_BASE)
01198 #define DMA2_Stream0        ((DMA_Stream_TypeDef *) DMA2_Stream0_BASE)
01199 #define DMA2_Stream1        ((DMA_Stream_TypeDef *) DMA2_Stream1_BASE)
01200 #define DMA2_Stream2        ((DMA_Stream_TypeDef *) DMA2_Stream2_BASE)
01201 #define DMA2_Stream3        ((DMA_Stream_TypeDef *) DMA2_Stream3_BASE)
01202 #define DMA2_Stream4        ((DMA_Stream_TypeDef *) DMA2_Stream4_BASE)
01203 #define DMA2_Stream5        ((DMA_Stream_TypeDef *) DMA2_Stream5_BASE)
01204 #define DMA2_Stream6        ((DMA_Stream_TypeDef *) DMA2_Stream6_BASE)
01205 #define DMA2_Stream7        ((DMA_Stream_TypeDef *) DMA2_Stream7_BASE)
01206 #define ETH                 ((ETH_TypeDef *) ETH_BASE)
01207 #define DCMI                ((DCMI_TypeDef *) DCMI_BASE)
01208 #define CRYP                ((CRYP_TypeDef *) CRYP_BASE)
01209 #define HASH                ((HASH_TypeDef *) HASH_BASE)
01210 #define RNG                 ((RNG_TypeDef *) RNG_BASE)
01211 #define FSMC_Bank1          ((FSMC_Bank1_TypeDef *) FSMC_Bank1_R_BASE)
01212 #define FSMC_Bank1E         ((FSMC_Bank1E_TypeDef *) FSMC_Bank1E_R_BASE)
01213 #define FSMC_Bank2          ((FSMC_Bank2_TypeDef *) FSMC_Bank2_R_BASE)
01214 #define FSMC_Bank3          ((FSMC_Bank3_TypeDef *) FSMC_Bank3_R_BASE)
01215 #define FSMC_Bank4          ((FSMC_Bank4_TypeDef *) FSMC_Bank4_R_BASE)
01216 #define DBGMCU              ((DBGMCU_TypeDef *) DBGMCU_BASE)
01217
01230 /*****************************************************************************/
01231 /*                         Peripheral Registers_Bits_Definition              */
01232 /*****************************************************************************/
01233
```

```
01234 /******************************************************************************/
01235 /*                                                                            */
01236 /*                         Analog to Digital Converter                        */
01237 /*                                                                            */
01238 /******************************************************************************/
01239 /********************  Bit definition for ADC_SR register  ********************/
01240 #define  ADC_SR_AWD                          ((uint8_t)0x01)
01241 #define  ADC_SR_EOC                          ((uint8_t)0x02)
01242 #define  ADC_SR_JEOC                         ((uint8_t)0x04)
01243 #define  ADC_SR_JSTRT                        ((uint8_t)0x08)
01244 #define  ADC_SR_STRT                         ((uint8_t)0x10)
01245 #define  ADC_SR_OVR                          ((uint8_t)0x20)
01247 /*******************  Bit definition for ADC_CR1 register  ********************/
01248 #define  ADC_CR1_AWDCH                       ((uint32_t)0x0000001F)
01249 #define  ADC_CR1_AWDCH_0                     ((uint32_t)0x00000001)
01250 #define  ADC_CR1_AWDCH_1                     ((uint32_t)0x00000002)
01251 #define  ADC_CR1_AWDCH_2                     ((uint32_t)0x00000004)
01252 #define  ADC_CR1_AWDCH_3                     ((uint32_t)0x00000008)
01253 #define  ADC_CR1_AWDCH_4                     ((uint32_t)0x00000010)
01254 #define  ADC_CR1_EOCIE                       ((uint32_t)0x00000020)
01255 #define  ADC_CR1_AWDIE                       ((uint32_t)0x00000040)
01256 #define  ADC_CR1_JEOCIE                      ((uint32_t)0x00000080)
01257 #define  ADC_CR1_SCAN                        ((uint32_t)0x00000100)
01258 #define  ADC_CR1_AWDSGL                      ((uint32_t)0x00000200)
01259 #define  ADC_CR1_JAUTO                       ((uint32_t)0x00000400)
01260 #define  ADC_CR1_DISCEN                      ((uint32_t)0x00000800)
01261 #define  ADC_CR1_JDISCEN                     ((uint32_t)0x00001000)
01262 #define  ADC_CR1_DISCNUM                     ((uint32_t)0x0000E000)
01263 #define  ADC_CR1_DISCNUM_0                   ((uint32_t)0x00002000)
01264 #define  ADC_CR1_DISCNUM_1                   ((uint32_t)0x00004000)
01265 #define  ADC_CR1_DISCNUM_2                   ((uint32_t)0x00008000)
01266 #define  ADC_CR1_JAWDEN                      ((uint32_t)0x00400000)
01267 #define  ADC_CR1_AWDEN                       ((uint32_t)0x00800000)
01268 #define  ADC_CR1_RES                         ((uint32_t)0x03000000)
01269 #define  ADC_CR1_RES_0                       ((uint32_t)0x01000000)
01270 #define  ADC_CR1_RES_1                       ((uint32_t)0x02000000)
01271 #define  ADC_CR1_OVRIE                       ((uint32_t)0x04000000)
01273 /*******************  Bit definition for ADC_CR2 register  ********************/
01274 #define  ADC_CR2_ADON                        ((uint32_t)0x00000001)
01275 #define  ADC_CR2_CONT                        ((uint32_t)0x00000002)
01276 #define  ADC_CR2_DMA                         ((uint32_t)0x00000100)
01277 #define  ADC_CR2_DDS                         ((uint32_t)0x00000200)
01278 #define  ADC_CR2_EOCS                        ((uint32_t)0x00000400)
01279 #define  ADC_CR2_ALIGN                       ((uint32_t)0x00000800)
01280 #define  ADC_CR2_JEXTSEL                     ((uint32_t)0x000F0000)
01281 #define  ADC_CR2_JEXTSEL_0                   ((uint32_t)0x00010000)
01282 #define  ADC_CR2_JEXTSEL_1                   ((uint32_t)0x00020000)
01283 #define  ADC_CR2_JEXTSEL_2                   ((uint32_t)0x00040000)
01284 #define  ADC_CR2_JEXTSEL_3                   ((uint32_t)0x00080000)
01285 #define  ADC_CR2_JEXTEN                      ((uint32_t)0x00300000)
01286 #define  ADC_CR2_JEXTEN_0                    ((uint32_t)0x00100000)
01287 #define  ADC_CR2_JEXTEN_1                    ((uint32_t)0x00200000)
01288 #define  ADC_CR2_JSWSTART                    ((uint32_t)0x00400000)
01289 #define  ADC_CR2_EXTSEL                      ((uint32_t)0x0F000000)
01290 #define  ADC_CR2_EXTSEL_0                    ((uint32_t)0x01000000)
01291 #define  ADC_CR2_EXTSEL_1                    ((uint32_t)0x02000000)
01292 #define  ADC_CR2_EXTSEL_2                    ((uint32_t)0x04000000)
01293 #define  ADC_CR2_EXTSEL_3                    ((uint32_t)0x08000000)
01294 #define  ADC_CR2_EXTEN                       ((uint32_t)0x30000000)
01295 #define  ADC_CR2_EXTEN_0                     ((uint32_t)0x10000000)
01296 #define  ADC_CR2_EXTEN_1                     ((uint32_t)0x20000000)
01297 #define  ADC_CR2_SWSTART                     ((uint32_t)0x40000000)
01299 /******************  Bit definition for ADC_SMPR1 register  *******************/
01300 #define  ADC_SMPR1_SMP10                     ((uint32_t)0x00000007)
01301 #define  ADC_SMPR1_SMP10_0                   ((uint32_t)0x00000001)
01302 #define  ADC_SMPR1_SMP10_1                   ((uint32_t)0x00000002)
01303 #define  ADC_SMPR1_SMP10_2                   ((uint32_t)0x00000004)
01304 #define  ADC_SMPR1_SMP11                     ((uint32_t)0x00000038)
01305 #define  ADC_SMPR1_SMP11_0                   ((uint32_t)0x00000008)
01306 #define  ADC_SMPR1_SMP11_1                   ((uint32_t)0x00000010)
01307 #define  ADC_SMPR1_SMP11_2                   ((uint32_t)0x00000020)
01308 #define  ADC_SMPR1_SMP12                     ((uint32_t)0x000001C0)
01309 #define  ADC_SMPR1_SMP12_0                   ((uint32_t)0x00000040)
01310 #define  ADC_SMPR1_SMP12_1                   ((uint32_t)0x00000080)
01311 #define  ADC_SMPR1_SMP12_2                   ((uint32_t)0x00000100)
01312 #define  ADC_SMPR1_SMP13                     ((uint32_t)0x00000E00)
01313 #define  ADC_SMPR1_SMP13_0                   ((uint32_t)0x00000200)
01314 #define  ADC_SMPR1_SMP13_1                   ((uint32_t)0x00000400)
01315 #define  ADC_SMPR1_SMP13_2                   ((uint32_t)0x00000800)
01316 #define  ADC_SMPR1_SMP14                     ((uint32_t)0x00007000)
01317 #define  ADC_SMPR1_SMP14_0                   ((uint32_t)0x00001000)
01318 #define  ADC_SMPR1_SMP14_1                   ((uint32_t)0x00002000)
01319 #define  ADC_SMPR1_SMP14_2                   ((uint32_t)0x00004000)
01320 #define  ADC_SMPR1_SMP15                     ((uint32_t)0x00038000)
01321 #define  ADC_SMPR1_SMP15_0                   ((uint32_t)0x00008000)
01322 #define  ADC_SMPR1_SMP15_1                   ((uint32_t)0x00010000)
01323 #define  ADC_SMPR1_SMP15_2                   ((uint32_t)0x00020000)
```

```
01324 #define  ADC_SMPR1_SMP16                     ((uint32_t)0x001C0000)
01325 #define  ADC_SMPR1_SMP16_0                   ((uint32_t)0x00040000)
01326 #define  ADC_SMPR1_SMP16_1                   ((uint32_t)0x00080000)
01327 #define  ADC_SMPR1_SMP16_2                   ((uint32_t)0x00100000)
01328 #define  ADC_SMPR1_SMP17                     ((uint32_t)0x00E00000)
01329 #define  ADC_SMPR1_SMP17_0                   ((uint32_t)0x00200000)
01330 #define  ADC_SMPR1_SMP17_1                   ((uint32_t)0x00400000)
01331 #define  ADC_SMPR1_SMP17_2                   ((uint32_t)0x00800000)
01332 #define  ADC_SMPR1_SMP18                     ((uint32_t)0x07000000)
01333 #define  ADC_SMPR1_SMP18_0                   ((uint32_t)0x01000000)
01334 #define  ADC_SMPR1_SMP18_1                   ((uint32_t)0x02000000)
01335 #define  ADC_SMPR1_SMP18_2                   ((uint32_t)0x04000000)
01337 /******************  Bit definition for ADC_SMPR2 register  ******************/
01338 #define  ADC_SMPR2_SMP0                      ((uint32_t)0x00000007)
01339 #define  ADC_SMPR2_SMP0_0                    ((uint32_t)0x00000001)
01340 #define  ADC_SMPR2_SMP0_1                    ((uint32_t)0x00000002)
01341 #define  ADC_SMPR2_SMP0_2                    ((uint32_t)0x00000004)
01342 #define  ADC_SMPR2_SMP1                      ((uint32_t)0x00000038)
01343 #define  ADC_SMPR2_SMP1_0                    ((uint32_t)0x00000008)
01344 #define  ADC_SMPR2_SMP1_1                    ((uint32_t)0x00000010)
01345 #define  ADC_SMPR2_SMP1_2                    ((uint32_t)0x00000020)
01346 #define  ADC_SMPR2_SMP2                      ((uint32_t)0x000001C0)
01347 #define  ADC_SMPR2_SMP2_0                    ((uint32_t)0x00000040)
01348 #define  ADC_SMPR2_SMP2_1                    ((uint32_t)0x00000080)
01349 #define  ADC_SMPR2_SMP2_2                    ((uint32_t)0x00000100)
01350 #define  ADC_SMPR2_SMP3                      ((uint32_t)0x00000E00)
01351 #define  ADC_SMPR2_SMP3_0                    ((uint32_t)0x00000200)
01352 #define  ADC_SMPR2_SMP3_1                    ((uint32_t)0x00000400)
01353 #define  ADC_SMPR2_SMP3_2                    ((uint32_t)0x00000800)
01354 #define  ADC_SMPR2_SMP4                      ((uint32_t)0x00007000)
01355 #define  ADC_SMPR2_SMP4_0                    ((uint32_t)0x00001000)
01356 #define  ADC_SMPR2_SMP4_1                    ((uint32_t)0x00002000)
01357 #define  ADC_SMPR2_SMP4_2                    ((uint32_t)0x00004000)
01358 #define  ADC_SMPR2_SMP5                      ((uint32_t)0x00038000)
01359 #define  ADC_SMPR2_SMP5_0                    ((uint32_t)0x00008000)
01360 #define  ADC_SMPR2_SMP5_1                    ((uint32_t)0x00010000)
01361 #define  ADC_SMPR2_SMP5_2                    ((uint32_t)0x00020000)
01362 #define  ADC_SMPR2_SMP6                      ((uint32_t)0x001C0000)
01363 #define  ADC_SMPR2_SMP6_0                    ((uint32_t)0x00040000)
01364 #define  ADC_SMPR2_SMP6_1                    ((uint32_t)0x00080000)
01365 #define  ADC_SMPR2_SMP6_2                    ((uint32_t)0x00100000)
01366 #define  ADC_SMPR2_SMP7                      ((uint32_t)0x00E00000)
01367 #define  ADC_SMPR2_SMP7_0                    ((uint32_t)0x00200000)
01368 #define  ADC_SMPR2_SMP7_1                    ((uint32_t)0x00400000)
01369 #define  ADC_SMPR2_SMP7_2                    ((uint32_t)0x00800000)
01370 #define  ADC_SMPR2_SMP8                      ((uint32_t)0x07000000)
01371 #define  ADC_SMPR2_SMP8_0                    ((uint32_t)0x01000000)
01372 #define  ADC_SMPR2_SMP8_1                    ((uint32_t)0x02000000)
01373 #define  ADC_SMPR2_SMP8_2                    ((uint32_t)0x04000000)
01374 #define  ADC_SMPR2_SMP9                      ((uint32_t)0x38000000)
01375 #define  ADC_SMPR2_SMP9_0                    ((uint32_t)0x08000000)
01376 #define  ADC_SMPR2_SMP9_1                    ((uint32_t)0x10000000)
01377 #define  ADC_SMPR2_SMP9_2                    ((uint32_t)0x20000000)
01379 /******************  Bit definition for ADC_JOFR1 register  ******************/
01380 #define  ADC_JOFR1_JOFFSET1                  ((uint16_t)0x0FFF)
01382 /******************  Bit definition for ADC_JOFR2 register  ******************/
01383 #define  ADC_JOFR2_JOFFSET2                  ((uint16_t)0x0FFF)
01385 /******************  Bit definition for ADC_JOFR3 register  ******************/
01386 #define  ADC_JOFR3_JOFFSET3                  ((uint16_t)0x0FFF)
01388 /******************  Bit definition for ADC_JOFR4 register  ******************/
01389 #define  ADC_JOFR4_JOFFSET4                  ((uint16_t)0x0FFF)
01391 /*******************  Bit definition for ADC_HTR register  ********************/
01392 #define  ADC_HTR_HT                          ((uint16_t)0x0FFF)
01394 /*******************  Bit definition for ADC_LTR register  ********************/
01395 #define  ADC_LTR_LT                          ((uint16_t)0x0FFF)
01397 /*******************  Bit definition for ADC_SQR1 register  *******************/
01398 #define  ADC_SQR1_SQ13                       ((uint32_t)0x0000001F)
01399 #define  ADC_SQR1_SQ13_0                     ((uint32_t)0x00000001)
01400 #define  ADC_SQR1_SQ13_1                     ((uint32_t)0x00000002)
01401 #define  ADC_SQR1_SQ13_2                     ((uint32_t)0x00000004)
01402 #define  ADC_SQR1_SQ13_3                     ((uint32_t)0x00000008)
01403 #define  ADC_SQR1_SQ13_4                     ((uint32_t)0x00000010)
01404 #define  ADC_SQR1_SQ14                       ((uint32_t)0x000003E0)
01405 #define  ADC_SQR1_SQ14_0                     ((uint32_t)0x00000020)
01406 #define  ADC_SQR1_SQ14_1                     ((uint32_t)0x00000040)
01407 #define  ADC_SQR1_SQ14_2                     ((uint32_t)0x00000080)
01408 #define  ADC_SQR1_SQ14_3                     ((uint32_t)0x00000100)
01409 #define  ADC_SQR1_SQ14_4                     ((uint32_t)0x00000200)
01410 #define  ADC_SQR1_SQ15                       ((uint32_t)0x00007C00)
01411 #define  ADC_SQR1_SQ15_0                     ((uint32_t)0x00000400)
01412 #define  ADC_SQR1_SQ15_1                     ((uint32_t)0x00000800)
01413 #define  ADC_SQR1_SQ15_2                     ((uint32_t)0x00001000)
01414 #define  ADC_SQR1_SQ15_3                     ((uint32_t)0x00002000)
01415 #define  ADC_SQR1_SQ15_4                     ((uint32_t)0x00004000)
01416 #define  ADC_SQR1_SQ16                       ((uint32_t)0x000F8000)
01417 #define  ADC_SQR1_SQ16_0                     ((uint32_t)0x00008000)
01418 #define  ADC_SQR1_SQ16_1                     ((uint32_t)0x00010000)
```

```
01419 #define  ADC_SQR1_SQ16_2                     ((uint32_t)0x00020000)
01420 #define  ADC_SQR1_SQ16_3                     ((uint32_t)0x00040000)
01421 #define  ADC_SQR1_SQ16_4                     ((uint32_t)0x00080000)
01422 #define  ADC_SQR1_L                          ((uint32_t)0x00F00000)
01423 #define  ADC_SQR1_L_0                        ((uint32_t)0x00100000)
01424 #define  ADC_SQR1_L_1                        ((uint32_t)0x00200000)
01425 #define  ADC_SQR1_L_2                        ((uint32_t)0x00400000)
01426 #define  ADC_SQR1_L_3                        ((uint32_t)0x00800000)
01428 /********************  Bit definition for ADC_SQR2 register  ********************/
01429 #define  ADC_SQR2_SQ7                        ((uint32_t)0x0000001F)
01430 #define  ADC_SQR2_SQ7_0                      ((uint32_t)0x00000001)
01431 #define  ADC_SQR2_SQ7_1                      ((uint32_t)0x00000002)
01432 #define  ADC_SQR2_SQ7_2                      ((uint32_t)0x00000004)
01433 #define  ADC_SQR2_SQ7_3                      ((uint32_t)0x00000008)
01434 #define  ADC_SQR2_SQ7_4                      ((uint32_t)0x00000010)
01435 #define  ADC_SQR2_SQ8                        ((uint32_t)0x000003E0)
01436 #define  ADC_SQR2_SQ8_0                      ((uint32_t)0x00000020)
01437 #define  ADC_SQR2_SQ8_1                      ((uint32_t)0x00000040)
01438 #define  ADC_SQR2_SQ8_2                      ((uint32_t)0x00000080)
01439 #define  ADC_SQR2_SQ8_3                      ((uint32_t)0x00000100)
01440 #define  ADC_SQR2_SQ8_4                      ((uint32_t)0x00000200)
01441 #define  ADC_SQR2_SQ9                        ((uint32_t)0x00007C00)
01442 #define  ADC_SQR2_SQ9_0                      ((uint32_t)0x00000400)
01443 #define  ADC_SQR2_SQ9_1                      ((uint32_t)0x00000800)
01444 #define  ADC_SQR2_SQ9_2                      ((uint32_t)0x00001000)
01445 #define  ADC_SQR2_SQ9_3                      ((uint32_t)0x00002000)
01446 #define  ADC_SQR2_SQ9_4                      ((uint32_t)0x00004000)
01447 #define  ADC_SQR2_SQ10                       ((uint32_t)0x000F8000)
01448 #define  ADC_SQR2_SQ10_0                     ((uint32_t)0x00008000)
01449 #define  ADC_SQR2_SQ10_1                     ((uint32_t)0x00010000)
01450 #define  ADC_SQR2_SQ10_2                     ((uint32_t)0x00020000)
01451 #define  ADC_SQR2_SQ10_3                     ((uint32_t)0x00040000)
01452 #define  ADC_SQR2_SQ10_4                     ((uint32_t)0x00080000)
01453 #define  ADC_SQR2_SQ11                       ((uint32_t)0x01F00000)
01454 #define  ADC_SQR2_SQ11_0                     ((uint32_t)0x00100000)
01455 #define  ADC_SQR2_SQ11_1                     ((uint32_t)0x00200000)
01456 #define  ADC_SQR2_SQ11_2                     ((uint32_t)0x00400000)
01457 #define  ADC_SQR2_SQ11_3                     ((uint32_t)0x00800000)
01458 #define  ADC_SQR2_SQ11_4                     ((uint32_t)0x01000000)
01459 #define  ADC_SQR2_SQ12                       ((uint32_t)0x3E000000)
01460 #define  ADC_SQR2_SQ12_0                     ((uint32_t)0x02000000)
01461 #define  ADC_SQR2_SQ12_1                     ((uint32_t)0x04000000)
01462 #define  ADC_SQR2_SQ12_2                     ((uint32_t)0x08000000)
01463 #define  ADC_SQR2_SQ12_3                     ((uint32_t)0x10000000)
01464 #define  ADC_SQR2_SQ12_4                     ((uint32_t)0x20000000)
01466 /********************  Bit definition for ADC_SQR3 register  ********************/
01467 #define  ADC_SQR3_SQ1                        ((uint32_t)0x0000001F)
01468 #define  ADC_SQR3_SQ1_0                      ((uint32_t)0x00000001)
01469 #define  ADC_SQR3_SQ1_1                      ((uint32_t)0x00000002)
01470 #define  ADC_SQR3_SQ1_2                      ((uint32_t)0x00000004)
01471 #define  ADC_SQR3_SQ1_3                      ((uint32_t)0x00000008)
01472 #define  ADC_SQR3_SQ1_4                      ((uint32_t)0x00000010)
01473 #define  ADC_SQR3_SQ2                        ((uint32_t)0x000003E0)
01474 #define  ADC_SQR3_SQ2_0                      ((uint32_t)0x00000020)
01475 #define  ADC_SQR3_SQ2_1                      ((uint32_t)0x00000040)
01476 #define  ADC_SQR3_SQ2_2                      ((uint32_t)0x00000080)
01477 #define  ADC_SQR3_SQ2_3                      ((uint32_t)0x00000100)
01478 #define  ADC_SQR3_SQ2_4                      ((uint32_t)0x00000200)
01479 #define  ADC_SQR3_SQ3                        ((uint32_t)0x00007C00)
01480 #define  ADC_SQR3_SQ3_0                      ((uint32_t)0x00000400)
01481 #define  ADC_SQR3_SQ3_1                      ((uint32_t)0x00000800)
01482 #define  ADC_SQR3_SQ3_2                      ((uint32_t)0x00001000)
01483 #define  ADC_SQR3_SQ3_3                      ((uint32_t)0x00002000)
01484 #define  ADC_SQR3_SQ3_4                      ((uint32_t)0x00004000)
01485 #define  ADC_SQR3_SQ4                        ((uint32_t)0x000F8000)
01486 #define  ADC_SQR3_SQ4_0                      ((uint32_t)0x00008000)
01487 #define  ADC_SQR3_SQ4_1                      ((uint32_t)0x00010000)
01488 #define  ADC_SQR3_SQ4_2                      ((uint32_t)0x00020000)
01489 #define  ADC_SQR3_SQ4_3                      ((uint32_t)0x00040000)
01490 #define  ADC_SQR3_SQ4_4                      ((uint32_t)0x00080000)
01491 #define  ADC_SQR3_SQ5                        ((uint32_t)0x01F00000)
01492 #define  ADC_SQR3_SQ5_0                      ((uint32_t)0x00100000)
01493 #define  ADC_SQR3_SQ5_1                      ((uint32_t)0x00200000)
01494 #define  ADC_SQR3_SQ5_2                      ((uint32_t)0x00400000)
01495 #define  ADC_SQR3_SQ5_3                      ((uint32_t)0x00800000)
01496 #define  ADC_SQR3_SQ5_4                      ((uint32_t)0x01000000)
01497 #define  ADC_SQR3_SQ6                        ((uint32_t)0x3E000000)
01498 #define  ADC_SQR3_SQ6_0                      ((uint32_t)0x02000000)
01499 #define  ADC_SQR3_SQ6_1                      ((uint32_t)0x04000000)
01500 #define  ADC_SQR3_SQ6_2                      ((uint32_t)0x08000000)
01501 #define  ADC_SQR3_SQ6_3                      ((uint32_t)0x10000000)
01502 #define  ADC_SQR3_SQ6_4                      ((uint32_t)0x20000000)
01504 /********************  Bit definition for ADC_JSQR register  ********************/
01505 #define  ADC_JSQR_JSQ1                       ((uint32_t)0x0000001F)
01506 #define  ADC_JSQR_JSQ1_0                     ((uint32_t)0x00000001)
01507 #define  ADC_JSQR_JSQ1_1                     ((uint32_t)0x00000002)
01508 #define  ADC_JSQR_JSQ1_2                     ((uint32_t)0x00000004)
```

```
01509 #define  ADC_JSQR_JSQ1_3                     ((uint32_t)0x00000008)
01510 #define  ADC_JSQR_JSQ1_4                     ((uint32_t)0x00000010)
01511 #define  ADC_JSQR_JSQ2                       ((uint32_t)0x000003E0)
01512 #define  ADC_JSQR_JSQ2_0                     ((uint32_t)0x00000020)
01513 #define  ADC_JSQR_JSQ2_1                     ((uint32_t)0x00000040)
01514 #define  ADC_JSQR_JSQ2_2                     ((uint32_t)0x00000080)
01515 #define  ADC_JSQR_JSQ2_3                     ((uint32_t)0x00000100)
01516 #define  ADC_JSQR_JSQ2_4                     ((uint32_t)0x00000200)
01517 #define  ADC_JSQR_JSQ3                       ((uint32_t)0x00007C00)
01518 #define  ADC_JSQR_JSQ3_0                     ((uint32_t)0x00000400)
01519 #define  ADC_JSQR_JSQ3_1                     ((uint32_t)0x00000800)
01520 #define  ADC_JSQR_JSQ3_2                     ((uint32_t)0x00001000)
01521 #define  ADC_JSQR_JSQ3_3                     ((uint32_t)0x00002000)
01522 #define  ADC_JSQR_JSQ3_4                     ((uint32_t)0x00004000)
01523 #define  ADC_JSQR_JSQ4                       ((uint32_t)0x000F8000)
01524 #define  ADC_JSQR_JSQ4_0                     ((uint32_t)0x00008000)
01525 #define  ADC_JSQR_JSQ4_1                     ((uint32_t)0x00010000)
01526 #define  ADC_JSQR_JSQ4_2                     ((uint32_t)0x00020000)
01527 #define  ADC_JSQR_JSQ4_3                     ((uint32_t)0x00040000)
01528 #define  ADC_JSQR_JSQ4_4                     ((uint32_t)0x00080000)
01529 #define  ADC_JSQR_JL                         ((uint32_t)0x00300000)
01530 #define  ADC_JSQR_JL_0                       ((uint32_t)0x00100000)
01531 #define  ADC_JSQR_JL_1                       ((uint32_t)0x00200000)
01533 /*******************  Bit definition for ADC_JDR1 register  *******************/
01534 #define  ADC_JDR1_JDATA                      ((uint16_t)0xFFFF)
01536 /*******************  Bit definition for ADC_JDR2 register  *******************/
01537 #define  ADC_JDR2_JDATA                      ((uint16_t)0xFFFF)
01539 /*******************  Bit definition for ADC_JDR3 register  *******************/
01540 #define  ADC_JDR3_JDATA                      ((uint16_t)0xFFFF)
01542 /*******************  Bit definition for ADC_JDR4 register  *******************/
01543 #define  ADC_JDR4_JDATA                      ((uint16_t)0xFFFF)
01545 /********************  Bit definition for ADC_DR register  ********************/
01546 #define  ADC_DR_DATA                         ((uint32_t)0x0000FFFF)
01547 #define  ADC_DR_ADC2DATA                     ((uint32_t)0xFFFF0000)
01549 /*******************  Bit definition for ADC_CSR register  *******************/
01550 #define  ADC_CSR_AWD1                        ((uint32_t)0x00000001)
01551 #define  ADC_CSR_EOC1                        ((uint32_t)0x00000002)
01552 #define  ADC_CSR_JEOC1                       ((uint32_t)0x00000004)
01553 #define  ADC_CSR_JSTRT1                      ((uint32_t)0x00000008)
01554 #define  ADC_CSR_STRT1                       ((uint32_t)0x00000010)
01555 #define  ADC_CSR_DOVR1                       ((uint32_t)0x00000020)
01556 #define  ADC_CSR_AWD2                        ((uint32_t)0x00000100)
01557 #define  ADC_CSR_EOC2                        ((uint32_t)0x00000200)
01558 #define  ADC_CSR_JEOC2                       ((uint32_t)0x00000400)
01559 #define  ADC_CSR_JSTRT2                      ((uint32_t)0x00000800)
01560 #define  ADC_CSR_STRT2                       ((uint32_t)0x00001000)
01561 #define  ADC_CSR_DOVR2                       ((uint32_t)0x00002000)
01562 #define  ADC_CSR_AWD3                        ((uint32_t)0x00010000)
01563 #define  ADC_CSR_EOC3                        ((uint32_t)0x00020000)
01564 #define  ADC_CSR_JEOC3                       ((uint32_t)0x00040000)
01565 #define  ADC_CSR_JSTRT3                      ((uint32_t)0x00080000)
01566 #define  ADC_CSR_STRT3                       ((uint32_t)0x00100000)
01567 #define  ADC_CSR_DOVR3                       ((uint32_t)0x00200000)
01569 /*******************  Bit definition for ADC_CCR register  *******************/
01570 #define  ADC_CCR_MULTI                       ((uint32_t)0x0000001F)
01571 #define  ADC_CCR_MULTI_0                     ((uint32_t)0x00000001)
01572 #define  ADC_CCR_MULTI_1                     ((uint32_t)0x00000002)
01573 #define  ADC_CCR_MULTI_2                     ((uint32_t)0x00000004)
01574 #define  ADC_CCR_MULTI_3                     ((uint32_t)0x00000008)
01575 #define  ADC_CCR_MULTI_4                     ((uint32_t)0x00000010)
01576 #define  ADC_CCR_DELAY                       ((uint32_t)0x00000F00)
01577 #define  ADC_CCR_DELAY_0                     ((uint32_t)0x00000100)
01578 #define  ADC_CCR_DELAY_1                     ((uint32_t)0x00000200)
01579 #define  ADC_CCR_DELAY_2                     ((uint32_t)0x00000400)
01580 #define  ADC_CCR_DELAY_3                     ((uint32_t)0x00000800)
01581 #define  ADC_CCR_DDS                         ((uint32_t)0x00002000)
01582 #define  ADC_CCR_DMA                         ((uint32_t)0x0000C000)
01583 #define  ADC_CCR_DMA_0                       ((uint32_t)0x00004000)
01584 #define  ADC_CCR_DMA_1                       ((uint32_t)0x00008000)
01585 #define  ADC_CCR_ADCPRE                      ((uint32_t)0x00030000)
01586 #define  ADC_CCR_ADCPRE_0                    ((uint32_t)0x00010000)
01587 #define  ADC_CCR_ADCPRE_1                    ((uint32_t)0x00020000)
01588 #define  ADC_CCR_VBATE                       ((uint32_t)0x00400000)
01589 #define  ADC_CCR_TSVREFE                     ((uint32_t)0x00800000)
01591 /*******************  Bit definition for ADC_CDR register  *******************/
01592 #define  ADC_CDR_DATA1                      ((uint32_t)0x0000FFFF)
01593 #define  ADC_CDR_DATA2                      ((uint32_t)0xFFFF0000)
01595 /******************************************************************************/
01596 /*                                                                            */
01597 /*                          Controller Area Network                           */
01598 /*                                                                            */
01599 /******************************************************************************/
01601 /*******************  Bit definition for CAN_MCR register  *******************/
01602 #define  CAN_MCR_INRQ                        ((uint16_t)0x0001)
01603 #define  CAN_MCR_SLEEP                       ((uint16_t)0x0002)
01604 #define  CAN_MCR_TXFP                        ((uint16_t)0x0004)
01605 #define  CAN_MCR_RFLM                        ((uint16_t)0x0008)
```

```
01606 #define  CAN_MCR_NART                        ((uint16_t)0x0010)
01607 #define  CAN_MCR_AWUM                        ((uint16_t)0x0020)
01608 #define  CAN_MCR_ABOM                        ((uint16_t)0x0040)
01609 #define  CAN_MCR_TTCM                        ((uint16_t)0x0080)
01610 #define  CAN_MCR_RESET                       ((uint16_t)0x8000)
01612 /*******************  Bit definition for CAN_MSR register  *******************/
01613 #define  CAN_MSR_INAK                        ((uint16_t)0x0001)
01614 #define  CAN_MSR_SLAK                        ((uint16_t)0x0002)
01615 #define  CAN_MSR_ERRI                        ((uint16_t)0x0004)
01616 #define  CAN_MSR_WKUI                        ((uint16_t)0x0008)
01617 #define  CAN_MSR_SLAKI                       ((uint16_t)0x0010)
01618 #define  CAN_MSR_TXM                         ((uint16_t)0x0100)
01619 #define  CAN_MSR_RXM                         ((uint16_t)0x0200)
01620 #define  CAN_MSR_SAMP                        ((uint16_t)0x0400)
01621 #define  CAN_MSR_RX                          ((uint16_t)0x0800)
01623 /*******************  Bit definition for CAN_TSR register  *******************/
01624 #define  CAN_TSR_RQCP0                       ((uint32_t)0x00000001)
01625 #define  CAN_TSR_TXOK0                       ((uint32_t)0x00000002)
01626 #define  CAN_TSR_ALST0                       ((uint32_t)0x00000004)
01627 #define  CAN_TSR_TERR0                       ((uint32_t)0x00000008)
01628 #define  CAN_TSR_ABRQ0                       ((uint32_t)0x00000080)
01629 #define  CAN_TSR_RQCP1                       ((uint32_t)0x00000100)
01630 #define  CAN_TSR_TXOK1                       ((uint32_t)0x00000200)
01631 #define  CAN_TSR_ALST1                       ((uint32_t)0x00000400)
01632 #define  CAN_TSR_TERR1                       ((uint32_t)0x00000800)
01633 #define  CAN_TSR_ABRQ1                       ((uint32_t)0x00008000)
01634 #define  CAN_TSR_RQCP2                       ((uint32_t)0x00010000)
01635 #define  CAN_TSR_TXOK2                       ((uint32_t)0x00020000)
01636 #define  CAN_TSR_ALST2                       ((uint32_t)0x00040000)
01637 #define  CAN_TSR_TERR2                       ((uint32_t)0x00080000)
01638 #define  CAN_TSR_ABRQ2                       ((uint32_t)0x00800000)
01639 #define  CAN_TSR_CODE                        ((uint32_t)0x03000000)
01641 #define  CAN_TSR_TME                         ((uint32_t)0x1C000000)
01642 #define  CAN_TSR_TME0                        ((uint32_t)0x04000000)
01643 #define  CAN_TSR_TME1                        ((uint32_t)0x08000000)
01644 #define  CAN_TSR_TME2                        ((uint32_t)0x10000000)
01646 #define  CAN_TSR_LOW                         ((uint32_t)0xE0000000)
01647 #define  CAN_TSR_LOW0                        ((uint32_t)0x20000000)
01648 #define  CAN_TSR_LOW1                        ((uint32_t)0x40000000)
01649 #define  CAN_TSR_LOW2                        ((uint32_t)0x80000000)
01651 /*******************  Bit definition for CAN_RF0R register  *******************/
01652 #define  CAN_RF0R_FMP0                       ((uint8_t)0x03)
01653 #define  CAN_RF0R_FULL0                      ((uint8_t)0x08)
01654 #define  CAN_RF0R_FOVR0                      ((uint8_t)0x10)
01655 #define  CAN_RF0R_RFOM0                      ((uint8_t)0x20)
01657 /*******************  Bit definition for CAN_RF1R register  *******************/
01658 #define  CAN_RF1R_FMP1                       ((uint8_t)0x03)
01659 #define  CAN_RF1R_FULL1                      ((uint8_t)0x08)
01660 #define  CAN_RF1R_FOVR1                      ((uint8_t)0x10)
01661 #define  CAN_RF1R_RFOM1                      ((uint8_t)0x20)
01663 /********************  Bit definition for CAN_IER register  *******************/
01664 #define  CAN_IER_TMEIE                       ((uint32_t)0x00000001)
01665 #define  CAN_IER_FMPIE0                      ((uint32_t)0x00000002)
01666 #define  CAN_IER_FFIE0                       ((uint32_t)0x00000004)
01667 #define  CAN_IER_FOVIE0                      ((uint32_t)0x00000008)
01668 #define  CAN_IER_FMPIE1                      ((uint32_t)0x00000010)
01669 #define  CAN_IER_FFIE1                       ((uint32_t)0x00000020)
01670 #define  CAN_IER_FOVIE1                      ((uint32_t)0x00000040)
01671 #define  CAN_IER_EWGIE                       ((uint32_t)0x00000100)
01672 #define  CAN_IER_EPVIE                       ((uint32_t)0x00000200)
01673 #define  CAN_IER_BOFIE                       ((uint32_t)0x00000400)
01674 #define  CAN_IER_LECIE                       ((uint32_t)0x00000800)
01675 #define  CAN_IER_ERRIE                       ((uint32_t)0x00008000)
01676 #define  CAN_IER_WKUIE                       ((uint32_t)0x00010000)
01677 #define  CAN_IER_SLKIE                       ((uint32_t)0x00020000)
01679 /********************  Bit definition for CAN_ESR register  *******************/
01680 #define  CAN_ESR_EWGF                        ((uint32_t)0x00000001)
01681 #define  CAN_ESR_EPVF                        ((uint32_t)0x00000002)
01682 #define  CAN_ESR_BOFF                        ((uint32_t)0x00000004)
01684 #define  CAN_ESR_LEC                         ((uint32_t)0x00000070)
01685 #define  CAN_ESR_LEC_0                       ((uint32_t)0x00000010)
01686 #define  CAN_ESR_LEC_1                       ((uint32_t)0x00000020)
01687 #define  CAN_ESR_LEC_2                       ((uint32_t)0x00000040)
01689 #define  CAN_ESR_TEC                         ((uint32_t)0x00FF0000)
01690 #define  CAN_ESR_REC                         ((uint32_t)0xFF000000)
01692 /*******************  Bit definition for CAN_BTR register  *******************/
01693 #define  CAN_BTR_BRP                         ((uint32_t)0x000003FF)
01694 #define  CAN_BTR_TS1                         ((uint32_t)0x000F0000)
01695 #define  CAN_BTR_TS2                         ((uint32_t)0x00700000)
01696 #define  CAN_BTR_SJW                         ((uint32_t)0x03000000)
01697 #define  CAN_BTR_LBKM                        ((uint32_t)0x40000000)
01698 #define  CAN_BTR_SILM                        ((uint32_t)0x80000000)
01701 /*****************  Bit definition for CAN_TI0R register  *******************/
01702 #define  CAN_TI0R_TXRQ                       ((uint32_t)0x00000001)
01703 #define  CAN_TI0R_RTR                        ((uint32_t)0x00000002)
01704 #define  CAN_TI0R_IDE                        ((uint32_t)0x00000004)
01705 #define  CAN_TI0R_EXID                       ((uint32_t)0x001FFFF8)
```

```
01706 #define  CAN_TI0R_STID                       ((uint32_t)0xFFE00000)
01708 /*****************  Bit definition for CAN_TDT0R register  *****************/
01709 #define  CAN_TDT0R_DLC                       ((uint32_t)0x0000000F)
01710 #define  CAN_TDT0R_TGT                       ((uint32_t)0x00000100)
01711 #define  CAN_TDT0R_TIME                      ((uint32_t)0xFFFF0000)
01713 /*****************  Bit definition for CAN_TDL0R register  *****************/
01714 #define  CAN_TDL0R_DATA0                     ((uint32_t)0x000000FF)
01715 #define  CAN_TDL0R_DATA1                     ((uint32_t)0x0000FF00)
01716 #define  CAN_TDL0R_DATA2                     ((uint32_t)0x00FF0000)
01717 #define  CAN_TDL0R_DATA3                     ((uint32_t)0xFF000000)
01719 /*****************  Bit definition for CAN_TDH0R register  *****************/
01720 #define  CAN_TDH0R_DATA4                     ((uint32_t)0x000000FF)
01721 #define  CAN_TDH0R_DATA5                     ((uint32_t)0x0000FF00)
01722 #define  CAN_TDH0R_DATA6                     ((uint32_t)0x00FF0000)
01723 #define  CAN_TDH0R_DATA7                     ((uint32_t)0xFF000000)
01725 /*****************  Bit definition for CAN_TI1R register  *****************/
01726 #define  CAN_TI1R_TXRQ                       ((uint32_t)0x00000001)
01727 #define  CAN_TI1R_RTR                        ((uint32_t)0x00000002)
01728 #define  CAN_TI1R_IDE                        ((uint32_t)0x00000004)
01729 #define  CAN_TI1R_EXID                       ((uint32_t)0x001FFFF8)
01730 #define  CAN_TI1R_STID                       ((uint32_t)0xFFE00000)
01732 /*****************  Bit definition for CAN_TDT1R register  *****************/
01733 #define  CAN_TDT1R_DLC                       ((uint32_t)0x0000000F)
01734 #define  CAN_TDT1R_TGT                       ((uint32_t)0x00000100)
01735 #define  CAN_TDT1R_TIME                      ((uint32_t)0xFFFF0000)
01737 /*****************  Bit definition for CAN_TDL1R register  *****************/
01738 #define  CAN_TDL1R_DATA0                     ((uint32_t)0x000000FF)
01739 #define  CAN_TDL1R_DATA1                     ((uint32_t)0x0000FF00)
01740 #define  CAN_TDL1R_DATA2                     ((uint32_t)0x00FF0000)
01741 #define  CAN_TDL1R_DATA3                     ((uint32_t)0xFF000000)
01743 /*****************  Bit definition for CAN_TDH1R register  *****************/
01744 #define  CAN_TDH1R_DATA4                     ((uint32_t)0x000000FF)
01745 #define  CAN_TDH1R_DATA5                     ((uint32_t)0x0000FF00)
01746 #define  CAN_TDH1R_DATA6                     ((uint32_t)0x00FF0000)
01747 #define  CAN_TDH1R_DATA7                     ((uint32_t)0xFF000000)
01749 /*****************  Bit definition for CAN_TI2R register  *****************/
01750 #define  CAN_TI2R_TXRQ                       ((uint32_t)0x00000001)
01751 #define  CAN_TI2R_RTR                        ((uint32_t)0x00000002)
01752 #define  CAN_TI2R_IDE                        ((uint32_t)0x00000004)
01753 #define  CAN_TI2R_EXID                       ((uint32_t)0x001FFFF8)
01754 #define  CAN_TI2R_STID                       ((uint32_t)0xFFE00000)
01756 /*****************  Bit definition for CAN_TDT2R register  *****************/
01757 #define  CAN_TDT2R_DLC                       ((uint32_t)0x0000000F)
01758 #define  CAN_TDT2R_TGT                       ((uint32_t)0x00000100)
01759 #define  CAN_TDT2R_TIME                      ((uint32_t)0xFFFF0000)
01761 /*****************  Bit definition for CAN_TDL2R register  *****************/
01762 #define  CAN_TDL2R_DATA0                     ((uint32_t)0x000000FF)
01763 #define  CAN_TDL2R_DATA1                     ((uint32_t)0x0000FF00)
01764 #define  CAN_TDL2R_DATA2                     ((uint32_t)0x00FF0000)
01765 #define  CAN_TDL2R_DATA3                     ((uint32_t)0xFF000000)
01767 /*****************  Bit definition for CAN_TDH2R register  *****************/
01768 #define  CAN_TDH2R_DATA4                     ((uint32_t)0x000000FF)
01769 #define  CAN_TDH2R_DATA5                     ((uint32_t)0x0000FF00)
01770 #define  CAN_TDH2R_DATA6                     ((uint32_t)0x00FF0000)
01771 #define  CAN_TDH2R_DATA7                     ((uint32_t)0xFF000000)
01773 /*****************  Bit definition for CAN_RI0R register  *****************/
01774 #define  CAN_RI0R_RTR                        ((uint32_t)0x00000002)
01775 #define  CAN_RI0R_IDE                        ((uint32_t)0x00000004)
01776 #define  CAN_RI0R_EXID                       ((uint32_t)0x001FFFF8)
01777 #define  CAN_RI0R_STID                       ((uint32_t)0xFFE00000)
01779 /*****************  Bit definition for CAN_RDT0R register  *****************/
01780 #define  CAN_RDT0R_DLC                       ((uint32_t)0x0000000F)
01781 #define  CAN_RDT0R_FMI                       ((uint32_t)0x0000FF00)
01782 #define  CAN_RDT0R_TIME                      ((uint32_t)0xFFFF0000)
01784 /*****************  Bit definition for CAN_RDL0R register  *****************/
01785 #define  CAN_RDL0R_DATA0                     ((uint32_t)0x000000FF)
01786 #define  CAN_RDL0R_DATA1                     ((uint32_t)0x0000FF00)
01787 #define  CAN_RDL0R_DATA2                     ((uint32_t)0x00FF0000)
01788 #define  CAN_RDL0R_DATA3                     ((uint32_t)0xFF000000)
01790 /*****************  Bit definition for CAN_RDH0R register  *****************/
01791 #define  CAN_RDH0R_DATA4                     ((uint32_t)0x000000FF)
01792 #define  CAN_RDH0R_DATA5                     ((uint32_t)0x0000FF00)
01793 #define  CAN_RDH0R_DATA6                     ((uint32_t)0x00FF0000)
01794 #define  CAN_RDH0R_DATA7                     ((uint32_t)0xFF000000)
01796 /*****************  Bit definition for CAN_RI1R register  *****************/
01797 #define  CAN_RI1R_RTR                        ((uint32_t)0x00000002)
01798 #define  CAN_RI1R_IDE                        ((uint32_t)0x00000004)
01799 #define  CAN_RI1R_EXID                       ((uint32_t)0x001FFFF8)
01800 #define  CAN_RI1R_STID                       ((uint32_t)0xFFE00000)
01802 /*****************  Bit definition for CAN_RDT1R register  *****************/
01803 #define  CAN_RDT1R_DLC                       ((uint32_t)0x0000000F)
01804 #define  CAN_RDT1R_FMI                       ((uint32_t)0x0000FF00)
01805 #define  CAN_RDT1R_TIME                      ((uint32_t)0xFFFF0000)
01807 /*****************  Bit definition for CAN_RDL1R register  *****************/
01808 #define  CAN_RDL1R_DATA0                     ((uint32_t)0x000000FF)
01809 #define  CAN_RDL1R_DATA1                     ((uint32_t)0x0000FF00)
01810 #define  CAN_RDL1R_DATA2                     ((uint32_t)0x00FF0000)
```

```
01811 #define  CAN_RDL1R_DATA3                     ((uint32_t)0xFF000000)
01813 /*******************  Bit definition for CAN_RDH1R register  *****************/
01814 #define  CAN_RDH1R_DATA4                     ((uint32_t)0x000000FF)
01815 #define  CAN_RDH1R_DATA5                     ((uint32_t)0x0000FF00)
01816 #define  CAN_RDH1R_DATA6                     ((uint32_t)0x00FF0000)
01817 #define  CAN_RDH1R_DATA7                     ((uint32_t)0xFF000000)
01820 /*******************  Bit definition for CAN_FMR register  *******************/
01821 #define  CAN_FMR_FINIT                       ((uint8_t)0x01)
01823 /*******************  Bit definition for CAN_FM1R register  ******************/
01824 #define  CAN_FM1R_FBM                        ((uint16_t)0x3FFF)
01825 #define  CAN_FM1R_FBM0                       ((uint16_t)0x0001)
01826 #define  CAN_FM1R_FBM1                       ((uint16_t)0x0002)
01827 #define  CAN_FM1R_FBM2                       ((uint16_t)0x0004)
01828 #define  CAN_FM1R_FBM3                       ((uint16_t)0x0008)
01829 #define  CAN_FM1R_FBM4                       ((uint16_t)0x0010)
01830 #define  CAN_FM1R_FBM5                       ((uint16_t)0x0020)
01831 #define  CAN_FM1R_FBM6                       ((uint16_t)0x0040)
01832 #define  CAN_FM1R_FBM7                       ((uint16_t)0x0080)
01833 #define  CAN_FM1R_FBM8                       ((uint16_t)0x0100)
01834 #define  CAN_FM1R_FBM9                       ((uint16_t)0x0200)
01835 #define  CAN_FM1R_FBM10                      ((uint16_t)0x0400)
01836 #define  CAN_FM1R_FBM11                      ((uint16_t)0x0800)
01837 #define  CAN_FM1R_FBM12                      ((uint16_t)0x1000)
01838 #define  CAN_FM1R_FBM13                      ((uint16_t)0x2000)
01840 /*******************  Bit definition for CAN_FS1R register  ******************/
01841 #define  CAN_FS1R_FSC                        ((uint16_t)0x3FFF)
01842 #define  CAN_FS1R_FSC0                       ((uint16_t)0x0001)
01843 #define  CAN_FS1R_FSC1                       ((uint16_t)0x0002)
01844 #define  CAN_FS1R_FSC2                       ((uint16_t)0x0004)
01845 #define  CAN_FS1R_FSC3                       ((uint16_t)0x0008)
01846 #define  CAN_FS1R_FSC4                       ((uint16_t)0x0010)
01847 #define  CAN_FS1R_FSC5                       ((uint16_t)0x0020)
01848 #define  CAN_FS1R_FSC6                       ((uint16_t)0x0040)
01849 #define  CAN_FS1R_FSC7                       ((uint16_t)0x0080)
01850 #define  CAN_FS1R_FSC8                       ((uint16_t)0x0100)
01851 #define  CAN_FS1R_FSC9                       ((uint16_t)0x0200)
01852 #define  CAN_FS1R_FSC10                      ((uint16_t)0x0400)
01853 #define  CAN_FS1R_FSC11                      ((uint16_t)0x0800)
01854 #define  CAN_FS1R_FSC12                      ((uint16_t)0x1000)
01855 #define  CAN_FS1R_FSC13                      ((uint16_t)0x2000)
01857 /******************  Bit definition for CAN_FFA1R register  ******************/
01858 #define  CAN_FFA1R_FFA                       ((uint16_t)0x3FFF)
01859 #define  CAN_FFA1R_FFA0                      ((uint16_t)0x0001)
01860 #define  CAN_FFA1R_FFA1                      ((uint16_t)0x0002)
01861 #define  CAN_FFA1R_FFA2                      ((uint16_t)0x0004)
01862 #define  CAN_FFA1R_FFA3                      ((uint16_t)0x0008)
01863 #define  CAN_FFA1R_FFA4                      ((uint16_t)0x0010)
01864 #define  CAN_FFA1R_FFA5                      ((uint16_t)0x0020)
01865 #define  CAN_FFA1R_FFA6                      ((uint16_t)0x0040)
01866 #define  CAN_FFA1R_FFA7                      ((uint16_t)0x0080)
01867 #define  CAN_FFA1R_FFA8                      ((uint16_t)0x0100)
01868 #define  CAN_FFA1R_FFA9                      ((uint16_t)0x0200)
01869 #define  CAN_FFA1R_FFA10                     ((uint16_t)0x0400)
01870 #define  CAN_FFA1R_FFA11                     ((uint16_t)0x0800)
01871 #define  CAN_FFA1R_FFA12                     ((uint16_t)0x1000)
01872 #define  CAN_FFA1R_FFA13                     ((uint16_t)0x2000)
01874 /*******************  Bit definition for CAN_FA1R register  ******************/
01875 #define  CAN_FA1R_FACT                       ((uint16_t)0x3FFF)
01876 #define  CAN_FA1R_FACT0                      ((uint16_t)0x0001)
01877 #define  CAN_FA1R_FACT1                      ((uint16_t)0x0002)
01878 #define  CAN_FA1R_FACT2                      ((uint16_t)0x0004)
01879 #define  CAN_FA1R_FACT3                      ((uint16_t)0x0008)
01880 #define  CAN_FA1R_FACT4                      ((uint16_t)0x0010)
01881 #define  CAN_FA1R_FACT5                      ((uint16_t)0x0020)
01882 #define  CAN_FA1R_FACT6                      ((uint16_t)0x0040)
01883 #define  CAN_FA1R_FACT7                      ((uint16_t)0x0080)
01884 #define  CAN_FA1R_FACT8                      ((uint16_t)0x0100)
01885 #define  CAN_FA1R_FACT9                      ((uint16_t)0x0200)
01886 #define  CAN_FA1R_FACT10                     ((uint16_t)0x0400)
01887 #define  CAN_FA1R_FACT11                     ((uint16_t)0x0800)
01888 #define  CAN_FA1R_FACT12                     ((uint16_t)0x1000)
01889 #define  CAN_FA1R_FACT13                     ((uint16_t)0x2000)
01891 /*******************  Bit definition for CAN_F0R1 register  ******************/
01892 #define  CAN_F0R1_FB0                        ((uint32_t)0x00000001)
01893 #define  CAN_F0R1_FB1                        ((uint32_t)0x00000002)
01894 #define  CAN_F0R1_FB2                        ((uint32_t)0x00000004)
01895 #define  CAN_F0R1_FB3                        ((uint32_t)0x00000008)
01896 #define  CAN_F0R1_FB4                        ((uint32_t)0x00000010)
01897 #define  CAN_F0R1_FB5                        ((uint32_t)0x00000020)
01898 #define  CAN_F0R1_FB6                        ((uint32_t)0x00000040)
01899 #define  CAN_F0R1_FB7                        ((uint32_t)0x00000080)
01900 #define  CAN_F0R1_FB8                        ((uint32_t)0x00000100)
01901 #define  CAN_F0R1_FB9                        ((uint32_t)0x00000200)
01902 #define  CAN_F0R1_FB10                       ((uint32_t)0x00000400)
01903 #define  CAN_F0R1_FB11                       ((uint32_t)0x00000800)
01904 #define  CAN_F0R1_FB12                       ((uint32_t)0x00001000)
01905 #define  CAN_F0R1_FB13                       ((uint32_t)0x00002000)
```

```
01906 #define  CAN_F0R1_FB14                       ((uint32_t)0x00004000)
01907 #define  CAN_F0R1_FB15                       ((uint32_t)0x00008000)
01908 #define  CAN_F0R1_FB16                       ((uint32_t)0x00010000)
01909 #define  CAN_F0R1_FB17                       ((uint32_t)0x00020000)
01910 #define  CAN_F0R1_FB18                       ((uint32_t)0x00040000)
01911 #define  CAN_F0R1_FB19                       ((uint32_t)0x00080000)
01912 #define  CAN_F0R1_FB20                       ((uint32_t)0x00100000)
01913 #define  CAN_F0R1_FB21                       ((uint32_t)0x00200000)
01914 #define  CAN_F0R1_FB22                       ((uint32_t)0x00400000)
01915 #define  CAN_F0R1_FB23                       ((uint32_t)0x00800000)
01916 #define  CAN_F0R1_FB24                       ((uint32_t)0x01000000)
01917 #define  CAN_F0R1_FB25                       ((uint32_t)0x02000000)
01918 #define  CAN_F0R1_FB26                       ((uint32_t)0x04000000)
01919 #define  CAN_F0R1_FB27                       ((uint32_t)0x08000000)
01920 #define  CAN_F0R1_FB28                       ((uint32_t)0x10000000)
01921 #define  CAN_F0R1_FB29                       ((uint32_t)0x20000000)
01922 #define  CAN_F0R1_FB30                       ((uint32_t)0x40000000)
01923 #define  CAN_F0R1_FB31                       ((uint32_t)0x80000000)
01925 /*******************  Bit definition for CAN_F1R1 register  *******************/
01926 #define  CAN_F1R1_FB0                        ((uint32_t)0x00000001)
01927 #define  CAN_F1R1_FB1                        ((uint32_t)0x00000002)
01928 #define  CAN_F1R1_FB2                        ((uint32_t)0x00000004)
01929 #define  CAN_F1R1_FB3                        ((uint32_t)0x00000008)
01930 #define  CAN_F1R1_FB4                        ((uint32_t)0x00000010)
01931 #define  CAN_F1R1_FB5                        ((uint32_t)0x00000020)
01932 #define  CAN_F1R1_FB6                        ((uint32_t)0x00000040)
01933 #define  CAN_F1R1_FB7                        ((uint32_t)0x00000080)
01934 #define  CAN_F1R1_FB8                        ((uint32_t)0x00000100)
01935 #define  CAN_F1R1_FB9                        ((uint32_t)0x00000200)
01936 #define  CAN_F1R1_FB10                       ((uint32_t)0x00000400)
01937 #define  CAN_F1R1_FB11                       ((uint32_t)0x00000800)
01938 #define  CAN_F1R1_FB12                       ((uint32_t)0x00001000)
01939 #define  CAN_F1R1_FB13                       ((uint32_t)0x00002000)
01940 #define  CAN_F1R1_FB14                       ((uint32_t)0x00004000)
01941 #define  CAN_F1R1_FB15                       ((uint32_t)0x00008000)
01942 #define  CAN_F1R1_FB16                       ((uint32_t)0x00010000)
01943 #define  CAN_F1R1_FB17                       ((uint32_t)0x00020000)
01944 #define  CAN_F1R1_FB18                       ((uint32_t)0x00040000)
01945 #define  CAN_F1R1_FB19                       ((uint32_t)0x00080000)
01946 #define  CAN_F1R1_FB20                       ((uint32_t)0x00100000)
01947 #define  CAN_F1R1_FB21                       ((uint32_t)0x00200000)
01948 #define  CAN_F1R1_FB22                       ((uint32_t)0x00400000)
01949 #define  CAN_F1R1_FB23                       ((uint32_t)0x00800000)
01950 #define  CAN_F1R1_FB24                       ((uint32_t)0x01000000)
01951 #define  CAN_F1R1_FB25                       ((uint32_t)0x02000000)
01952 #define  CAN_F1R1_FB26                       ((uint32_t)0x04000000)
01953 #define  CAN_F1R1_FB27                       ((uint32_t)0x08000000)
01954 #define  CAN_F1R1_FB28                       ((uint32_t)0x10000000)
01955 #define  CAN_F1R1_FB29                       ((uint32_t)0x20000000)
01956 #define  CAN_F1R1_FB30                       ((uint32_t)0x40000000)
01957 #define  CAN_F1R1_FB31                       ((uint32_t)0x80000000)
01959 /*******************  Bit definition for CAN_F2R1 register  *******************/
01960 #define  CAN_F2R1_FB0                        ((uint32_t)0x00000001)
01961 #define  CAN_F2R1_FB1                        ((uint32_t)0x00000002)
01962 #define  CAN_F2R1_FB2                        ((uint32_t)0x00000004)
01963 #define  CAN_F2R1_FB3                        ((uint32_t)0x00000008)
01964 #define  CAN_F2R1_FB4                        ((uint32_t)0x00000010)
01965 #define  CAN_F2R1_FB5                        ((uint32_t)0x00000020)
01966 #define  CAN_F2R1_FB6                        ((uint32_t)0x00000040)
01967 #define  CAN_F2R1_FB7                        ((uint32_t)0x00000080)
01968 #define  CAN_F2R1_FB8                        ((uint32_t)0x00000100)
01969 #define  CAN_F2R1_FB9                        ((uint32_t)0x00000200)
01970 #define  CAN_F2R1_FB10                       ((uint32_t)0x00000400)
01971 #define  CAN_F2R1_FB11                       ((uint32_t)0x00000800)
01972 #define  CAN_F2R1_FB12                       ((uint32_t)0x00001000)
01973 #define  CAN_F2R1_FB13                       ((uint32_t)0x00002000)
01974 #define  CAN_F2R1_FB14                       ((uint32_t)0x00004000)
01975 #define  CAN_F2R1_FB15                       ((uint32_t)0x00008000)
01976 #define  CAN_F2R1_FB16                       ((uint32_t)0x00010000)
01977 #define  CAN_F2R1_FB17                       ((uint32_t)0x00020000)
01978 #define  CAN_F2R1_FB18                       ((uint32_t)0x00040000)
01979 #define  CAN_F2R1_FB19                       ((uint32_t)0x00080000)
01980 #define  CAN_F2R1_FB20                       ((uint32_t)0x00100000)
01981 #define  CAN_F2R1_FB21                       ((uint32_t)0x00200000)
01982 #define  CAN_F2R1_FB22                       ((uint32_t)0x00400000)
01983 #define  CAN_F2R1_FB23                       ((uint32_t)0x00800000)
01984 #define  CAN_F2R1_FB24                       ((uint32_t)0x01000000)
01985 #define  CAN_F2R1_FB25                       ((uint32_t)0x02000000)
01986 #define  CAN_F2R1_FB26                       ((uint32_t)0x04000000)
01987 #define  CAN_F2R1_FB27                       ((uint32_t)0x08000000)
01988 #define  CAN_F2R1_FB28                       ((uint32_t)0x10000000)
01989 #define  CAN_F2R1_FB29                       ((uint32_t)0x20000000)
01990 #define  CAN_F2R1_FB30                       ((uint32_t)0x40000000)
01991 #define  CAN_F2R1_FB31                       ((uint32_t)0x80000000)
01993 /*******************  Bit definition for CAN_F3R1 register  *******************/
01994 #define  CAN_F3R1_FB0                        ((uint32_t)0x00000001)
01995 #define  CAN_F3R1_FB1                        ((uint32_t)0x00000002)
```

```
01996 #define  CAN_F3R1_FB2                        ((uint32_t)0x00000004)
01997 #define  CAN_F3R1_FB3                        ((uint32_t)0x00000008)
01998 #define  CAN_F3R1_FB4                        ((uint32_t)0x00000010)
01999 #define  CAN_F3R1_FB5                        ((uint32_t)0x00000020)
02000 #define  CAN_F3R1_FB6                        ((uint32_t)0x00000040)
02001 #define  CAN_F3R1_FB7                        ((uint32_t)0x00000080)
02002 #define  CAN_F3R1_FB8                        ((uint32_t)0x00000100)
02003 #define  CAN_F3R1_FB9                        ((uint32_t)0x00000200)
02004 #define  CAN_F3R1_FB10                       ((uint32_t)0x00000400)
02005 #define  CAN_F3R1_FB11                       ((uint32_t)0x00000800)
02006 #define  CAN_F3R1_FB12                       ((uint32_t)0x00001000)
02007 #define  CAN_F3R1_FB13                       ((uint32_t)0x00002000)
02008 #define  CAN_F3R1_FB14                       ((uint32_t)0x00004000)
02009 #define  CAN_F3R1_FB15                       ((uint32_t)0x00008000)
02010 #define  CAN_F3R1_FB16                       ((uint32_t)0x00010000)
02011 #define  CAN_F3R1_FB17                       ((uint32_t)0x00020000)
02012 #define  CAN_F3R1_FB18                       ((uint32_t)0x00040000)
02013 #define  CAN_F3R1_FB19                       ((uint32_t)0x00080000)
02014 #define  CAN_F3R1_FB20                       ((uint32_t)0x00100000)
02015 #define  CAN_F3R1_FB21                       ((uint32_t)0x00200000)
02016 #define  CAN_F3R1_FB22                       ((uint32_t)0x00400000)
02017 #define  CAN_F3R1_FB23                       ((uint32_t)0x00800000)
02018 #define  CAN_F3R1_FB24                       ((uint32_t)0x01000000)
02019 #define  CAN_F3R1_FB25                       ((uint32_t)0x02000000)
02020 #define  CAN_F3R1_FB26                       ((uint32_t)0x04000000)
02021 #define  CAN_F3R1_FB27                       ((uint32_t)0x08000000)
02022 #define  CAN_F3R1_FB28                       ((uint32_t)0x10000000)
02023 #define  CAN_F3R1_FB29                       ((uint32_t)0x20000000)
02024 #define  CAN_F3R1_FB30                       ((uint32_t)0x40000000)
02025 #define  CAN_F3R1_FB31                       ((uint32_t)0x80000000)
02027 /*******************  Bit definition for CAN_F4R1 register  *******************/
02028 #define  CAN_F4R1_FB0                        ((uint32_t)0x00000001)
02029 #define  CAN_F4R1_FB1                        ((uint32_t)0x00000002)
02030 #define  CAN_F4R1_FB2                        ((uint32_t)0x00000004)
02031 #define  CAN_F4R1_FB3                        ((uint32_t)0x00000008)
02032 #define  CAN_F4R1_FB4                        ((uint32_t)0x00000010)
02033 #define  CAN_F4R1_FB5                        ((uint32_t)0x00000020)
02034 #define  CAN_F4R1_FB6                        ((uint32_t)0x00000040)
02035 #define  CAN_F4R1_FB7                        ((uint32_t)0x00000080)
02036 #define  CAN_F4R1_FB8                        ((uint32_t)0x00000100)
02037 #define  CAN_F4R1_FB9                        ((uint32_t)0x00000200)
02038 #define  CAN_F4R1_FB10                       ((uint32_t)0x00000400)
02039 #define  CAN_F4R1_FB11                       ((uint32_t)0x00000800)
02040 #define  CAN_F4R1_FB12                       ((uint32_t)0x00001000)
02041 #define  CAN_F4R1_FB13                       ((uint32_t)0x00002000)
02042 #define  CAN_F4R1_FB14                       ((uint32_t)0x00004000)
02043 #define  CAN_F4R1_FB15                       ((uint32_t)0x00008000)
02044 #define  CAN_F4R1_FB16                       ((uint32_t)0x00010000)
02045 #define  CAN_F4R1_FB17                       ((uint32_t)0x00020000)
02046 #define  CAN_F4R1_FB18                       ((uint32_t)0x00040000)
02047 #define  CAN_F4R1_FB19                       ((uint32_t)0x00080000)
02048 #define  CAN_F4R1_FB20                       ((uint32_t)0x00100000)
02049 #define  CAN_F4R1_FB21                       ((uint32_t)0x00200000)
02050 #define  CAN_F4R1_FB22                       ((uint32_t)0x00400000)
02051 #define  CAN_F4R1_FB23                       ((uint32_t)0x00800000)
02052 #define  CAN_F4R1_FB24                       ((uint32_t)0x01000000)
02053 #define  CAN_F4R1_FB25                       ((uint32_t)0x02000000)
02054 #define  CAN_F4R1_FB26                       ((uint32_t)0x04000000)
02055 #define  CAN_F4R1_FB27                       ((uint32_t)0x08000000)
02056 #define  CAN_F4R1_FB28                       ((uint32_t)0x10000000)
02057 #define  CAN_F4R1_FB29                       ((uint32_t)0x20000000)
02058 #define  CAN_F4R1_FB30                       ((uint32_t)0x40000000)
02059 #define  CAN_F4R1_FB31                       ((uint32_t)0x80000000)
02061 /*******************  Bit definition for CAN_F5R1 register  *******************/
02062 #define  CAN_F5R1_FB0                        ((uint32_t)0x00000001)
02063 #define  CAN_F5R1_FB1                        ((uint32_t)0x00000002)
02064 #define  CAN_F5R1_FB2                        ((uint32_t)0x00000004)
02065 #define  CAN_F5R1_FB3                        ((uint32_t)0x00000008)
02066 #define  CAN_F5R1_FB4                        ((uint32_t)0x00000010)
02067 #define  CAN_F5R1_FB5                        ((uint32_t)0x00000020)
02068 #define  CAN_F5R1_FB6                        ((uint32_t)0x00000040)
02069 #define  CAN_F5R1_FB7                        ((uint32_t)0x00000080)
02070 #define  CAN_F5R1_FB8                        ((uint32_t)0x00000100)
02071 #define  CAN_F5R1_FB9                        ((uint32_t)0x00000200)
02072 #define  CAN_F5R1_FB10                       ((uint32_t)0x00000400)
02073 #define  CAN_F5R1_FB11                       ((uint32_t)0x00000800)
02074 #define  CAN_F5R1_FB12                       ((uint32_t)0x00001000)
02075 #define  CAN_F5R1_FB13                       ((uint32_t)0x00002000)
02076 #define  CAN_F5R1_FB14                       ((uint32_t)0x00004000)
02077 #define  CAN_F5R1_FB15                       ((uint32_t)0x00008000)
02078 #define  CAN_F5R1_FB16                       ((uint32_t)0x00010000)
02079 #define  CAN_F5R1_FB17                       ((uint32_t)0x00020000)
02080 #define  CAN_F5R1_FB18                       ((uint32_t)0x00040000)
02081 #define  CAN_F5R1_FB19                       ((uint32_t)0x00080000)
02082 #define  CAN_F5R1_FB20                       ((uint32_t)0x00100000)
02083 #define  CAN_F5R1_FB21                       ((uint32_t)0x00200000)
02084 #define  CAN_F5R1_FB22                       ((uint32_t)0x00400000)
```

```
02085 #define  CAN_F5R1_FB23                       ((uint32_t)0x00800000)
02086 #define  CAN_F5R1_FB24                       ((uint32_t)0x01000000)
02087 #define  CAN_F5R1_FB25                       ((uint32_t)0x02000000)
02088 #define  CAN_F5R1_FB26                       ((uint32_t)0x04000000)
02089 #define  CAN_F5R1_FB27                       ((uint32_t)0x08000000)
02090 #define  CAN_F5R1_FB28                       ((uint32_t)0x10000000)
02091 #define  CAN_F5R1_FB29                       ((uint32_t)0x20000000)
02092 #define  CAN_F5R1_FB30                       ((uint32_t)0x40000000)
02093 #define  CAN_F5R1_FB31                       ((uint32_t)0x80000000)
02095 /*******************  Bit definition for CAN_F6R1 register  *******************/
02096 #define  CAN_F6R1_FB0                        ((uint32_t)0x00000001)
02097 #define  CAN_F6R1_FB1                        ((uint32_t)0x00000002)
02098 #define  CAN_F6R1_FB2                        ((uint32_t)0x00000004)
02099 #define  CAN_F6R1_FB3                        ((uint32_t)0x00000008)
02100 #define  CAN_F6R1_FB4                        ((uint32_t)0x00000010)
02101 #define  CAN_F6R1_FB5                        ((uint32_t)0x00000020)
02102 #define  CAN_F6R1_FB6                        ((uint32_t)0x00000040)
02103 #define  CAN_F6R1_FB7                        ((uint32_t)0x00000080)
02104 #define  CAN_F6R1_FB8                        ((uint32_t)0x00000100)
02105 #define  CAN_F6R1_FB9                        ((uint32_t)0x00000200)
02106 #define  CAN_F6R1_FB10                       ((uint32_t)0x00000400)
02107 #define  CAN_F6R1_FB11                       ((uint32_t)0x00000800)
02108 #define  CAN_F6R1_FB12                       ((uint32_t)0x00001000)
02109 #define  CAN_F6R1_FB13                       ((uint32_t)0x00002000)
02110 #define  CAN_F6R1_FB14                       ((uint32_t)0x00004000)
02111 #define  CAN_F6R1_FB15                       ((uint32_t)0x00008000)
02112 #define  CAN_F6R1_FB16                       ((uint32_t)0x00010000)
02113 #define  CAN_F6R1_FB17                       ((uint32_t)0x00020000)
02114 #define  CAN_F6R1_FB18                       ((uint32_t)0x00040000)
02115 #define  CAN_F6R1_FB19                       ((uint32_t)0x00080000)
02116 #define  CAN_F6R1_FB20                       ((uint32_t)0x00100000)
02117 #define  CAN_F6R1_FB21                       ((uint32_t)0x00200000)
02118 #define  CAN_F6R1_FB22                       ((uint32_t)0x00400000)
02119 #define  CAN_F6R1_FB23                       ((uint32_t)0x00800000)
02120 #define  CAN_F6R1_FB24                       ((uint32_t)0x01000000)
02121 #define  CAN_F6R1_FB25                       ((uint32_t)0x02000000)
02122 #define  CAN_F6R1_FB26                       ((uint32_t)0x04000000)
02123 #define  CAN_F6R1_FB27                       ((uint32_t)0x08000000)
02124 #define  CAN_F6R1_FB28                       ((uint32_t)0x10000000)
02125 #define  CAN_F6R1_FB29                       ((uint32_t)0x20000000)
02126 #define  CAN_F6R1_FB30                       ((uint32_t)0x40000000)
02127 #define  CAN_F6R1_FB31                       ((uint32_t)0x80000000)
02129 /*******************  Bit definition for CAN_F7R1 register  *******************/
02130 #define  CAN_F7R1_FB0                        ((uint32_t)0x00000001)
02131 #define  CAN_F7R1_FB1                        ((uint32_t)0x00000002)
02132 #define  CAN_F7R1_FB2                        ((uint32_t)0x00000004)
02133 #define  CAN_F7R1_FB3                        ((uint32_t)0x00000008)
02134 #define  CAN_F7R1_FB4                        ((uint32_t)0x00000010)
02135 #define  CAN_F7R1_FB5                        ((uint32_t)0x00000020)
02136 #define  CAN_F7R1_FB6                        ((uint32_t)0x00000040)
02137 #define  CAN_F7R1_FB7                        ((uint32_t)0x00000080)
02138 #define  CAN_F7R1_FB8                        ((uint32_t)0x00000100)
02139 #define  CAN_F7R1_FB9                        ((uint32_t)0x00000200)
02140 #define  CAN_F7R1_FB10                       ((uint32_t)0x00000400)
02141 #define  CAN_F7R1_FB11                       ((uint32_t)0x00000800)
02142 #define  CAN_F7R1_FB12                       ((uint32_t)0x00001000)
02143 #define  CAN_F7R1_FB13                       ((uint32_t)0x00002000)
02144 #define  CAN_F7R1_FB14                       ((uint32_t)0x00004000)
02145 #define  CAN_F7R1_FB15                       ((uint32_t)0x00008000)
02146 #define  CAN_F7R1_FB16                       ((uint32_t)0x00010000)
02147 #define  CAN_F7R1_FB17                       ((uint32_t)0x00020000)
02148 #define  CAN_F7R1_FB18                       ((uint32_t)0x00040000)
02149 #define  CAN_F7R1_FB19                       ((uint32_t)0x00080000)
02150 #define  CAN_F7R1_FB20                       ((uint32_t)0x00100000)
02151 #define  CAN_F7R1_FB21                       ((uint32_t)0x00200000)
02152 #define  CAN_F7R1_FB22                       ((uint32_t)0x00400000)
02153 #define  CAN_F7R1_FB23                       ((uint32_t)0x00800000)
02154 #define  CAN_F7R1_FB24                       ((uint32_t)0x01000000)
02155 #define  CAN_F7R1_FB25                       ((uint32_t)0x02000000)
02156 #define  CAN_F7R1_FB26                       ((uint32_t)0x04000000)
02157 #define  CAN_F7R1_FB27                       ((uint32_t)0x08000000)
02158 #define  CAN_F7R1_FB28                       ((uint32_t)0x10000000)
02159 #define  CAN_F7R1_FB29                       ((uint32_t)0x20000000)
02160 #define  CAN_F7R1_FB30                       ((uint32_t)0x40000000)
02161 #define  CAN_F7R1_FB31                       ((uint32_t)0x80000000)
02163 /*******************  Bit definition for CAN_F8R1 register  *******************/
02164 #define  CAN_F8R1_FB0                        ((uint32_t)0x00000001)
02165 #define  CAN_F8R1_FB1                        ((uint32_t)0x00000002)
02166 #define  CAN_F8R1_FB2                        ((uint32_t)0x00000004)
02167 #define  CAN_F8R1_FB3                        ((uint32_t)0x00000008)
02168 #define  CAN_F8R1_FB4                        ((uint32_t)0x00000010)
02169 #define  CAN_F8R1_FB5                        ((uint32_t)0x00000020)
02170 #define  CAN_F8R1_FB6                        ((uint32_t)0x00000040)
02171 #define  CAN_F8R1_FB7                        ((uint32_t)0x00000080)
02172 #define  CAN_F8R1_FB8                        ((uint32_t)0x00000100)
02173 #define  CAN_F8R1_FB9                        ((uint32_t)0x00000200)
02174 #define  CAN_F8R1_FB10                       ((uint32_t)0x00000400)
```

```
02175 #define  CAN_F8R1_FB11                       ((uint32_t)0x00000800)
02176 #define  CAN_F8R1_FB12                       ((uint32_t)0x00001000)
02177 #define  CAN_F8R1_FB13                       ((uint32_t)0x00002000)
02178 #define  CAN_F8R1_FB14                       ((uint32_t)0x00004000)
02179 #define  CAN_F8R1_FB15                       ((uint32_t)0x00008000)
02180 #define  CAN_F8R1_FB16                       ((uint32_t)0x00010000)
02181 #define  CAN_F8R1_FB17                       ((uint32_t)0x00020000)
02182 #define  CAN_F8R1_FB18                       ((uint32_t)0x00040000)
02183 #define  CAN_F8R1_FB19                       ((uint32_t)0x00080000)
02184 #define  CAN_F8R1_FB20                       ((uint32_t)0x00100000)
02185 #define  CAN_F8R1_FB21                       ((uint32_t)0x00200000)
02186 #define  CAN_F8R1_FB22                       ((uint32_t)0x00400000)
02187 #define  CAN_F8R1_FB23                       ((uint32_t)0x00800000)
02188 #define  CAN_F8R1_FB24                       ((uint32_t)0x01000000)
02189 #define  CAN_F8R1_FB25                       ((uint32_t)0x02000000)
02190 #define  CAN_F8R1_FB26                       ((uint32_t)0x04000000)
02191 #define  CAN_F8R1_FB27                       ((uint32_t)0x08000000)
02192 #define  CAN_F8R1_FB28                       ((uint32_t)0x10000000)
02193 #define  CAN_F8R1_FB29                       ((uint32_t)0x20000000)
02194 #define  CAN_F8R1_FB30                       ((uint32_t)0x40000000)
02195 #define  CAN_F8R1_FB31                       ((uint32_t)0x80000000)
02197 /*******************  Bit definition for CAN_F9R1 register  *******************/
02198 #define  CAN_F9R1_FB0                        ((uint32_t)0x00000001)
02199 #define  CAN_F9R1_FB1                        ((uint32_t)0x00000002)
02200 #define  CAN_F9R1_FB2                        ((uint32_t)0x00000004)
02201 #define  CAN_F9R1_FB3                        ((uint32_t)0x00000008)
02202 #define  CAN_F9R1_FB4                        ((uint32_t)0x00000010)
02203 #define  CAN_F9R1_FB5                        ((uint32_t)0x00000020)
02204 #define  CAN_F9R1_FB6                        ((uint32_t)0x00000040)
02205 #define  CAN_F9R1_FB7                        ((uint32_t)0x00000080)
02206 #define  CAN_F9R1_FB8                        ((uint32_t)0x00000100)
02207 #define  CAN_F9R1_FB9                        ((uint32_t)0x00000200)
02208 #define  CAN_F9R1_FB10                       ((uint32_t)0x00000400)
02209 #define  CAN_F9R1_FB11                       ((uint32_t)0x00000800)
02210 #define  CAN_F9R1_FB12                       ((uint32_t)0x00001000)
02211 #define  CAN_F9R1_FB13                       ((uint32_t)0x00002000)
02212 #define  CAN_F9R1_FB14                       ((uint32_t)0x00004000)
02213 #define  CAN_F9R1_FB15                       ((uint32_t)0x00008000)
02214 #define  CAN_F9R1_FB16                       ((uint32_t)0x00010000)
02215 #define  CAN_F9R1_FB17                       ((uint32_t)0x00020000)
02216 #define  CAN_F9R1_FB18                       ((uint32_t)0x00040000)
02217 #define  CAN_F9R1_FB19                       ((uint32_t)0x00080000)
02218 #define  CAN_F9R1_FB20                       ((uint32_t)0x00100000)
02219 #define  CAN_F9R1_FB21                       ((uint32_t)0x00200000)
02220 #define  CAN_F9R1_FB22                       ((uint32_t)0x00400000)
02221 #define  CAN_F9R1_FB23                       ((uint32_t)0x00800000)
02222 #define  CAN_F9R1_FB24                       ((uint32_t)0x01000000)
02223 #define  CAN_F9R1_FB25                       ((uint32_t)0x02000000)
02224 #define  CAN_F9R1_FB26                       ((uint32_t)0x04000000)
02225 #define  CAN_F9R1_FB27                       ((uint32_t)0x08000000)
02226 #define  CAN_F9R1_FB28                       ((uint32_t)0x10000000)
02227 #define  CAN_F9R1_FB29                       ((uint32_t)0x20000000)
02228 #define  CAN_F9R1_FB30                       ((uint32_t)0x40000000)
02229 #define  CAN_F9R1_FB31                       ((uint32_t)0x80000000)
02231 /*******************  Bit definition for CAN_F10R1 register  ******************/
02232 #define  CAN_F10R1_FB0                       ((uint32_t)0x00000001)
02233 #define  CAN_F10R1_FB1                       ((uint32_t)0x00000002)
02234 #define  CAN_F10R1_FB2                       ((uint32_t)0x00000004)
02235 #define  CAN_F10R1_FB3                       ((uint32_t)0x00000008)
02236 #define  CAN_F10R1_FB4                       ((uint32_t)0x00000010)
02237 #define  CAN_F10R1_FB5                       ((uint32_t)0x00000020)
02238 #define  CAN_F10R1_FB6                       ((uint32_t)0x00000040)
02239 #define  CAN_F10R1_FB7                       ((uint32_t)0x00000080)
02240 #define  CAN_F10R1_FB8                       ((uint32_t)0x00000100)
02241 #define  CAN_F10R1_FB9                       ((uint32_t)0x00000200)
02242 #define  CAN_F10R1_FB10                      ((uint32_t)0x00000400)
02243 #define  CAN_F10R1_FB11                      ((uint32_t)0x00000800)
02244 #define  CAN_F10R1_FB12                      ((uint32_t)0x00001000)
02245 #define  CAN_F10R1_FB13                      ((uint32_t)0x00002000)
02246 #define  CAN_F10R1_FB14                      ((uint32_t)0x00004000)
02247 #define  CAN_F10R1_FB15                      ((uint32_t)0x00008000)
02248 #define  CAN_F10R1_FB16                      ((uint32_t)0x00010000)
02249 #define  CAN_F10R1_FB17                      ((uint32_t)0x00020000)
02250 #define  CAN_F10R1_FB18                      ((uint32_t)0x00040000)
02251 #define  CAN_F10R1_FB19                      ((uint32_t)0x00080000)
02252 #define  CAN_F10R1_FB20                      ((uint32_t)0x00100000)
02253 #define  CAN_F10R1_FB21                      ((uint32_t)0x00200000)
02254 #define  CAN_F10R1_FB22                      ((uint32_t)0x00400000)
02255 #define  CAN_F10R1_FB23                      ((uint32_t)0x00800000)
02256 #define  CAN_F10R1_FB24                      ((uint32_t)0x01000000)
02257 #define  CAN_F10R1_FB25                      ((uint32_t)0x02000000)
02258 #define  CAN_F10R1_FB26                      ((uint32_t)0x04000000)
02259 #define  CAN_F10R1_FB27                      ((uint32_t)0x08000000)
02260 #define  CAN_F10R1_FB28                      ((uint32_t)0x10000000)
02261 #define  CAN_F10R1_FB29                      ((uint32_t)0x20000000)
02262 #define  CAN_F10R1_FB30                      ((uint32_t)0x40000000)
02263 #define  CAN_F10R1_FB31                      ((uint32_t)0x80000000)
```

```
02265 /******************  Bit definition for CAN_F11R1 register  ******************/
02266 #define  CAN_F11R1_FB0                       ((uint32_t)0x00000001)
02267 #define  CAN_F11R1_FB1                       ((uint32_t)0x00000002)
02268 #define  CAN_F11R1_FB2                       ((uint32_t)0x00000004)
02269 #define  CAN_F11R1_FB3                       ((uint32_t)0x00000008)
02270 #define  CAN_F11R1_FB4                       ((uint32_t)0x00000010)
02271 #define  CAN_F11R1_FB5                       ((uint32_t)0x00000020)
02272 #define  CAN_F11R1_FB6                       ((uint32_t)0x00000040)
02273 #define  CAN_F11R1_FB7                       ((uint32_t)0x00000080)
02274 #define  CAN_F11R1_FB8                       ((uint32_t)0x00000100)
02275 #define  CAN_F11R1_FB9                       ((uint32_t)0x00000200)
02276 #define  CAN_F11R1_FB10                      ((uint32_t)0x00000400)
02277 #define  CAN_F11R1_FB11                      ((uint32_t)0x00000800)
02278 #define  CAN_F11R1_FB12                      ((uint32_t)0x00001000)
02279 #define  CAN_F11R1_FB13                      ((uint32_t)0x00002000)
02280 #define  CAN_F11R1_FB14                      ((uint32_t)0x00004000)
02281 #define  CAN_F11R1_FB15                      ((uint32_t)0x00008000)
02282 #define  CAN_F11R1_FB16                      ((uint32_t)0x00010000)
02283 #define  CAN_F11R1_FB17                      ((uint32_t)0x00020000)
02284 #define  CAN_F11R1_FB18                      ((uint32_t)0x00040000)
02285 #define  CAN_F11R1_FB19                      ((uint32_t)0x00080000)
02286 #define  CAN_F11R1_FB20                      ((uint32_t)0x00100000)
02287 #define  CAN_F11R1_FB21                      ((uint32_t)0x00200000)
02288 #define  CAN_F11R1_FB22                      ((uint32_t)0x00400000)
02289 #define  CAN_F11R1_FB23                      ((uint32_t)0x00800000)
02290 #define  CAN_F11R1_FB24                      ((uint32_t)0x01000000)
02291 #define  CAN_F11R1_FB25                      ((uint32_t)0x02000000)
02292 #define  CAN_F11R1_FB26                      ((uint32_t)0x04000000)
02293 #define  CAN_F11R1_FB27                      ((uint32_t)0x08000000)
02294 #define  CAN_F11R1_FB28                      ((uint32_t)0x10000000)
02295 #define  CAN_F11R1_FB29                      ((uint32_t)0x20000000)
02296 #define  CAN_F11R1_FB30                      ((uint32_t)0x40000000)
02297 #define  CAN_F11R1_FB31                      ((uint32_t)0x80000000)
02299 /******************  Bit definition for CAN_F12R1 register  ******************/
02300 #define  CAN_F12R1_FB0                       ((uint32_t)0x00000001)
02301 #define  CAN_F12R1_FB1                       ((uint32_t)0x00000002)
02302 #define  CAN_F12R1_FB2                       ((uint32_t)0x00000004)
02303 #define  CAN_F12R1_FB3                       ((uint32_t)0x00000008)
02304 #define  CAN_F12R1_FB4                       ((uint32_t)0x00000010)
02305 #define  CAN_F12R1_FB5                       ((uint32_t)0x00000020)
02306 #define  CAN_F12R1_FB6                       ((uint32_t)0x00000040)
02307 #define  CAN_F12R1_FB7                       ((uint32_t)0x00000080)
02308 #define  CAN_F12R1_FB8                       ((uint32_t)0x00000100)
02309 #define  CAN_F12R1_FB9                       ((uint32_t)0x00000200)
02310 #define  CAN_F12R1_FB10                      ((uint32_t)0x00000400)
02311 #define  CAN_F12R1_FB11                      ((uint32_t)0x00000800)
02312 #define  CAN_F12R1_FB12                      ((uint32_t)0x00001000)
02313 #define  CAN_F12R1_FB13                      ((uint32_t)0x00002000)
02314 #define  CAN_F12R1_FB14                      ((uint32_t)0x00004000)
02315 #define  CAN_F12R1_FB15                      ((uint32_t)0x00008000)
02316 #define  CAN_F12R1_FB16                      ((uint32_t)0x00010000)
02317 #define  CAN_F12R1_FB17                      ((uint32_t)0x00020000)
02318 #define  CAN_F12R1_FB18                      ((uint32_t)0x00040000)
02319 #define  CAN_F12R1_FB19                      ((uint32_t)0x00080000)
02320 #define  CAN_F12R1_FB20                      ((uint32_t)0x00100000)
02321 #define  CAN_F12R1_FB21                      ((uint32_t)0x00200000)
02322 #define  CAN_F12R1_FB22                      ((uint32_t)0x00400000)
02323 #define  CAN_F12R1_FB23                      ((uint32_t)0x00800000)
02324 #define  CAN_F12R1_FB24                      ((uint32_t)0x01000000)
02325 #define  CAN_F12R1_FB25                      ((uint32_t)0x02000000)
02326 #define  CAN_F12R1_FB26                      ((uint32_t)0x04000000)
02327 #define  CAN_F12R1_FB27                      ((uint32_t)0x08000000)
02328 #define  CAN_F12R1_FB28                      ((uint32_t)0x10000000)
02329 #define  CAN_F12R1_FB29                      ((uint32_t)0x20000000)
02330 #define  CAN_F12R1_FB30                      ((uint32_t)0x40000000)
02331 #define  CAN_F12R1_FB31                      ((uint32_t)0x80000000)
02333 /******************  Bit definition for CAN_F13R1 register  ******************/
02334 #define  CAN_F13R1_FB0                       ((uint32_t)0x00000001)
02335 #define  CAN_F13R1_FB1                       ((uint32_t)0x00000002)
02336 #define  CAN_F13R1_FB2                       ((uint32_t)0x00000004)
02337 #define  CAN_F13R1_FB3                       ((uint32_t)0x00000008)
02338 #define  CAN_F13R1_FB4                       ((uint32_t)0x00000010)
02339 #define  CAN_F13R1_FB5                       ((uint32_t)0x00000020)
02340 #define  CAN_F13R1_FB6                       ((uint32_t)0x00000040)
02341 #define  CAN_F13R1_FB7                       ((uint32_t)0x00000080)
02342 #define  CAN_F13R1_FB8                       ((uint32_t)0x00000100)
02343 #define  CAN_F13R1_FB9                       ((uint32_t)0x00000200)
02344 #define  CAN_F13R1_FB10                      ((uint32_t)0x00000400)
02345 #define  CAN_F13R1_FB11                      ((uint32_t)0x00000800)
02346 #define  CAN_F13R1_FB12                      ((uint32_t)0x00001000)
02347 #define  CAN_F13R1_FB13                      ((uint32_t)0x00002000)
02348 #define  CAN_F13R1_FB14                      ((uint32_t)0x00004000)
02349 #define  CAN_F13R1_FB15                      ((uint32_t)0x00008000)
02350 #define  CAN_F13R1_FB16                      ((uint32_t)0x00010000)
02351 #define  CAN_F13R1_FB17                      ((uint32_t)0x00020000)
02352 #define  CAN_F13R1_FB18                      ((uint32_t)0x00040000)
02353 #define  CAN_F13R1_FB19                      ((uint32_t)0x00080000)
```

```
02354 #define  CAN_F13R1_FB20                      ((uint32_t)0x00100000)
02355 #define  CAN_F13R1_FB21                      ((uint32_t)0x00200000)
02356 #define  CAN_F13R1_FB22                      ((uint32_t)0x00400000)
02357 #define  CAN_F13R1_FB23                      ((uint32_t)0x00800000)
02358 #define  CAN_F13R1_FB24                      ((uint32_t)0x01000000)
02359 #define  CAN_F13R1_FB25                      ((uint32_t)0x02000000)
02360 #define  CAN_F13R1_FB26                      ((uint32_t)0x04000000)
02361 #define  CAN_F13R1_FB27                      ((uint32_t)0x08000000)
02362 #define  CAN_F13R1_FB28                      ((uint32_t)0x10000000)
02363 #define  CAN_F13R1_FB29                      ((uint32_t)0x20000000)
02364 #define  CAN_F13R1_FB30                      ((uint32_t)0x40000000)
02365 #define  CAN_F13R1_FB31                      ((uint32_t)0x80000000)
02367 /*******************  Bit definition for CAN_F0R2 register  *******************/
02368 #define  CAN_F0R2_FB0                        ((uint32_t)0x00000001)
02369 #define  CAN_F0R2_FB1                        ((uint32_t)0x00000002)
02370 #define  CAN_F0R2_FB2                        ((uint32_t)0x00000004)
02371 #define  CAN_F0R2_FB3                        ((uint32_t)0x00000008)
02372 #define  CAN_F0R2_FB4                        ((uint32_t)0x00000010)
02373 #define  CAN_F0R2_FB5                        ((uint32_t)0x00000020)
02374 #define  CAN_F0R2_FB6                        ((uint32_t)0x00000040)
02375 #define  CAN_F0R2_FB7                        ((uint32_t)0x00000080)
02376 #define  CAN_F0R2_FB8                        ((uint32_t)0x00000100)
02377 #define  CAN_F0R2_FB9                        ((uint32_t)0x00000200)
02378 #define  CAN_F0R2_FB10                       ((uint32_t)0x00000400)
02379 #define  CAN_F0R2_FB11                       ((uint32_t)0x00000800)
02380 #define  CAN_F0R2_FB12                       ((uint32_t)0x00001000)
02381 #define  CAN_F0R2_FB13                       ((uint32_t)0x00002000)
02382 #define  CAN_F0R2_FB14                       ((uint32_t)0x00004000)
02383 #define  CAN_F0R2_FB15                       ((uint32_t)0x00008000)
02384 #define  CAN_F0R2_FB16                       ((uint32_t)0x00010000)
02385 #define  CAN_F0R2_FB17                       ((uint32_t)0x00020000)
02386 #define  CAN_F0R2_FB18                       ((uint32_t)0x00040000)
02387 #define  CAN_F0R2_FB19                       ((uint32_t)0x00080000)
02388 #define  CAN_F0R2_FB20                       ((uint32_t)0x00100000)
02389 #define  CAN_F0R2_FB21                       ((uint32_t)0x00200000)
02390 #define  CAN_F0R2_FB22                       ((uint32_t)0x00400000)
02391 #define  CAN_F0R2_FB23                       ((uint32_t)0x00800000)
02392 #define  CAN_F0R2_FB24                       ((uint32_t)0x01000000)
02393 #define  CAN_F0R2_FB25                       ((uint32_t)0x02000000)
02394 #define  CAN_F0R2_FB26                       ((uint32_t)0x04000000)
02395 #define  CAN_F0R2_FB27                       ((uint32_t)0x08000000)
02396 #define  CAN_F0R2_FB28                       ((uint32_t)0x10000000)
02397 #define  CAN_F0R2_FB29                       ((uint32_t)0x20000000)
02398 #define  CAN_F0R2_FB30                       ((uint32_t)0x40000000)
02399 #define  CAN_F0R2_FB31                       ((uint32_t)0x80000000)
02401 /*******************  Bit definition for CAN_F1R2 register  *******************/
02402 #define  CAN_F1R2_FB0                        ((uint32_t)0x00000001)
02403 #define  CAN_F1R2_FB1                        ((uint32_t)0x00000002)
02404 #define  CAN_F1R2_FB2                        ((uint32_t)0x00000004)
02405 #define  CAN_F1R2_FB3                        ((uint32_t)0x00000008)
02406 #define  CAN_F1R2_FB4                        ((uint32_t)0x00000010)
02407 #define  CAN_F1R2_FB5                        ((uint32_t)0x00000020)
02408 #define  CAN_F1R2_FB6                        ((uint32_t)0x00000040)
02409 #define  CAN_F1R2_FB7                        ((uint32_t)0x00000080)
02410 #define  CAN_F1R2_FB8                        ((uint32_t)0x00000100)
02411 #define  CAN_F1R2_FB9                        ((uint32_t)0x00000200)
02412 #define  CAN_F1R2_FB10                       ((uint32_t)0x00000400)
02413 #define  CAN_F1R2_FB11                       ((uint32_t)0x00000800)
02414 #define  CAN_F1R2_FB12                       ((uint32_t)0x00001000)
02415 #define  CAN_F1R2_FB13                       ((uint32_t)0x00002000)
02416 #define  CAN_F1R2_FB14                       ((uint32_t)0x00004000)
02417 #define  CAN_F1R2_FB15                       ((uint32_t)0x00008000)
02418 #define  CAN_F1R2_FB16                       ((uint32_t)0x00010000)
02419 #define  CAN_F1R2_FB17                       ((uint32_t)0x00020000)
02420 #define  CAN_F1R2_FB18                       ((uint32_t)0x00040000)
02421 #define  CAN_F1R2_FB19                       ((uint32_t)0x00080000)
02422 #define  CAN_F1R2_FB20                       ((uint32_t)0x00100000)
02423 #define  CAN_F1R2_FB21                       ((uint32_t)0x00200000)
02424 #define  CAN_F1R2_FB22                       ((uint32_t)0x00400000)
02425 #define  CAN_F1R2_FB23                       ((uint32_t)0x00800000)
02426 #define  CAN_F1R2_FB24                       ((uint32_t)0x01000000)
02427 #define  CAN_F1R2_FB25                       ((uint32_t)0x02000000)
02428 #define  CAN_F1R2_FB26                       ((uint32_t)0x04000000)
02429 #define  CAN_F1R2_FB27                       ((uint32_t)0x08000000)
02430 #define  CAN_F1R2_FB28                       ((uint32_t)0x10000000)
02431 #define  CAN_F1R2_FB29                       ((uint32_t)0x20000000)
02432 #define  CAN_F1R2_FB30                       ((uint32_t)0x40000000)
02433 #define  CAN_F1R2_FB31                       ((uint32_t)0x80000000)
02435 /*******************  Bit definition for CAN_F2R2 register  *******************/
02436 #define  CAN_F2R2_FB0                        ((uint32_t)0x00000001)
02437 #define  CAN_F2R2_FB1                        ((uint32_t)0x00000002)
02438 #define  CAN_F2R2_FB2                        ((uint32_t)0x00000004)
02439 #define  CAN_F2R2_FB3                        ((uint32_t)0x00000008)
02440 #define  CAN_F2R2_FB4                        ((uint32_t)0x00000010)
02441 #define  CAN_F2R2_FB5                        ((uint32_t)0x00000020)
02442 #define  CAN_F2R2_FB6                        ((uint32_t)0x00000040)
02443 #define  CAN_F2R2_FB7                        ((uint32_t)0x00000080)
```

```
02444 #define  CAN_F2R2_FB8                        ((uint32_t)0x00000100)
02445 #define  CAN_F2R2_FB9                        ((uint32_t)0x00000200)
02446 #define  CAN_F2R2_FB10                       ((uint32_t)0x00000400)
02447 #define  CAN_F2R2_FB11                       ((uint32_t)0x00000800)
02448 #define  CAN_F2R2_FB12                       ((uint32_t)0x00001000)
02449 #define  CAN_F2R2_FB13                       ((uint32_t)0x00002000)
02450 #define  CAN_F2R2_FB14                       ((uint32_t)0x00004000)
02451 #define  CAN_F2R2_FB15                       ((uint32_t)0x00008000)
02452 #define  CAN_F2R2_FB16                       ((uint32_t)0x00010000)
02453 #define  CAN_F2R2_FB17                       ((uint32_t)0x00020000)
02454 #define  CAN_F2R2_FB18                       ((uint32_t)0x00040000)
02455 #define  CAN_F2R2_FB19                       ((uint32_t)0x00080000)
02456 #define  CAN_F2R2_FB20                       ((uint32_t)0x00100000)
02457 #define  CAN_F2R2_FB21                       ((uint32_t)0x00200000)
02458 #define  CAN_F2R2_FB22                       ((uint32_t)0x00400000)
02459 #define  CAN_F2R2_FB23                       ((uint32_t)0x00800000)
02460 #define  CAN_F2R2_FB24                       ((uint32_t)0x01000000)
02461 #define  CAN_F2R2_FB25                       ((uint32_t)0x02000000)
02462 #define  CAN_F2R2_FB26                       ((uint32_t)0x04000000)
02463 #define  CAN_F2R2_FB27                       ((uint32_t)0x08000000)
02464 #define  CAN_F2R2_FB28                       ((uint32_t)0x10000000)
02465 #define  CAN_F2R2_FB29                       ((uint32_t)0x20000000)
02466 #define  CAN_F2R2_FB30                       ((uint32_t)0x40000000)
02467 #define  CAN_F2R2_FB31                       ((uint32_t)0x80000000)
02469 /*******************  Bit definition for CAN_F3R2 register  *******************/
02470 #define  CAN_F3R2_FB0                        ((uint32_t)0x00000001)
02471 #define  CAN_F3R2_FB1                        ((uint32_t)0x00000002)
02472 #define  CAN_F3R2_FB2                        ((uint32_t)0x00000004)
02473 #define  CAN_F3R2_FB3                        ((uint32_t)0x00000008)
02474 #define  CAN_F3R2_FB4                        ((uint32_t)0x00000010)
02475 #define  CAN_F3R2_FB5                        ((uint32_t)0x00000020)
02476 #define  CAN_F3R2_FB6                        ((uint32_t)0x00000040)
02477 #define  CAN_F3R2_FB7                        ((uint32_t)0x00000080)
02478 #define  CAN_F3R2_FB8                        ((uint32_t)0x00000100)
02479 #define  CAN_F3R2_FB9                        ((uint32_t)0x00000200)
02480 #define  CAN_F3R2_FB10                       ((uint32_t)0x00000400)
02481 #define  CAN_F3R2_FB11                       ((uint32_t)0x00000800)
02482 #define  CAN_F3R2_FB12                       ((uint32_t)0x00001000)
02483 #define  CAN_F3R2_FB13                       ((uint32_t)0x00002000)
02484 #define  CAN_F3R2_FB14                       ((uint32_t)0x00004000)
02485 #define  CAN_F3R2_FB15                       ((uint32_t)0x00008000)
02486 #define  CAN_F3R2_FB16                       ((uint32_t)0x00010000)
02487 #define  CAN_F3R2_FB17                       ((uint32_t)0x00020000)
02488 #define  CAN_F3R2_FB18                       ((uint32_t)0x00040000)
02489 #define  CAN_F3R2_FB19                       ((uint32_t)0x00080000)
02490 #define  CAN_F3R2_FB20                       ((uint32_t)0x00100000)
02491 #define  CAN_F3R2_FB21                       ((uint32_t)0x00200000)
02492 #define  CAN_F3R2_FB22                       ((uint32_t)0x00400000)
02493 #define  CAN_F3R2_FB23                       ((uint32_t)0x00800000)
02494 #define  CAN_F3R2_FB24                       ((uint32_t)0x01000000)
02495 #define  CAN_F3R2_FB25                       ((uint32_t)0x02000000)
02496 #define  CAN_F3R2_FB26                       ((uint32_t)0x04000000)
02497 #define  CAN_F3R2_FB27                       ((uint32_t)0x08000000)
02498 #define  CAN_F3R2_FB28                       ((uint32_t)0x10000000)
02499 #define  CAN_F3R2_FB29                       ((uint32_t)0x20000000)
02500 #define  CAN_F3R2_FB30                       ((uint32_t)0x40000000)
02501 #define  CAN_F3R2_FB31                       ((uint32_t)0x80000000)
02503 /*******************  Bit definition for CAN_F4R2 register  *******************/
02504 #define  CAN_F4R2_FB0                        ((uint32_t)0x00000001)
02505 #define  CAN_F4R2_FB1                        ((uint32_t)0x00000002)
02506 #define  CAN_F4R2_FB2                        ((uint32_t)0x00000004)
02507 #define  CAN_F4R2_FB3                        ((uint32_t)0x00000008)
02508 #define  CAN_F4R2_FB4                        ((uint32_t)0x00000010)
02509 #define  CAN_F4R2_FB5                        ((uint32_t)0x00000020)
02510 #define  CAN_F4R2_FB6                        ((uint32_t)0x00000040)
02511 #define  CAN_F4R2_FB7                        ((uint32_t)0x00000080)
02512 #define  CAN_F4R2_FB8                        ((uint32_t)0x00000100)
02513 #define  CAN_F4R2_FB9                        ((uint32_t)0x00000200)
02514 #define  CAN_F4R2_FB10                       ((uint32_t)0x00000400)
02515 #define  CAN_F4R2_FB11                       ((uint32_t)0x00000800)
02516 #define  CAN_F4R2_FB12                       ((uint32_t)0x00001000)
02517 #define  CAN_F4R2_FB13                       ((uint32_t)0x00002000)
02518 #define  CAN_F4R2_FB14                       ((uint32_t)0x00004000)
02519 #define  CAN_F4R2_FB15                       ((uint32_t)0x00008000)
02520 #define  CAN_F4R2_FB16                       ((uint32_t)0x00010000)
02521 #define  CAN_F4R2_FB17                       ((uint32_t)0x00020000)
02522 #define  CAN_F4R2_FB18                       ((uint32_t)0x00040000)
02523 #define  CAN_F4R2_FB19                       ((uint32_t)0x00080000)
02524 #define  CAN_F4R2_FB20                       ((uint32_t)0x00100000)
02525 #define  CAN_F4R2_FB21                       ((uint32_t)0x00200000)
02526 #define  CAN_F4R2_FB22                       ((uint32_t)0x00400000)
02527 #define  CAN_F4R2_FB23                       ((uint32_t)0x00800000)
02528 #define  CAN_F4R2_FB24                       ((uint32_t)0x01000000)
02529 #define  CAN_F4R2_FB25                       ((uint32_t)0x02000000)
02530 #define  CAN_F4R2_FB26                       ((uint32_t)0x04000000)
02531 #define  CAN_F4R2_FB27                       ((uint32_t)0x08000000)
02532 #define  CAN_F4R2_FB28                       ((uint32_t)0x10000000)
```

```
02533 #define  CAN_F4R2_FB29                       ((uint32_t)0x20000000)
02534 #define  CAN_F4R2_FB30                       ((uint32_t)0x40000000)
02535 #define  CAN_F4R2_FB31                       ((uint32_t)0x80000000)
02537 /*******************  Bit definition for CAN_F5R2 register  *******************/
02538 #define  CAN_F5R2_FB0                        ((uint32_t)0x00000001)
02539 #define  CAN_F5R2_FB1                        ((uint32_t)0x00000002)
02540 #define  CAN_F5R2_FB2                        ((uint32_t)0x00000004)
02541 #define  CAN_F5R2_FB3                        ((uint32_t)0x00000008)
02542 #define  CAN_F5R2_FB4                        ((uint32_t)0x00000010)
02543 #define  CAN_F5R2_FB5                        ((uint32_t)0x00000020)
02544 #define  CAN_F5R2_FB6                        ((uint32_t)0x00000040)
02545 #define  CAN_F5R2_FB7                        ((uint32_t)0x00000080)
02546 #define  CAN_F5R2_FB8                        ((uint32_t)0x00000100)
02547 #define  CAN_F5R2_FB9                        ((uint32_t)0x00000200)
02548 #define  CAN_F5R2_FB10                       ((uint32_t)0x00000400)
02549 #define  CAN_F5R2_FB11                       ((uint32_t)0x00000800)
02550 #define  CAN_F5R2_FB12                       ((uint32_t)0x00001000)
02551 #define  CAN_F5R2_FB13                       ((uint32_t)0x00002000)
02552 #define  CAN_F5R2_FB14                       ((uint32_t)0x00004000)
02553 #define  CAN_F5R2_FB15                       ((uint32_t)0x00008000)
02554 #define  CAN_F5R2_FB16                       ((uint32_t)0x00010000)
02555 #define  CAN_F5R2_FB17                       ((uint32_t)0x00020000)
02556 #define  CAN_F5R2_FB18                       ((uint32_t)0x00040000)
02557 #define  CAN_F5R2_FB19                       ((uint32_t)0x00080000)
02558 #define  CAN_F5R2_FB20                       ((uint32_t)0x00100000)
02559 #define  CAN_F5R2_FB21                       ((uint32_t)0x00200000)
02560 #define  CAN_F5R2_FB22                       ((uint32_t)0x00400000)
02561 #define  CAN_F5R2_FB23                       ((uint32_t)0x00800000)
02562 #define  CAN_F5R2_FB24                       ((uint32_t)0x01000000)
02563 #define  CAN_F5R2_FB25                       ((uint32_t)0x02000000)
02564 #define  CAN_F5R2_FB26                       ((uint32_t)0x04000000)
02565 #define  CAN_F5R2_FB27                       ((uint32_t)0x08000000)
02566 #define  CAN_F5R2_FB28                       ((uint32_t)0x10000000)
02567 #define  CAN_F5R2_FB29                       ((uint32_t)0x20000000)
02568 #define  CAN_F5R2_FB30                       ((uint32_t)0x40000000)
02569 #define  CAN_F5R2_FB31                       ((uint32_t)0x80000000)
02571 /*******************  Bit definition for CAN_F6R2 register  *******************/
02572 #define  CAN_F6R2_FB0                        ((uint32_t)0x00000001)
02573 #define  CAN_F6R2_FB1                        ((uint32_t)0x00000002)
02574 #define  CAN_F6R2_FB2                        ((uint32_t)0x00000004)
02575 #define  CAN_F6R2_FB3                        ((uint32_t)0x00000008)
02576 #define  CAN_F6R2_FB4                        ((uint32_t)0x00000010)
02577 #define  CAN_F6R2_FB5                        ((uint32_t)0x00000020)
02578 #define  CAN_F6R2_FB6                        ((uint32_t)0x00000040)
02579 #define  CAN_F6R2_FB7                        ((uint32_t)0x00000080)
02580 #define  CAN_F6R2_FB8                        ((uint32_t)0x00000100)
02581 #define  CAN_F6R2_FB9                        ((uint32_t)0x00000200)
02582 #define  CAN_F6R2_FB10                       ((uint32_t)0x00000400)
02583 #define  CAN_F6R2_FB11                       ((uint32_t)0x00000800)
02584 #define  CAN_F6R2_FB12                       ((uint32_t)0x00001000)
02585 #define  CAN_F6R2_FB13                       ((uint32_t)0x00002000)
02586 #define  CAN_F6R2_FB14                       ((uint32_t)0x00004000)
02587 #define  CAN_F6R2_FB15                       ((uint32_t)0x00008000)
02588 #define  CAN_F6R2_FB16                       ((uint32_t)0x00010000)
02589 #define  CAN_F6R2_FB17                       ((uint32_t)0x00020000)
02590 #define  CAN_F6R2_FB18                       ((uint32_t)0x00040000)
02591 #define  CAN_F6R2_FB19                       ((uint32_t)0x00080000)
02592 #define  CAN_F6R2_FB20                       ((uint32_t)0x00100000)
02593 #define  CAN_F6R2_FB21                       ((uint32_t)0x00200000)
02594 #define  CAN_F6R2_FB22                       ((uint32_t)0x00400000)
02595 #define  CAN_F6R2_FB23                       ((uint32_t)0x00800000)
02596 #define  CAN_F6R2_FB24                       ((uint32_t)0x01000000)
02597 #define  CAN_F6R2_FB25                       ((uint32_t)0x02000000)
02598 #define  CAN_F6R2_FB26                       ((uint32_t)0x04000000)
02599 #define  CAN_F6R2_FB27                       ((uint32_t)0x08000000)
02600 #define  CAN_F6R2_FB28                       ((uint32_t)0x10000000)
02601 #define  CAN_F6R2_FB29                       ((uint32_t)0x20000000)
02602 #define  CAN_F6R2_FB30                       ((uint32_t)0x40000000)
02603 #define  CAN_F6R2_FB31                       ((uint32_t)0x80000000)
02605 /*******************  Bit definition for CAN_F7R2 register  *******************/
02606 #define  CAN_F7R2_FB0                        ((uint32_t)0x00000001)
02607 #define  CAN_F7R2_FB1                        ((uint32_t)0x00000002)
02608 #define  CAN_F7R2_FB2                        ((uint32_t)0x00000004)
02609 #define  CAN_F7R2_FB3                        ((uint32_t)0x00000008)
02610 #define  CAN_F7R2_FB4                        ((uint32_t)0x00000010)
02611 #define  CAN_F7R2_FB5                        ((uint32_t)0x00000020)
02612 #define  CAN_F7R2_FB6                        ((uint32_t)0x00000040)
02613 #define  CAN_F7R2_FB7                        ((uint32_t)0x00000080)
02614 #define  CAN_F7R2_FB8                        ((uint32_t)0x00000100)
02615 #define  CAN_F7R2_FB9                        ((uint32_t)0x00000200)
02616 #define  CAN_F7R2_FB10                       ((uint32_t)0x00000400)
02617 #define  CAN_F7R2_FB11                       ((uint32_t)0x00000800)
02618 #define  CAN_F7R2_FB12                       ((uint32_t)0x00001000)
02619 #define  CAN_F7R2_FB13                       ((uint32_t)0x00002000)
02620 #define  CAN_F7R2_FB14                       ((uint32_t)0x00004000)
02621 #define  CAN_F7R2_FB15                       ((uint32_t)0x00008000)
02622 #define  CAN_F7R2_FB16                       ((uint32_t)0x00010000)
```

```
02623 #define  CAN_F7R2_FB17                       ((uint32_t)0x00020000)
02624 #define  CAN_F7R2_FB18                       ((uint32_t)0x00040000)
02625 #define  CAN_F7R2_FB19                       ((uint32_t)0x00080000)
02626 #define  CAN_F7R2_FB20                       ((uint32_t)0x00100000)
02627 #define  CAN_F7R2_FB21                       ((uint32_t)0x00200000)
02628 #define  CAN_F7R2_FB22                       ((uint32_t)0x00400000)
02629 #define  CAN_F7R2_FB23                       ((uint32_t)0x00800000)
02630 #define  CAN_F7R2_FB24                       ((uint32_t)0x01000000)
02631 #define  CAN_F7R2_FB25                       ((uint32_t)0x02000000)
02632 #define  CAN_F7R2_FB26                       ((uint32_t)0x04000000)
02633 #define  CAN_F7R2_FB27                       ((uint32_t)0x08000000)
02634 #define  CAN_F7R2_FB28                       ((uint32_t)0x10000000)
02635 #define  CAN_F7R2_FB29                       ((uint32_t)0x20000000)
02636 #define  CAN_F7R2_FB30                       ((uint32_t)0x40000000)
02637 #define  CAN_F7R2_FB31                       ((uint32_t)0x80000000)
02639 /*******************  Bit definition for CAN_F8R2 register  *******************/
02640 #define  CAN_F8R2_FB0                        ((uint32_t)0x00000001)
02641 #define  CAN_F8R2_FB1                        ((uint32_t)0x00000002)
02642 #define  CAN_F8R2_FB2                        ((uint32_t)0x00000004)
02643 #define  CAN_F8R2_FB3                        ((uint32_t)0x00000008)
02644 #define  CAN_F8R2_FB4                        ((uint32_t)0x00000010)
02645 #define  CAN_F8R2_FB5                        ((uint32_t)0x00000020)
02646 #define  CAN_F8R2_FB6                        ((uint32_t)0x00000040)
02647 #define  CAN_F8R2_FB7                        ((uint32_t)0x00000080)
02648 #define  CAN_F8R2_FB8                        ((uint32_t)0x00000100)
02649 #define  CAN_F8R2_FB9                        ((uint32_t)0x00000200)
02650 #define  CAN_F8R2_FB10                       ((uint32_t)0x00000400)
02651 #define  CAN_F8R2_FB11                       ((uint32_t)0x00000800)
02652 #define  CAN_F8R2_FB12                       ((uint32_t)0x00001000)
02653 #define  CAN_F8R2_FB13                       ((uint32_t)0x00002000)
02654 #define  CAN_F8R2_FB14                       ((uint32_t)0x00004000)
02655 #define  CAN_F8R2_FB15                       ((uint32_t)0x00008000)
02656 #define  CAN_F8R2_FB16                       ((uint32_t)0x00010000)
02657 #define  CAN_F8R2_FB17                       ((uint32_t)0x00020000)
02658 #define  CAN_F8R2_FB18                       ((uint32_t)0x00040000)
02659 #define  CAN_F8R2_FB19                       ((uint32_t)0x00080000)
02660 #define  CAN_F8R2_FB20                       ((uint32_t)0x00100000)
02661 #define  CAN_F8R2_FB21                       ((uint32_t)0x00200000)
02662 #define  CAN_F8R2_FB22                       ((uint32_t)0x00400000)
02663 #define  CAN_F8R2_FB23                       ((uint32_t)0x00800000)
02664 #define  CAN_F8R2_FB24                       ((uint32_t)0x01000000)
02665 #define  CAN_F8R2_FB25                       ((uint32_t)0x02000000)
02666 #define  CAN_F8R2_FB26                       ((uint32_t)0x04000000)
02667 #define  CAN_F8R2_FB27                       ((uint32_t)0x08000000)
02668 #define  CAN_F8R2_FB28                       ((uint32_t)0x10000000)
02669 #define  CAN_F8R2_FB29                       ((uint32_t)0x20000000)
02670 #define  CAN_F8R2_FB30                       ((uint32_t)0x40000000)
02671 #define  CAN_F8R2_FB31                       ((uint32_t)0x80000000)
02673 /*******************  Bit definition for CAN_F9R2 register  *******************/
02674 #define  CAN_F9R2_FB0                        ((uint32_t)0x00000001)
02675 #define  CAN_F9R2_FB1                        ((uint32_t)0x00000002)
02676 #define  CAN_F9R2_FB2                        ((uint32_t)0x00000004)
02677 #define  CAN_F9R2_FB3                        ((uint32_t)0x00000008)
02678 #define  CAN_F9R2_FB4                        ((uint32_t)0x00000010)
02679 #define  CAN_F9R2_FB5                        ((uint32_t)0x00000020)
02680 #define  CAN_F9R2_FB6                        ((uint32_t)0x00000040)
02681 #define  CAN_F9R2_FB7                        ((uint32_t)0x00000080)
02682 #define  CAN_F9R2_FB8                        ((uint32_t)0x00000100)
02683 #define  CAN_F9R2_FB9                        ((uint32_t)0x00000200)
02684 #define  CAN_F9R2_FB10                       ((uint32_t)0x00000400)
02685 #define  CAN_F9R2_FB11                       ((uint32_t)0x00000800)
02686 #define  CAN_F9R2_FB12                       ((uint32_t)0x00001000)
02687 #define  CAN_F9R2_FB13                       ((uint32_t)0x00002000)
02688 #define  CAN_F9R2_FB14                       ((uint32_t)0x00004000)
02689 #define  CAN_F9R2_FB15                       ((uint32_t)0x00008000)
02690 #define  CAN_F9R2_FB16                       ((uint32_t)0x00010000)
02691 #define  CAN_F9R2_FB17                       ((uint32_t)0x00020000)
02692 #define  CAN_F9R2_FB18                       ((uint32_t)0x00040000)
02693 #define  CAN_F9R2_FB19                       ((uint32_t)0x00080000)
02694 #define  CAN_F9R2_FB20                       ((uint32_t)0x00100000)
02695 #define  CAN_F9R2_FB21                       ((uint32_t)0x00200000)
02696 #define  CAN_F9R2_FB22                       ((uint32_t)0x00400000)
02697 #define  CAN_F9R2_FB23                       ((uint32_t)0x00800000)
02698 #define  CAN_F9R2_FB24                       ((uint32_t)0x01000000)
02699 #define  CAN_F9R2_FB25                       ((uint32_t)0x02000000)
02700 #define  CAN_F9R2_FB26                       ((uint32_t)0x04000000)
02701 #define  CAN_F9R2_FB27                       ((uint32_t)0x08000000)
02702 #define  CAN_F9R2_FB28                       ((uint32_t)0x10000000)
02703 #define  CAN_F9R2_FB29                       ((uint32_t)0x20000000)
02704 #define  CAN_F9R2_FB30                       ((uint32_t)0x40000000)
02705 #define  CAN_F9R2_FB31                       ((uint32_t)0x80000000)
02707 /*******************  Bit definition for CAN_F10R2 register  *******************/
02708 #define  CAN_F10R2_FB0                       ((uint32_t)0x00000001)
02709 #define  CAN_F10R2_FB1                       ((uint32_t)0x00000002)
02710 #define  CAN_F10R2_FB2                       ((uint32_t)0x00000004)
02711 #define  CAN_F10R2_FB3                       ((uint32_t)0x00000008)
02712 #define  CAN_F10R2_FB4                       ((uint32_t)0x00000010)
```

```
02713 #define  CAN_F10R2_FB5                       ((uint32_t)0x00000020)
02714 #define  CAN_F10R2_FB6                       ((uint32_t)0x00000040)
02715 #define  CAN_F10R2_FB7                       ((uint32_t)0x00000080)
02716 #define  CAN_F10R2_FB8                       ((uint32_t)0x00000100)
02717 #define  CAN_F10R2_FB9                       ((uint32_t)0x00000200)
02718 #define  CAN_F10R2_FB10                      ((uint32_t)0x00000400)
02719 #define  CAN_F10R2_FB11                      ((uint32_t)0x00000800)
02720 #define  CAN_F10R2_FB12                      ((uint32_t)0x00001000)
02721 #define  CAN_F10R2_FB13                      ((uint32_t)0x00002000)
02722 #define  CAN_F10R2_FB14                      ((uint32_t)0x00004000)
02723 #define  CAN_F10R2_FB15                      ((uint32_t)0x00008000)
02724 #define  CAN_F10R2_FB16                      ((uint32_t)0x00010000)
02725 #define  CAN_F10R2_FB17                      ((uint32_t)0x00020000)
02726 #define  CAN_F10R2_FB18                      ((uint32_t)0x00040000)
02727 #define  CAN_F10R2_FB19                      ((uint32_t)0x00080000)
02728 #define  CAN_F10R2_FB20                      ((uint32_t)0x00100000)
02729 #define  CAN_F10R2_FB21                      ((uint32_t)0x00200000)
02730 #define  CAN_F10R2_FB22                      ((uint32_t)0x00400000)
02731 #define  CAN_F10R2_FB23                      ((uint32_t)0x00800000)
02732 #define  CAN_F10R2_FB24                      ((uint32_t)0x01000000)
02733 #define  CAN_F10R2_FB25                      ((uint32_t)0x02000000)
02734 #define  CAN_F10R2_FB26                      ((uint32_t)0x04000000)
02735 #define  CAN_F10R2_FB27                      ((uint32_t)0x08000000)
02736 #define  CAN_F10R2_FB28                      ((uint32_t)0x10000000)
02737 #define  CAN_F10R2_FB29                      ((uint32_t)0x20000000)
02738 #define  CAN_F10R2_FB30                      ((uint32_t)0x40000000)
02739 #define  CAN_F10R2_FB31                      ((uint32_t)0x80000000)
02741 /*******************  Bit definition for CAN_F11R2 register  ******************/
02742 #define  CAN_F11R2_FB0                       ((uint32_t)0x00000001)
02743 #define  CAN_F11R2_FB1                       ((uint32_t)0x00000002)
02744 #define  CAN_F11R2_FB2                       ((uint32_t)0x00000004)
02745 #define  CAN_F11R2_FB3                       ((uint32_t)0x00000008)
02746 #define  CAN_F11R2_FB4                       ((uint32_t)0x00000010)
02747 #define  CAN_F11R2_FB5                       ((uint32_t)0x00000020)
02748 #define  CAN_F11R2_FB6                       ((uint32_t)0x00000040)
02749 #define  CAN_F11R2_FB7                       ((uint32_t)0x00000080)
02750 #define  CAN_F11R2_FB8                       ((uint32_t)0x00000100)
02751 #define  CAN_F11R2_FB9                       ((uint32_t)0x00000200)
02752 #define  CAN_F11R2_FB10                      ((uint32_t)0x00000400)
02753 #define  CAN_F11R2_FB11                      ((uint32_t)0x00000800)
02754 #define  CAN_F11R2_FB12                      ((uint32_t)0x00001000)
02755 #define  CAN_F11R2_FB13                      ((uint32_t)0x00002000)
02756 #define  CAN_F11R2_FB14                      ((uint32_t)0x00004000)
02757 #define  CAN_F11R2_FB15                      ((uint32_t)0x00008000)
02758 #define  CAN_F11R2_FB16                      ((uint32_t)0x00010000)
02759 #define  CAN_F11R2_FB17                      ((uint32_t)0x00020000)
02760 #define  CAN_F11R2_FB18                      ((uint32_t)0x00040000)
02761 #define  CAN_F11R2_FB19                      ((uint32_t)0x00080000)
02762 #define  CAN_F11R2_FB20                      ((uint32_t)0x00100000)
02763 #define  CAN_F11R2_FB21                      ((uint32_t)0x00200000)
02764 #define  CAN_F11R2_FB22                      ((uint32_t)0x00400000)
02765 #define  CAN_F11R2_FB23                      ((uint32_t)0x00800000)
02766 #define  CAN_F11R2_FB24                      ((uint32_t)0x01000000)
02767 #define  CAN_F11R2_FB25                      ((uint32_t)0x02000000)
02768 #define  CAN_F11R2_FB26                      ((uint32_t)0x04000000)
02769 #define  CAN_F11R2_FB27                      ((uint32_t)0x08000000)
02770 #define  CAN_F11R2_FB28                      ((uint32_t)0x10000000)
02771 #define  CAN_F11R2_FB29                      ((uint32_t)0x20000000)
02772 #define  CAN_F11R2_FB30                      ((uint32_t)0x40000000)
02773 #define  CAN_F11R2_FB31                      ((uint32_t)0x80000000)
02775 /*******************  Bit definition for CAN_F12R2 register  ******************/
02776 #define  CAN_F12R2_FB0                       ((uint32_t)0x00000001)
02777 #define  CAN_F12R2_FB1                       ((uint32_t)0x00000002)
02778 #define  CAN_F12R2_FB2                       ((uint32_t)0x00000004)
02779 #define  CAN_F12R2_FB3                       ((uint32_t)0x00000008)
02780 #define  CAN_F12R2_FB4                       ((uint32_t)0x00000010)
02781 #define  CAN_F12R2_FB5                       ((uint32_t)0x00000020)
02782 #define  CAN_F12R2_FB6                       ((uint32_t)0x00000040)
02783 #define  CAN_F12R2_FB7                       ((uint32_t)0x00000080)
02784 #define  CAN_F12R2_FB8                       ((uint32_t)0x00000100)
02785 #define  CAN_F12R2_FB9                       ((uint32_t)0x00000200)
02786 #define  CAN_F12R2_FB10                      ((uint32_t)0x00000400)
02787 #define  CAN_F12R2_FB11                      ((uint32_t)0x00000800)
02788 #define  CAN_F12R2_FB12                      ((uint32_t)0x00001000)
02789 #define  CAN_F12R2_FB13                      ((uint32_t)0x00002000)
02790 #define  CAN_F12R2_FB14                      ((uint32_t)0x00004000)
02791 #define  CAN_F12R2_FB15                      ((uint32_t)0x00008000)
02792 #define  CAN_F12R2_FB16                      ((uint32_t)0x00010000)
02793 #define  CAN_F12R2_FB17                      ((uint32_t)0x00020000)
02794 #define  CAN_F12R2_FB18                      ((uint32_t)0x00040000)
02795 #define  CAN_F12R2_FB19                      ((uint32_t)0x00080000)
02796 #define  CAN_F12R2_FB20                      ((uint32_t)0x00100000)
02797 #define  CAN_F12R2_FB21                      ((uint32_t)0x00200000)
02798 #define  CAN_F12R2_FB22                      ((uint32_t)0x00400000)
02799 #define  CAN_F12R2_FB23                      ((uint32_t)0x00800000)
02800 #define  CAN_F12R2_FB24                      ((uint32_t)0x01000000)
02801 #define  CAN_F12R2_FB25                      ((uint32_t)0x02000000)
```

```
02802 #define  CAN_F12R2_FB26                      ((uint32_t)0x04000000)
02803 #define  CAN_F12R2_FB27                      ((uint32_t)0x08000000)
02804 #define  CAN_F12R2_FB28                      ((uint32_t)0x10000000)
02805 #define  CAN_F12R2_FB29                      ((uint32_t)0x20000000)
02806 #define  CAN_F12R2_FB30                      ((uint32_t)0x40000000)
02807 #define  CAN_F12R2_FB31                      ((uint32_t)0x80000000)
02809 /*******************  Bit definition for CAN_F13R2 register  ******************/
02810 #define  CAN_F13R2_FB0                       ((uint32_t)0x00000001)
02811 #define  CAN_F13R2_FB1                       ((uint32_t)0x00000002)
02812 #define  CAN_F13R2_FB2                       ((uint32_t)0x00000004)
02813 #define  CAN_F13R2_FB3                       ((uint32_t)0x00000008)
02814 #define  CAN_F13R2_FB4                       ((uint32_t)0x00000010)
02815 #define  CAN_F13R2_FB5                       ((uint32_t)0x00000020)
02816 #define  CAN_F13R2_FB6                       ((uint32_t)0x00000040)
02817 #define  CAN_F13R2_FB7                       ((uint32_t)0x00000080)
02818 #define  CAN_F13R2_FB8                       ((uint32_t)0x00000100)
02819 #define  CAN_F13R2_FB9                       ((uint32_t)0x00000200)
02820 #define  CAN_F13R2_FB10                      ((uint32_t)0x00000400)
02821 #define  CAN_F13R2_FB11                      ((uint32_t)0x00000800)
02822 #define  CAN_F13R2_FB12                      ((uint32_t)0x00001000)
02823 #define  CAN_F13R2_FB13                      ((uint32_t)0x00002000)
02824 #define  CAN_F13R2_FB14                      ((uint32_t)0x00004000)
02825 #define  CAN_F13R2_FB15                      ((uint32_t)0x00008000)
02826 #define  CAN_F13R2_FB16                      ((uint32_t)0x00010000)
02827 #define  CAN_F13R2_FB17                      ((uint32_t)0x00020000)
02828 #define  CAN_F13R2_FB18                      ((uint32_t)0x00040000)
02829 #define  CAN_F13R2_FB19                      ((uint32_t)0x00080000)
02830 #define  CAN_F13R2_FB20                      ((uint32_t)0x00100000)
02831 #define  CAN_F13R2_FB21                      ((uint32_t)0x00200000)
02832 #define  CAN_F13R2_FB22                      ((uint32_t)0x00400000)
02833 #define  CAN_F13R2_FB23                      ((uint32_t)0x00800000)
02834 #define  CAN_F13R2_FB24                      ((uint32_t)0x01000000)
02835 #define  CAN_F13R2_FB25                      ((uint32_t)0x02000000)
02836 #define  CAN_F13R2_FB26                      ((uint32_t)0x04000000)
02837 #define  CAN_F13R2_FB27                      ((uint32_t)0x08000000)
02838 #define  CAN_F13R2_FB28                      ((uint32_t)0x10000000)
02839 #define  CAN_F13R2_FB29                      ((uint32_t)0x20000000)
02840 #define  CAN_F13R2_FB30                      ((uint32_t)0x40000000)
02841 #define  CAN_F13R2_FB31                      ((uint32_t)0x80000000)
02843 /******************************************************************************/
02844 /*                                                                            */
02845 /*                          CRC calculation unit                              */
02846 /*                                                                            */
02847 /******************************************************************************/
02848 /*******************  Bit definition for CRC_DR register  *********************/
02849 #define  CRC_DR_DR                           ((uint32_t)0xFFFFFFFF)
02852 /*******************  Bit definition for CRC_IDR register  ********************/
02853 #define  CRC_IDR_IDR                         ((uint8_t)0xFF)
02856 /********************  Bit definition for CRC_CR register  ********************/
02857 #define  CRC_CR_RESET                        ((uint8_t)0x01)
02859 /******************************************************************************/
02860 /*                                                                            */
02861 /*                            Crypto Processor                                */
02862 /*                                                                            */
02863 /******************************************************************************/
02864 /******************** Bits definition for CRYP_CR register  ********************/
02865 #define CRYP_CR_ALGODIR                      ((uint32_t)0x00000004)
02866
02867 #define CRYP_CR_ALGOMODE                     ((uint32_t)0x00000038)
02868 #define CRYP_CR_ALGOMODE_0                   ((uint32_t)0x00000008)
02869 #define CRYP_CR_ALGOMODE_1                   ((uint32_t)0x00000010)
02870 #define CRYP_CR_ALGOMODE_2                   ((uint32_t)0x00000020)
02871 #define CRYP_CR_ALGOMODE_TDES_ECB            ((uint32_t)0x00000000)
02872 #define CRYP_CR_ALGOMODE_TDES_CBC            ((uint32_t)0x00000008)
02873 #define CRYP_CR_ALGOMODE_DES_ECB             ((uint32_t)0x00000010)
02874 #define CRYP_CR_ALGOMODE_DES_CBC             ((uint32_t)0x00000018)
02875 #define CRYP_CR_ALGOMODE_AES_ECB             ((uint32_t)0x00000020)
02876 #define CRYP_CR_ALGOMODE_AES_CBC             ((uint32_t)0x00000028)
02877 #define CRYP_CR_ALGOMODE_AES_CTR             ((uint32_t)0x00000030)
02878 #define CRYP_CR_ALGOMODE_AES_KEY             ((uint32_t)0x00000038)
02879
02880 #define CRYP_CR_DATATYPE                     ((uint32_t)0x000000C0)
02881 #define CRYP_CR_DATATYPE_0                   ((uint32_t)0x00000040)
02882 #define CRYP_CR_DATATYPE_1                   ((uint32_t)0x00000080)
02883 #define CRYP_CR_KEYSIZE                      ((uint32_t)0x00000300)
02884 #define CRYP_CR_KEYSIZE_0                    ((uint32_t)0x00000100)
02885 #define CRYP_CR_KEYSIZE_1                    ((uint32_t)0x00000200)
02886 #define CRYP_CR_FFLUSH                       ((uint32_t)0x00004000)
02887 #define CRYP_CR_CRYPEN                       ((uint32_t)0x00008000)
02888 /******************** Bits definition for CRYP_SR register  *******************/
02889 #define CRYP_SR_IFEM                         ((uint32_t)0x00000001)
02890 #define CRYP_SR_IFNF                         ((uint32_t)0x00000002)
02891 #define CRYP_SR_OFNE                         ((uint32_t)0x00000004)
02892 #define CRYP_SR_OFFU                         ((uint32_t)0x00000008)
02893 #define CRYP_SR_BUSY                         ((uint32_t)0x00000010)
02894 /******************** Bits definition for CRYP_DMACR register  ******************/
02895 #define CRYP_DMACR_DIEN                      ((uint32_t)0x00000001)
```

```
02896 #define CRYP_DMACR_DOEN                     ((uint32_t)0x00000002)
02897 /*****************  Bits definition for CRYP_IMSCR register  *****************/
02898 #define CRYP_IMSCR_INIM                     ((uint32_t)0x00000001)
02899 #define CRYP_IMSCR_OUTIM                    ((uint32_t)0x00000002)
02900 /******************* Bits definition for CRYP_RISR register  *******************/
02901 #define CRYP_RISR_OUTRIS                    ((uint32_t)0x00000001)
02902 #define CRYP_RISR_INRIS                     ((uint32_t)0x00000002)
02903 /******************* Bits definition for CRYP_MISR register  *******************/
02904 #define CRYP_MISR_INMIS                     ((uint32_t)0x00000001)
02905 #define CRYP_MISR_OUTMIS                    ((uint32_t)0x00000002)
02906
02907 /******************************************************************************/
02908 /*                                                                            */
02909 /*                    Digital to Analog Converter                             */
02910 /*                                                                            */
02911 /******************************************************************************/
02912 /********************  Bit definition for DAC_CR register  ********************/
02913 #define  DAC_CR_EN1                         ((uint32_t)0x00000001)
02914 #define  DAC_CR_BOFF1                       ((uint32_t)0x00000002)
02915 #define  DAC_CR_TEN1                        ((uint32_t)0x00000004)
02917 #define  DAC_CR_TSEL1                       ((uint32_t)0x00000038)
02918 #define  DAC_CR_TSEL1_0                     ((uint32_t)0x00000008)
02919 #define  DAC_CR_TSEL1_1                     ((uint32_t)0x00000010)
02920 #define  DAC_CR_TSEL1_2                     ((uint32_t)0x00000020)
02922 #define  DAC_CR_WAVE1                       ((uint32_t)0x000000C0)
02923 #define  DAC_CR_WAVE1_0                     ((uint32_t)0x00000040)
02924 #define  DAC_CR_WAVE1_1                     ((uint32_t)0x00000080)
02926 #define  DAC_CR_MAMP1                       ((uint32_t)0x00000F00)
02927 #define  DAC_CR_MAMP1_0                     ((uint32_t)0x00000100)
02928 #define  DAC_CR_MAMP1_1                     ((uint32_t)0x00000200)
02929 #define  DAC_CR_MAMP1_2                     ((uint32_t)0x00000400)
02930 #define  DAC_CR_MAMP1_3                     ((uint32_t)0x00000800)
02932 #define  DAC_CR_DMAEN1                      ((uint32_t)0x00001000)
02933 #define  DAC_CR_EN2                         ((uint32_t)0x00010000)
02934 #define  DAC_CR_BOFF2                       ((uint32_t)0x00020000)
02935 #define  DAC_CR_TEN2                        ((uint32_t)0x00040000)
02937 #define  DAC_CR_TSEL2                       ((uint32_t)0x00380000)
02938 #define  DAC_CR_TSEL2_0                     ((uint32_t)0x00080000)
02939 #define  DAC_CR_TSEL2_1                     ((uint32_t)0x00100000)
02940 #define  DAC_CR_TSEL2_2                     ((uint32_t)0x00200000)
02942 #define  DAC_CR_WAVE2                       ((uint32_t)0x00C00000)
02943 #define  DAC_CR_WAVE2_0                     ((uint32_t)0x00400000)
02944 #define  DAC_CR_WAVE2_1                     ((uint32_t)0x00800000)
02946 #define  DAC_CR_MAMP2                       ((uint32_t)0x0F000000)
02947 #define  DAC_CR_MAMP2_0                     ((uint32_t)0x01000000)
02948 #define  DAC_CR_MAMP2_1                     ((uint32_t)0x02000000)
02949 #define  DAC_CR_MAMP2_2                     ((uint32_t)0x04000000)
02950 #define  DAC_CR_MAMP2_3                     ((uint32_t)0x08000000)
02952 #define  DAC_CR_DMAEN2                      ((uint32_t)0x10000000)
02954 /*****************  Bit definition for DAC_SWTRIGR register  ******************/
02955 #define  DAC_SWTRIGR_SWTRIG1                ((uint8_t)0x01)
02956 #define  DAC_SWTRIGR_SWTRIG2                ((uint8_t)0x02)
02958 /*****************  Bit definition for DAC_DHR12R1 register  ******************/
02959 #define  DAC_DHR12R1_DACC1DHR               ((uint16_t)0x0FFF)
02961 /*****************  Bit definition for DAC_DHR12L1 register  ******************/
02962 #define  DAC_DHR12L1_DACC1DHR               ((uint16_t)0xFFF0)
02964 /******************  Bit definition for DAC_DHR8R1 register  ******************/
02965 #define  DAC_DHR8R1_DACC1DHR                ((uint8_t)0xFF)
02967 /*****************  Bit definition for DAC_DHR12R2 register  ******************/
02968 #define  DAC_DHR12R2_DACC2DHR               ((uint16_t)0x0FFF)
02970 /*****************  Bit definition for DAC_DHR12L2 register  ******************/
02971 #define  DAC_DHR12L2_DACC2DHR               ((uint16_t)0xFFF0)
02973 /******************  Bit definition for DAC_DHR8R2 register  ******************/
02974 #define  DAC_DHR8R2_DACC2DHR                ((uint8_t)0xFF)
02976 /*****************  Bit definition for DAC_DHR12RD register  ******************/
02977 #define  DAC_DHR12RD_DACC1DHR               ((uint32_t)0x00000FFF)
02978 #define  DAC_DHR12RD_DACC2DHR               ((uint32_t)0x0FFF0000)
02980 /*****************  Bit definition for DAC_DHR12LD register  ******************/
02981 #define  DAC_DHR12LD_DACC1DHR               ((uint32_t)0x0000FFF0)
02982 #define  DAC_DHR12LD_DACC2DHR               ((uint32_t)0xFFF00000)
02984 /******************  Bit definition for DAC_DHR8RD register  ******************/
02985 #define  DAC_DHR8RD_DACC1DHR                ((uint16_t)0x00FF)
02986 #define  DAC_DHR8RD_DACC2DHR                ((uint16_t)0xFF00)
02988 /*******************  Bit definition for DAC_DOR1 register  *******************/
02989 #define  DAC_DOR1_DACC1DOR                  ((uint16_t)0x0FFF)
02991 /*******************  Bit definition for DAC_DOR2 register  *******************/
02992 #define  DAC_DOR2_DACC2DOR                  ((uint16_t)0x0FFF)
02994 /********************  Bit definition for DAC_SR register  ********************/
02995 #define  DAC_SR_DMAUDR1                     ((uint32_t)0x00002000)
02996 #define  DAC_SR_DMAUDR2                     ((uint32_t)0x20000000)
02998 /******************************************************************************/
02999 /*                                                                            */
03000 /*                                 Debug MCU                                  */
03001 /*                                                                            */
03002 /******************************************************************************/
03003
03004 /******************************************************************************/
```

```
03005 /*                                                                          */
03006 /*                                  DCMI                                    */
03007 /*                                                                          */
03008 /****************************************************************************/
03009 /********************  Bits definition for DCMI_CR register  ****************/
03010 #define DCMI_CR_CAPTURE                      ((uint32_t)0x00000001)
03011 #define DCMI_CR_CM                           ((uint32_t)0x00000002)
03012 #define DCMI_CR_CROP                         ((uint32_t)0x00000004)
03013 #define DCMI_CR_JPEG                         ((uint32_t)0x00000008)
03014 #define DCMI_CR_ESS                          ((uint32_t)0x00000010)
03015 #define DCMI_CR_PCKPOL                       ((uint32_t)0x00000020)
03016 #define DCMI_CR_HSPOL                        ((uint32_t)0x00000040)
03017 #define DCMI_CR_VSPOL                        ((uint32_t)0x00000080)
03018 #define DCMI_CR_FCRC_0                       ((uint32_t)0x00000100)
03019 #define DCMI_CR_FCRC_1                       ((uint32_t)0x00000200)
03020 #define DCMI_CR_EDM_0                        ((uint32_t)0x00000400)
03021 #define DCMI_CR_EDM_1                        ((uint32_t)0x00000800)
03022 #define DCMI_CR_CRE                          ((uint32_t)0x00001000)
03023 #define DCMI_CR_ENABLE                       ((uint32_t)0x00004000)
03024
03025 /********************  Bits definition for DCMI_SR register  ****************/
03026 #define DCMI_SR_HSYNC                        ((uint32_t)0x00000001)
03027 #define DCMI_SR_VSYNC                        ((uint32_t)0x00000002)
03028 #define DCMI_SR_FNE                          ((uint32_t)0x00000004)
03029
03030 /********************  Bits definition for DCMI_RISR register  ***************/
03031 #define DCMI_RISR_FRAME_RIS                  ((uint32_t)0x00000001)
03032 #define DCMI_RISR_OVF_RIS                    ((uint32_t)0x00000002)
03033 #define DCMI_RISR_ERR_RIS                    ((uint32_t)0x00000004)
03034 #define DCMI_RISR_VSYNC_RIS                  ((uint32_t)0x00000008)
03035 #define DCMI_RISR_LINE_RIS                   ((uint32_t)0x00000010)
03036
03037 /********************  Bits definition for DCMI_IER register  ****************/
03038 #define DCMI_IER_FRAME_IE                    ((uint32_t)0x00000001)
03039 #define DCMI_IER_OVF_IE                      ((uint32_t)0x00000002)
03040 #define DCMI_IER_ERR_IE                      ((uint32_t)0x00000004)
03041 #define DCMI_IER_VSYNC_IE                    ((uint32_t)0x00000008)
03042 #define DCMI_IER_LINE_IE                     ((uint32_t)0x00000010)
03043
03044 /********************  Bits definition for DCMI_MISR register  ***************/
03045 #define DCMI_MISR_FRAME_MIS                  ((uint32_t)0x00000001)
03046 #define DCMI_MISR_OVF_MIS                    ((uint32_t)0x00000002)
03047 #define DCMI_MISR_ERR_MIS                    ((uint32_t)0x00000004)
03048 #define DCMI_MISR_VSYNC_MIS                  ((uint32_t)0x00000008)
03049 #define DCMI_MISR_LINE_MIS                   ((uint32_t)0x00000010)
03050
03051 /********************  Bits definition for DCMI_ICR register  ****************/
03052 #define DCMI_ICR_FRAME_ISC                   ((uint32_t)0x00000001)
03053 #define DCMI_ICR_OVF_ISC                     ((uint32_t)0x00000002)
03054 #define DCMI_ICR_ERR_ISC                     ((uint32_t)0x00000004)
03055 #define DCMI_ICR_VSYNC_ISC                   ((uint32_t)0x00000008)
03056 #define DCMI_ICR_LINE_ISC                    ((uint32_t)0x00000010)
03057
03058 /****************************************************************************/
03059 /*                                                                          */
03060 /*                             DMA Controller                               */
03061 /*                                                                          */
03062 /****************************************************************************/
03063 /********************  Bits definition for DMA_SxCR register  ****************/
03064 #define DMA_SxCR_CHSEL                       ((uint32_t)0x0E000000)
03065 #define DMA_SxCR_CHSEL_0                     ((uint32_t)0x02000000)
03066 #define DMA_SxCR_CHSEL_1                     ((uint32_t)0x04000000)
03067 #define DMA_SxCR_CHSEL_2                     ((uint32_t)0x08000000)
03068 #define DMA_SxCR_MBURST                      ((uint32_t)0x01800000)
03069 #define DMA_SxCR_MBURST_0                    ((uint32_t)0x00800000)
03070 #define DMA_SxCR_MBURST_1                    ((uint32_t)0x01000000)
03071 #define DMA_SxCR_PBURST                      ((uint32_t)0x00600000)
03072 #define DMA_SxCR_PBURST_0                    ((uint32_t)0x00200000)
03073 #define DMA_SxCR_PBURST_1                    ((uint32_t)0x00400000)
03074 #define DMA_SxCR_ACK                         ((uint32_t)0x00100000)
03075 #define DMA_SxCR_CT                          ((uint32_t)0x00080000)
03076 #define DMA_SxCR_DBM                         ((uint32_t)0x00040000)
03077 #define DMA_SxCR_PL                          ((uint32_t)0x00030000)
03078 #define DMA_SxCR_PL_0                        ((uint32_t)0x00010000)
03079 #define DMA_SxCR_PL_1                        ((uint32_t)0x00020000)
03080 #define DMA_SxCR_PINCOS                      ((uint32_t)0x00008000)
03081 #define DMA_SxCR_MSIZE                       ((uint32_t)0x00006000)
03082 #define DMA_SxCR_MSIZE_0                     ((uint32_t)0x00002000)
03083 #define DMA_SxCR_MSIZE_1                     ((uint32_t)0x00004000)
03084 #define DMA_SxCR_PSIZE                       ((uint32_t)0x00001800)
03085 #define DMA_SxCR_PSIZE_0                     ((uint32_t)0x00000800)
03086 #define DMA_SxCR_PSIZE_1                     ((uint32_t)0x00001000)
03087 #define DMA_SxCR_MINC                        ((uint32_t)0x00000400)
03088 #define DMA_SxCR_PINC                        ((uint32_t)0x00000200)
03089 #define DMA_SxCR_CIRC                        ((uint32_t)0x00000100)
03090 #define DMA_SxCR_DIR                         ((uint32_t)0x000000C0)
03091 #define DMA_SxCR_DIR_0                       ((uint32_t)0x00000040)
```

```
03092 #define DMA_SxCR_DIR_1                          ((uint32_t)0x00000080)
03093 #define DMA_SxCR_PFCTRL                         ((uint32_t)0x00000020)
03094 #define DMA_SxCR_TCIE                           ((uint32_t)0x00000010)
03095 #define DMA_SxCR_HTIE                           ((uint32_t)0x00000008)
03096 #define DMA_SxCR_TEIE                           ((uint32_t)0x00000004)
03097 #define DMA_SxCR_DMEIE                          ((uint32_t)0x00000002)
03098 #define DMA_SxCR_EN                             ((uint32_t)0x00000001)
03099
03100 /********************  Bits definition for DMA_SxCNDTR register  **************/
03101 #define DMA_SxNDT                               ((uint32_t)0x0000FFFF)
03102 #define DMA_SxNDT_0                             ((uint32_t)0x00000001)
03103 #define DMA_SxNDT_1                             ((uint32_t)0x00000002)
03104 #define DMA_SxNDT_2                             ((uint32_t)0x00000004)
03105 #define DMA_SxNDT_3                             ((uint32_t)0x00000008)
03106 #define DMA_SxNDT_4                             ((uint32_t)0x00000010)
03107 #define DMA_SxNDT_5                             ((uint32_t)0x00000020)
03108 #define DMA_SxNDT_6                             ((uint32_t)0x00000040)
03109 #define DMA_SxNDT_7                             ((uint32_t)0x00000080)
03110 #define DMA_SxNDT_8                             ((uint32_t)0x00000100)
03111 #define DMA_SxNDT_9                             ((uint32_t)0x00000200)
03112 #define DMA_SxNDT_10                            ((uint32_t)0x00000400)
03113 #define DMA_SxNDT_11                            ((uint32_t)0x00000800)
03114 #define DMA_SxNDT_12                            ((uint32_t)0x00001000)
03115 #define DMA_SxNDT_13                            ((uint32_t)0x00002000)
03116 #define DMA_SxNDT_14                            ((uint32_t)0x00004000)
03117 #define DMA_SxNDT_15                            ((uint32_t)0x00008000)
03118
03119 /********************  Bits definition for DMA_SxFCR register  ****************/
03120 #define DMA_SxFCR_FEIE                          ((uint32_t)0x00000080)
03121 #define DMA_SxFCR_FS                            ((uint32_t)0x00000038)
03122 #define DMA_SxFCR_FS_0                          ((uint32_t)0x00000008)
03123 #define DMA_SxFCR_FS_1                          ((uint32_t)0x00000010)
03124 #define DMA_SxFCR_FS_2                          ((uint32_t)0x00000020)
03125 #define DMA_SxFCR_DMDIS                         ((uint32_t)0x00000004)
03126 #define DMA_SxFCR_FTH                           ((uint32_t)0x00000003)
03127 #define DMA_SxFCR_FTH_0                         ((uint32_t)0x00000001)
03128 #define DMA_SxFCR_FTH_1                         ((uint32_t)0x00000002)
03129
03130 /********************  Bits definition for DMA_LISR register  *****************/
03131 #define DMA_LISR_TCIF3                          ((uint32_t)0x08000000)
03132 #define DMA_LISR_HTIF3                          ((uint32_t)0x04000000)
03133 #define DMA_LISR_TEIF3                          ((uint32_t)0x02000000)
03134 #define DMA_LISR_DMEIF3                         ((uint32_t)0x01000000)
03135 #define DMA_LISR_FEIF3                          ((uint32_t)0x00400000)
03136 #define DMA_LISR_TCIF2                          ((uint32_t)0x00200000)
03137 #define DMA_LISR_HTIF2                          ((uint32_t)0x00100000)
03138 #define DMA_LISR_TEIF2                          ((uint32_t)0x00080000)
03139 #define DMA_LISR_DMEIF2                         ((uint32_t)0x00040000)
03140 #define DMA_LISR_FEIF2                          ((uint32_t)0x00010000)
03141 #define DMA_LISR_TCIF1                          ((uint32_t)0x00000800)
03142 #define DMA_LISR_HTIF1                          ((uint32_t)0x00000400)
03143 #define DMA_LISR_TEIF1                          ((uint32_t)0x00000200)
03144 #define DMA_LISR_DMEIF1                         ((uint32_t)0x00000100)
03145 #define DMA_LISR_FEIF1                          ((uint32_t)0x00000040)
03146 #define DMA_LISR_TCIF0                          ((uint32_t)0x00000020)
03147 #define DMA_LISR_HTIF0                          ((uint32_t)0x00000010)
03148 #define DMA_LISR_TEIF0                          ((uint32_t)0x00000008)
03149 #define DMA_LISR_DMEIF0                         ((uint32_t)0x00000004)
03150 #define DMA_LISR_FEIF0                          ((uint32_t)0x00000001)
03151
03152 /********************  Bits definition for DMA_HISR register  *****************/
03153 #define DMA_HISR_TCIF7                          ((uint32_t)0x08000000)
03154 #define DMA_HISR_HTIF7                          ((uint32_t)0x04000000)
03155 #define DMA_HISR_TEIF7                          ((uint32_t)0x02000000)
03156 #define DMA_HISR_DMEIF7                         ((uint32_t)0x01000000)
03157 #define DMA_HISR_FEIF7                          ((uint32_t)0x00400000)
03158 #define DMA_HISR_TCIF6                          ((uint32_t)0x00200000)
03159 #define DMA_HISR_HTIF6                          ((uint32_t)0x00100000)
03160 #define DMA_HISR_TEIF6                          ((uint32_t)0x00080000)
03161 #define DMA_HISR_DMEIF6                         ((uint32_t)0x00040000)
03162 #define DMA_HISR_FEIF6                          ((uint32_t)0x00010000)
03163 #define DMA_HISR_TCIF5                          ((uint32_t)0x00000800)
03164 #define DMA_HISR_HTIF5                          ((uint32_t)0x00000400)
03165 #define DMA_HISR_TEIF5                          ((uint32_t)0x00000200)
03166 #define DMA_HISR_DMEIF5                         ((uint32_t)0x00000100)
03167 #define DMA_HISR_FEIF5                          ((uint32_t)0x00000040)
03168 #define DMA_HISR_TCIF4                          ((uint32_t)0x00000020)
03169 #define DMA_HISR_HTIF4                          ((uint32_t)0x00000010)
03170 #define DMA_HISR_TEIF4                          ((uint32_t)0x00000008)
03171 #define DMA_HISR_DMEIF4                         ((uint32_t)0x00000004)
03172 #define DMA_HISR_FEIF4                          ((uint32_t)0x00000001)
03173
03174 /********************  Bits definition for DMA_LIFCR register  ****************/
03175 #define DMA_LIFCR_CTCIF3                        ((uint32_t)0x08000000)
03176 #define DMA_LIFCR_CHTIF3                        ((uint32_t)0x04000000)
03177 #define DMA_LIFCR_CTEIF3                        ((uint32_t)0x02000000)
03178 #define DMA_LIFCR_CDMEIF3                       ((uint32_t)0x01000000)
```

```
03179 #define DMA_LIFCR_CFEIF3                    ((uint32_t)0x00400000)
03180 #define DMA_LIFCR_CTCIF2                    ((uint32_t)0x00200000)
03181 #define DMA_LIFCR_CHTIF2                    ((uint32_t)0x00100000)
03182 #define DMA_LIFCR_CTEIF2                    ((uint32_t)0x00080000)
03183 #define DMA_LIFCR_CDMEIF2                   ((uint32_t)0x00040000)
03184 #define DMA_LIFCR_CFEIF2                    ((uint32_t)0x00010000)
03185 #define DMA_LIFCR_CTCIF1                    ((uint32_t)0x00000800)
03186 #define DMA_LIFCR_CHTIF1                    ((uint32_t)0x00000400)
03187 #define DMA_LIFCR_CTEIF1                    ((uint32_t)0x00000200)
03188 #define DMA_LIFCR_CDMEIF1                   ((uint32_t)0x00000100)
03189 #define DMA_LIFCR_CFEIF1                    ((uint32_t)0x00000040)
03190 #define DMA_LIFCR_CTCIF0                    ((uint32_t)0x00000020)
03191 #define DMA_LIFCR_CHTIF0                    ((uint32_t)0x00000010)
03192 #define DMA_LIFCR_CTEIF0                    ((uint32_t)0x00000008)
03193 #define DMA_LIFCR_CDMEIF0                   ((uint32_t)0x00000004)
03194 #define DMA_LIFCR_CFEIF0                    ((uint32_t)0x00000001)
03195
03196 /*******************  Bits definition for DMA_HIFCR  register  ****************/
03197 #define DMA_HIFCR_CTCIF7                    ((uint32_t)0x08000000)
03198 #define DMA_HIFCR_CHTIF7                    ((uint32_t)0x04000000)
03199 #define DMA_HIFCR_CTEIF7                    ((uint32_t)0x02000000)
03200 #define DMA_HIFCR_CDMEIF7                   ((uint32_t)0x01000000)
03201 #define DMA_HIFCR_CFEIF7                    ((uint32_t)0x00400000)
03202 #define DMA_HIFCR_CTCIF6                    ((uint32_t)0x00200000)
03203 #define DMA_HIFCR_CHTIF6                    ((uint32_t)0x00100000)
03204 #define DMA_HIFCR_CTEIF6                    ((uint32_t)0x00080000)
03205 #define DMA_HIFCR_CDMEIF6                   ((uint32_t)0x00040000)
03206 #define DMA_HIFCR_CFEIF6                    ((uint32_t)0x00010000)
03207 #define DMA_HIFCR_CTCIF5                    ((uint32_t)0x00000800)
03208 #define DMA_HIFCR_CHTIF5                    ((uint32_t)0x00000400)
03209 #define DMA_HIFCR_CTEIF5                    ((uint32_t)0x00000200)
03210 #define DMA_HIFCR_CDMEIF5                   ((uint32_t)0x00000100)
03211 #define DMA_HIFCR_CFEIF5                    ((uint32_t)0x00000040)
03212 #define DMA_HIFCR_CTCIF4                    ((uint32_t)0x00000020)
03213 #define DMA_HIFCR_CHTIF4                    ((uint32_t)0x00000010)
03214 #define DMA_HIFCR_CTEIF4                    ((uint32_t)0x00000008)
03215 #define DMA_HIFCR_CDMEIF4                   ((uint32_t)0x00000004)
03216 #define DMA_HIFCR_CFEIF4                    ((uint32_t)0x00000001)
03217
03218 /******************************************************************************/
03219 /*                                                                            */
03220 /*                    External Interrupt/Event Controller                     */
03221 /*                                                                            */
03222 /******************************************************************************/
03223 /*******************  Bit definition for EXTI_IMR register  *******************/
03224 #define  EXTI_IMR_MR0                       ((uint32_t)0x00000001)
03225 #define  EXTI_IMR_MR1                       ((uint32_t)0x00000002)
03226 #define  EXTI_IMR_MR2                       ((uint32_t)0x00000004)
03227 #define  EXTI_IMR_MR3                       ((uint32_t)0x00000008)
03228 #define  EXTI_IMR_MR4                       ((uint32_t)0x00000010)
03229 #define  EXTI_IMR_MR5                       ((uint32_t)0x00000020)
03230 #define  EXTI_IMR_MR6                       ((uint32_t)0x00000040)
03231 #define  EXTI_IMR_MR7                       ((uint32_t)0x00000080)
03232 #define  EXTI_IMR_MR8                       ((uint32_t)0x00000100)
03233 #define  EXTI_IMR_MR9                       ((uint32_t)0x00000200)
03234 #define  EXTI_IMR_MR10                      ((uint32_t)0x00000400)
03235 #define  EXTI_IMR_MR11                      ((uint32_t)0x00000800)
03236 #define  EXTI_IMR_MR12                      ((uint32_t)0x00001000)
03237 #define  EXTI_IMR_MR13                      ((uint32_t)0x00002000)
03238 #define  EXTI_IMR_MR14                      ((uint32_t)0x00004000)
03239 #define  EXTI_IMR_MR15                      ((uint32_t)0x00008000)
03240 #define  EXTI_IMR_MR16                      ((uint32_t)0x00010000)
03241 #define  EXTI_IMR_MR17                      ((uint32_t)0x00020000)
03242 #define  EXTI_IMR_MR18                      ((uint32_t)0x00040000)
03243 #define  EXTI_IMR_MR19                      ((uint32_t)0x00080000)
03245 /*******************  Bit definition for EXTI_EMR register  *******************/
03246 #define  EXTI_EMR_MR0                       ((uint32_t)0x00000001)
03247 #define  EXTI_EMR_MR1                       ((uint32_t)0x00000002)
03248 #define  EXTI_EMR_MR2                       ((uint32_t)0x00000004)
03249 #define  EXTI_EMR_MR3                       ((uint32_t)0x00000008)
03250 #define  EXTI_EMR_MR4                       ((uint32_t)0x00000010)
03251 #define  EXTI_EMR_MR5                       ((uint32_t)0x00000020)
03252 #define  EXTI_EMR_MR6                       ((uint32_t)0x00000040)
03253 #define  EXTI_EMR_MR7                       ((uint32_t)0x00000080)
03254 #define  EXTI_EMR_MR8                       ((uint32_t)0x00000100)
03255 #define  EXTI_EMR_MR9                       ((uint32_t)0x00000200)
03256 #define  EXTI_EMR_MR10                      ((uint32_t)0x00000400)
03257 #define  EXTI_EMR_MR11                      ((uint32_t)0x00000800)
03258 #define  EXTI_EMR_MR12                      ((uint32_t)0x00001000)
03259 #define  EXTI_EMR_MR13                      ((uint32_t)0x00002000)
03260 #define  EXTI_EMR_MR14                      ((uint32_t)0x00004000)
03261 #define  EXTI_EMR_MR15                      ((uint32_t)0x00008000)
03262 #define  EXTI_EMR_MR16                      ((uint32_t)0x00010000)
03263 #define  EXTI_EMR_MR17                      ((uint32_t)0x00020000)
03264 #define  EXTI_EMR_MR18                      ((uint32_t)0x00040000)
03265 #define  EXTI_EMR_MR19                      ((uint32_t)0x00080000)
03267 /******************  Bit definition for EXTI_RTSR register  *******************/
```

```
03268 #define  EXTI_RTSR_TR0                       ((uint32_t)0x00000001)
03269 #define  EXTI_RTSR_TR1                       ((uint32_t)0x00000002)
03270 #define  EXTI_RTSR_TR2                       ((uint32_t)0x00000004)
03271 #define  EXTI_RTSR_TR3                       ((uint32_t)0x00000008)
03272 #define  EXTI_RTSR_TR4                       ((uint32_t)0x00000010)
03273 #define  EXTI_RTSR_TR5                       ((uint32_t)0x00000020)
03274 #define  EXTI_RTSR_TR6                       ((uint32_t)0x00000040)
03275 #define  EXTI_RTSR_TR7                       ((uint32_t)0x00000080)
03276 #define  EXTI_RTSR_TR8                       ((uint32_t)0x00000100)
03277 #define  EXTI_RTSR_TR9                       ((uint32_t)0x00000200)
03278 #define  EXTI_RTSR_TR10                      ((uint32_t)0x00000400)
03279 #define  EXTI_RTSR_TR11                      ((uint32_t)0x00000800)
03280 #define  EXTI_RTSR_TR12                      ((uint32_t)0x00001000)
03281 #define  EXTI_RTSR_TR13                      ((uint32_t)0x00002000)
03282 #define  EXTI_RTSR_TR14                      ((uint32_t)0x00004000)
03283 #define  EXTI_RTSR_TR15                      ((uint32_t)0x00008000)
03284 #define  EXTI_RTSR_TR16                      ((uint32_t)0x00010000)
03285 #define  EXTI_RTSR_TR17                      ((uint32_t)0x00020000)
03286 #define  EXTI_RTSR_TR18                      ((uint32_t)0x00040000)
03287 #define  EXTI_RTSR_TR19                      ((uint32_t)0x00080000)
03289 /******************  Bit definition for EXTI_FTSR register  ******************/
03290 #define  EXTI_FTSR_TR0                       ((uint32_t)0x00000001)
03291 #define  EXTI_FTSR_TR1                       ((uint32_t)0x00000002)
03292 #define  EXTI_FTSR_TR2                       ((uint32_t)0x00000004)
03293 #define  EXTI_FTSR_TR3                       ((uint32_t)0x00000008)
03294 #define  EXTI_FTSR_TR4                       ((uint32_t)0x00000010)
03295 #define  EXTI_FTSR_TR5                       ((uint32_t)0x00000020)
03296 #define  EXTI_FTSR_TR6                       ((uint32_t)0x00000040)
03297 #define  EXTI_FTSR_TR7                       ((uint32_t)0x00000080)
03298 #define  EXTI_FTSR_TR8                       ((uint32_t)0x00000100)
03299 #define  EXTI_FTSR_TR9                       ((uint32_t)0x00000200)
03300 #define  EXTI_FTSR_TR10                      ((uint32_t)0x00000400)
03301 #define  EXTI_FTSR_TR11                      ((uint32_t)0x00000800)
03302 #define  EXTI_FTSR_TR12                      ((uint32_t)0x00001000)
03303 #define  EXTI_FTSR_TR13                      ((uint32_t)0x00002000)
03304 #define  EXTI_FTSR_TR14                      ((uint32_t)0x00004000)
03305 #define  EXTI_FTSR_TR15                      ((uint32_t)0x00008000)
03306 #define  EXTI_FTSR_TR16                      ((uint32_t)0x00010000)
03307 #define  EXTI_FTSR_TR17                      ((uint32_t)0x00020000)
03308 #define  EXTI_FTSR_TR18                      ((uint32_t)0x00040000)
03309 #define  EXTI_FTSR_TR19                      ((uint32_t)0x00080000)
03311 /******************  Bit definition for EXTI_SWIER register  ******************/
03312 #define  EXTI_SWIER_SWIER0                   ((uint32_t)0x00000001)
03313 #define  EXTI_SWIER_SWIER1                   ((uint32_t)0x00000002)
03314 #define  EXTI_SWIER_SWIER2                   ((uint32_t)0x00000004)
03315 #define  EXTI_SWIER_SWIER3                   ((uint32_t)0x00000008)
03316 #define  EXTI_SWIER_SWIER4                   ((uint32_t)0x00000010)
03317 #define  EXTI_SWIER_SWIER5                   ((uint32_t)0x00000020)
03318 #define  EXTI_SWIER_SWIER6                   ((uint32_t)0x00000040)
03319 #define  EXTI_SWIER_SWIER7                   ((uint32_t)0x00000080)
03320 #define  EXTI_SWIER_SWIER8                   ((uint32_t)0x00000100)
03321 #define  EXTI_SWIER_SWIER9                   ((uint32_t)0x00000200)
03322 #define  EXTI_SWIER_SWIER10                  ((uint32_t)0x00000400)
03323 #define  EXTI_SWIER_SWIER11                  ((uint32_t)0x00000800)
03324 #define  EXTI_SWIER_SWIER12                  ((uint32_t)0x00001000)
03325 #define  EXTI_SWIER_SWIER13                  ((uint32_t)0x00002000)
03326 #define  EXTI_SWIER_SWIER14                  ((uint32_t)0x00004000)
03327 #define  EXTI_SWIER_SWIER15                  ((uint32_t)0x00008000)
03328 #define  EXTI_SWIER_SWIER16                  ((uint32_t)0x00010000)
03329 #define  EXTI_SWIER_SWIER17                  ((uint32_t)0x00020000)
03330 #define  EXTI_SWIER_SWIER18                  ((uint32_t)0x00040000)
03331 #define  EXTI_SWIER_SWIER19                  ((uint32_t)0x00080000)
03333 /******************  Bit definition for EXTI_PR register  ******************/
03334 #define  EXTI_PR_PR0                         ((uint32_t)0x00000001)
03335 #define  EXTI_PR_PR1                         ((uint32_t)0x00000002)
03336 #define  EXTI_PR_PR2                         ((uint32_t)0x00000004)
03337 #define  EXTI_PR_PR3                         ((uint32_t)0x00000008)
03338 #define  EXTI_PR_PR4                         ((uint32_t)0x00000010)
03339 #define  EXTI_PR_PR5                         ((uint32_t)0x00000020)
03340 #define  EXTI_PR_PR6                         ((uint32_t)0x00000040)
03341 #define  EXTI_PR_PR7                         ((uint32_t)0x00000080)
03342 #define  EXTI_PR_PR8                         ((uint32_t)0x00000100)
03343 #define  EXTI_PR_PR9                         ((uint32_t)0x00000200)
03344 #define  EXTI_PR_PR10                        ((uint32_t)0x00000400)
03345 #define  EXTI_PR_PR11                        ((uint32_t)0x00000800)
03346 #define  EXTI_PR_PR12                        ((uint32_t)0x00001000)
03347 #define  EXTI_PR_PR13                        ((uint32_t)0x00002000)
03348 #define  EXTI_PR_PR14                        ((uint32_t)0x00004000)
03349 #define  EXTI_PR_PR15                        ((uint32_t)0x00008000)
03350 #define  EXTI_PR_PR16                        ((uint32_t)0x00010000)
03351 #define  EXTI_PR_PR17                        ((uint32_t)0x00020000)
03352 #define  EXTI_PR_PR18                        ((uint32_t)0x00040000)
03353 #define  EXTI_PR_PR19                        ((uint32_t)0x00080000)
03355 /******************************************************************************/
03356 /*                                                                            */
03357 /*                                    FLASH                                   */
03358 /*                                                                            */
```

```
03359 /******************************************************************************/
03360 /*******************  Bits definition for FLASH_ACR register  *****************/
03361 #define FLASH_ACR_LATENCY                    ((uint32_t)0x00000007)
03362 #define FLASH_ACR_LATENCY_0WS                ((uint32_t)0x00000000)
03363 #define FLASH_ACR_LATENCY_1WS                ((uint32_t)0x00000001)
03364 #define FLASH_ACR_LATENCY_2WS                ((uint32_t)0x00000002)
03365 #define FLASH_ACR_LATENCY_3WS                ((uint32_t)0x00000003)
03366 #define FLASH_ACR_LATENCY_4WS                ((uint32_t)0x00000004)
03367 #define FLASH_ACR_LATENCY_5WS                ((uint32_t)0x00000005)
03368 #define FLASH_ACR_LATENCY_6WS                ((uint32_t)0x00000006)
03369 #define FLASH_ACR_LATENCY_7WS                ((uint32_t)0x00000007)
03370
03371 #define FLASH_ACR_PRFTEN                     ((uint32_t)0x00000100)
03372 #define FLASH_ACR_ICEN                       ((uint32_t)0x00000200)
03373 #define FLASH_ACR_DCEN                       ((uint32_t)0x00000400)
03374 #define FLASH_ACR_ICRST                      ((uint32_t)0x00000800)
03375 #define FLASH_ACR_DCRST                      ((uint32_t)0x00001000)
03376 #define FLASH_ACR_BYTE0_ADDRESS              ((uint32_t)0x40023C00)
03377 #define FLASH_ACR_BYTE2_ADDRESS              ((uint32_t)0x40023C03)
03378
03379 /*******************  Bits definition for FLASH_SR register  ******************/
03380 #define FLASH_SR_EOP                         ((uint32_t)0x00000001)
03381 #define FLASH_SR_SOP                         ((uint32_t)0x00000002)
03382 #define FLASH_SR_WRPERR                      ((uint32_t)0x00000010)
03383 #define FLASH_SR_PGAERR                      ((uint32_t)0x00000020)
03384 #define FLASH_SR_PGPERR                      ((uint32_t)0x00000040)
03385 #define FLASH_SR_PGSERR                      ((uint32_t)0x00000080)
03386 #define FLASH_SR_BSY                         ((uint32_t)0x00010000)
03387
03388 /*******************  Bits definition for FLASH_CR register  ******************/
03389 #define FLASH_CR_PG                          ((uint32_t)0x00000001)
03390 #define FLASH_CR_SER                         ((uint32_t)0x00000002)
03391 #define FLASH_CR_MER                         ((uint32_t)0x00000004)
03392 #define FLASH_CR_SNB_0                       ((uint32_t)0x00000008)
03393 #define FLASH_CR_SNB_1                       ((uint32_t)0x00000010)
03394 #define FLASH_CR_SNB_2                       ((uint32_t)0x00000020)
03395 #define FLASH_CR_SNB_3                       ((uint32_t)0x00000040)
03396 #define FLASH_CR_PSIZE_0                     ((uint32_t)0x00000100)
03397 #define FLASH_CR_PSIZE_1                     ((uint32_t)0x00000200)
03398 #define FLASH_CR_STRT                        ((uint32_t)0x00010000)
03399 #define FLASH_CR_EOPIE                       ((uint32_t)0x01000000)
03400 #define FLASH_CR_LOCK                        ((uint32_t)0x80000000)
03401
03402 /*******************  Bits definition for FLASH_OPTCR register  ***************/
03403 #define FLASH_OPTCR_OPTLOCK                  ((uint32_t)0x00000001)
03404 #define FLASH_OPTCR_OPTSTRT                  ((uint32_t)0x00000002)
03405 #define FLASH_OPTCR_BOR_LEV_0                ((uint32_t)0x00000004)
03406 #define FLASH_OPTCR_BOR_LEV_1                ((uint32_t)0x00000008)
03407 #define FLASH_OPTCR_BOR_LEV                  ((uint32_t)0x0000000C)
03408 #define FLASH_OPTCR_WDG_SW                   ((uint32_t)0x00000020)
03409 #define FLASH_OPTCR_nRST_STOP                ((uint32_t)0x00000040)
03410 #define FLASH_OPTCR_nRST_STDBY               ((uint32_t)0x00000080)
03411 #define FLASH_OPTCR_RDP_0                    ((uint32_t)0x00000100)
03412 #define FLASH_OPTCR_RDP_1                    ((uint32_t)0x00000200)
03413 #define FLASH_OPTCR_RDP_2                    ((uint32_t)0x00000400)
03414 #define FLASH_OPTCR_RDP_3                    ((uint32_t)0x00000800)
03415 #define FLASH_OPTCR_RDP_4                    ((uint32_t)0x00001000)
03416 #define FLASH_OPTCR_RDP_5                    ((uint32_t)0x00002000)
03417 #define FLASH_OPTCR_RDP_6                    ((uint32_t)0x00004000)
03418 #define FLASH_OPTCR_RDP_7                    ((uint32_t)0x00008000)
03419 #define FLASH_OPTCR_nWRP_0                   ((uint32_t)0x00010000)
03420 #define FLASH_OPTCR_nWRP_1                   ((uint32_t)0x00020000)
03421 #define FLASH_OPTCR_nWRP_2                   ((uint32_t)0x00040000)
03422 #define FLASH_OPTCR_nWRP_3                   ((uint32_t)0x00080000)
03423 #define FLASH_OPTCR_nWRP_4                   ((uint32_t)0x00100000)
03424 #define FLASH_OPTCR_nWRP_5                   ((uint32_t)0x00200000)
03425 #define FLASH_OPTCR_nWRP_6                   ((uint32_t)0x00400000)
03426 #define FLASH_OPTCR_nWRP_7                   ((uint32_t)0x00800000)
03427 #define FLASH_OPTCR_nWRP_8                   ((uint32_t)0x01000000)
03428 #define FLASH_OPTCR_nWRP_9                   ((uint32_t)0x02000000)
03429 #define FLASH_OPTCR_nWRP_10                  ((uint32_t)0x04000000)
03430 #define FLASH_OPTCR_nWRP_11                  ((uint32_t)0x08000000)
03431
03432 /******************************************************************************/
03433 /*                                                                            */
03434 /*                      Flexible Static Memory Controller                     */
03435 /*                                                                            */
03436 /******************************************************************************/
03437 /******************  Bit definition for FSMC_BCR1 register  *******************/
03438 #define  FSMC_BCR1_MBKEN                     ((uint32_t)0x00000001)
03439 #define  FSMC_BCR1_MUXEN                     ((uint32_t)0x00000002)
03441 #define  FSMC_BCR1_MTYP                      ((uint32_t)0x0000000C)
03442 #define  FSMC_BCR1_MTYP_0                    ((uint32_t)0x00000004)
03443 #define  FSMC_BCR1_MTYP_1                    ((uint32_t)0x00000008)
03445 #define  FSMC_BCR1_MWID                      ((uint32_t)0x00000030)
03446 #define  FSMC_BCR1_MWID_0                    ((uint32_t)0x00000010)
03447 #define  FSMC_BCR1_MWID_1                    ((uint32_t)0x00000020)
```

```
03449 #define  FSMC_BCR1_FACCEN                    ((uint32_t)0x00000040)
03450 #define  FSMC_BCR1_BURSTEN                   ((uint32_t)0x00000100)
03451 #define  FSMC_BCR1_WAITPOL                   ((uint32_t)0x00000200)
03452 #define  FSMC_BCR1_WRAPMOD                   ((uint32_t)0x00000400)
03453 #define  FSMC_BCR1_WAITCFG                   ((uint32_t)0x00000800)
03454 #define  FSMC_BCR1_WREN                      ((uint32_t)0x00001000)
03455 #define  FSMC_BCR1_WAITEN                    ((uint32_t)0x00002000)
03456 #define  FSMC_BCR1_EXTMOD                    ((uint32_t)0x00004000)
03457 #define  FSMC_BCR1_ASYNCWAIT                 ((uint32_t)0x00008000)
03458 #define  FSMC_BCR1_CBURSTRW                  ((uint32_t)0x00080000)
03460 /******************  Bit definition for FSMC_BCR2 register  *******************/
03461 #define  FSMC_BCR2_MBKEN                     ((uint32_t)0x00000001)
03462 #define  FSMC_BCR2_MUXEN                     ((uint32_t)0x00000002)
03464 #define  FSMC_BCR2_MTYP                      ((uint32_t)0x0000000C)
03465 #define  FSMC_BCR2_MTYP_0                    ((uint32_t)0x00000004)
03466 #define  FSMC_BCR2_MTYP_1                    ((uint32_t)0x00000008)
03468 #define  FSMC_BCR2_MWID                      ((uint32_t)0x00000030)
03469 #define  FSMC_BCR2_MWID_0                    ((uint32_t)0x00000010)
03470 #define  FSMC_BCR2_MWID_1                    ((uint32_t)0x00000020)
03472 #define  FSMC_BCR2_FACCEN                    ((uint32_t)0x00000040)
03473 #define  FSMC_BCR2_BURSTEN                   ((uint32_t)0x00000100)
03474 #define  FSMC_BCR2_WAITPOL                   ((uint32_t)0x00000200)
03475 #define  FSMC_BCR2_WRAPMOD                   ((uint32_t)0x00000400)
03476 #define  FSMC_BCR2_WAITCFG                   ((uint32_t)0x00000800)
03477 #define  FSMC_BCR2_WREN                      ((uint32_t)0x00001000)
03478 #define  FSMC_BCR2_WAITEN                    ((uint32_t)0x00002000)
03479 #define  FSMC_BCR2_EXTMOD                    ((uint32_t)0x00004000)
03480 #define  FSMC_BCR2_ASYNCWAIT                 ((uint32_t)0x00008000)
03481 #define  FSMC_BCR2_CBURSTRW                  ((uint32_t)0x00080000)
03483 /******************  Bit definition for FSMC_BCR3 register  *******************/
03484 #define  FSMC_BCR3_MBKEN                     ((uint32_t)0x00000001)
03485 #define  FSMC_BCR3_MUXEN                     ((uint32_t)0x00000002)
03487 #define  FSMC_BCR3_MTYP                      ((uint32_t)0x0000000C)
03488 #define  FSMC_BCR3_MTYP_0                    ((uint32_t)0x00000004)
03489 #define  FSMC_BCR3_MTYP_1                    ((uint32_t)0x00000008)
03491 #define  FSMC_BCR3_MWID                      ((uint32_t)0x00000030)
03492 #define  FSMC_BCR3_MWID_0                    ((uint32_t)0x00000010)
03493 #define  FSMC_BCR3_MWID_1                    ((uint32_t)0x00000020)
03495 #define  FSMC_BCR3_FACCEN                    ((uint32_t)0x00000040)
03496 #define  FSMC_BCR3_BURSTEN                   ((uint32_t)0x00000100)
03497 #define  FSMC_BCR3_WAITPOL                   ((uint32_t)0x00000200)
03498 #define  FSMC_BCR3_WRAPMOD                   ((uint32_t)0x00000400)
03499 #define  FSMC_BCR3_WAITCFG                   ((uint32_t)0x00000800)
03500 #define  FSMC_BCR3_WREN                      ((uint32_t)0x00001000)
03501 #define  FSMC_BCR3_WAITEN                    ((uint32_t)0x00002000)
03502 #define  FSMC_BCR3_EXTMOD                    ((uint32_t)0x00004000)
03503 #define  FSMC_BCR3_ASYNCWAIT                 ((uint32_t)0x00008000)
03504 #define  FSMC_BCR3_CBURSTRW                  ((uint32_t)0x00080000)
03506 /******************  Bit definition for FSMC_BCR4 register  *******************/
03507 #define  FSMC_BCR4_MBKEN                     ((uint32_t)0x00000001)
03508 #define  FSMC_BCR4_MUXEN                     ((uint32_t)0x00000002)
03510 #define  FSMC_BCR4_MTYP                      ((uint32_t)0x0000000C)
03511 #define  FSMC_BCR4_MTYP_0                    ((uint32_t)0x00000004)
03512 #define  FSMC_BCR4_MTYP_1                    ((uint32_t)0x00000008)
03514 #define  FSMC_BCR4_MWID                      ((uint32_t)0x00000030)
03515 #define  FSMC_BCR4_MWID_0                    ((uint32_t)0x00000010)
03516 #define  FSMC_BCR4_MWID_1                    ((uint32_t)0x00000020)
03518 #define  FSMC_BCR4_FACCEN                    ((uint32_t)0x00000040)
03519 #define  FSMC_BCR4_BURSTEN                   ((uint32_t)0x00000100)
03520 #define  FSMC_BCR4_WAITPOL                   ((uint32_t)0x00000200)
03521 #define  FSMC_BCR4_WRAPMOD                   ((uint32_t)0x00000400)
03522 #define  FSMC_BCR4_WAITCFG                   ((uint32_t)0x00000800)
03523 #define  FSMC_BCR4_WREN                      ((uint32_t)0x00001000)
03524 #define  FSMC_BCR4_WAITEN                    ((uint32_t)0x00002000)
03525 #define  FSMC_BCR4_EXTMOD                    ((uint32_t)0x00004000)
03526 #define  FSMC_BCR4_ASYNCWAIT                 ((uint32_t)0x00008000)
03527 #define  FSMC_BCR4_CBURSTRW                  ((uint32_t)0x00080000)
03529 /******************  Bit definition for FSMC_BTR1 register  *******************/
03530 #define  FSMC_BTR1_ADDSET                    ((uint32_t)0x0000000F)
03531 #define  FSMC_BTR1_ADDSET_0                  ((uint32_t)0x00000001)
03532 #define  FSMC_BTR1_ADDSET_1                  ((uint32_t)0x00000002)
03533 #define  FSMC_BTR1_ADDSET_2                  ((uint32_t)0x00000004)
03534 #define  FSMC_BTR1_ADDSET_3                  ((uint32_t)0x00000008)
03536 #define  FSMC_BTR1_ADDHLD                    ((uint32_t)0x000000F0)
03537 #define  FSMC_BTR1_ADDHLD_0                  ((uint32_t)0x00000010)
03538 #define  FSMC_BTR1_ADDHLD_1                  ((uint32_t)0x00000020)
03539 #define  FSMC_BTR1_ADDHLD_2                  ((uint32_t)0x00000040)
03540 #define  FSMC_BTR1_ADDHLD_3                  ((uint32_t)0x00000080)
03542 #define  FSMC_BTR1_DATAST                    ((uint32_t)0x0000FF00)
03543 #define  FSMC_BTR1_DATAST_0                  ((uint32_t)0x00000100)
03544 #define  FSMC_BTR1_DATAST_1                  ((uint32_t)0x00000200)
03545 #define  FSMC_BTR1_DATAST_2                  ((uint32_t)0x00000400)
03546 #define  FSMC_BTR1_DATAST_3                  ((uint32_t)0x00000800)
03548 #define  FSMC_BTR1_BUSTURN                   ((uint32_t)0x000F0000)
03549 #define  FSMC_BTR1_BUSTURN_0                 ((uint32_t)0x00010000)
03550 #define  FSMC_BTR1_BUSTURN_1                 ((uint32_t)0x00020000)
03551 #define  FSMC_BTR1_BUSTURN_2                 ((uint32_t)0x00040000)
```

```
03552 #define  FSMC_BTR1_BUSTURN_3                ((uint32_t)0x00080000)
03554 #define  FSMC_BTR1_CLKDIV                   ((uint32_t)0x00F00000)
03555 #define  FSMC_BTR1_CLKDIV_0                 ((uint32_t)0x00100000)
03556 #define  FSMC_BTR1_CLKDIV_1                 ((uint32_t)0x00200000)
03557 #define  FSMC_BTR1_CLKDIV_2                 ((uint32_t)0x00400000)
03558 #define  FSMC_BTR1_CLKDIV_3                 ((uint32_t)0x00800000)
03560 #define  FSMC_BTR1_DATLAT                   ((uint32_t)0x0F000000)
03561 #define  FSMC_BTR1_DATLAT_0                 ((uint32_t)0x01000000)
03562 #define  FSMC_BTR1_DATLAT_1                 ((uint32_t)0x02000000)
03563 #define  FSMC_BTR1_DATLAT_2                 ((uint32_t)0x04000000)
03564 #define  FSMC_BTR1_DATLAT_3                 ((uint32_t)0x08000000)
03566 #define  FSMC_BTR1_ACCMOD                   ((uint32_t)0x30000000)
03567 #define  FSMC_BTR1_ACCMOD_0                 ((uint32_t)0x10000000)
03568 #define  FSMC_BTR1_ACCMOD_1                 ((uint32_t)0x20000000)
03570 /******************  Bit definition for FSMC_BTR2 register  *******************/
03571 #define  FSMC_BTR2_ADDSET                   ((uint32_t)0x0000000F)
03572 #define  FSMC_BTR2_ADDSET_0                 ((uint32_t)0x00000001)
03573 #define  FSMC_BTR2_ADDSET_1                 ((uint32_t)0x00000002)
03574 #define  FSMC_BTR2_ADDSET_2                 ((uint32_t)0x00000004)
03575 #define  FSMC_BTR2_ADDSET_3                 ((uint32_t)0x00000008)
03577 #define  FSMC_BTR2_ADDHLD                   ((uint32_t)0x000000F0)
03578 #define  FSMC_BTR2_ADDHLD_0                 ((uint32_t)0x00000010)
03579 #define  FSMC_BTR2_ADDHLD_1                 ((uint32_t)0x00000020)
03580 #define  FSMC_BTR2_ADDHLD_2                 ((uint32_t)0x00000040)
03581 #define  FSMC_BTR2_ADDHLD_3                 ((uint32_t)0x00000080)
03583 #define  FSMC_BTR2_DATAST                   ((uint32_t)0x0000FF00)
03584 #define  FSMC_BTR2_DATAST_0                 ((uint32_t)0x00000100)
03585 #define  FSMC_BTR2_DATAST_1                 ((uint32_t)0x00000200)
03586 #define  FSMC_BTR2_DATAST_2                 ((uint32_t)0x00000400)
03587 #define  FSMC_BTR2_DATAST_3                 ((uint32_t)0x00000800)
03589 #define  FSMC_BTR2_BUSTURN                  ((uint32_t)0x000F0000)
03590 #define  FSMC_BTR2_BUSTURN_0                ((uint32_t)0x00010000)
03591 #define  FSMC_BTR2_BUSTURN_1                ((uint32_t)0x00020000)
03592 #define  FSMC_BTR2_BUSTURN_2                ((uint32_t)0x00040000)
03593 #define  FSMC_BTR2_BUSTURN_3                ((uint32_t)0x00080000)
03595 #define  FSMC_BTR2_CLKDIV                   ((uint32_t)0x00F00000)
03596 #define  FSMC_BTR2_CLKDIV_0                 ((uint32_t)0x00100000)
03597 #define  FSMC_BTR2_CLKDIV_1                 ((uint32_t)0x00200000)
03598 #define  FSMC_BTR2_CLKDIV_2                 ((uint32_t)0x00400000)
03599 #define  FSMC_BTR2_CLKDIV_3                 ((uint32_t)0x00800000)
03601 #define  FSMC_BTR2_DATLAT                   ((uint32_t)0x0F000000)
03602 #define  FSMC_BTR2_DATLAT_0                 ((uint32_t)0x01000000)
03603 #define  FSMC_BTR2_DATLAT_1                 ((uint32_t)0x02000000)
03604 #define  FSMC_BTR2_DATLAT_2                 ((uint32_t)0x04000000)
03605 #define  FSMC_BTR2_DATLAT_3                 ((uint32_t)0x08000000)
03607 #define  FSMC_BTR2_ACCMOD                   ((uint32_t)0x30000000)
03608 #define  FSMC_BTR2_ACCMOD_0                 ((uint32_t)0x10000000)
03609 #define  FSMC_BTR2_ACCMOD_1                 ((uint32_t)0x20000000)
03611 /******************  Bit definition for FSMC_BTR3 register  *******************/
03612 #define  FSMC_BTR3_ADDSET                   ((uint32_t)0x0000000F)
03613 #define  FSMC_BTR3_ADDSET_0                 ((uint32_t)0x00000001)
03614 #define  FSMC_BTR3_ADDSET_1                 ((uint32_t)0x00000002)
03615 #define  FSMC_BTR3_ADDSET_2                 ((uint32_t)0x00000004)
03616 #define  FSMC_BTR3_ADDSET_3                 ((uint32_t)0x00000008)
03618 #define  FSMC_BTR3_ADDHLD                   ((uint32_t)0x000000F0)
03619 #define  FSMC_BTR3_ADDHLD_0                 ((uint32_t)0x00000010)
03620 #define  FSMC_BTR3_ADDHLD_1                 ((uint32_t)0x00000020)
03621 #define  FSMC_BTR3_ADDHLD_2                 ((uint32_t)0x00000040)
03622 #define  FSMC_BTR3_ADDHLD_3                 ((uint32_t)0x00000080)
03624 #define  FSMC_BTR3_DATAST                   ((uint32_t)0x0000FF00)
03625 #define  FSMC_BTR3_DATAST_0                 ((uint32_t)0x00000100)
03626 #define  FSMC_BTR3_DATAST_1                 ((uint32_t)0x00000200)
03627 #define  FSMC_BTR3_DATAST_2                 ((uint32_t)0x00000400)
03628 #define  FSMC_BTR3_DATAST_3                 ((uint32_t)0x00000800)
03630 #define  FSMC_BTR3_BUSTURN                  ((uint32_t)0x000F0000)
03631 #define  FSMC_BTR3_BUSTURN_0                ((uint32_t)0x00010000)
03632 #define  FSMC_BTR3_BUSTURN_1                ((uint32_t)0x00020000)
03633 #define  FSMC_BTR3_BUSTURN_2                ((uint32_t)0x00040000)
03634 #define  FSMC_BTR3_BUSTURN_3                ((uint32_t)0x00080000)
03636 #define  FSMC_BTR3_CLKDIV                   ((uint32_t)0x00F00000)
03637 #define  FSMC_BTR3_CLKDIV_0                 ((uint32_t)0x00100000)
03638 #define  FSMC_BTR3_CLKDIV_1                 ((uint32_t)0x00200000)
03639 #define  FSMC_BTR3_CLKDIV_2                 ((uint32_t)0x00400000)
03640 #define  FSMC_BTR3_CLKDIV_3                 ((uint32_t)0x00800000)
03642 #define  FSMC_BTR3_DATLAT                   ((uint32_t)0x0F000000)
03643 #define  FSMC_BTR3_DATLAT_0                 ((uint32_t)0x01000000)
03644 #define  FSMC_BTR3_DATLAT_1                 ((uint32_t)0x02000000)
03645 #define  FSMC_BTR3_DATLAT_2                 ((uint32_t)0x04000000)
03646 #define  FSMC_BTR3_DATLAT_3                 ((uint32_t)0x08000000)
03648 #define  FSMC_BTR3_ACCMOD                   ((uint32_t)0x30000000)
03649 #define  FSMC_BTR3_ACCMOD_0                 ((uint32_t)0x10000000)
03650 #define  FSMC_BTR3_ACCMOD_1                 ((uint32_t)0x20000000)
03652 /******************  Bit definition for FSMC_BTR4 register  *******************/
03653 #define  FSMC_BTR4_ADDSET                   ((uint32_t)0x0000000F)
03654 #define  FSMC_BTR4_ADDSET_0                 ((uint32_t)0x00000001)
03655 #define  FSMC_BTR4_ADDSET_1                 ((uint32_t)0x00000002)
03656 #define  FSMC_BTR4_ADDSET_2                 ((uint32_t)0x00000004)
```

```
03657 #define  FSMC_BTR4_ADDSET_3                  ((uint32_t)0x00000008)
03659 #define  FSMC_BTR4_ADDHLD                    ((uint32_t)0x000000F0)
03660 #define  FSMC_BTR4_ADDHLD_0                  ((uint32_t)0x00000010)
03661 #define  FSMC_BTR4_ADDHLD_1                  ((uint32_t)0x00000020)
03662 #define  FSMC_BTR4_ADDHLD_2                  ((uint32_t)0x00000040)
03663 #define  FSMC_BTR4_ADDHLD_3                  ((uint32_t)0x00000080)
03665 #define  FSMC_BTR4_DATAST                    ((uint32_t)0x0000FF00)
03666 #define  FSMC_BTR4_DATAST_0                  ((uint32_t)0x00000100)
03667 #define  FSMC_BTR4_DATAST_1                  ((uint32_t)0x00000200)
03668 #define  FSMC_BTR4_DATAST_2                  ((uint32_t)0x00000400)
03669 #define  FSMC_BTR4_DATAST_3                  ((uint32_t)0x00000800)
03671 #define  FSMC_BTR4_BUSTURN                   ((uint32_t)0x000F0000)
03672 #define  FSMC_BTR4_BUSTURN_0                 ((uint32_t)0x00010000)
03673 #define  FSMC_BTR4_BUSTURN_1                 ((uint32_t)0x00020000)
03674 #define  FSMC_BTR4_BUSTURN_2                 ((uint32_t)0x00040000)
03675 #define  FSMC_BTR4_BUSTURN_3                 ((uint32_t)0x00080000)
03677 #define  FSMC_BTR4_CLKDIV                    ((uint32_t)0x00F00000)
03678 #define  FSMC_BTR4_CLKDIV_0                  ((uint32_t)0x00100000)
03679 #define  FSMC_BTR4_CLKDIV_1                  ((uint32_t)0x00200000)
03680 #define  FSMC_BTR4_CLKDIV_2                  ((uint32_t)0x00400000)
03681 #define  FSMC_BTR4_CLKDIV_3                  ((uint32_t)0x00800000)
03683 #define  FSMC_BTR4_DATLAT                    ((uint32_t)0x0F000000)
03684 #define  FSMC_BTR4_DATLAT_0                  ((uint32_t)0x01000000)
03685 #define  FSMC_BTR4_DATLAT_1                  ((uint32_t)0x02000000)
03686 #define  FSMC_BTR4_DATLAT_2                  ((uint32_t)0x04000000)
03687 #define  FSMC_BTR4_DATLAT_3                  ((uint32_t)0x08000000)
03689 #define  FSMC_BTR4_ACCMOD                    ((uint32_t)0x30000000)
03690 #define  FSMC_BTR4_ACCMOD_0                  ((uint32_t)0x10000000)
03691 #define  FSMC_BTR4_ACCMOD_1                  ((uint32_t)0x20000000)
03693 /******************  Bit definition for FSMC_BWTR1 register  ******************/
03694 #define  FSMC_BWTR1_ADDSET                   ((uint32_t)0x0000000F)
03695 #define  FSMC_BWTR1_ADDSET_0                 ((uint32_t)0x00000001)
03696 #define  FSMC_BWTR1_ADDSET_1                 ((uint32_t)0x00000002)
03697 #define  FSMC_BWTR1_ADDSET_2                 ((uint32_t)0x00000004)
03698 #define  FSMC_BWTR1_ADDSET_3                 ((uint32_t)0x00000008)
03700 #define  FSMC_BWTR1_ADDHLD                   ((uint32_t)0x000000F0)
03701 #define  FSMC_BWTR1_ADDHLD_0                 ((uint32_t)0x00000010)
03702 #define  FSMC_BWTR1_ADDHLD_1                 ((uint32_t)0x00000020)
03703 #define  FSMC_BWTR1_ADDHLD_2                 ((uint32_t)0x00000040)
03704 #define  FSMC_BWTR1_ADDHLD_3                 ((uint32_t)0x00000080)
03706 #define  FSMC_BWTR1_DATAST                   ((uint32_t)0x0000FF00)
03707 #define  FSMC_BWTR1_DATAST_0                 ((uint32_t)0x00000100)
03708 #define  FSMC_BWTR1_DATAST_1                 ((uint32_t)0x00000200)
03709 #define  FSMC_BWTR1_DATAST_2                 ((uint32_t)0x00000400)
03710 #define  FSMC_BWTR1_DATAST_3                 ((uint32_t)0x00000800)
03712 #define  FSMC_BWTR1_CLKDIV                   ((uint32_t)0x00F00000)
03713 #define  FSMC_BWTR1_CLKDIV_0                 ((uint32_t)0x00100000)
03714 #define  FSMC_BWTR1_CLKDIV_1                 ((uint32_t)0x00200000)
03715 #define  FSMC_BWTR1_CLKDIV_2                 ((uint32_t)0x00400000)
03716 #define  FSMC_BWTR1_CLKDIV_3                 ((uint32_t)0x00800000)
03718 #define  FSMC_BWTR1_DATLAT                   ((uint32_t)0x0F000000)
03719 #define  FSMC_BWTR1_DATLAT_0                 ((uint32_t)0x01000000)
03720 #define  FSMC_BWTR1_DATLAT_1                 ((uint32_t)0x02000000)
03721 #define  FSMC_BWTR1_DATLAT_2                 ((uint32_t)0x04000000)
03722 #define  FSMC_BWTR1_DATLAT_3                 ((uint32_t)0x08000000)
03724 #define  FSMC_BWTR1_ACCMOD                   ((uint32_t)0x30000000)
03725 #define  FSMC_BWTR1_ACCMOD_0                 ((uint32_t)0x10000000)
03726 #define  FSMC_BWTR1_ACCMOD_1                 ((uint32_t)0x20000000)
03728 /******************  Bit definition for FSMC_BWTR2 register  ******************/
03729 #define  FSMC_BWTR2_ADDSET                   ((uint32_t)0x0000000F)
03730 #define  FSMC_BWTR2_ADDSET_0                 ((uint32_t)0x00000001)
03731 #define  FSMC_BWTR2_ADDSET_1                 ((uint32_t)0x00000002)
03732 #define  FSMC_BWTR2_ADDSET_2                 ((uint32_t)0x00000004)
03733 #define  FSMC_BWTR2_ADDSET_3                 ((uint32_t)0x00000008)
03735 #define  FSMC_BWTR2_ADDHLD                   ((uint32_t)0x000000F0)
03736 #define  FSMC_BWTR2_ADDHLD_0                 ((uint32_t)0x00000010)
03737 #define  FSMC_BWTR2_ADDHLD_1                 ((uint32_t)0x00000020)
03738 #define  FSMC_BWTR2_ADDHLD_2                 ((uint32_t)0x00000040)
03739 #define  FSMC_BWTR2_ADDHLD_3                 ((uint32_t)0x00000080)
03741 #define  FSMC_BWTR2_DATAST                   ((uint32_t)0x0000FF00)
03742 #define  FSMC_BWTR2_DATAST_0                 ((uint32_t)0x00000100)
03743 #define  FSMC_BWTR2_DATAST_1                 ((uint32_t)0x00000200)
03744 #define  FSMC_BWTR2_DATAST_2                 ((uint32_t)0x00000400)
03745 #define  FSMC_BWTR2_DATAST_3                 ((uint32_t)0x00000800)
03747 #define  FSMC_BWTR2_CLKDIV                   ((uint32_t)0x00F00000)
03748 #define  FSMC_BWTR2_CLKDIV_0                 ((uint32_t)0x00100000)
03749 #define  FSMC_BWTR2_CLKDIV_1                 ((uint32_t)0x00200000)
03750 #define  FSMC_BWTR2_CLKDIV_2                 ((uint32_t)0x00400000)
03751 #define  FSMC_BWTR2_CLKDIV_3                 ((uint32_t)0x00800000)
03753 #define  FSMC_BWTR2_DATLAT                   ((uint32_t)0x0F000000)
03754 #define  FSMC_BWTR2_DATLAT_0                 ((uint32_t)0x01000000)
03755 #define  FSMC_BWTR2_DATLAT_1                 ((uint32_t)0x02000000)
03756 #define  FSMC_BWTR2_DATLAT_2                 ((uint32_t)0x04000000)
03757 #define  FSMC_BWTR2_DATLAT_3                 ((uint32_t)0x08000000)
03759 #define  FSMC_BWTR2_ACCMOD                   ((uint32_t)0x30000000)
03760 #define  FSMC_BWTR2_ACCMOD_0                 ((uint32_t)0x10000000)
03761 #define  FSMC_BWTR2_ACCMOD_1                 ((uint32_t)0x20000000)
```

```
03763 /******************  Bit definition for FSMC_BWTR3 register  ******************/
03764 #define  FSMC_BWTR3_ADDSET                   ((uint32_t)0x0000000F)
03765 #define  FSMC_BWTR3_ADDSET_0                 ((uint32_t)0x00000001)
03766 #define  FSMC_BWTR3_ADDSET_1                 ((uint32_t)0x00000002)
03767 #define  FSMC_BWTR3_ADDSET_2                 ((uint32_t)0x00000004)
03768 #define  FSMC_BWTR3_ADDSET_3                 ((uint32_t)0x00000008)
03770 #define  FSMC_BWTR3_ADDHLD                   ((uint32_t)0x000000F0)
03771 #define  FSMC_BWTR3_ADDHLD_0                 ((uint32_t)0x00000010)
03772 #define  FSMC_BWTR3_ADDHLD_1                 ((uint32_t)0x00000020)
03773 #define  FSMC_BWTR3_ADDHLD_2                 ((uint32_t)0x00000040)
03774 #define  FSMC_BWTR3_ADDHLD_3                 ((uint32_t)0x00000080)
03776 #define  FSMC_BWTR3_DATAST                   ((uint32_t)0x0000FF00)
03777 #define  FSMC_BWTR3_DATAST_0                 ((uint32_t)0x00000100)
03778 #define  FSMC_BWTR3_DATAST_1                 ((uint32_t)0x00000200)
03779 #define  FSMC_BWTR3_DATAST_2                 ((uint32_t)0x00000400)
03780 #define  FSMC_BWTR3_DATAST_3                 ((uint32_t)0x00000800)
03782 #define  FSMC_BWTR3_CLKDIV                   ((uint32_t)0x00F00000)
03783 #define  FSMC_BWTR3_CLKDIV_0                 ((uint32_t)0x00100000)
03784 #define  FSMC_BWTR3_CLKDIV_1                 ((uint32_t)0x00200000)
03785 #define  FSMC_BWTR3_CLKDIV_2                 ((uint32_t)0x00400000)
03786 #define  FSMC_BWTR3_CLKDIV_3                 ((uint32_t)0x00800000)
03788 #define  FSMC_BWTR3_DATLAT                   ((uint32_t)0x0F000000)
03789 #define  FSMC_BWTR3_DATLAT_0                 ((uint32_t)0x01000000)
03790 #define  FSMC_BWTR3_DATLAT_1                 ((uint32_t)0x02000000)
03791 #define  FSMC_BWTR3_DATLAT_2                 ((uint32_t)0x04000000)
03792 #define  FSMC_BWTR3_DATLAT_3                 ((uint32_t)0x08000000)
03794 #define  FSMC_BWTR3_ACCMOD                   ((uint32_t)0x30000000)
03795 #define  FSMC_BWTR3_ACCMOD_0                 ((uint32_t)0x10000000)
03796 #define  FSMC_BWTR3_ACCMOD_1                 ((uint32_t)0x20000000)
03798 /******************  Bit definition for FSMC_BWTR4 register  ******************/
03799 #define  FSMC_BWTR4_ADDSET                   ((uint32_t)0x0000000F)
03800 #define  FSMC_BWTR4_ADDSET_0                 ((uint32_t)0x00000001)
03801 #define  FSMC_BWTR4_ADDSET_1                 ((uint32_t)0x00000002)
03802 #define  FSMC_BWTR4_ADDSET_2                 ((uint32_t)0x00000004)
03803 #define  FSMC_BWTR4_ADDSET_3                 ((uint32_t)0x00000008)
03805 #define  FSMC_BWTR4_ADDHLD                   ((uint32_t)0x000000F0)
03806 #define  FSMC_BWTR4_ADDHLD_0                 ((uint32_t)0x00000010)
03807 #define  FSMC_BWTR4_ADDHLD_1                 ((uint32_t)0x00000020)
03808 #define  FSMC_BWTR4_ADDHLD_2                 ((uint32_t)0x00000040)
03809 #define  FSMC_BWTR4_ADDHLD_3                 ((uint32_t)0x00000080)
03811 #define  FSMC_BWTR4_DATAST                   ((uint32_t)0x0000FF00)
03812 #define  FSMC_BWTR4_DATAST_0                 ((uint32_t)0x00000100)
03813 #define  FSMC_BWTR4_DATAST_1                 ((uint32_t)0x00000200)
03814 #define  FSMC_BWTR4_DATAST_2                 ((uint32_t)0x00000400)
03815 #define  FSMC_BWTR4_DATAST_3                 ((uint32_t)0x00000800)
03817 #define  FSMC_BWTR4_CLKDIV                   ((uint32_t)0x00F00000)
03818 #define  FSMC_BWTR4_CLKDIV_0                 ((uint32_t)0x00100000)
03819 #define  FSMC_BWTR4_CLKDIV_1                 ((uint32_t)0x00200000)
03820 #define  FSMC_BWTR4_CLKDIV_2                 ((uint32_t)0x00400000)
03821 #define  FSMC_BWTR4_CLKDIV_3                 ((uint32_t)0x00800000)
03823 #define  FSMC_BWTR4_DATLAT                   ((uint32_t)0x0F000000)
03824 #define  FSMC_BWTR4_DATLAT_0                 ((uint32_t)0x01000000)
03825 #define  FSMC_BWTR4_DATLAT_1                 ((uint32_t)0x02000000)
03826 #define  FSMC_BWTR4_DATLAT_2                 ((uint32_t)0x04000000)
03827 #define  FSMC_BWTR4_DATLAT_3                 ((uint32_t)0x08000000)
03829 #define  FSMC_BWTR4_ACCMOD                   ((uint32_t)0x30000000)
03830 #define  FSMC_BWTR4_ACCMOD_0                 ((uint32_t)0x10000000)
03831 #define  FSMC_BWTR4_ACCMOD_1                 ((uint32_t)0x20000000)
03833 /******************  Bit definition for FSMC_PCR2 register  *******************/
03834 #define  FSMC_PCR2_PWAITEN                   ((uint32_t)0x00000002)
03835 #define  FSMC_PCR2_PBKEN                     ((uint32_t)0x00000004)
03836 #define  FSMC_PCR2_PTYP                      ((uint32_t)0x00000008)
03838 #define  FSMC_PCR2_PWID                      ((uint32_t)0x00000030)
03839 #define  FSMC_PCR2_PWID_0                    ((uint32_t)0x00000010)
03840 #define  FSMC_PCR2_PWID_1                    ((uint32_t)0x00000020)
03842 #define  FSMC_PCR2_ECCEN                     ((uint32_t)0x00000040)
03844 #define  FSMC_PCR2_TCLR                      ((uint32_t)0x00001E00)
03845 #define  FSMC_PCR2_TCLR_0                    ((uint32_t)0x00000200)
03846 #define  FSMC_PCR2_TCLR_1                    ((uint32_t)0x00000400)
03847 #define  FSMC_PCR2_TCLR_2                    ((uint32_t)0x00000800)
03848 #define  FSMC_PCR2_TCLR_3                    ((uint32_t)0x00001000)
03850 #define  FSMC_PCR2_TAR                       ((uint32_t)0x0001E000)
03851 #define  FSMC_PCR2_TAR_0                     ((uint32_t)0x00002000)
03852 #define  FSMC_PCR2_TAR_1                     ((uint32_t)0x00004000)
03853 #define  FSMC_PCR2_TAR_2                     ((uint32_t)0x00008000)
03854 #define  FSMC_PCR2_TAR_3                     ((uint32_t)0x00010000)
03856 #define  FSMC_PCR2_ECCPS                     ((uint32_t)0x000E0000)
03857 #define  FSMC_PCR2_ECCPS_0                   ((uint32_t)0x00020000)
03858 #define  FSMC_PCR2_ECCPS_1                   ((uint32_t)0x00040000)
03859 #define  FSMC_PCR2_ECCPS_2                   ((uint32_t)0x00080000)
03861 /******************  Bit definition for FSMC_PCR3 register  *******************/
03862 #define  FSMC_PCR3_PWAITEN                   ((uint32_t)0x00000002)
03863 #define  FSMC_PCR3_PBKEN                     ((uint32_t)0x00000004)
03864 #define  FSMC_PCR3_PTYP                      ((uint32_t)0x00000008)
03866 #define  FSMC_PCR3_PWID                      ((uint32_t)0x00000030)
03867 #define  FSMC_PCR3_PWID_0                    ((uint32_t)0x00000010)
03868 #define  FSMC_PCR3_PWID_1                    ((uint32_t)0x00000020)
```

```
03870 #define  FSMC_PCR3_ECCEN                     ((uint32_t)0x00000040)
03872 #define  FSMC_PCR3_TCLR                      ((uint32_t)0x00001E00)
03873 #define  FSMC_PCR3_TCLR_0                    ((uint32_t)0x00000200)
03874 #define  FSMC_PCR3_TCLR_1                    ((uint32_t)0x00000400)
03875 #define  FSMC_PCR3_TCLR_2                    ((uint32_t)0x00000800)
03876 #define  FSMC_PCR3_TCLR_3                    ((uint32_t)0x00001000)
03878 #define  FSMC_PCR3_TAR                       ((uint32_t)0x0001E000)
03879 #define  FSMC_PCR3_TAR_0                     ((uint32_t)0x00002000)
03880 #define  FSMC_PCR3_TAR_1                     ((uint32_t)0x00004000)
03881 #define  FSMC_PCR3_TAR_2                     ((uint32_t)0x00008000)
03882 #define  FSMC_PCR3_TAR_3                     ((uint32_t)0x00010000)
03884 #define  FSMC_PCR3_ECCPS                     ((uint32_t)0x000E0000)
03885 #define  FSMC_PCR3_ECCPS_0                   ((uint32_t)0x00020000)
03886 #define  FSMC_PCR3_ECCPS_1                   ((uint32_t)0x00040000)
03887 #define  FSMC_PCR3_ECCPS_2                   ((uint32_t)0x00080000)
03889 /******************  Bit definition for FSMC_PCR4 register  ******************/
03890 #define  FSMC_PCR4_PWAITEN                   ((uint32_t)0x00000002)
03891 #define  FSMC_PCR4_PBKEN                     ((uint32_t)0x00000004)
03892 #define  FSMC_PCR4_PTYP                      ((uint32_t)0x00000008)
03894 #define  FSMC_PCR4_PWID                      ((uint32_t)0x00000030)
03895 #define  FSMC_PCR4_PWID_0                    ((uint32_t)0x00000010)
03896 #define  FSMC_PCR4_PWID_1                    ((uint32_t)0x00000020)
03898 #define  FSMC_PCR4_ECCEN                     ((uint32_t)0x00000040)
03900 #define  FSMC_PCR4_TCLR                      ((uint32_t)0x00001E00)
03901 #define  FSMC_PCR4_TCLR_0                    ((uint32_t)0x00000200)
03902 #define  FSMC_PCR4_TCLR_1                    ((uint32_t)0x00000400)
03903 #define  FSMC_PCR4_TCLR_2                    ((uint32_t)0x00000800)
03904 #define  FSMC_PCR4_TCLR_3                    ((uint32_t)0x00001000)
03906 #define  FSMC_PCR4_TAR                       ((uint32_t)0x0001E000)
03907 #define  FSMC_PCR4_TAR_0                     ((uint32_t)0x00002000)
03908 #define  FSMC_PCR4_TAR_1                     ((uint32_t)0x00004000)
03909 #define  FSMC_PCR4_TAR_2                     ((uint32_t)0x00008000)
03910 #define  FSMC_PCR4_TAR_3                     ((uint32_t)0x00010000)
03912 #define  FSMC_PCR4_ECCPS                     ((uint32_t)0x000E0000)
03913 #define  FSMC_PCR4_ECCPS_0                   ((uint32_t)0x00020000)
03914 #define  FSMC_PCR4_ECCPS_1                   ((uint32_t)0x00040000)
03915 #define  FSMC_PCR4_ECCPS_2                   ((uint32_t)0x00080000)
03917 /******************  Bit definition for FSMC_SR2 register  ******************/
03918 #define  FSMC_SR2_IRS                        ((uint8_t)0x01)
03919 #define  FSMC_SR2_ILS                        ((uint8_t)0x02)
03920 #define  FSMC_SR2_IFS                        ((uint8_t)0x04)
03921 #define  FSMC_SR2_IREN                       ((uint8_t)0x08)
03922 #define  FSMC_SR2_ILEN                       ((uint8_t)0x10)
03923 #define  FSMC_SR2_IFEN                       ((uint8_t)0x20)
03924 #define  FSMC_SR2_FEMPT                      ((uint8_t)0x40)
03926 /******************  Bit definition for FSMC_SR3 register  ******************/
03927 #define  FSMC_SR3_IRS                        ((uint8_t)0x01)
03928 #define  FSMC_SR3_ILS                        ((uint8_t)0x02)
03929 #define  FSMC_SR3_IFS                        ((uint8_t)0x04)
03930 #define  FSMC_SR3_IREN                       ((uint8_t)0x08)
03931 #define  FSMC_SR3_ILEN                       ((uint8_t)0x10)
03932 #define  FSMC_SR3_IFEN                       ((uint8_t)0x20)
03933 #define  FSMC_SR3_FEMPT                      ((uint8_t)0x40)
03935 /******************  Bit definition for FSMC_SR4 register  ******************/
03936 #define  FSMC_SR4_IRS                        ((uint8_t)0x01)
03937 #define  FSMC_SR4_ILS                        ((uint8_t)0x02)
03938 #define  FSMC_SR4_IFS                        ((uint8_t)0x04)
03939 #define  FSMC_SR4_IREN                       ((uint8_t)0x08)
03940 #define  FSMC_SR4_ILEN                       ((uint8_t)0x10)
03941 #define  FSMC_SR4_IFEN                       ((uint8_t)0x20)
03942 #define  FSMC_SR4_FEMPT                      ((uint8_t)0x40)
03944 /******************  Bit definition for FSMC_PMEM2 register  ******************/
03945 #define  FSMC_PMEM2_MEMSET2                  ((uint32_t)0x000000FF)
03946 #define  FSMC_PMEM2_MEMSET2_0                ((uint32_t)0x00000001)
03947 #define  FSMC_PMEM2_MEMSET2_1                ((uint32_t)0x00000002)
03948 #define  FSMC_PMEM2_MEMSET2_2                ((uint32_t)0x00000004)
03949 #define  FSMC_PMEM2_MEMSET2_3                ((uint32_t)0x00000008)
03950 #define  FSMC_PMEM2_MEMSET2_4                ((uint32_t)0x00000010)
03951 #define  FSMC_PMEM2_MEMSET2_5                ((uint32_t)0x00000020)
03952 #define  FSMC_PMEM2_MEMSET2_6                ((uint32_t)0x00000040)
03953 #define  FSMC_PMEM2_MEMSET2_7                ((uint32_t)0x00000080)
03955 #define  FSMC_PMEM2_MEMWAIT2                 ((uint32_t)0x0000FF00)
03956 #define  FSMC_PMEM2_MEMWAIT2_0               ((uint32_t)0x00000100)
03957 #define  FSMC_PMEM2_MEMWAIT2_1               ((uint32_t)0x00000200)
03958 #define  FSMC_PMEM2_MEMWAIT2_2               ((uint32_t)0x00000400)
03959 #define  FSMC_PMEM2_MEMWAIT2_3               ((uint32_t)0x00000800)
03960 #define  FSMC_PMEM2_MEMWAIT2_4               ((uint32_t)0x00001000)
03961 #define  FSMC_PMEM2_MEMWAIT2_5               ((uint32_t)0x00002000)
03962 #define  FSMC_PMEM2_MEMWAIT2_6               ((uint32_t)0x00004000)
03963 #define  FSMC_PMEM2_MEMWAIT2_7               ((uint32_t)0x00008000)
03965 #define  FSMC_PMEM2_MEMHOLD2                 ((uint32_t)0x00FF0000)
03966 #define  FSMC_PMEM2_MEMHOLD2_0               ((uint32_t)0x00010000)
03967 #define  FSMC_PMEM2_MEMHOLD2_1               ((uint32_t)0x00020000)
03968 #define  FSMC_PMEM2_MEMHOLD2_2               ((uint32_t)0x00040000)
03969 #define  FSMC_PMEM2_MEMHOLD2_3               ((uint32_t)0x00080000)
03970 #define  FSMC_PMEM2_MEMHOLD2_4               ((uint32_t)0x00100000)
03971 #define  FSMC_PMEM2_MEMHOLD2_5               ((uint32_t)0x00200000)
```

```
03972 #define  FSMC_PMEM2_MEMHOLD2_6              ((uint32_t)0x00400000)
03973 #define  FSMC_PMEM2_MEMHOLD2_7              ((uint32_t)0x00800000)
03975 #define  FSMC_PMEM2_MEMHIZ2                 ((uint32_t)0xFF000000)
03976 #define  FSMC_PMEM2_MEMHIZ2_0               ((uint32_t)0x01000000)
03977 #define  FSMC_PMEM2_MEMHIZ2_1               ((uint32_t)0x02000000)
03978 #define  FSMC_PMEM2_MEMHIZ2_2               ((uint32_t)0x04000000)
03979 #define  FSMC_PMEM2_MEMHIZ2_3               ((uint32_t)0x08000000)
03980 #define  FSMC_PMEM2_MEMHIZ2_4               ((uint32_t)0x10000000)
03981 #define  FSMC_PMEM2_MEMHIZ2_5               ((uint32_t)0x20000000)
03982 #define  FSMC_PMEM2_MEMHIZ2_6               ((uint32_t)0x40000000)
03983 #define  FSMC_PMEM2_MEMHIZ2_7               ((uint32_t)0x80000000)
03985 /******************  Bit definition for FSMC_PMEM3 register  ******************/
03986 #define  FSMC_PMEM3_MEMSET3                 ((uint32_t)0x000000FF)
03987 #define  FSMC_PMEM3_MEMSET3_0               ((uint32_t)0x00000001)
03988 #define  FSMC_PMEM3_MEMSET3_1               ((uint32_t)0x00000002)
03989 #define  FSMC_PMEM3_MEMSET3_2               ((uint32_t)0x00000004)
03990 #define  FSMC_PMEM3_MEMSET3_3               ((uint32_t)0x00000008)
03991 #define  FSMC_PMEM3_MEMSET3_4               ((uint32_t)0x00000010)
03992 #define  FSMC_PMEM3_MEMSET3_5               ((uint32_t)0x00000020)
03993 #define  FSMC_PMEM3_MEMSET3_6               ((uint32_t)0x00000040)
03994 #define  FSMC_PMEM3_MEMSET3_7               ((uint32_t)0x00000080)
03996 #define  FSMC_PMEM3_MEMWAIT3                ((uint32_t)0x0000FF00)
03997 #define  FSMC_PMEM3_MEMWAIT3_0              ((uint32_t)0x00000100)
03998 #define  FSMC_PMEM3_MEMWAIT3_1              ((uint32_t)0x00000200)
03999 #define  FSMC_PMEM3_MEMWAIT3_2              ((uint32_t)0x00000400)
04000 #define  FSMC_PMEM3_MEMWAIT3_3              ((uint32_t)0x00000800)
04001 #define  FSMC_PMEM3_MEMWAIT3_4              ((uint32_t)0x00001000)
04002 #define  FSMC_PMEM3_MEMWAIT3_5              ((uint32_t)0x00002000)
04003 #define  FSMC_PMEM3_MEMWAIT3_6              ((uint32_t)0x00004000)
04004 #define  FSMC_PMEM3_MEMWAIT3_7              ((uint32_t)0x00008000)
04006 #define  FSMC_PMEM3_MEMHOLD3                ((uint32_t)0x00FF0000)
04007 #define  FSMC_PMEM3_MEMHOLD3_0              ((uint32_t)0x00010000)
04008 #define  FSMC_PMEM3_MEMHOLD3_1              ((uint32_t)0x00020000)
04009 #define  FSMC_PMEM3_MEMHOLD3_2              ((uint32_t)0x00040000)
04010 #define  FSMC_PMEM3_MEMHOLD3_3              ((uint32_t)0x00080000)
04011 #define  FSMC_PMEM3_MEMHOLD3_4              ((uint32_t)0x00100000)
04012 #define  FSMC_PMEM3_MEMHOLD3_5              ((uint32_t)0x00200000)
04013 #define  FSMC_PMEM3_MEMHOLD3_6              ((uint32_t)0x00400000)
04014 #define  FSMC_PMEM3_MEMHOLD3_7              ((uint32_t)0x00800000)
04016 #define  FSMC_PMEM3_MEMHIZ3                 ((uint32_t)0xFF000000)
04017 #define  FSMC_PMEM3_MEMHIZ3_0               ((uint32_t)0x01000000)
04018 #define  FSMC_PMEM3_MEMHIZ3_1               ((uint32_t)0x02000000)
04019 #define  FSMC_PMEM3_MEMHIZ3_2               ((uint32_t)0x04000000)
04020 #define  FSMC_PMEM3_MEMHIZ3_3               ((uint32_t)0x08000000)
04021 #define  FSMC_PMEM3_MEMHIZ3_4               ((uint32_t)0x10000000)
04022 #define  FSMC_PMEM3_MEMHIZ3_5               ((uint32_t)0x20000000)
04023 #define  FSMC_PMEM3_MEMHIZ3_6               ((uint32_t)0x40000000)
04024 #define  FSMC_PMEM3_MEMHIZ3_7               ((uint32_t)0x80000000)
04026 /******************  Bit definition for FSMC_PMEM4 register  ******************/
04027 #define  FSMC_PMEM4_MEMSET4                 ((uint32_t)0x000000FF)
04028 #define  FSMC_PMEM4_MEMSET4_0               ((uint32_t)0x00000001)
04029 #define  FSMC_PMEM4_MEMSET4_1               ((uint32_t)0x00000002)
04030 #define  FSMC_PMEM4_MEMSET4_2               ((uint32_t)0x00000004)
04031 #define  FSMC_PMEM4_MEMSET4_3               ((uint32_t)0x00000008)
04032 #define  FSMC_PMEM4_MEMSET4_4               ((uint32_t)0x00000010)
04033 #define  FSMC_PMEM4_MEMSET4_5               ((uint32_t)0x00000020)
04034 #define  FSMC_PMEM4_MEMSET4_6               ((uint32_t)0x00000040)
04035 #define  FSMC_PMEM4_MEMSET4_7               ((uint32_t)0x00000080)
04037 #define  FSMC_PMEM4_MEMWAIT4                ((uint32_t)0x0000FF00)
04038 #define  FSMC_PMEM4_MEMWAIT4_0              ((uint32_t)0x00000100)
04039 #define  FSMC_PMEM4_MEMWAIT4_1              ((uint32_t)0x00000200)
04040 #define  FSMC_PMEM4_MEMWAIT4_2              ((uint32_t)0x00000400)
04041 #define  FSMC_PMEM4_MEMWAIT4_3              ((uint32_t)0x00000800)
04042 #define  FSMC_PMEM4_MEMWAIT4_4              ((uint32_t)0x00001000)
04043 #define  FSMC_PMEM4_MEMWAIT4_5              ((uint32_t)0x00002000)
04044 #define  FSMC_PMEM4_MEMWAIT4_6              ((uint32_t)0x00004000)
04045 #define  FSMC_PMEM4_MEMWAIT4_7              ((uint32_t)0x00008000)
04047 #define  FSMC_PMEM4_MEMHOLD4                ((uint32_t)0x00FF0000)
04048 #define  FSMC_PMEM4_MEMHOLD4_0              ((uint32_t)0x00010000)
04049 #define  FSMC_PMEM4_MEMHOLD4_1              ((uint32_t)0x00020000)
04050 #define  FSMC_PMEM4_MEMHOLD4_2              ((uint32_t)0x00040000)
04051 #define  FSMC_PMEM4_MEMHOLD4_3              ((uint32_t)0x00080000)
04052 #define  FSMC_PMEM4_MEMHOLD4_4              ((uint32_t)0x00100000)
04053 #define  FSMC_PMEM4_MEMHOLD4_5              ((uint32_t)0x00200000)
04054 #define  FSMC_PMEM4_MEMHOLD4_6              ((uint32_t)0x00400000)
04055 #define  FSMC_PMEM4_MEMHOLD4_7              ((uint32_t)0x00800000)
04057 #define  FSMC_PMEM4_MEMHIZ4                 ((uint32_t)0xFF000000)
04058 #define  FSMC_PMEM4_MEMHIZ4_0               ((uint32_t)0x01000000)
04059 #define  FSMC_PMEM4_MEMHIZ4_1               ((uint32_t)0x02000000)
04060 #define  FSMC_PMEM4_MEMHIZ4_2               ((uint32_t)0x04000000)
04061 #define  FSMC_PMEM4_MEMHIZ4_3               ((uint32_t)0x08000000)
04062 #define  FSMC_PMEM4_MEMHIZ4_4               ((uint32_t)0x10000000)
04063 #define  FSMC_PMEM4_MEMHIZ4_5               ((uint32_t)0x20000000)
04064 #define  FSMC_PMEM4_MEMHIZ4_6               ((uint32_t)0x40000000)
04065 #define  FSMC_PMEM4_MEMHIZ4_7               ((uint32_t)0x80000000)
04067 /******************  Bit definition for FSMC_PATT2 register  ******************/
04068 #define  FSMC_PATT2_ATTSET2                 ((uint32_t)0x000000FF)
```

```
04069 #define  FSMC_PATT2_ATTSET2_0               ((uint32_t)0x00000001)
04070 #define  FSMC_PATT2_ATTSET2_1               ((uint32_t)0x00000002)
04071 #define  FSMC_PATT2_ATTSET2_2               ((uint32_t)0x00000004)
04072 #define  FSMC_PATT2_ATTSET2_3               ((uint32_t)0x00000008)
04073 #define  FSMC_PATT2_ATTSET2_4               ((uint32_t)0x00000010)
04074 #define  FSMC_PATT2_ATTSET2_5               ((uint32_t)0x00000020)
04075 #define  FSMC_PATT2_ATTSET2_6               ((uint32_t)0x00000040)
04076 #define  FSMC_PATT2_ATTSET2_7               ((uint32_t)0x00000080)
04078 #define  FSMC_PATT2_ATTWAIT2                ((uint32_t)0x0000FF00)
04079 #define  FSMC_PATT2_ATTWAIT2_0              ((uint32_t)0x00000100)
04080 #define  FSMC_PATT2_ATTWAIT2_1              ((uint32_t)0x00000200)
04081 #define  FSMC_PATT2_ATTWAIT2_2              ((uint32_t)0x00000400)
04082 #define  FSMC_PATT2_ATTWAIT2_3              ((uint32_t)0x00000800)
04083 #define  FSMC_PATT2_ATTWAIT2_4              ((uint32_t)0x00001000)
04084 #define  FSMC_PATT2_ATTWAIT2_5              ((uint32_t)0x00002000)
04085 #define  FSMC_PATT2_ATTWAIT2_6              ((uint32_t)0x00004000)
04086 #define  FSMC_PATT2_ATTWAIT2_7              ((uint32_t)0x00008000)
04088 #define  FSMC_PATT2_ATTHOLD2                ((uint32_t)0x00FF0000)
04089 #define  FSMC_PATT2_ATTHOLD2_0              ((uint32_t)0x00010000)
04090 #define  FSMC_PATT2_ATTHOLD2_1              ((uint32_t)0x00020000)
04091 #define  FSMC_PATT2_ATTHOLD2_2              ((uint32_t)0x00040000)
04092 #define  FSMC_PATT2_ATTHOLD2_3              ((uint32_t)0x00080000)
04093 #define  FSMC_PATT2_ATTHOLD2_4              ((uint32_t)0x00100000)
04094 #define  FSMC_PATT2_ATTHOLD2_5              ((uint32_t)0x00200000)
04095 #define  FSMC_PATT2_ATTHOLD2_6              ((uint32_t)0x00400000)
04096 #define  FSMC_PATT2_ATTHOLD2_7              ((uint32_t)0x00800000)
04098 #define  FSMC_PATT2_ATTHIZ2                 ((uint32_t)0xFF000000)
04099 #define  FSMC_PATT2_ATTHIZ2_0               ((uint32_t)0x01000000)
04100 #define  FSMC_PATT2_ATTHIZ2_1               ((uint32_t)0x02000000)
04101 #define  FSMC_PATT2_ATTHIZ2_2               ((uint32_t)0x04000000)
04102 #define  FSMC_PATT2_ATTHIZ2_3               ((uint32_t)0x08000000)
04103 #define  FSMC_PATT2_ATTHIZ2_4               ((uint32_t)0x10000000)
04104 #define  FSMC_PATT2_ATTHIZ2_5               ((uint32_t)0x20000000)
04105 #define  FSMC_PATT2_ATTHIZ2_6               ((uint32_t)0x40000000)
04106 #define  FSMC_PATT2_ATTHIZ2_7               ((uint32_t)0x80000000)
04108 /******************  Bit definition for FSMC_PATT3 register  ******************/
04109 #define  FSMC_PATT3_ATTSET3                 ((uint32_t)0x000000FF)
04110 #define  FSMC_PATT3_ATTSET3_0               ((uint32_t)0x00000001)
04111 #define  FSMC_PATT3_ATTSET3_1               ((uint32_t)0x00000002)
04112 #define  FSMC_PATT3_ATTSET3_2               ((uint32_t)0x00000004)
04113 #define  FSMC_PATT3_ATTSET3_3               ((uint32_t)0x00000008)
04114 #define  FSMC_PATT3_ATTSET3_4               ((uint32_t)0x00000010)
04115 #define  FSMC_PATT3_ATTSET3_5               ((uint32_t)0x00000020)
04116 #define  FSMC_PATT3_ATTSET3_6               ((uint32_t)0x00000040)
04117 #define  FSMC_PATT3_ATTSET3_7               ((uint32_t)0x00000080)
04119 #define  FSMC_PATT3_ATTWAIT3                ((uint32_t)0x0000FF00)
04120 #define  FSMC_PATT3_ATTWAIT3_0              ((uint32_t)0x00000100)
04121 #define  FSMC_PATT3_ATTWAIT3_1              ((uint32_t)0x00000200)
04122 #define  FSMC_PATT3_ATTWAIT3_2              ((uint32_t)0x00000400)
04123 #define  FSMC_PATT3_ATTWAIT3_3              ((uint32_t)0x00000800)
04124 #define  FSMC_PATT3_ATTWAIT3_4              ((uint32_t)0x00001000)
04125 #define  FSMC_PATT3_ATTWAIT3_5              ((uint32_t)0x00002000)
04126 #define  FSMC_PATT3_ATTWAIT3_6              ((uint32_t)0x00004000)
04127 #define  FSMC_PATT3_ATTWAIT3_7              ((uint32_t)0x00008000)
04129 #define  FSMC_PATT3_ATTHOLD3                ((uint32_t)0x00FF0000)
04130 #define  FSMC_PATT3_ATTHOLD3_0              ((uint32_t)0x00010000)
04131 #define  FSMC_PATT3_ATTHOLD3_1              ((uint32_t)0x00020000)
04132 #define  FSMC_PATT3_ATTHOLD3_2              ((uint32_t)0x00040000)
04133 #define  FSMC_PATT3_ATTHOLD3_3              ((uint32_t)0x00080000)
04134 #define  FSMC_PATT3_ATTHOLD3_4              ((uint32_t)0x00100000)
04135 #define  FSMC_PATT3_ATTHOLD3_5              ((uint32_t)0x00200000)
04136 #define  FSMC_PATT3_ATTHOLD3_6              ((uint32_t)0x00400000)
04137 #define  FSMC_PATT3_ATTHOLD3_7              ((uint32_t)0x00800000)
04139 #define  FSMC_PATT3_ATTHIZ3                 ((uint32_t)0xFF000000)
04140 #define  FSMC_PATT3_ATTHIZ3_0               ((uint32_t)0x01000000)
04141 #define  FSMC_PATT3_ATTHIZ3_1               ((uint32_t)0x02000000)
04142 #define  FSMC_PATT3_ATTHIZ3_2               ((uint32_t)0x04000000)
04143 #define  FSMC_PATT3_ATTHIZ3_3               ((uint32_t)0x08000000)
04144 #define  FSMC_PATT3_ATTHIZ3_4               ((uint32_t)0x10000000)
04145 #define  FSMC_PATT3_ATTHIZ3_5               ((uint32_t)0x20000000)
04146 #define  FSMC_PATT3_ATTHIZ3_6               ((uint32_t)0x40000000)
04147 #define  FSMC_PATT3_ATTHIZ3_7               ((uint32_t)0x80000000)
04149 /******************  Bit definition for FSMC_PATT4 register  ******************/
04150 #define  FSMC_PATT4_ATTSET4                 ((uint32_t)0x000000FF)
04151 #define  FSMC_PATT4_ATTSET4_0               ((uint32_t)0x00000001)
04152 #define  FSMC_PATT4_ATTSET4_1               ((uint32_t)0x00000002)
04153 #define  FSMC_PATT4_ATTSET4_2               ((uint32_t)0x00000004)
04154 #define  FSMC_PATT4_ATTSET4_3               ((uint32_t)0x00000008)
04155 #define  FSMC_PATT4_ATTSET4_4               ((uint32_t)0x00000010)
04156 #define  FSMC_PATT4_ATTSET4_5               ((uint32_t)0x00000020)
04157 #define  FSMC_PATT4_ATTSET4_6               ((uint32_t)0x00000040)
04158 #define  FSMC_PATT4_ATTSET4_7               ((uint32_t)0x00000080)
04160 #define  FSMC_PATT4_ATTWAIT4                ((uint32_t)0x0000FF00)
04161 #define  FSMC_PATT4_ATTWAIT4_0              ((uint32_t)0x00000100)
04162 #define  FSMC_PATT4_ATTWAIT4_1              ((uint32_t)0x00000200)
04163 #define  FSMC_PATT4_ATTWAIT4_2              ((uint32_t)0x00000400)
04164 #define  FSMC_PATT4_ATTWAIT4_3              ((uint32_t)0x00000800)
```

```
04165 #define  FSMC_PATT4_ATTWAIT4_4              ((uint32_t)0x00001000)
04166 #define  FSMC_PATT4_ATTWAIT4_5              ((uint32_t)0x00002000)
04167 #define  FSMC_PATT4_ATTWAIT4_6              ((uint32_t)0x00004000)
04168 #define  FSMC_PATT4_ATTWAIT4_7              ((uint32_t)0x00008000)
04170 #define  FSMC_PATT4_ATTHOLD4                ((uint32_t)0x00FF0000)
04171 #define  FSMC_PATT4_ATTHOLD4_0              ((uint32_t)0x00010000)
04172 #define  FSMC_PATT4_ATTHOLD4_1              ((uint32_t)0x00020000)
04173 #define  FSMC_PATT4_ATTHOLD4_2              ((uint32_t)0x00040000)
04174 #define  FSMC_PATT4_ATTHOLD4_3              ((uint32_t)0x00080000)
04175 #define  FSMC_PATT4_ATTHOLD4_4              ((uint32_t)0x00100000)
04176 #define  FSMC_PATT4_ATTHOLD4_5              ((uint32_t)0x00200000)
04177 #define  FSMC_PATT4_ATTHOLD4_6              ((uint32_t)0x00400000)
04178 #define  FSMC_PATT4_ATTHOLD4_7              ((uint32_t)0x00800000)
04180 #define  FSMC_PATT4_ATTHIZ4                 ((uint32_t)0xFF000000)
04181 #define  FSMC_PATT4_ATTHIZ4_0               ((uint32_t)0x01000000)
04182 #define  FSMC_PATT4_ATTHIZ4_1               ((uint32_t)0x02000000)
04183 #define  FSMC_PATT4_ATTHIZ4_2               ((uint32_t)0x04000000)
04184 #define  FSMC_PATT4_ATTHIZ4_3               ((uint32_t)0x08000000)
04185 #define  FSMC_PATT4_ATTHIZ4_4               ((uint32_t)0x10000000)
04186 #define  FSMC_PATT4_ATTHIZ4_5               ((uint32_t)0x20000000)
04187 #define  FSMC_PATT4_ATTHIZ4_6               ((uint32_t)0x40000000)
04188 #define  FSMC_PATT4_ATTHIZ4_7               ((uint32_t)0x80000000)
04190 /******************  Bit definition for FSMC_PIO4 register  *******************/
04191 #define  FSMC_PIO4_IOSET4                   ((uint32_t)0x000000FF)
04192 #define  FSMC_PIO4_IOSET4_0                 ((uint32_t)0x00000001)
04193 #define  FSMC_PIO4_IOSET4_1                 ((uint32_t)0x00000002)
04194 #define  FSMC_PIO4_IOSET4_2                 ((uint32_t)0x00000004)
04195 #define  FSMC_PIO4_IOSET4_3                 ((uint32_t)0x00000008)
04196 #define  FSMC_PIO4_IOSET4_4                 ((uint32_t)0x00000010)
04197 #define  FSMC_PIO4_IOSET4_5                 ((uint32_t)0x00000020)
04198 #define  FSMC_PIO4_IOSET4_6                 ((uint32_t)0x00000040)
04199 #define  FSMC_PIO4_IOSET4_7                 ((uint32_t)0x00000080)
04201 #define  FSMC_PIO4_IOWAIT4                  ((uint32_t)0x0000FF00)
04202 #define  FSMC_PIO4_IOWAIT4_0                ((uint32_t)0x00000100)
04203 #define  FSMC_PIO4_IOWAIT4_1                ((uint32_t)0x00000200)
04204 #define  FSMC_PIO4_IOWAIT4_2                ((uint32_t)0x00000400)
04205 #define  FSMC_PIO4_IOWAIT4_3                ((uint32_t)0x00000800)
04206 #define  FSMC_PIO4_IOWAIT4_4                ((uint32_t)0x00001000)
04207 #define  FSMC_PIO4_IOWAIT4_5                ((uint32_t)0x00002000)
04208 #define  FSMC_PIO4_IOWAIT4_6                ((uint32_t)0x00004000)
04209 #define  FSMC_PIO4_IOWAIT4_7                ((uint32_t)0x00008000)
04211 #define  FSMC_PIO4_IOHOLD4                  ((uint32_t)0x00FF0000)
04212 #define  FSMC_PIO4_IOHOLD4_0                ((uint32_t)0x00010000)
04213 #define  FSMC_PIO4_IOHOLD4_1                ((uint32_t)0x00020000)
04214 #define  FSMC_PIO4_IOHOLD4_2                ((uint32_t)0x00040000)
04215 #define  FSMC_PIO4_IOHOLD4_3                ((uint32_t)0x00080000)
04216 #define  FSMC_PIO4_IOHOLD4_4                ((uint32_t)0x00100000)
04217 #define  FSMC_PIO4_IOHOLD4_5                ((uint32_t)0x00200000)
04218 #define  FSMC_PIO4_IOHOLD4_6                ((uint32_t)0x00400000)
04219 #define  FSMC_PIO4_IOHOLD4_7                ((uint32_t)0x00800000)
04221 #define  FSMC_PIO4_IOHIZ4                   ((uint32_t)0xFF000000)
04222 #define  FSMC_PIO4_IOHIZ4_0                 ((uint32_t)0x01000000)
04223 #define  FSMC_PIO4_IOHIZ4_1                 ((uint32_t)0x02000000)
04224 #define  FSMC_PIO4_IOHIZ4_2                 ((uint32_t)0x04000000)
04225 #define  FSMC_PIO4_IOHIZ4_3                 ((uint32_t)0x08000000)
04226 #define  FSMC_PIO4_IOHIZ4_4                 ((uint32_t)0x10000000)
04227 #define  FSMC_PIO4_IOHIZ4_5                 ((uint32_t)0x20000000)
04228 #define  FSMC_PIO4_IOHIZ4_6                 ((uint32_t)0x40000000)
04229 #define  FSMC_PIO4_IOHIZ4_7                 ((uint32_t)0x80000000)
04231 /******************  Bit definition for FSMC_ECCR2 register  ******************/
04232 #define  FSMC_ECCR2_ECC2                    ((uint32_t)0xFFFFFFFF)
04234 /******************  Bit definition for FSMC_ECCR3 register  ******************/
04235 #define  FSMC_ECCR3_ECC3                    ((uint32_t)0xFFFFFFFF)
04237 /******************************************************************************/
04238 /*                                                                            */
04239 /*                             General Purpose I/O                            */
04240 /*                                                                            */
04241 /******************************************************************************/
04242 /******************  Bits definition for GPIO_MODER register  *****************/
04243 #define GPIO_MODER_MODER0                   ((uint32_t)0x00000003)
04244 #define GPIO_MODER_MODER0_0                 ((uint32_t)0x00000001)
04245 #define GPIO_MODER_MODER0_1                 ((uint32_t)0x00000002)
04246
04247 #define GPIO_MODER_MODER1                   ((uint32_t)0x0000000C)
04248 #define GPIO_MODER_MODER1_0                 ((uint32_t)0x00000004)
04249 #define GPIO_MODER_MODER1_1                 ((uint32_t)0x00000008)
04250
04251 #define GPIO_MODER_MODER2                   ((uint32_t)0x00000030)
04252 #define GPIO_MODER_MODER2_0                 ((uint32_t)0x00000010)
04253 #define GPIO_MODER_MODER2_1                 ((uint32_t)0x00000020)
04254
04255 #define GPIO_MODER_MODER3                   ((uint32_t)0x000000C0)
04256 #define GPIO_MODER_MODER3_0                 ((uint32_t)0x00000040)
04257 #define GPIO_MODER_MODER3_1                 ((uint32_t)0x00000080)
04258
04259 #define GPIO_MODER_MODER4                   ((uint32_t)0x00000300)
04260 #define GPIO_MODER_MODER4_0                 ((uint32_t)0x00000100)
```

```
04261 #define GPIO_MODER_MODER4_1                 ((uint32_t)0x00000200)
04262
04263 #define GPIO_MODER_MODER5                   ((uint32_t)0x00000C00)
04264 #define GPIO_MODER_MODER5_0                 ((uint32_t)0x00000400)
04265 #define GPIO_MODER_MODER5_1                 ((uint32_t)0x00000800)
04266
04267 #define GPIO_MODER_MODER6                   ((uint32_t)0x00003000)
04268 #define GPIO_MODER_MODER6_0                 ((uint32_t)0x00001000)
04269 #define GPIO_MODER_MODER6_1                 ((uint32_t)0x00002000)
04270
04271 #define GPIO_MODER_MODER7                   ((uint32_t)0x0000C000)
04272 #define GPIO_MODER_MODER7_0                 ((uint32_t)0x00004000)
04273 #define GPIO_MODER_MODER7_1                 ((uint32_t)0x00008000)
04274
04275 #define GPIO_MODER_MODER8                   ((uint32_t)0x00030000)
04276 #define GPIO_MODER_MODER8_0                 ((uint32_t)0x00010000)
04277 #define GPIO_MODER_MODER8_1                 ((uint32_t)0x00020000)
04278
04279 #define GPIO_MODER_MODER9                   ((uint32_t)0x000C0000)
04280 #define GPIO_MODER_MODER9_0                 ((uint32_t)0x00040000)
04281 #define GPIO_MODER_MODER9_1                 ((uint32_t)0x00080000)
04282
04283 #define GPIO_MODER_MODER10                  ((uint32_t)0x00300000)
04284 #define GPIO_MODER_MODER10_0                ((uint32_t)0x00100000)
04285 #define GPIO_MODER_MODER10_1                ((uint32_t)0x00200000)
04286
04287 #define GPIO_MODER_MODER11                  ((uint32_t)0x00C00000)
04288 #define GPIO_MODER_MODER11_0                ((uint32_t)0x00400000)
04289 #define GPIO_MODER_MODER11_1                ((uint32_t)0x00800000)
04290
04291 #define GPIO_MODER_MODER12                  ((uint32_t)0x03000000)
04292 #define GPIO_MODER_MODER12_0                ((uint32_t)0x01000000)
04293 #define GPIO_MODER_MODER12_1                ((uint32_t)0x02000000)
04294
04295 #define GPIO_MODER_MODER13                  ((uint32_t)0x0C000000)
04296 #define GPIO_MODER_MODER13_0                ((uint32_t)0x04000000)
04297 #define GPIO_MODER_MODER13_1                ((uint32_t)0x08000000)
04298
04299 #define GPIO_MODER_MODER14                  ((uint32_t)0x30000000)
04300 #define GPIO_MODER_MODER14_0                ((uint32_t)0x10000000)
04301 #define GPIO_MODER_MODER14_1                ((uint32_t)0x20000000)
04302
04303 #define GPIO_MODER_MODER15                  ((uint32_t)0xC0000000)
04304 #define GPIO_MODER_MODER15_0                ((uint32_t)0x40000000)
04305 #define GPIO_MODER_MODER15_1                ((uint32_t)0x80000000)
04306
04307 /******************  Bits definition for GPIO_OTYPER register  ****************/
04308 #define GPIO_OTYPER_OT_0                    ((uint32_t)0x00000001)
04309 #define GPIO_OTYPER_OT_1                    ((uint32_t)0x00000002)
04310 #define GPIO_OTYPER_OT_2                    ((uint32_t)0x00000004)
04311 #define GPIO_OTYPER_OT_3                    ((uint32_t)0x00000008)
04312 #define GPIO_OTYPER_OT_4                    ((uint32_t)0x00000010)
04313 #define GPIO_OTYPER_OT_5                    ((uint32_t)0x00000020)
04314 #define GPIO_OTYPER_OT_6                    ((uint32_t)0x00000040)
04315 #define GPIO_OTYPER_OT_7                    ((uint32_t)0x00000080)
04316 #define GPIO_OTYPER_OT_8                    ((uint32_t)0x00000100)
04317 #define GPIO_OTYPER_OT_9                    ((uint32_t)0x00000200)
04318 #define GPIO_OTYPER_OT_10                   ((uint32_t)0x00000400)
04319 #define GPIO_OTYPER_OT_11                   ((uint32_t)0x00000800)
04320 #define GPIO_OTYPER_OT_12                   ((uint32_t)0x00001000)
04321 #define GPIO_OTYPER_OT_13                   ((uint32_t)0x00002000)
04322 #define GPIO_OTYPER_OT_14                   ((uint32_t)0x00004000)
04323 #define GPIO_OTYPER_OT_15                   ((uint32_t)0x00008000)
04324
04325 /******************  Bits definition for GPIO_OSPEEDR register  ***************/
04326 #define GPIO_OSPEEDER_OSPEEDR0              ((uint32_t)0x00000003)
04327 #define GPIO_OSPEEDER_OSPEEDR0_0            ((uint32_t)0x00000001)
04328 #define GPIO_OSPEEDER_OSPEEDR0_1            ((uint32_t)0x00000002)
04329
04330 #define GPIO_OSPEEDER_OSPEEDR1              ((uint32_t)0x0000000C)
04331 #define GPIO_OSPEEDER_OSPEEDR1_0            ((uint32_t)0x00000004)
04332 #define GPIO_OSPEEDER_OSPEEDR1_1            ((uint32_t)0x00000008)
04333
04334 #define GPIO_OSPEEDER_OSPEEDR2              ((uint32_t)0x00000030)
04335 #define GPIO_OSPEEDER_OSPEEDR2_0            ((uint32_t)0x00000010)
04336 #define GPIO_OSPEEDER_OSPEEDR2_1            ((uint32_t)0x00000020)
04337
04338 #define GPIO_OSPEEDER_OSPEEDR3              ((uint32_t)0x000000C0)
04339 #define GPIO_OSPEEDER_OSPEEDR3_0            ((uint32_t)0x00000040)
04340 #define GPIO_OSPEEDER_OSPEEDR3_1            ((uint32_t)0x00000080)
04341
04342 #define GPIO_OSPEEDER_OSPEEDR4              ((uint32_t)0x00000300)
04343 #define GPIO_OSPEEDER_OSPEEDR4_0            ((uint32_t)0x00000100)
04344 #define GPIO_OSPEEDER_OSPEEDR4_1            ((uint32_t)0x00000200)
04345
04346 #define GPIO_OSPEEDER_OSPEEDR5              ((uint32_t)0x00000C00)
04347 #define GPIO_OSPEEDER_OSPEEDR5_0            ((uint32_t)0x00000400)
```

```
04348 #define GPIO_OSPEEDER_OSPEEDR5_1              ((uint32_t)0x00000800)
04349
04350 #define GPIO_OSPEEDER_OSPEEDR6                ((uint32_t)0x00003000)
04351 #define GPIO_OSPEEDER_OSPEEDR6_0              ((uint32_t)0x00001000)
04352 #define GPIO_OSPEEDER_OSPEEDR6_1              ((uint32_t)0x00002000)
04353
04354 #define GPIO_OSPEEDER_OSPEEDR7                ((uint32_t)0x0000C000)
04355 #define GPIO_OSPEEDER_OSPEEDR7_0              ((uint32_t)0x00004000)
04356 #define GPIO_OSPEEDER_OSPEEDR7_1              ((uint32_t)0x00008000)
04357
04358 #define GPIO_OSPEEDER_OSPEEDR8                ((uint32_t)0x00030000)
04359 #define GPIO_OSPEEDER_OSPEEDR8_0              ((uint32_t)0x00010000)
04360 #define GPIO_OSPEEDER_OSPEEDR8_1              ((uint32_t)0x00020000)
04361
04362 #define GPIO_OSPEEDER_OSPEEDR9                ((uint32_t)0x000C0000)
04363 #define GPIO_OSPEEDER_OSPEEDR9_0              ((uint32_t)0x00040000)
04364 #define GPIO_OSPEEDER_OSPEEDR9_1              ((uint32_t)0x00080000)
04365
04366 #define GPIO_OSPEEDER_OSPEEDR10               ((uint32_t)0x00300000)
04367 #define GPIO_OSPEEDER_OSPEEDR10_0             ((uint32_t)0x00100000)
04368 #define GPIO_OSPEEDER_OSPEEDR10_1             ((uint32_t)0x00200000)
04369
04370 #define GPIO_OSPEEDER_OSPEEDR11               ((uint32_t)0x00C00000)
04371 #define GPIO_OSPEEDER_OSPEEDR11_0             ((uint32_t)0x00400000)
04372 #define GPIO_OSPEEDER_OSPEEDR11_1             ((uint32_t)0x00800000)
04373
04374 #define GPIO_OSPEEDER_OSPEEDR12               ((uint32_t)0x03000000)
04375 #define GPIO_OSPEEDER_OSPEEDR12_0             ((uint32_t)0x01000000)
04376 #define GPIO_OSPEEDER_OSPEEDR12_1             ((uint32_t)0x02000000)
04377
04378 #define GPIO_OSPEEDER_OSPEEDR13               ((uint32_t)0x0C000000)
04379 #define GPIO_OSPEEDER_OSPEEDR13_0             ((uint32_t)0x04000000)
04380 #define GPIO_OSPEEDER_OSPEEDR13_1             ((uint32_t)0x08000000)
04381
04382 #define GPIO_OSPEEDER_OSPEEDR14               ((uint32_t)0x30000000)
04383 #define GPIO_OSPEEDER_OSPEEDR14_0             ((uint32_t)0x10000000)
04384 #define GPIO_OSPEEDER_OSPEEDR14_1             ((uint32_t)0x20000000)
04385
04386 #define GPIO_OSPEEDER_OSPEEDR15               ((uint32_t)0xC0000000)
04387 #define GPIO_OSPEEDER_OSPEEDR15_0             ((uint32_t)0x40000000)
04388 #define GPIO_OSPEEDER_OSPEEDR15_1             ((uint32_t)0x80000000)
04389
04390 /******************  Bits definition for GPIO_PUPDR register  *****************/
04391 #define GPIO_PUPDR_PUPDR0                     ((uint32_t)0x00000003)
04392 #define GPIO_PUPDR_PUPDR0_0                   ((uint32_t)0x00000001)
04393 #define GPIO_PUPDR_PUPDR0_1                   ((uint32_t)0x00000002)
04394
04395 #define GPIO_PUPDR_PUPDR1                     ((uint32_t)0x0000000C)
04396 #define GPIO_PUPDR_PUPDR1_0                   ((uint32_t)0x00000004)
04397 #define GPIO_PUPDR_PUPDR1_1                   ((uint32_t)0x00000008)
04398
04399 #define GPIO_PUPDR_PUPDR2                     ((uint32_t)0x00000030)
04400 #define GPIO_PUPDR_PUPDR2_0                   ((uint32_t)0x00000010)
04401 #define GPIO_PUPDR_PUPDR2_1                   ((uint32_t)0x00000020)
04402
04403 #define GPIO_PUPDR_PUPDR3                     ((uint32_t)0x000000C0)
04404 #define GPIO_PUPDR_PUPDR3_0                   ((uint32_t)0x00000040)
04405 #define GPIO_PUPDR_PUPDR3_1                   ((uint32_t)0x00000080)
04406
04407 #define GPIO_PUPDR_PUPDR4                     ((uint32_t)0x00000300)
04408 #define GPIO_PUPDR_PUPDR4_0                   ((uint32_t)0x00000100)
04409 #define GPIO_PUPDR_PUPDR4_1                   ((uint32_t)0x00000200)
04410
04411 #define GPIO_PUPDR_PUPDR5                     ((uint32_t)0x00000C00)
04412 #define GPIO_PUPDR_PUPDR5_0                   ((uint32_t)0x00000400)
04413 #define GPIO_PUPDR_PUPDR5_1                   ((uint32_t)0x00000800)
04414
04415 #define GPIO_PUPDR_PUPDR6                     ((uint32_t)0x00003000)
04416 #define GPIO_PUPDR_PUPDR6_0                   ((uint32_t)0x00001000)
04417 #define GPIO_PUPDR_PUPDR6_1                   ((uint32_t)0x00002000)
04418
04419 #define GPIO_PUPDR_PUPDR7                     ((uint32_t)0x0000C000)
04420 #define GPIO_PUPDR_PUPDR7_0                   ((uint32_t)0x00004000)
04421 #define GPIO_PUPDR_PUPDR7_1                   ((uint32_t)0x00008000)
04422
04423 #define GPIO_PUPDR_PUPDR8                     ((uint32_t)0x00030000)
04424 #define GPIO_PUPDR_PUPDR8_0                   ((uint32_t)0x00010000)
04425 #define GPIO_PUPDR_PUPDR8_1                   ((uint32_t)0x00020000)
04426
04427 #define GPIO_PUPDR_PUPDR9                     ((uint32_t)0x000C0000)
04428 #define GPIO_PUPDR_PUPDR9_0                   ((uint32_t)0x00040000)
04429 #define GPIO_PUPDR_PUPDR9_1                   ((uint32_t)0x00080000)
04430
04431 #define GPIO_PUPDR_PUPDR10                    ((uint32_t)0x00300000)
04432 #define GPIO_PUPDR_PUPDR10_0                  ((uint32_t)0x00100000)
04433 #define GPIO_PUPDR_PUPDR10_1                  ((uint32_t)0x00200000)
04434
```

```
04435 #define GPIO_PUPDR_PUPDR11                  ((uint32_t)0x00C00000)
04436 #define GPIO_PUPDR_PUPDR11_0                ((uint32_t)0x00400000)
04437 #define GPIO_PUPDR_PUPDR11_1                ((uint32_t)0x00800000)
04438
04439 #define GPIO_PUPDR_PUPDR12                  ((uint32_t)0x03000000)
04440 #define GPIO_PUPDR_PUPDR12_0                ((uint32_t)0x01000000)
04441 #define GPIO_PUPDR_PUPDR12_1                ((uint32_t)0x02000000)
04442
04443 #define GPIO_PUPDR_PUPDR13                  ((uint32_t)0x0C000000)
04444 #define GPIO_PUPDR_PUPDR13_0                ((uint32_t)0x04000000)
04445 #define GPIO_PUPDR_PUPDR13_1                ((uint32_t)0x08000000)
04446
04447 #define GPIO_PUPDR_PUPDR14                  ((uint32_t)0x30000000)
04448 #define GPIO_PUPDR_PUPDR14_0                ((uint32_t)0x10000000)
04449 #define GPIO_PUPDR_PUPDR14_1                ((uint32_t)0x20000000)
04450
04451 #define GPIO_PUPDR_PUPDR15                  ((uint32_t)0xC0000000)
04452 #define GPIO_PUPDR_PUPDR15_0                ((uint32_t)0x40000000)
04453 #define GPIO_PUPDR_PUPDR15_1                ((uint32_t)0x80000000)
04454
04455 /******************  Bits definition for GPIO_IDR register  *******************/
04456 #define GPIO_IDR_IDR_0                      ((uint32_t)0x00000001)
04457 #define GPIO_IDR_IDR_1                      ((uint32_t)0x00000002)
04458 #define GPIO_IDR_IDR_2                      ((uint32_t)0x00000004)
04459 #define GPIO_IDR_IDR_3                      ((uint32_t)0x00000008)
04460 #define GPIO_IDR_IDR_4                      ((uint32_t)0x00000010)
04461 #define GPIO_IDR_IDR_5                      ((uint32_t)0x00000020)
04462 #define GPIO_IDR_IDR_6                      ((uint32_t)0x00000040)
04463 #define GPIO_IDR_IDR_7                      ((uint32_t)0x00000080)
04464 #define GPIO_IDR_IDR_8                      ((uint32_t)0x00000100)
04465 #define GPIO_IDR_IDR_9                      ((uint32_t)0x00000200)
04466 #define GPIO_IDR_IDR_10                     ((uint32_t)0x00000400)
04467 #define GPIO_IDR_IDR_11                     ((uint32_t)0x00000800)
04468 #define GPIO_IDR_IDR_12                     ((uint32_t)0x00001000)
04469 #define GPIO_IDR_IDR_13                     ((uint32_t)0x00002000)
04470 #define GPIO_IDR_IDR_14                     ((uint32_t)0x00004000)
04471 #define GPIO_IDR_IDR_15                     ((uint32_t)0x00008000)
04472 /* Old GPIO_IDR register bits definition, maintained for legacy purpose */
04473 #define GPIO_OTYPER_IDR_0                   GPIO_IDR_IDR_0
04474 #define GPIO_OTYPER_IDR_1                   GPIO_IDR_IDR_1
04475 #define GPIO_OTYPER_IDR_2                   GPIO_IDR_IDR_2
04476 #define GPIO_OTYPER_IDR_3                   GPIO_IDR_IDR_3
04477 #define GPIO_OTYPER_IDR_4                   GPIO_IDR_IDR_4
04478 #define GPIO_OTYPER_IDR_5                   GPIO_IDR_IDR_5
04479 #define GPIO_OTYPER_IDR_6                   GPIO_IDR_IDR_6
04480 #define GPIO_OTYPER_IDR_7                   GPIO_IDR_IDR_7
04481 #define GPIO_OTYPER_IDR_8                   GPIO_IDR_IDR_8
04482 #define GPIO_OTYPER_IDR_9                   GPIO_IDR_IDR_9
04483 #define GPIO_OTYPER_IDR_10                  GPIO_IDR_IDR_10
04484 #define GPIO_OTYPER_IDR_11                  GPIO_IDR_IDR_11
04485 #define GPIO_OTYPER_IDR_12                  GPIO_IDR_IDR_12
04486 #define GPIO_OTYPER_IDR_13                  GPIO_IDR_IDR_13
04487 #define GPIO_OTYPER_IDR_14                  GPIO_IDR_IDR_14
04488 #define GPIO_OTYPER_IDR_15                  GPIO_IDR_IDR_15
04489
04490 /******************  Bits definition for GPIO_ODR register  *******************/
04491 #define GPIO_ODR_ODR_0                      ((uint32_t)0x00000001)
04492 #define GPIO_ODR_ODR_1                      ((uint32_t)0x00000002)
04493 #define GPIO_ODR_ODR_2                      ((uint32_t)0x00000004)
04494 #define GPIO_ODR_ODR_3                      ((uint32_t)0x00000008)
04495 #define GPIO_ODR_ODR_4                      ((uint32_t)0x00000010)
04496 #define GPIO_ODR_ODR_5                      ((uint32_t)0x00000020)
04497 #define GPIO_ODR_ODR_6                      ((uint32_t)0x00000040)
04498 #define GPIO_ODR_ODR_7                      ((uint32_t)0x00000080)
04499 #define GPIO_ODR_ODR_8                      ((uint32_t)0x00000100)
04500 #define GPIO_ODR_ODR_9                      ((uint32_t)0x00000200)
04501 #define GPIO_ODR_ODR_10                     ((uint32_t)0x00000400)
04502 #define GPIO_ODR_ODR_11                     ((uint32_t)0x00000800)
04503 #define GPIO_ODR_ODR_12                     ((uint32_t)0x00001000)
04504 #define GPIO_ODR_ODR_13                     ((uint32_t)0x00002000)
04505 #define GPIO_ODR_ODR_14                     ((uint32_t)0x00004000)
04506 #define GPIO_ODR_ODR_15                     ((uint32_t)0x00008000)
04507 /* Old GPIO_ODR register bits definition, maintained for legacy purpose */
04508 #define GPIO_OTYPER_ODR_0                   GPIO_ODR_ODR_0
04509 #define GPIO_OTYPER_ODR_1                   GPIO_ODR_ODR_1
04510 #define GPIO_OTYPER_ODR_2                   GPIO_ODR_ODR_2
04511 #define GPIO_OTYPER_ODR_3                   GPIO_ODR_ODR_3
04512 #define GPIO_OTYPER_ODR_4                   GPIO_ODR_ODR_4
04513 #define GPIO_OTYPER_ODR_5                   GPIO_ODR_ODR_5
04514 #define GPIO_OTYPER_ODR_6                   GPIO_ODR_ODR_6
04515 #define GPIO_OTYPER_ODR_7                   GPIO_ODR_ODR_7
04516 #define GPIO_OTYPER_ODR_8                   GPIO_ODR_ODR_8
04517 #define GPIO_OTYPER_ODR_9                   GPIO_ODR_ODR_9
04518 #define GPIO_OTYPER_ODR_10                  GPIO_ODR_ODR_10
04519 #define GPIO_OTYPER_ODR_11                  GPIO_ODR_ODR_11
04520 #define GPIO_OTYPER_ODR_12                  GPIO_ODR_ODR_12
04521 #define GPIO_OTYPER_ODR_13                  GPIO_ODR_ODR_13
```

```
04522 #define GPIO_OTYPER_ODR_14                  GPIO_ODR_ODR_14
04523 #define GPIO_OTYPER_ODR_15                  GPIO_ODR_ODR_15
04524
04525 /******************  Bits definition for GPIO_BSRR register  ******************/
04526 #define GPIO_BSRR_BS_0                      ((uint32_t)0x00000001)
04527 #define GPIO_BSRR_BS_1                      ((uint32_t)0x00000002)
04528 #define GPIO_BSRR_BS_2                      ((uint32_t)0x00000004)
04529 #define GPIO_BSRR_BS_3                      ((uint32_t)0x00000008)
04530 #define GPIO_BSRR_BS_4                      ((uint32_t)0x00000010)
04531 #define GPIO_BSRR_BS_5                      ((uint32_t)0x00000020)
04532 #define GPIO_BSRR_BS_6                      ((uint32_t)0x00000040)
04533 #define GPIO_BSRR_BS_7                      ((uint32_t)0x00000080)
04534 #define GPIO_BSRR_BS_8                      ((uint32_t)0x00000100)
04535 #define GPIO_BSRR_BS_9                      ((uint32_t)0x00000200)
04536 #define GPIO_BSRR_BS_10                     ((uint32_t)0x00000400)
04537 #define GPIO_BSRR_BS_11                     ((uint32_t)0x00000800)
04538 #define GPIO_BSRR_BS_12                     ((uint32_t)0x00001000)
04539 #define GPIO_BSRR_BS_13                     ((uint32_t)0x00002000)
04540 #define GPIO_BSRR_BS_14                     ((uint32_t)0x00004000)
04541 #define GPIO_BSRR_BS_15                     ((uint32_t)0x00008000)
04542 #define GPIO_BSRR_BR_0                      ((uint32_t)0x00010000)
04543 #define GPIO_BSRR_BR_1                      ((uint32_t)0x00020000)
04544 #define GPIO_BSRR_BR_2                      ((uint32_t)0x00040000)
04545 #define GPIO_BSRR_BR_3                      ((uint32_t)0x00080000)
04546 #define GPIO_BSRR_BR_4                      ((uint32_t)0x00100000)
04547 #define GPIO_BSRR_BR_5                      ((uint32_t)0x00200000)
04548 #define GPIO_BSRR_BR_6                      ((uint32_t)0x00400000)
04549 #define GPIO_BSRR_BR_7                      ((uint32_t)0x00800000)
04550 #define GPIO_BSRR_BR_8                      ((uint32_t)0x01000000)
04551 #define GPIO_BSRR_BR_9                      ((uint32_t)0x02000000)
04552 #define GPIO_BSRR_BR_10                     ((uint32_t)0x04000000)
04553 #define GPIO_BSRR_BR_11                     ((uint32_t)0x08000000)
04554 #define GPIO_BSRR_BR_12                     ((uint32_t)0x10000000)
04555 #define GPIO_BSRR_BR_13                     ((uint32_t)0x20000000)
04556 #define GPIO_BSRR_BR_14                     ((uint32_t)0x40000000)
04557 #define GPIO_BSRR_BR_15                     ((uint32_t)0x80000000)
04558
04559 /******************************************************************************/
04560 /*                                                                            */
04561 /*                                    HASH                                    */
04562 /*                                                                            */
04563 /******************************************************************************/
04564 /******************  Bits definition for HASH_CR register  ********************/
04565 #define HASH_CR_INIT                        ((uint32_t)0x00000004)
04566 #define HASH_CR_DMAE                        ((uint32_t)0x00000008)
04567 #define HASH_CR_DATATYPE                    ((uint32_t)0x00000030)
04568 #define HASH_CR_DATATYPE_0                  ((uint32_t)0x00000010)
04569 #define HASH_CR_DATATYPE_1                  ((uint32_t)0x00000020)
04570 #define HASH_CR_MODE                        ((uint32_t)0x00000040)
04571 #define HASH_CR_ALGO                        ((uint32_t)0x00000080)
04572 #define HASH_CR_NBW                         ((uint32_t)0x00000F00)
04573 #define HASH_CR_NBW_0                       ((uint32_t)0x00000100)
04574 #define HASH_CR_NBW_1                       ((uint32_t)0x00000200)
04575 #define HASH_CR_NBW_2                       ((uint32_t)0x00000400)
04576 #define HASH_CR_NBW_3                       ((uint32_t)0x00000800)
04577 #define HASH_CR_DINNE                       ((uint32_t)0x00001000)
04578 #define HASH_CR_LKEY                        ((uint32_t)0x00010000)
04579
04580 /******************  Bits definition for HASH_STR register  *******************/
04581 #define HASH_STR_NBW                        ((uint32_t)0x0000001F)
04582 #define HASH_STR_NBW_0                      ((uint32_t)0x00000001)
04583 #define HASH_STR_NBW_1                      ((uint32_t)0x00000002)
04584 #define HASH_STR_NBW_2                      ((uint32_t)0x00000004)
04585 #define HASH_STR_NBW_3                      ((uint32_t)0x00000008)
04586 #define HASH_STR_NBW_4                      ((uint32_t)0x00000010)
04587 #define HASH_STR_DCAL                       ((uint32_t)0x00000100)
04588
04589 /******************  Bits definition for HASH_IMR register  *******************/
04590 #define HASH_IMR_DINIM                      ((uint32_t)0x00000001)
04591 #define HASH_IMR_DCIM                       ((uint32_t)0x00000002)
04592
04593 /******************  Bits definition for HASH_SR register  ********************/
04594 #define HASH_SR_DINIS                       ((uint32_t)0x00000001)
04595 #define HASH_SR_DCIS                        ((uint32_t)0x00000002)
04596 #define HASH_SR_DMAS                        ((uint32_t)0x00000004)
04597 #define HASH_SR_BUSY                        ((uint32_t)0x00000008)
04598
04599 /******************************************************************************/
04600 /*                                                                            */
04601 /*                    Inter-integrated Circuit Interface                      */
04602 /*                                                                            */
04603 /******************************************************************************/
04604 /*******************  Bit definition for I2C_CR1 register  ********************/
04605 #define  I2C_CR1_PE                         ((uint16_t)0x0001)
04606 #define  I2C_CR1_SMBUS                      ((uint16_t)0x0002)
04607 #define  I2C_CR1_SMBTYPE                    ((uint16_t)0x0008)
04608 #define  I2C_CR1_ENARP                      ((uint16_t)0x0010)
```

```
04609 #define  I2C_CR1_ENPEC                       ((uint16_t)0x0020)
04610 #define  I2C_CR1_ENGC                        ((uint16_t)0x0040)
04611 #define  I2C_CR1_NOSTRETCH                   ((uint16_t)0x0080)
04612 #define  I2C_CR1_START                       ((uint16_t)0x0100)
04613 #define  I2C_CR1_STOP                        ((uint16_t)0x0200)
04614 #define  I2C_CR1_ACK                         ((uint16_t)0x0400)
04615 #define  I2C_CR1_POS                         ((uint16_t)0x0800)
04616 #define  I2C_CR1_PEC                         ((uint16_t)0x1000)
04617 #define  I2C_CR1_ALERT                       ((uint16_t)0x2000)
04618 #define  I2C_CR1_SWRST                       ((uint16_t)0x8000)
04620 /*******************  Bit definition for I2C_CR2 register  *******************/
04621 #define  I2C_CR2_FREQ                        ((uint16_t)0x003F)
04622 #define  I2C_CR2_FREQ_0                      ((uint16_t)0x0001)
04623 #define  I2C_CR2_FREQ_1                      ((uint16_t)0x0002)
04624 #define  I2C_CR2_FREQ_2                      ((uint16_t)0x0004)
04625 #define  I2C_CR2_FREQ_3                      ((uint16_t)0x0008)
04626 #define  I2C_CR2_FREQ_4                      ((uint16_t)0x0010)
04627 #define  I2C_CR2_FREQ_5                      ((uint16_t)0x0020)
04629 #define  I2C_CR2_ITERREN                     ((uint16_t)0x0100)
04630 #define  I2C_CR2_ITEVTEN                     ((uint16_t)0x0200)
04631 #define  I2C_CR2_ITBUFEN                     ((uint16_t)0x0400)
04632 #define  I2C_CR2_DMAEN                       ((uint16_t)0x0800)
04633 #define  I2C_CR2_LAST                        ((uint16_t)0x1000)
04635 /*******************  Bit definition for I2C_OAR1 register  *******************/
04636 #define  I2C_OAR1_ADD1_7                     ((uint16_t)0x00FE)
04637 #define  I2C_OAR1_ADD8_9                     ((uint16_t)0x0300)
04639 #define  I2C_OAR1_ADD0                       ((uint16_t)0x0001)
04640 #define  I2C_OAR1_ADD1                       ((uint16_t)0x0002)
04641 #define  I2C_OAR1_ADD2                       ((uint16_t)0x0004)
04642 #define  I2C_OAR1_ADD3                       ((uint16_t)0x0008)
04643 #define  I2C_OAR1_ADD4                       ((uint16_t)0x0010)
04644 #define  I2C_OAR1_ADD5                       ((uint16_t)0x0020)
04645 #define  I2C_OAR1_ADD6                       ((uint16_t)0x0040)
04646 #define  I2C_OAR1_ADD7                       ((uint16_t)0x0080)
04647 #define  I2C_OAR1_ADD8                       ((uint16_t)0x0100)
04648 #define  I2C_OAR1_ADD9                       ((uint16_t)0x0200)
04650 #define  I2C_OAR1_ADDMODE                    ((uint16_t)0x8000)
04652 /*******************  Bit definition for I2C_OAR2 register  *******************/
04653 #define  I2C_OAR2_ENDUAL                     ((uint8_t)0x01)
04654 #define  I2C_OAR2_ADD2                       ((uint8_t)0xFE)
04656 /*******************  Bit definition for I2C_DR register  ********************/
04657 #define  I2C_DR_DR                           ((uint8_t)0xFF)
04659 /*******************  Bit definition for I2C_SR1 register  *******************/
04660 #define  I2C_SR1_SB                          ((uint16_t)0x0001)
04661 #define  I2C_SR1_ADDR                        ((uint16_t)0x0002)
04662 #define  I2C_SR1_BTF                         ((uint16_t)0x0004)
04663 #define  I2C_SR1_ADD10                       ((uint16_t)0x0008)
04664 #define  I2C_SR1_STOPF                       ((uint16_t)0x0010)
04665 #define  I2C_SR1_RXNE                        ((uint16_t)0x0040)
04666 #define  I2C_SR1_TXE                         ((uint16_t)0x0080)
04667 #define  I2C_SR1_BERR                        ((uint16_t)0x0100)
04668 #define  I2C_SR1_ARLO                        ((uint16_t)0x0200)
04669 #define  I2C_SR1_AF                          ((uint16_t)0x0400)
04670 #define  I2C_SR1_OVR                         ((uint16_t)0x0800)
04671 #define  I2C_SR1_PECERR                      ((uint16_t)0x1000)
04672 #define  I2C_SR1_TIMEOUT                     ((uint16_t)0x4000)
04673 #define  I2C_SR1_SMBALERT                    ((uint16_t)0x8000)
04675 /*******************  Bit definition for I2C_SR2 register  *******************/
04676 #define  I2C_SR2_MSL                         ((uint16_t)0x0001)
04677 #define  I2C_SR2_BUSY                        ((uint16_t)0x0002)
04678 #define  I2C_SR2_TRA                         ((uint16_t)0x0004)
04679 #define  I2C_SR2_GENCALL                     ((uint16_t)0x0010)
04680 #define  I2C_SR2_SMBDEFAULT                  ((uint16_t)0x0020)
04681 #define  I2C_SR2_SMBHOST                     ((uint16_t)0x0040)
04682 #define  I2C_SR2_DUALF                       ((uint16_t)0x0080)
04683 #define  I2C_SR2_PEC                         ((uint16_t)0xFF00)
04685 /*******************  Bit definition for I2C_CCR register  *******************/
04686 #define  I2C_CCR_CCR                         ((uint16_t)0x0FFF)
04687 #define  I2C_CCR_DUTY                        ((uint16_t)0x4000)
04688 #define  I2C_CCR_FS                          ((uint16_t)0x8000)
04690 /*******************  Bit definition for I2C_TRISE register  *******************/
04691 #define  I2C_TRISE_TRISE                     ((uint8_t)0x3F)
04693 /******************************************************************************/
04694 /*                                                                            */
04695 /*                            Independent WATCHDOG                            */
04696 /*                                                                            */
04697 /******************************************************************************/
04698 /*******************  Bit definition for IWDG_KR register  *******************/
04699 #define  IWDG_KR_KEY                         ((uint16_t)0xFFFF)
04701 /*******************  Bit definition for IWDG_PR register  *******************/
04702 #define  IWDG_PR_PR                          ((uint8_t)0x07)
04703 #define  IWDG_PR_PR_0                        ((uint8_t)0x01)
04704 #define  IWDG_PR_PR_1                        ((uint8_t)0x02)
04705 #define  IWDG_PR_PR_2                        ((uint8_t)0x04)
04707 /*******************  Bit definition for IWDG_RLR register  *******************/
04708 #define  IWDG_RLR_RL                         ((uint16_t)0x0FFF)
04710 /*******************  Bit definition for IWDG_SR register  ********************/
```

```
04711 #define  IWDG_SR_PVU                         ((uint8_t)0x01)
04712 #define  IWDG_SR_RVU                         ((uint8_t)0x02)
04714 /******************************************************************************/
04715 /*                                                                            */
04716 /*                              Power Control                                 */
04717 /*                                                                            */
04718 /******************************************************************************/
04719 /********************  Bit definition for PWR_CR register  ********************/
04720 #define  PWR_CR_LPDS                         ((uint16_t)0x0001)
04721 #define  PWR_CR_PDDS                         ((uint16_t)0x0002)
04722 #define  PWR_CR_CWUF                         ((uint16_t)0x0004)
04723 #define  PWR_CR_CSBF                         ((uint16_t)0x0008)
04724 #define  PWR_CR_PVDE                         ((uint16_t)0x0010)
04726 #define  PWR_CR_PLS                          ((uint16_t)0x00E0)
04727 #define  PWR_CR_PLS_0                        ((uint16_t)0x0020)
04728 #define  PWR_CR_PLS_1                        ((uint16_t)0x0040)
04729 #define  PWR_CR_PLS_2                        ((uint16_t)0x0080)
04733 #define  PWR_CR_PLS_LEV0                     ((uint16_t)0x0000)
04734 #define  PWR_CR_PLS_LEV1                     ((uint16_t)0x0020)
04735 #define  PWR_CR_PLS_LEV2                     ((uint16_t)0x0040)
04736 #define  PWR_CR_PLS_LEV3                     ((uint16_t)0x0060)
04737 #define  PWR_CR_PLS_LEV4                     ((uint16_t)0x0080)
04738 #define  PWR_CR_PLS_LEV5                     ((uint16_t)0x00A0)
04739 #define  PWR_CR_PLS_LEV6                     ((uint16_t)0x00C0)
04740 #define  PWR_CR_PLS_LEV7                     ((uint16_t)0x00E0)
04742 #define  PWR_CR_DBP                          ((uint16_t)0x0100)
04743 #define  PWR_CR_FPDS                         ((uint16_t)0x0200)
04744 #define  PWR_CR_VOS                          ((uint16_t)0x4000)
04745 /* Legacy define */
04746 #define  PWR_CR_PMODE                        PWR_CR_VOS
04747
04748 /********************  Bit definition for PWR_CSR register  ********************/
04749 #define  PWR_CSR_WUF                         ((uint16_t)0x0001)
04750 #define  PWR_CSR_SBF                         ((uint16_t)0x0002)
04751 #define  PWR_CSR_PVDO                        ((uint16_t)0x0004)
04752 #define  PWR_CSR_BRR                         ((uint16_t)0x0008)
04753 #define  PWR_CSR_EWUP                        ((uint16_t)0x0100)
04754 #define  PWR_CSR_BRE                         ((uint16_t)0x0200)
04755 #define  PWR_CSR_VOSRDY                      ((uint16_t)0x4000)
04756 /* Legacy define */
04757 #define  PWR_CSR_REGRDY                      PWR_CSR_VOSRDY
04758
04759 /******************************************************************************/
04760 /*                                                                            */
04761 /*                         Reset and Clock Control                            */
04762 /*                                                                            */
04763 /******************************************************************************/
04764 /********************  Bit definition for RCC_CR register  ********************/
04765 #define  RCC_CR_HSION                        ((uint32_t)0x00000001)
04766 #define  RCC_CR_HSIRDY                       ((uint32_t)0x00000002)
04767
04768 #define  RCC_CR_HSITRIM                      ((uint32_t)0x000000F8)
04769 #define  RCC_CR_HSITRIM_0                    ((uint32_t)0x00000008)
04770 #define  RCC_CR_HSITRIM_1                    ((uint32_t)0x00000010)
04771 #define  RCC_CR_HSITRIM_2                    ((uint32_t)0x00000020)
04772 #define  RCC_CR_HSITRIM_3                    ((uint32_t)0x00000040)
04773 #define  RCC_CR_HSITRIM_4                    ((uint32_t)0x00000080)
04775 #define  RCC_CR_HSICAL                       ((uint32_t)0x0000FF00)
04776 #define  RCC_CR_HSICAL_0                     ((uint32_t)0x00000100)
04777 #define  RCC_CR_HSICAL_1                     ((uint32_t)0x00000200)
04778 #define  RCC_CR_HSICAL_2                     ((uint32_t)0x00000400)
04779 #define  RCC_CR_HSICAL_3                     ((uint32_t)0x00000800)
04780 #define  RCC_CR_HSICAL_4                     ((uint32_t)0x00001000)
04781 #define  RCC_CR_HSICAL_5                     ((uint32_t)0x00002000)
04782 #define  RCC_CR_HSICAL_6                     ((uint32_t)0x00004000)
04783 #define  RCC_CR_HSICAL_7                     ((uint32_t)0x00008000)
04785 #define  RCC_CR_HSEON                        ((uint32_t)0x00010000)
04786 #define  RCC_CR_HSERDY                       ((uint32_t)0x00020000)
04787 #define  RCC_CR_HSEBYP                       ((uint32_t)0x00040000)
04788 #define  RCC_CR_CSSON                        ((uint32_t)0x00080000)
04789 #define  RCC_CR_PLLON                        ((uint32_t)0x01000000)
04790 #define  RCC_CR_PLLRDY                       ((uint32_t)0x02000000)
04791 #define  RCC_CR_PLLI2SON                     ((uint32_t)0x04000000)
04792 #define  RCC_CR_PLLI2SRDY                    ((uint32_t)0x08000000)
04793
04794 /********************  Bit definition for RCC_PLLCFGR register  ***************/
04795 #define  RCC_PLLCFGR_PLLM                    ((uint32_t)0x0000003F)
04796 #define  RCC_PLLCFGR_PLLM_0                  ((uint32_t)0x00000001)
04797 #define  RCC_PLLCFGR_PLLM_1                  ((uint32_t)0x00000002)
04798 #define  RCC_PLLCFGR_PLLM_2                  ((uint32_t)0x00000004)
04799 #define  RCC_PLLCFGR_PLLM_3                  ((uint32_t)0x00000008)
04800 #define  RCC_PLLCFGR_PLLM_4                  ((uint32_t)0x00000010)
04801 #define  RCC_PLLCFGR_PLLM_5                  ((uint32_t)0x00000020)
04802
04803 #define  RCC_PLLCFGR_PLLN                     ((uint32_t)0x00007FC0)
04804 #define  RCC_PLLCFGR_PLLN_0                   ((uint32_t)0x00000040)
04805 #define  RCC_PLLCFGR_PLLN_1                   ((uint32_t)0x00000080)
```

```
04806 #define  RCC_PLLCFGR_PLLN_2                  ((uint32_t)0x00000100)
04807 #define  RCC_PLLCFGR_PLLN_3                  ((uint32_t)0x00000200)
04808 #define  RCC_PLLCFGR_PLLN_4                  ((uint32_t)0x00000400)
04809 #define  RCC_PLLCFGR_PLLN_5                  ((uint32_t)0x00000800)
04810 #define  RCC_PLLCFGR_PLLN_6                  ((uint32_t)0x00001000)
04811 #define  RCC_PLLCFGR_PLLN_7                  ((uint32_t)0x00002000)
04812 #define  RCC_PLLCFGR_PLLN_8                  ((uint32_t)0x00004000)
04813
04814 #define  RCC_PLLCFGR_PLLP                   ((uint32_t)0x00030000)
04815 #define  RCC_PLLCFGR_PLLP_0                 ((uint32_t)0x00010000)
04816 #define  RCC_PLLCFGR_PLLP_1                 ((uint32_t)0x00020000)
04817
04818 #define  RCC_PLLCFGR_PLLSRC                 ((uint32_t)0x00400000)
04819 #define  RCC_PLLCFGR_PLLSRC_HSE             ((uint32_t)0x00400000)
04820 #define  RCC_PLLCFGR_PLLSRC_HSI             ((uint32_t)0x00000000)
04821
04822 #define  RCC_PLLCFGR_PLLQ                   ((uint32_t)0x0F000000)
04823 #define  RCC_PLLCFGR_PLLQ_0                 ((uint32_t)0x01000000)
04824 #define  RCC_PLLCFGR_PLLQ_1                 ((uint32_t)0x02000000)
04825 #define  RCC_PLLCFGR_PLLQ_2                 ((uint32_t)0x04000000)
04826 #define  RCC_PLLCFGR_PLLQ_3                 ((uint32_t)0x08000000)
04827
04828 /********************  Bit definition for RCC_CFGR register  ******************/
04830 #define  RCC_CFGR_SW                        ((uint32_t)0x00000003)
04831 #define  RCC_CFGR_SW_0                      ((uint32_t)0x00000001)
04832 #define  RCC_CFGR_SW_1                      ((uint32_t)0x00000002)
04834 #define  RCC_CFGR_SW_HSI                    ((uint32_t)0x00000000)
04835 #define  RCC_CFGR_SW_HSE                    ((uint32_t)0x00000001)
04836 #define  RCC_CFGR_SW_PLL                    ((uint32_t)0x00000002)
04839 #define  RCC_CFGR_SWS                       ((uint32_t)0x0000000C)
04840 #define  RCC_CFGR_SWS_0                     ((uint32_t)0x00000004)
04841 #define  RCC_CFGR_SWS_1                     ((uint32_t)0x00000008)
04843 #define  RCC_CFGR_SWS_HSI                   ((uint32_t)0x00000000)
04844 #define  RCC_CFGR_SWS_HSE                   ((uint32_t)0x00000004)
04845 #define  RCC_CFGR_SWS_PLL                   ((uint32_t)0x00000008)
04848 #define  RCC_CFGR_HPRE                      ((uint32_t)0x000000F0)
04849 #define  RCC_CFGR_HPRE_0                    ((uint32_t)0x00000010)
04850 #define  RCC_CFGR_HPRE_1                    ((uint32_t)0x00000020)
04851 #define  RCC_CFGR_HPRE_2                    ((uint32_t)0x00000040)
04852 #define  RCC_CFGR_HPRE_3                    ((uint32_t)0x00000080)
04854 #define  RCC_CFGR_HPRE_DIV1                 ((uint32_t)0x00000000)
04855 #define  RCC_CFGR_HPRE_DIV2                 ((uint32_t)0x00000080)
04856 #define  RCC_CFGR_HPRE_DIV4                 ((uint32_t)0x00000090)
04857 #define  RCC_CFGR_HPRE_DIV8                 ((uint32_t)0x000000A0)
04858 #define  RCC_CFGR_HPRE_DIV16                ((uint32_t)0x000000B0)
04859 #define  RCC_CFGR_HPRE_DIV64                ((uint32_t)0x000000C0)
04860 #define  RCC_CFGR_HPRE_DIV128               ((uint32_t)0x000000D0)
04861 #define  RCC_CFGR_HPRE_DIV256               ((uint32_t)0x000000E0)
04862 #define  RCC_CFGR_HPRE_DIV512               ((uint32_t)0x000000F0)
04865 #define  RCC_CFGR_PPRE1                     ((uint32_t)0x00001C00)
04866 #define  RCC_CFGR_PPRE1_0                   ((uint32_t)0x00000400)
04867 #define  RCC_CFGR_PPRE1_1                   ((uint32_t)0x00000800)
04868 #define  RCC_CFGR_PPRE1_2                   ((uint32_t)0x00001000)
04870 #define  RCC_CFGR_PPRE1_DIV1                ((uint32_t)0x00000000)
04871 #define  RCC_CFGR_PPRE1_DIV2                ((uint32_t)0x00001000)
04872 #define  RCC_CFGR_PPRE1_DIV4                ((uint32_t)0x00001400)
04873 #define  RCC_CFGR_PPRE1_DIV8                ((uint32_t)0x00001800)
04874 #define  RCC_CFGR_PPRE1_DIV16               ((uint32_t)0x00001C00)
04877 #define  RCC_CFGR_PPRE2                     ((uint32_t)0x0000E000)
04878 #define  RCC_CFGR_PPRE2_0                   ((uint32_t)0x00002000)
04879 #define  RCC_CFGR_PPRE2_1                   ((uint32_t)0x00004000)
04880 #define  RCC_CFGR_PPRE2_2                   ((uint32_t)0x00008000)
04882 #define  RCC_CFGR_PPRE2_DIV1                ((uint32_t)0x00000000)
04883 #define  RCC_CFGR_PPRE2_DIV2                ((uint32_t)0x00008000)
04884 #define  RCC_CFGR_PPRE2_DIV4                ((uint32_t)0x0000A000)
04885 #define  RCC_CFGR_PPRE2_DIV8                ((uint32_t)0x0000C000)
04886 #define  RCC_CFGR_PPRE2_DIV16               ((uint32_t)0x0000E000)
04889 #define  RCC_CFGR_RTCPRE                    ((uint32_t)0x001F0000)
04890 #define  RCC_CFGR_RTCPRE_0                  ((uint32_t)0x00010000)
04891 #define  RCC_CFGR_RTCPRE_1                  ((uint32_t)0x00020000)
04892 #define  RCC_CFGR_RTCPRE_2                  ((uint32_t)0x00040000)
04893 #define  RCC_CFGR_RTCPRE_3                  ((uint32_t)0x00080000)
04894 #define  RCC_CFGR_RTCPRE_4                  ((uint32_t)0x00100000)
04895
04897 #define  RCC_CFGR_MCO1                      ((uint32_t)0x00600000)
04898 #define  RCC_CFGR_MCO1_0                    ((uint32_t)0x00200000)
04899 #define  RCC_CFGR_MCO1_1                    ((uint32_t)0x00400000)
04900
04901 #define  RCC_CFGR_I2SSRC                    ((uint32_t)0x00800000)
04902
04903 #define  RCC_CFGR_MCO1PRE                   ((uint32_t)0x07000000)
04904 #define  RCC_CFGR_MCO1PRE_0                 ((uint32_t)0x01000000)
04905 #define  RCC_CFGR_MCO1PRE_1                 ((uint32_t)0x02000000)
04906 #define  RCC_CFGR_MCO1PRE_2                 ((uint32_t)0x04000000)
04907
04908 #define  RCC_CFGR_MCO2PRE                   ((uint32_t)0x38000000)
04909 #define  RCC_CFGR_MCO2PRE_0                 ((uint32_t)0x08000000)
```

```
04910 #define  RCC_CFGR_MCO2PRE_1                  ((uint32_t)0x10000000)
04911 #define  RCC_CFGR_MCO2PRE_2                  ((uint32_t)0x20000000)
04912
04913 #define  RCC_CFGR_MCO2                       ((uint32_t)0xC0000000)
04914 #define  RCC_CFGR_MCO2_0                     ((uint32_t)0x40000000)
04915 #define  RCC_CFGR_MCO2_1                     ((uint32_t)0x80000000)
04916
04917 /********************  Bit definition for RCC_CIR register  ******************/
04918 #define  RCC_CIR_LSIRDYF                     ((uint32_t)0x00000001)
04919 #define  RCC_CIR_LSERDYF                     ((uint32_t)0x00000002)
04920 #define  RCC_CIR_HSIRDYF                     ((uint32_t)0x00000004)
04921 #define  RCC_CIR_HSERDYF                     ((uint32_t)0x00000008)
04922 #define  RCC_CIR_PLLRDYF                     ((uint32_t)0x00000010)
04923 #define  RCC_CIR_PLLI2SRDYF                  ((uint32_t)0x00000020)
04924 #define  RCC_CIR_CSSF                        ((uint32_t)0x00000080)
04925 #define  RCC_CIR_LSIRDYIE                    ((uint32_t)0x00000100)
04926 #define  RCC_CIR_LSERDYIE                    ((uint32_t)0x00000200)
04927 #define  RCC_CIR_HSIRDYIE                    ((uint32_t)0x00000400)
04928 #define  RCC_CIR_HSERDYIE                    ((uint32_t)0x00000800)
04929 #define  RCC_CIR_PLLRDYIE                    ((uint32_t)0x00001000)
04930 #define  RCC_CIR_PLLI2SRDYIE                 ((uint32_t)0x00002000)
04931 #define  RCC_CIR_LSIRDYC                     ((uint32_t)0x00010000)
04932 #define  RCC_CIR_LSERDYC                     ((uint32_t)0x00020000)
04933 #define  RCC_CIR_HSIRDYC                     ((uint32_t)0x00040000)
04934 #define  RCC_CIR_HSERDYC                     ((uint32_t)0x00080000)
04935 #define  RCC_CIR_PLLRDYC                     ((uint32_t)0x00100000)
04936 #define  RCC_CIR_PLLI2SRDYC                  ((uint32_t)0x00200000)
04937 #define  RCC_CIR_CSSC                        ((uint32_t)0x00800000)
04938
04939 /********************  Bit definition for RCC_AHB1RSTR register  **************/
04940 #define  RCC_AHB1RSTR_GPIOARST               ((uint32_t)0x00000001)
04941 #define  RCC_AHB1RSTR_GPIOBRST               ((uint32_t)0x00000002)
04942 #define  RCC_AHB1RSTR_GPIOCRST               ((uint32_t)0x00000004)
04943 #define  RCC_AHB1RSTR_GPIODRST               ((uint32_t)0x00000008)
04944 #define  RCC_AHB1RSTR_GPIOERST               ((uint32_t)0x00000010)
04945 #define  RCC_AHB1RSTR_GPIOFRST               ((uint32_t)0x00000020)
04946 #define  RCC_AHB1RSTR_GPIOGRST               ((uint32_t)0x00000040)
04947 #define  RCC_AHB1RSTR_GPIOHRST               ((uint32_t)0x00000080)
04948 #define  RCC_AHB1RSTR_GPIOIRST               ((uint32_t)0x00000100)
04949 #define  RCC_AHB1RSTR_CRCRST                 ((uint32_t)0x00001000)
04950 #define  RCC_AHB1RSTR_DMA1RST                ((uint32_t)0x00200000)
04951 #define  RCC_AHB1RSTR_DMA2RST                ((uint32_t)0x00400000)
04952 #define  RCC_AHB1RSTR_ETHMACRST              ((uint32_t)0x02000000)
04953 #define  RCC_AHB1RSTR_OTGHRST                ((uint32_t)0x10000000)
04954
04955 /********************  Bit definition for RCC_AHB2RSTR register  **************/
04956 #define  RCC_AHB2RSTR_DCMIRST                ((uint32_t)0x00000001)
04957 #define  RCC_AHB2RSTR_CRYPRST                ((uint32_t)0x00000010)
04958 #define  RCC_AHB2RSTR_HSAHRST                ((uint32_t)0x00000020)
04959 #define  RCC_AHB2RSTR_RNGRST                 ((uint32_t)0x00000040)
04960 #define  RCC_AHB2RSTR_OTGFSRST               ((uint32_t)0x00000080)
04961
04962 /********************  Bit definition for RCC_AHB3RSTR register  **************/
04963 #define  RCC_AHB3RSTR_FSMCRST                ((uint32_t)0x00000001)
04964
04965 /********************  Bit definition for RCC_APB1RSTR register  **************/
04966 #define  RCC_APB1RSTR_TIM2RST                ((uint32_t)0x00000001)
04967 #define  RCC_APB1RSTR_TIM3RST                ((uint32_t)0x00000002)
04968 #define  RCC_APB1RSTR_TIM4RST                ((uint32_t)0x00000004)
04969 #define  RCC_APB1RSTR_TIM5RST                ((uint32_t)0x00000008)
04970 #define  RCC_APB1RSTR_TIM6RST                ((uint32_t)0x00000010)
04971 #define  RCC_APB1RSTR_TIM7RST                ((uint32_t)0x00000020)
04972 #define  RCC_APB1RSTR_TIM12RST               ((uint32_t)0x00000040)
04973 #define  RCC_APB1RSTR_TIM13RST               ((uint32_t)0x00000080)
04974 #define  RCC_APB1RSTR_TIM14RST               ((uint32_t)0x00000100)
04975 #define  RCC_APB1RSTR_WWDGEN                 ((uint32_t)0x00000800)
04976 #define  RCC_APB1RSTR_SPI2RST                ((uint32_t)0x00008000)
04977 #define  RCC_APB1RSTR_SPI3RST                ((uint32_t)0x00010000)
04978 #define  RCC_APB1RSTR_USART2RST              ((uint32_t)0x00020000)
04979 #define  RCC_APB1RSTR_USART3RST              ((uint32_t)0x00040000)
04980 #define  RCC_APB1RSTR_UART4RST               ((uint32_t)0x00080000)
04981 #define  RCC_APB1RSTR_UART5RST               ((uint32_t)0x00100000)
04982 #define  RCC_APB1RSTR_I2C1RST                ((uint32_t)0x00200000)
04983 #define  RCC_APB1RSTR_I2C2RST                ((uint32_t)0x00400000)
04984 #define  RCC_APB1RSTR_I2C3RST                ((uint32_t)0x00800000)
04985 #define  RCC_APB1RSTR_CAN1RST                ((uint32_t)0x02000000)
04986 #define  RCC_APB1RSTR_CAN2RST                ((uint32_t)0x04000000)
04987 #define  RCC_APB1RSTR_PWRRST                 ((uint32_t)0x10000000)
04988 #define  RCC_APB1RSTR_DACRST                 ((uint32_t)0x20000000)
04989
04990 /********************  Bit definition for RCC_APB2RSTR register  **************/
04991 #define  RCC_APB2RSTR_TIM1RST                ((uint32_t)0x00000001)
04992 #define  RCC_APB2RSTR_TIM8RST                ((uint32_t)0x00000002)
04993 #define  RCC_APB2RSTR_USART1RST              ((uint32_t)0x00000010)
04994 #define  RCC_APB2RSTR_USART6RST              ((uint32_t)0x00000020)
04995 #define  RCC_APB2RSTR_ADCRST                 ((uint32_t)0x00000100)
04996 #define  RCC_APB2RSTR_SDIORST                ((uint32_t)0x00000800)
```

```
04997 #define  RCC_APB2RSTR_SPI1RST               ((uint32_t)0x00001000)
04998 #define  RCC_APB2RSTR_SYSCFGRST             ((uint32_t)0x00004000)
04999 #define  RCC_APB2RSTR_TIM9RST               ((uint32_t)0x00010000)
05000 #define  RCC_APB2RSTR_TIM10RST              ((uint32_t)0x00020000)
05001 #define  RCC_APB2RSTR_TIM11RST              ((uint32_t)0x00040000)
05002 /* Old SPI1RST bit definition, maintained for legacy purpose */
05003 #define  RCC_APB2RSTR_SPI1                  RCC_APB2RSTR_SPI1RST
05004
05005 /*******************  Bit definition for RCC_AHB1ENR register  ***************/
05006 #define  RCC_AHB1ENR_GPIOAEN                ((uint32_t)0x00000001)
05007 #define  RCC_AHB1ENR_GPIOBEN                ((uint32_t)0x00000002)
05008 #define  RCC_AHB1ENR_GPIOCEN                ((uint32_t)0x00000004)
05009 #define  RCC_AHB1ENR_GPIODEN                ((uint32_t)0x00000008)
05010 #define  RCC_AHB1ENR_GPIOEEN                ((uint32_t)0x00000010)
05011 #define  RCC_AHB1ENR_GPIOFEN                ((uint32_t)0x00000020)
05012 #define  RCC_AHB1ENR_GPIOGEN                ((uint32_t)0x00000040)
05013 #define  RCC_AHB1ENR_GPIOHEN                ((uint32_t)0x00000080)
05014 #define  RCC_AHB1ENR_GPIOIEN                ((uint32_t)0x00000100)
05015 #define  RCC_AHB1ENR_CRCEN                  ((uint32_t)0x00001000)
05016 #define  RCC_AHB1ENR_BKPSRAMEN              ((uint32_t)0x00040000)
05017 #define  RCC_AHB1ENR_CCMDATARAMEN           ((uint32_t)0x00100000)
05018 #define  RCC_AHB1ENR_DMA1EN                 ((uint32_t)0x00200000)
05019 #define  RCC_AHB1ENR_DMA2EN                 ((uint32_t)0x00400000)
05020 #define  RCC_AHB1ENR_ETHMACEN               ((uint32_t)0x02000000)
05021 #define  RCC_AHB1ENR_ETHMACTXEN             ((uint32_t)0x04000000)
05022 #define  RCC_AHB1ENR_ETHMACRXEN             ((uint32_t)0x08000000)
05023 #define  RCC_AHB1ENR_ETHMACPTPEN            ((uint32_t)0x10000000)
05024 #define  RCC_AHB1ENR_OTGHSEN                ((uint32_t)0x20000000)
05025 #define  RCC_AHB1ENR_OTGHSULPIEN            ((uint32_t)0x40000000)
05026
05027 /*******************  Bit definition for RCC_AHB2ENR register  ***************/
05028 #define  RCC_AHB2ENR_DCMIEN                 ((uint32_t)0x00000001)
05029 #define  RCC_AHB2ENR_CRYPEN                 ((uint32_t)0x00000010)
05030 #define  RCC_AHB2ENR_HASHEN                 ((uint32_t)0x00000020)
05031 #define  RCC_AHB2ENR_RNGEN                  ((uint32_t)0x00000040)
05032 #define  RCC_AHB2ENR_OTGFSEN                ((uint32_t)0x00000080)
05033
05034 /*******************  Bit definition for RCC_AHB3ENR register  ***************/
05035 #define  RCC_AHB3ENR_FSMCEN                 ((uint32_t)0x00000001)
05036
05037 /*******************  Bit definition for RCC_APB1ENR register  ***************/
05038 #define  RCC_APB1ENR_TIM2EN                 ((uint32_t)0x00000001)
05039 #define  RCC_APB1ENR_TIM3EN                 ((uint32_t)0x00000002)
05040 #define  RCC_APB1ENR_TIM4EN                 ((uint32_t)0x00000004)
05041 #define  RCC_APB1ENR_TIM5EN                 ((uint32_t)0x00000008)
05042 #define  RCC_APB1ENR_TIM6EN                 ((uint32_t)0x00000010)
05043 #define  RCC_APB1ENR_TIM7EN                 ((uint32_t)0x00000020)
05044 #define  RCC_APB1ENR_TIM12EN                ((uint32_t)0x00000040)
05045 #define  RCC_APB1ENR_TIM13EN                ((uint32_t)0x00000080)
05046 #define  RCC_APB1ENR_TIM14EN                ((uint32_t)0x00000100)
05047 #define  RCC_APB1ENR_WWDGEN                 ((uint32_t)0x00000800)
05048 #define  RCC_APB1ENR_SPI2EN                 ((uint32_t)0x00004000)
05049 #define  RCC_APB1ENR_SPI3EN                 ((uint32_t)0x00008000)
05050 #define  RCC_APB1ENR_USART2EN               ((uint32_t)0x00020000)
05051 #define  RCC_APB1ENR_USART3EN               ((uint32_t)0x00040000)
05052 #define  RCC_APB1ENR_UART4EN                ((uint32_t)0x00080000)
05053 #define  RCC_APB1ENR_UART5EN                ((uint32_t)0x00100000)
05054 #define  RCC_APB1ENR_I2C1EN                 ((uint32_t)0x00200000)
05055 #define  RCC_APB1ENR_I2C2EN                 ((uint32_t)0x00400000)
05056 #define  RCC_APB1ENR_I2C3EN                 ((uint32_t)0x00800000)
05057 #define  RCC_APB1ENR_CAN1EN                 ((uint32_t)0x02000000)
05058 #define  RCC_APB1ENR_CAN2EN                 ((uint32_t)0x04000000)
05059 #define  RCC_APB1ENR_PWREN                  ((uint32_t)0x10000000)
05060 #define  RCC_APB1ENR_DACEN                  ((uint32_t)0x20000000)
05061
05062 /*******************  Bit definition for RCC_APB2ENR register  ***************/
05063 #define  RCC_APB2ENR_TIM1EN                 ((uint32_t)0x00000001)
05064 #define  RCC_APB2ENR_TIM8EN                 ((uint32_t)0x00000002)
05065 #define  RCC_APB2ENR_USART1EN               ((uint32_t)0x00000010)
05066 #define  RCC_APB2ENR_USART6EN               ((uint32_t)0x00000020)
05067 #define  RCC_APB2ENR_ADC1EN                 ((uint32_t)0x00000100)
05068 #define  RCC_APB2ENR_ADC2EN                 ((uint32_t)0x00000200)
05069 #define  RCC_APB2ENR_ADC3EN                 ((uint32_t)0x00000400)
05070 #define  RCC_APB2ENR_SDIOEN                 ((uint32_t)0x00000800)
05071 #define  RCC_APB2ENR_SPI1EN                 ((uint32_t)0x00001000)
05072 #define  RCC_APB2ENR_SYSCFGEN               ((uint32_t)0x00004000)
05073 #define  RCC_APB2ENR_TIM11EN                ((uint32_t)0x00040000)
05074 #define  RCC_APB2ENR_TIM10EN                ((uint32_t)0x00020000)
05075 #define  RCC_APB2ENR_TIM9EN                 ((uint32_t)0x00010000)
05076
05077 /*******************  Bit definition for RCC_AHB1LPENR register  *************/
05078 #define  RCC_AHB1LPENR_GPIOALPEN            ((uint32_t)0x00000001)
05079 #define  RCC_AHB1LPENR_GPIOBLPEN            ((uint32_t)0x00000002)
05080 #define  RCC_AHB1LPENR_GPIOCLPEN            ((uint32_t)0x00000004)
05081 #define  RCC_AHB1LPENR_GPIODLPEN            ((uint32_t)0x00000008)
05082 #define  RCC_AHB1LPENR_GPIOELPEN            ((uint32_t)0x00000010)
05083 #define  RCC_AHB1LPENR_GPIOFLPEN            ((uint32_t)0x00000020)
```

```
05084 #define  RCC_AHB1LPENR_GPIOGLPEN             ((uint32_t)0x00000040)
05085 #define  RCC_AHB1LPENR_GPIOHLPEN             ((uint32_t)0x00000080)
05086 #define  RCC_AHB1LPENR_GPIOILPEN             ((uint32_t)0x00000100)
05087 #define  RCC_AHB1LPENR_CRCLPEN               ((uint32_t)0x00001000)
05088 #define  RCC_AHB1LPENR_FLITFLPEN             ((uint32_t)0x00008000)
05089 #define  RCC_AHB1LPENR_SRAM1LPEN             ((uint32_t)0x00010000)
05090 #define  RCC_AHB1LPENR_SRAM2LPEN             ((uint32_t)0x00020000)
05091 #define  RCC_AHB1LPENR_BKPSRAMLPEN           ((uint32_t)0x00040000)
05092 #define  RCC_AHB1LPENR_DMA1LPEN              ((uint32_t)0x00200000)
05093 #define  RCC_AHB1LPENR_DMA2LPEN              ((uint32_t)0x00400000)
05094 #define  RCC_AHB1LPENR_ETHMACLPEN            ((uint32_t)0x02000000)
05095 #define  RCC_AHB1LPENR_ETHMACTXLPEN          ((uint32_t)0x04000000)
05096 #define  RCC_AHB1LPENR_ETHMACRXLPEN          ((uint32_t)0x08000000)
05097 #define  RCC_AHB1LPENR_ETHMACPTPLPEN         ((uint32_t)0x10000000)
05098 #define  RCC_AHB1LPENR_OTGHSLPEN             ((uint32_t)0x20000000)
05099 #define  RCC_AHB1LPENR_OTGHSULPILPEN         ((uint32_t)0x40000000)
05100
05101 /********************  Bit definition for RCC_AHB2LPENR register  *************/
05102 #define  RCC_AHB2LPENR_DCMILPEN              ((uint32_t)0x00000001)
05103 #define  RCC_AHB2LPENR_CRYPLPEN              ((uint32_t)0x00000010)
05104 #define  RCC_AHB2LPENR_HASHLPEN              ((uint32_t)0x00000020)
05105 #define  RCC_AHB2LPENR_RNGLPEN               ((uint32_t)0x00000040)
05106 #define  RCC_AHB2LPENR_OTGFSLPEN             ((uint32_t)0x00000080)
05107
05108 /********************  Bit definition for RCC_AHB3LPENR register  *************/
05109 #define  RCC_AHB3LPENR_FSMCLPEN              ((uint32_t)0x00000001)
05110
05111 /********************  Bit definition for RCC_APB1LPENR register  *************/
05112 #define  RCC_APB1LPENR_TIM2LPEN              ((uint32_t)0x00000001)
05113 #define  RCC_APB1LPENR_TIM3LPEN              ((uint32_t)0x00000002)
05114 #define  RCC_APB1LPENR_TIM4LPEN              ((uint32_t)0x00000004)
05115 #define  RCC_APB1LPENR_TIM5LPEN              ((uint32_t)0x00000008)
05116 #define  RCC_APB1LPENR_TIM6LPEN              ((uint32_t)0x00000010)
05117 #define  RCC_APB1LPENR_TIM7LPEN              ((uint32_t)0x00000020)
05118 #define  RCC_APB1LPENR_TIM12LPEN             ((uint32_t)0x00000040)
05119 #define  RCC_APB1LPENR_TIM13LPEN             ((uint32_t)0x00000080)
05120 #define  RCC_APB1LPENR_TIM14LPEN             ((uint32_t)0x00000100)
05121 #define  RCC_APB1LPENR_WWDGLPEN              ((uint32_t)0x00000800)
05122 #define  RCC_APB1LPENR_SPI2LPEN              ((uint32_t)0x00004000)
05123 #define  RCC_APB1LPENR_SPI3LPEN              ((uint32_t)0x00008000)
05124 #define  RCC_APB1LPENR_USART2LPEN            ((uint32_t)0x00020000)
05125 #define  RCC_APB1LPENR_USART3LPEN            ((uint32_t)0x00040000)
05126 #define  RCC_APB1LPENR_UART4LPEN             ((uint32_t)0x00080000)
05127 #define  RCC_APB1LPENR_UART5LPEN             ((uint32_t)0x00100000)
05128 #define  RCC_APB1LPENR_I2C1LPEN              ((uint32_t)0x00200000)
05129 #define  RCC_APB1LPENR_I2C2LPEN              ((uint32_t)0x00400000)
05130 #define  RCC_APB1LPENR_I2C3LPEN              ((uint32_t)0x00800000)
05131 #define  RCC_APB1LPENR_CAN1LPEN              ((uint32_t)0x02000000)
05132 #define  RCC_APB1LPENR_CAN2LPEN              ((uint32_t)0x04000000)
05133 #define  RCC_APB1LPENR_PWRLPEN               ((uint32_t)0x10000000)
05134 #define  RCC_APB1LPENR_DACLPEN               ((uint32_t)0x20000000)
05135
05136 /********************  Bit definition for RCC_APB2LPENR register  *************/
05137 #define  RCC_APB2LPENR_TIM1LPEN              ((uint32_t)0x00000001)
05138 #define  RCC_APB2LPENR_TIM8LPEN              ((uint32_t)0x00000002)
05139 #define  RCC_APB2LPENR_USART1LPEN            ((uint32_t)0x00000010)
05140 #define  RCC_APB2LPENR_USART6LPEN            ((uint32_t)0x00000020)
05141 #define  RCC_APB2LPENR_ADC1LPEN              ((uint32_t)0x00000100)
05142 #define  RCC_APB2LPENR_ADC2PEN               ((uint32_t)0x00000200)
05143 #define  RCC_APB2LPENR_ADC3LPEN              ((uint32_t)0x00000400)
05144 #define  RCC_APB2LPENR_SDIOLPEN              ((uint32_t)0x00000800)
05145 #define  RCC_APB2LPENR_SPI1LPEN              ((uint32_t)0x00001000)
05146 #define  RCC_APB2LPENR_SYSCFGLPEN            ((uint32_t)0x00004000)
05147 #define  RCC_APB2LPENR_TIM9LPEN              ((uint32_t)0x00010000)
05148 #define  RCC_APB2LPENR_TIM10LPEN             ((uint32_t)0x00020000)
05149 #define  RCC_APB2LPENR_TIM11LPEN             ((uint32_t)0x00040000)
05150
05151 /********************  Bit definition for RCC_BDCR register  ******************/
05152 #define  RCC_BDCR_LSEON                      ((uint32_t)0x00000001)
05153 #define  RCC_BDCR_LSERDY                     ((uint32_t)0x00000002)
05154 #define  RCC_BDCR_LSEBYP                     ((uint32_t)0x00000004)
05155
05156 #define  RCC_BDCR_RTCSEL                    ((uint32_t)0x00000300)
05157 #define  RCC_BDCR_RTCSEL_0                  ((uint32_t)0x00000100)
05158 #define  RCC_BDCR_RTCSEL_1                  ((uint32_t)0x00000200)
05159
05160 #define  RCC_BDCR_RTCEN                      ((uint32_t)0x00008000)
05161 #define  RCC_BDCR_BDRST                      ((uint32_t)0x00010000)
05162
05163 /********************  Bit definition for RCC_CSR register  *******************/
05164 #define  RCC_CSR_LSION                       ((uint32_t)0x00000001)
05165 #define  RCC_CSR_LSIRDY                      ((uint32_t)0x00000002)
05166 #define  RCC_CSR_RMVF                        ((uint32_t)0x01000000)
05167 #define  RCC_CSR_BORRSTF                     ((uint32_t)0x02000000)
05168 #define  RCC_CSR_PADRSTF                     ((uint32_t)0x04000000)
05169 #define  RCC_CSR_PORRSTF                     ((uint32_t)0x08000000)
05170 #define  RCC_CSR_SFTRSTF                     ((uint32_t)0x10000000)
```

```
05171 #define  RCC_CSR_WDGRSTF                     ((uint32_t)0x20000000)
05172 #define  RCC_CSR_WWDGRSTF                    ((uint32_t)0x40000000)
05173 #define  RCC_CSR_LPWRRSTF                    ((uint32_t)0x80000000)
05174
05175 /********************  Bit definition for RCC_SSCGR register  *****************/
05176 #define  RCC_SSCGR_MODPER                    ((uint32_t)0x00001FFF)
05177 #define  RCC_SSCGR_INCSTEP                   ((uint32_t)0x0FFFE000)
05178 #define  RCC_SSCGR_SPREADSEL                 ((uint32_t)0x40000000)
05179 #define  RCC_SSCGR_SSCGEN                    ((uint32_t)0x80000000)
05180
05181 /********************  Bit definition for RCC_PLLI2SCFGR register  ************/
05182 #define  RCC_PLLI2SCFGR_PLLI2SN              ((uint32_t)0x00007FC0)
05183 #define  RCC_PLLI2SCFGR_PLLI2SR              ((uint32_t)0x70000000)
05184
05185 /******************************************************************************/
05186 /*                                                                            */
05187 /*                                    RNG                                     */
05188 /*                                                                            */
05189 /******************************************************************************/
05190 /********************  Bits definition for RNG_CR register  *******************/
05191 #define RNG_CR_RNGEN                         ((uint32_t)0x00000004)
05192 #define RNG_CR_IE                            ((uint32_t)0x00000008)
05193
05194 /********************  Bits definition for RNG_SR register  *******************/
05195 #define RNG_SR_DRDY                          ((uint32_t)0x00000001)
05196 #define RNG_SR_CECS                          ((uint32_t)0x00000002)
05197 #define RNG_SR_SECS                          ((uint32_t)0x00000004)
05198 #define RNG_SR_CEIS                          ((uint32_t)0x00000020)
05199 #define RNG_SR_SEIS                          ((uint32_t)0x00000040)
05200
05201 /******************************************************************************/
05202 /*                                                                            */
05203 /*                          Real-Time Clock (RTC)                             */
05204 /*                                                                            */
05205 /******************************************************************************/
05206 /********************  Bits definition for RTC_TR register  *******************/
05207 #define RTC_TR_PM                            ((uint32_t)0x00400000)
05208 #define RTC_TR_HT                            ((uint32_t)0x00300000)
05209 #define RTC_TR_HT_0                          ((uint32_t)0x00100000)
05210 #define RTC_TR_HT_1                          ((uint32_t)0x00200000)
05211 #define RTC_TR_HU                            ((uint32_t)0x000F0000)
05212 #define RTC_TR_HU_0                          ((uint32_t)0x00010000)
05213 #define RTC_TR_HU_1                          ((uint32_t)0x00020000)
05214 #define RTC_TR_HU_2                          ((uint32_t)0x00040000)
05215 #define RTC_TR_HU_3                          ((uint32_t)0x00080000)
05216 #define RTC_TR_MNT                           ((uint32_t)0x00007000)
05217 #define RTC_TR_MNT_0                         ((uint32_t)0x00001000)
05218 #define RTC_TR_MNT_1                         ((uint32_t)0x00002000)
05219 #define RTC_TR_MNT_2                         ((uint32_t)0x00004000)
05220 #define RTC_TR_MNU                           ((uint32_t)0x00000F00)
05221 #define RTC_TR_MNU_0                         ((uint32_t)0x00000100)
05222 #define RTC_TR_MNU_1                         ((uint32_t)0x00000200)
05223 #define RTC_TR_MNU_2                         ((uint32_t)0x00000400)
05224 #define RTC_TR_MNU_3                         ((uint32_t)0x00000800)
05225 #define RTC_TR_ST                            ((uint32_t)0x00000070)
05226 #define RTC_TR_ST_0                          ((uint32_t)0x00000010)
05227 #define RTC_TR_ST_1                          ((uint32_t)0x00000020)
05228 #define RTC_TR_ST_2                          ((uint32_t)0x00000040)
05229 #define RTC_TR_SU                            ((uint32_t)0x0000000F)
05230 #define RTC_TR_SU_0                          ((uint32_t)0x00000001)
05231 #define RTC_TR_SU_1                          ((uint32_t)0x00000002)
05232 #define RTC_TR_SU_2                          ((uint32_t)0x00000004)
05233 #define RTC_TR_SU_3                          ((uint32_t)0x00000008)
05234
05235 /********************  Bits definition for RTC_DR register  *******************/
05236 #define RTC_DR_YT                            ((uint32_t)0x00F00000)
05237 #define RTC_DR_YT_0                          ((uint32_t)0x00100000)
05238 #define RTC_DR_YT_1                          ((uint32_t)0x00200000)
05239 #define RTC_DR_YT_2                          ((uint32_t)0x00400000)
05240 #define RTC_DR_YT_3                          ((uint32_t)0x00800000)
05241 #define RTC_DR_YU                            ((uint32_t)0x000F0000)
05242 #define RTC_DR_YU_0                          ((uint32_t)0x00010000)
05243 #define RTC_DR_YU_1                          ((uint32_t)0x00020000)
05244 #define RTC_DR_YU_2                          ((uint32_t)0x00040000)
05245 #define RTC_DR_YU_3                          ((uint32_t)0x00080000)
05246 #define RTC_DR_WDU                           ((uint32_t)0x0000E000)
05247 #define RTC_DR_WDU_0                         ((uint32_t)0x00002000)
05248 #define RTC_DR_WDU_1                         ((uint32_t)0x00004000)
05249 #define RTC_DR_WDU_2                         ((uint32_t)0x00008000)
05250 #define RTC_DR_MT                            ((uint32_t)0x00001000)
05251 #define RTC_DR_MU                            ((uint32_t)0x00000F00)
05252 #define RTC_DR_MU_0                          ((uint32_t)0x00000100)
05253 #define RTC_DR_MU_1                          ((uint32_t)0x00000200)
05254 #define RTC_DR_MU_2                          ((uint32_t)0x00000400)
05255 #define RTC_DR_MU_3                          ((uint32_t)0x00000800)
05256 #define RTC_DR_DT                            ((uint32_t)0x00000030)
05257 #define RTC_DR_DT_0                          ((uint32_t)0x00000010)
```

```
05258 #define RTC_DR_DT_1                          ((uint32_t)0x00000020)
05259 #define RTC_DR_DU                            ((uint32_t)0x0000000F)
05260 #define RTC_DR_DU_0                          ((uint32_t)0x00000001)
05261 #define RTC_DR_DU_1                          ((uint32_t)0x00000002)
05262 #define RTC_DR_DU_2                          ((uint32_t)0x00000004)
05263 #define RTC_DR_DU_3                          ((uint32_t)0x00000008)
05264
05265 /********************  Bits definition for RTC_CR register  *******************/
05266 #define RTC_CR_COE                           ((uint32_t)0x00800000)
05267 #define RTC_CR_OSEL                          ((uint32_t)0x00600000)
05268 #define RTC_CR_OSEL_0                        ((uint32_t)0x00200000)
05269 #define RTC_CR_OSEL_1                        ((uint32_t)0x00400000)
05270 #define RTC_CR_POL                           ((uint32_t)0x00100000)
05271 #define RTC_CR_COSEL                         ((uint32_t)0x00080000)
05272 #define RTC_CR_BCK                           ((uint32_t)0x00040000)
05273 #define RTC_CR_SUB1H                         ((uint32_t)0x00020000)
05274 #define RTC_CR_ADD1H                         ((uint32_t)0x00010000)
05275 #define RTC_CR_TSIE                          ((uint32_t)0x00008000)
05276 #define RTC_CR_WUTIE                         ((uint32_t)0x00004000)
05277 #define RTC_CR_ALRBIE                        ((uint32_t)0x00002000)
05278 #define RTC_CR_ALRAIE                        ((uint32_t)0x00001000)
05279 #define RTC_CR_TSE                           ((uint32_t)0x00000800)
05280 #define RTC_CR_WUTE                          ((uint32_t)0x00000400)
05281 #define RTC_CR_ALRBE                         ((uint32_t)0x00000200)
05282 #define RTC_CR_ALRAE                         ((uint32_t)0x00000100)
05283 #define RTC_CR_DCE                           ((uint32_t)0x00000080)
05284 #define RTC_CR_FMT                           ((uint32_t)0x00000040)
05285 #define RTC_CR_BYPSHAD                       ((uint32_t)0x00000020)
05286 #define RTC_CR_REFCKON                       ((uint32_t)0x00000010)
05287 #define RTC_CR_TSEDGE                        ((uint32_t)0x00000008)
05288 #define RTC_CR_WUCKSEL                       ((uint32_t)0x00000007)
05289 #define RTC_CR_WUCKSEL_0                     ((uint32_t)0x00000001)
05290 #define RTC_CR_WUCKSEL_1                     ((uint32_t)0x00000002)
05291 #define RTC_CR_WUCKSEL_2                     ((uint32_t)0x00000004)
05292
05293 /********************  Bits definition for RTC_ISR register  ******************/
05294 #define RTC_ISR_RECALPF                      ((uint32_t)0x00010000)
05295 #define RTC_ISR_TAMP1F                       ((uint32_t)0x00002000)
05296 #define RTC_ISR_TSOVF                        ((uint32_t)0x00001000)
05297 #define RTC_ISR_TSF                          ((uint32_t)0x00000800)
05298 #define RTC_ISR_WUTF                         ((uint32_t)0x00000400)
05299 #define RTC_ISR_ALRBF                        ((uint32_t)0x00000200)
05300 #define RTC_ISR_ALRAF                        ((uint32_t)0x00000100)
05301 #define RTC_ISR_INIT                         ((uint32_t)0x00000080)
05302 #define RTC_ISR_INITF                        ((uint32_t)0x00000040)
05303 #define RTC_ISR_RSF                          ((uint32_t)0x00000020)
05304 #define RTC_ISR_INITS                        ((uint32_t)0x00000010)
05305 #define RTC_ISR_SHPF                         ((uint32_t)0x00000008)
05306 #define RTC_ISR_WUTWF                        ((uint32_t)0x00000004)
05307 #define RTC_ISR_ALRBWF                       ((uint32_t)0x00000002)
05308 #define RTC_ISR_ALRAWF                       ((uint32_t)0x00000001)
05309
05310 /********************  Bits definition for RTC_PRER register  *****************/
05311 #define RTC_PRER_PREDIV_A                    ((uint32_t)0x007F0000)
05312 #define RTC_PRER_PREDIV_S                    ((uint32_t)0x00001FFF)
05313
05314 /********************  Bits definition for RTC_WUTR register  *****************/
05315 #define RTC_WUTR_WUT                         ((uint32_t)0x0000FFFF)
05316
05317 /********************  Bits definition for RTC_CALIBR register  ***************/
05318 #define RTC_CALIBR_DCS                       ((uint32_t)0x00000080)
05319 #define RTC_CALIBR_DC                        ((uint32_t)0x0000001F)
05320
05321 /********************  Bits definition for RTC_ALRMAR register  ***************/
05322 #define RTC_ALRMAR_MSK4                      ((uint32_t)0x80000000)
05323 #define RTC_ALRMAR_WDSEL                     ((uint32_t)0x40000000)
05324 #define RTC_ALRMAR_DT                        ((uint32_t)0x30000000)
05325 #define RTC_ALRMAR_DT_0                      ((uint32_t)0x10000000)
05326 #define RTC_ALRMAR_DT_1                      ((uint32_t)0x20000000)
05327 #define RTC_ALRMAR_DU                        ((uint32_t)0x0F000000)
05328 #define RTC_ALRMAR_DU_0                      ((uint32_t)0x01000000)
05329 #define RTC_ALRMAR_DU_1                      ((uint32_t)0x02000000)
05330 #define RTC_ALRMAR_DU_2                      ((uint32_t)0x04000000)
05331 #define RTC_ALRMAR_DU_3                      ((uint32_t)0x08000000)
05332 #define RTC_ALRMAR_MSK3                      ((uint32_t)0x00800000)
05333 #define RTC_ALRMAR_PM                        ((uint32_t)0x00400000)
05334 #define RTC_ALRMAR_HT                        ((uint32_t)0x00300000)
05335 #define RTC_ALRMAR_HT_0                      ((uint32_t)0x00100000)
05336 #define RTC_ALRMAR_HT_1                      ((uint32_t)0x00200000)
05337 #define RTC_ALRMAR_HU                        ((uint32_t)0x000F0000)
05338 #define RTC_ALRMAR_HU_0                      ((uint32_t)0x00010000)
05339 #define RTC_ALRMAR_HU_1                      ((uint32_t)0x00020000)
05340 #define RTC_ALRMAR_HU_2                      ((uint32_t)0x00040000)
05341 #define RTC_ALRMAR_HU_3                      ((uint32_t)0x00080000)
05342 #define RTC_ALRMAR_MSK2                      ((uint32_t)0x00008000)
05343 #define RTC_ALRMAR_MNT                       ((uint32_t)0x00007000)
05344 #define RTC_ALRMAR_MNT_0                     ((uint32_t)0x00001000)
```

```
05345 #define RTC_ALRMAR_MNT_1                    ((uint32_t)0x00002000)
05346 #define RTC_ALRMAR_MNT_2                    ((uint32_t)0x00004000)
05347 #define RTC_ALRMAR_MNU                      ((uint32_t)0x00000F00)
05348 #define RTC_ALRMAR_MNU_0                    ((uint32_t)0x00000100)
05349 #define RTC_ALRMAR_MNU_1                    ((uint32_t)0x00000200)
05350 #define RTC_ALRMAR_MNU_2                    ((uint32_t)0x00000400)
05351 #define RTC_ALRMAR_MNU_3                    ((uint32_t)0x00000800)
05352 #define RTC_ALRMAR_MSK1                     ((uint32_t)0x00000080)
05353 #define RTC_ALRMAR_ST                       ((uint32_t)0x00000070)
05354 #define RTC_ALRMAR_ST_0                     ((uint32_t)0x00000010)
05355 #define RTC_ALRMAR_ST_1                     ((uint32_t)0x00000020)
05356 #define RTC_ALRMAR_ST_2                     ((uint32_t)0x00000040)
05357 #define RTC_ALRMAR_SU                       ((uint32_t)0x0000000F)
05358 #define RTC_ALRMAR_SU_0                     ((uint32_t)0x00000001)
05359 #define RTC_ALRMAR_SU_1                     ((uint32_t)0x00000002)
05360 #define RTC_ALRMAR_SU_2                     ((uint32_t)0x00000004)
05361 #define RTC_ALRMAR_SU_3                     ((uint32_t)0x00000008)
05362
05363 /********************  Bits definition for RTC_ALRMBR register  ***************/
05364 #define RTC_ALRMBR_MSK4                     ((uint32_t)0x80000000)
05365 #define RTC_ALRMBR_WDSEL                    ((uint32_t)0x40000000)
05366 #define RTC_ALRMBR_DT                       ((uint32_t)0x30000000)
05367 #define RTC_ALRMBR_DT_0                     ((uint32_t)0x10000000)
05368 #define RTC_ALRMBR_DT_1                     ((uint32_t)0x20000000)
05369 #define RTC_ALRMBR_DU                       ((uint32_t)0x0F000000)
05370 #define RTC_ALRMBR_DU_0                     ((uint32_t)0x01000000)
05371 #define RTC_ALRMBR_DU_1                     ((uint32_t)0x02000000)
05372 #define RTC_ALRMBR_DU_2                     ((uint32_t)0x04000000)
05373 #define RTC_ALRMBR_DU_3                     ((uint32_t)0x08000000)
05374 #define RTC_ALRMBR_MSK3                     ((uint32_t)0x00800000)
05375 #define RTC_ALRMBR_PM                       ((uint32_t)0x00400000)
05376 #define RTC_ALRMBR_HT                       ((uint32_t)0x00300000)
05377 #define RTC_ALRMBR_HT_0                     ((uint32_t)0x00100000)
05378 #define RTC_ALRMBR_HT_1                     ((uint32_t)0x00200000)
05379 #define RTC_ALRMBR_HU                       ((uint32_t)0x000F0000)
05380 #define RTC_ALRMBR_HU_0                     ((uint32_t)0x00010000)
05381 #define RTC_ALRMBR_HU_1                     ((uint32_t)0x00020000)
05382 #define RTC_ALRMBR_HU_2                     ((uint32_t)0x00040000)
05383 #define RTC_ALRMBR_HU_3                     ((uint32_t)0x00080000)
05384 #define RTC_ALRMBR_MSK2                     ((uint32_t)0x00008000)
05385 #define RTC_ALRMBR_MNT                      ((uint32_t)0x00007000)
05386 #define RTC_ALRMBR_MNT_0                    ((uint32_t)0x00001000)
05387 #define RTC_ALRMBR_MNT_1                    ((uint32_t)0x00002000)
05388 #define RTC_ALRMBR_MNT_2                    ((uint32_t)0x00004000)
05389 #define RTC_ALRMBR_MNU                      ((uint32_t)0x00000F00)
05390 #define RTC_ALRMBR_MNU_0                    ((uint32_t)0x00000100)
05391 #define RTC_ALRMBR_MNU_1                    ((uint32_t)0x00000200)
05392 #define RTC_ALRMBR_MNU_2                    ((uint32_t)0x00000400)
05393 #define RTC_ALRMBR_MNU_3                    ((uint32_t)0x00000800)
05394 #define RTC_ALRMBR_MSK1                     ((uint32_t)0x00000080)
05395 #define RTC_ALRMBR_ST                       ((uint32_t)0x00000070)
05396 #define RTC_ALRMBR_ST_0                     ((uint32_t)0x00000010)
05397 #define RTC_ALRMBR_ST_1                     ((uint32_t)0x00000020)
05398 #define RTC_ALRMBR_ST_2                     ((uint32_t)0x00000040)
05399 #define RTC_ALRMBR_SU                       ((uint32_t)0x0000000F)
05400 #define RTC_ALRMBR_SU_0                     ((uint32_t)0x00000001)
05401 #define RTC_ALRMBR_SU_1                     ((uint32_t)0x00000002)
05402 #define RTC_ALRMBR_SU_2                     ((uint32_t)0x00000004)
05403 #define RTC_ALRMBR_SU_3                     ((uint32_t)0x00000008)
05404
05405 /********************  Bits definition for RTC_WPR register  ******************/
05406 #define RTC_WPR_KEY                         ((uint32_t)0x000000FF)
05407
05408 /********************  Bits definition for RTC_SSR register  ******************/
05409 #define RTC_SSR_SS                          ((uint32_t)0x0000FFFF)
05410
05411 /********************  Bits definition for RTC_SHIFTR register  ***************/
05412 #define RTC_SHIFTR_SUBFS                    ((uint32_t)0x00007FFF)
05413 #define RTC_SHIFTR_ADD1S                    ((uint32_t)0x80000000)
05414
05415 /********************  Bits definition for RTC_TSTR register  *****************/
05416 #define RTC_TSTR_PM                         ((uint32_t)0x00400000)
05417 #define RTC_TSTR_HT                         ((uint32_t)0x00300000)
05418 #define RTC_TSTR_HT_0                       ((uint32_t)0x00100000)
05419 #define RTC_TSTR_HT_1                       ((uint32_t)0x00200000)
05420 #define RTC_TSTR_HU                         ((uint32_t)0x000F0000)
05421 #define RTC_TSTR_HU_0                       ((uint32_t)0x00010000)
05422 #define RTC_TSTR_HU_1                       ((uint32_t)0x00020000)
05423 #define RTC_TSTR_HU_2                       ((uint32_t)0x00040000)
05424 #define RTC_TSTR_HU_3                       ((uint32_t)0x00080000)
05425 #define RTC_TSTR_MNT                        ((uint32_t)0x00007000)
05426 #define RTC_TSTR_MNT_0                      ((uint32_t)0x00001000)
05427 #define RTC_TSTR_MNT_1                      ((uint32_t)0x00002000)
05428 #define RTC_TSTR_MNT_2                      ((uint32_t)0x00004000)
05429 #define RTC_TSTR_MNU                        ((uint32_t)0x00000F00)
05430 #define RTC_TSTR_MNU_0                      ((uint32_t)0x00000100)
05431 #define RTC_TSTR_MNU_1                      ((uint32_t)0x00000200)
```

```
05432 #define RTC_TSTR_MNU_2                       ((uint32_t)0x00000400)
05433 #define RTC_TSTR_MNU_3                       ((uint32_t)0x00000800)
05434 #define RTC_TSTR_ST                          ((uint32_t)0x00000070)
05435 #define RTC_TSTR_ST_0                        ((uint32_t)0x00000010)
05436 #define RTC_TSTR_ST_1                        ((uint32_t)0x00000020)
05437 #define RTC_TSTR_ST_2                        ((uint32_t)0x00000040)
05438 #define RTC_TSTR_SU                          ((uint32_t)0x0000000F)
05439 #define RTC_TSTR_SU_0                        ((uint32_t)0x00000001)
05440 #define RTC_TSTR_SU_1                        ((uint32_t)0x00000002)
05441 #define RTC_TSTR_SU_2                        ((uint32_t)0x00000004)
05442 #define RTC_TSTR_SU_3                        ((uint32_t)0x00000008)
05443
05444 /********************  Bits definition for RTC_TSDR register  ****************/
05445 #define RTC_TSDR_WDU                         ((uint32_t)0x0000E000)
05446 #define RTC_TSDR_WDU_0                       ((uint32_t)0x00002000)
05447 #define RTC_TSDR_WDU_1                       ((uint32_t)0x00004000)
05448 #define RTC_TSDR_WDU_2                       ((uint32_t)0x00008000)
05449 #define RTC_TSDR_MT                          ((uint32_t)0x00001000)
05450 #define RTC_TSDR_MU                          ((uint32_t)0x00000F00)
05451 #define RTC_TSDR_MU_0                        ((uint32_t)0x00000100)
05452 #define RTC_TSDR_MU_1                        ((uint32_t)0x00000200)
05453 #define RTC_TSDR_MU_2                        ((uint32_t)0x00000400)
05454 #define RTC_TSDR_MU_3                        ((uint32_t)0x00000800)
05455 #define RTC_TSDR_DT                          ((uint32_t)0x00000030)
05456 #define RTC_TSDR_DT_0                        ((uint32_t)0x00000010)
05457 #define RTC_TSDR_DT_1                        ((uint32_t)0x00000020)
05458 #define RTC_TSDR_DU                          ((uint32_t)0x0000000F)
05459 #define RTC_TSDR_DU_0                        ((uint32_t)0x00000001)
05460 #define RTC_TSDR_DU_1                        ((uint32_t)0x00000002)
05461 #define RTC_TSDR_DU_2                        ((uint32_t)0x00000004)
05462 #define RTC_TSDR_DU_3                        ((uint32_t)0x00000008)
05463
05464 /********************  Bits definition for RTC_TSSSR register  ****************/
05465 #define RTC_TSSSR_SS                         ((uint32_t)0x0000FFFF)
05466
05467 /********************  Bits definition for RTC_CAL register  ****************/
05468 #define RTC_CALR_CALP                        ((uint32_t)0x00008000)
05469 #define RTC_CALR_CALW8                       ((uint32_t)0x00004000)
05470 #define RTC_CALR_CALW16                      ((uint32_t)0x00002000)
05471 #define RTC_CALR_CALM                        ((uint32_t)0x000001FF)
05472 #define RTC_CALR_CALM_0                      ((uint32_t)0x00000001)
05473 #define RTC_CALR_CALM_1                      ((uint32_t)0x00000002)
05474 #define RTC_CALR_CALM_2                      ((uint32_t)0x00000004)
05475 #define RTC_CALR_CALM_3                      ((uint32_t)0x00000008)
05476 #define RTC_CALR_CALM_4                      ((uint32_t)0x00000010)
05477 #define RTC_CALR_CALM_5                      ((uint32_t)0x00000020)
05478 #define RTC_CALR_CALM_6                      ((uint32_t)0x00000040)
05479 #define RTC_CALR_CALM_7                      ((uint32_t)0x00000080)
05480 #define RTC_CALR_CALM_8                      ((uint32_t)0x00000100)
05481
05482 /********************  Bits definition for RTC_TAFCR register  ****************/
05483 #define RTC_TAFCR_ALARMOUTTYPE               ((uint32_t)0x00040000)
05484 #define RTC_TAFCR_TSINSEL                    ((uint32_t)0x00020000)
05485 #define RTC_TAFCR_TAMPINSEL                  ((uint32_t)0x00010000)
05486 #define RTC_TAFCR_TAMPPUDIS                  ((uint32_t)0x00008000)
05487 #define RTC_TAFCR_TAMPPRCH                   ((uint32_t)0x00006000)
05488 #define RTC_TAFCR_TAMPPRCH_0                 ((uint32_t)0x00002000)
05489 #define RTC_TAFCR_TAMPPRCH_1                 ((uint32_t)0x00004000)
05490 #define RTC_TAFCR_TAMPFLT                    ((uint32_t)0x00001800)
05491 #define RTC_TAFCR_TAMPFLT_0                  ((uint32_t)0x00000800)
05492 #define RTC_TAFCR_TAMPFLT_1                  ((uint32_t)0x00001000)
05493 #define RTC_TAFCR_TAMPFREQ                   ((uint32_t)0x00000700)
05494 #define RTC_TAFCR_TAMPFREQ_0                 ((uint32_t)0x00000100)
05495 #define RTC_TAFCR_TAMPFREQ_1                 ((uint32_t)0x00000200)
05496 #define RTC_TAFCR_TAMPFREQ_2                 ((uint32_t)0x00000400)
05497 #define RTC_TAFCR_TAMPTS                     ((uint32_t)0x00000080)
05498 #define RTC_TAFCR_TAMPIE                     ((uint32_t)0x00000004)
05499 #define RTC_TAFCR_TAMP1TRG                   ((uint32_t)0x00000002)
05500 #define RTC_TAFCR_TAMP1E                     ((uint32_t)0x00000001)
05501
05502 /********************  Bits definition for RTC_ALRMASSR register  *************/
05503 #define RTC_ALRMASSR_MASKSS                  ((uint32_t)0x0F000000)
05504 #define RTC_ALRMASSR_MASKSS_0                ((uint32_t)0x01000000)
05505 #define RTC_ALRMASSR_MASKSS_1                ((uint32_t)0x02000000)
05506 #define RTC_ALRMASSR_MASKSS_2                ((uint32_t)0x04000000)
05507 #define RTC_ALRMASSR_MASKSS_3                ((uint32_t)0x08000000)
05508 #define RTC_ALRMASSR_SS                      ((uint32_t)0x00007FFF)
05509
05510 /********************  Bits definition for RTC_ALRMBSSR register  *************/
05511 #define RTC_ALRMBSSR_MASKSS                  ((uint32_t)0x0F000000)
05512 #define RTC_ALRMBSSR_MASKSS_0                ((uint32_t)0x01000000)
05513 #define RTC_ALRMBSSR_MASKSS_1                ((uint32_t)0x02000000)
05514 #define RTC_ALRMBSSR_MASKSS_2                ((uint32_t)0x04000000)
05515 #define RTC_ALRMBSSR_MASKSS_3                ((uint32_t)0x08000000)
05516 #define RTC_ALRMBSSR_SS                      ((uint32_t)0x00007FFF)
05517
05518 /********************  Bits definition for RTC_BKP0R register  ****************/
```

```
05519 #define RTC_BKP0R                           ((uint32_t)0xFFFFFFFF)
05520
05521 /********************  Bits definition for RTC_BKP1R register  ***************/
05522 #define RTC_BKP1R                           ((uint32_t)0xFFFFFFFF)
05523
05524 /********************  Bits definition for RTC_BKP2R register  ***************/
05525 #define RTC_BKP2R                           ((uint32_t)0xFFFFFFFF)
05526
05527 /********************  Bits definition for RTC_BKP3R register  ***************/
05528 #define RTC_BKP3R                           ((uint32_t)0xFFFFFFFF)
05529
05530 /********************  Bits definition for RTC_BKP4R register  ***************/
05531 #define RTC_BKP4R                           ((uint32_t)0xFFFFFFFF)
05532
05533 /********************  Bits definition for RTC_BKP5R register  ***************/
05534 #define RTC_BKP5R                           ((uint32_t)0xFFFFFFFF)
05535
05536 /********************  Bits definition for RTC_BKP6R register  ***************/
05537 #define RTC_BKP6R                           ((uint32_t)0xFFFFFFFF)
05538
05539 /********************  Bits definition for RTC_BKP7R register  ***************/
05540 #define RTC_BKP7R                           ((uint32_t)0xFFFFFFFF)
05541
05542 /********************  Bits definition for RTC_BKP8R register  ***************/
05543 #define RTC_BKP8R                           ((uint32_t)0xFFFFFFFF)
05544
05545 /********************  Bits definition for RTC_BKP9R register  ***************/
05546 #define RTC_BKP9R                           ((uint32_t)0xFFFFFFFF)
05547
05548 /********************  Bits definition for RTC_BKP10R register  **************/
05549 #define RTC_BKP10R                          ((uint32_t)0xFFFFFFFF)
05550
05551 /********************  Bits definition for RTC_BKP11R register  **************/
05552 #define RTC_BKP11R                          ((uint32_t)0xFFFFFFFF)
05553
05554 /********************  Bits definition for RTC_BKP12R register  **************/
05555 #define RTC_BKP12R                          ((uint32_t)0xFFFFFFFF)
05556
05557 /********************  Bits definition for RTC_BKP13R register  **************/
05558 #define RTC_BKP13R                          ((uint32_t)0xFFFFFFFF)
05559
05560 /********************  Bits definition for RTC_BKP14R register  **************/
05561 #define RTC_BKP14R                          ((uint32_t)0xFFFFFFFF)
05562
05563 /********************  Bits definition for RTC_BKP15R register  **************/
05564 #define RTC_BKP15R                          ((uint32_t)0xFFFFFFFF)
05565
05566 /********************  Bits definition for RTC_BKP16R register  **************/
05567 #define RTC_BKP16R                          ((uint32_t)0xFFFFFFFF)
05568
05569 /********************  Bits definition for RTC_BKP17R register  **************/
05570 #define RTC_BKP17R                          ((uint32_t)0xFFFFFFFF)
05571
05572 /********************  Bits definition for RTC_BKP18R register  **************/
05573 #define RTC_BKP18R                          ((uint32_t)0xFFFFFFFF)
05574
05575 /********************  Bits definition for RTC_BKP19R register  **************/
05576 #define RTC_BKP19R                          ((uint32_t)0xFFFFFFFF)
05577
05578 /******************************************************************************/
05579 /*                                                                            */
05580 /*                          SD host Interface                                 */
05581 /*                                                                            */
05582 /******************************************************************************/
05583 /******************  Bit definition for SDIO_POWER register  ******************/
05584 #define  SDIO_POWER_PWRCTRL                 ((uint8_t)0x03)
05585 #define  SDIO_POWER_PWRCTRL_0               ((uint8_t)0x01)
05586 #define  SDIO_POWER_PWRCTRL_1               ((uint8_t)0x02)
05588 /******************  Bit definition for SDIO_CLKCR register  ******************/
05589 #define  SDIO_CLKCR_CLKDIV                  ((uint16_t)0x00FF)
05590 #define  SDIO_CLKCR_CLKEN                   ((uint16_t)0x0100)
05591 #define  SDIO_CLKCR_PWRSAV                  ((uint16_t)0x0200)
05592 #define  SDIO_CLKCR_BYPASS                  ((uint16_t)0x0400)
05594 #define  SDIO_CLKCR_WIDBUS                  ((uint16_t)0x1800)
05595 #define  SDIO_CLKCR_WIDBUS_0                ((uint16_t)0x0800)
05596 #define  SDIO_CLKCR_WIDBUS_1                ((uint16_t)0x1000)
05598 #define  SDIO_CLKCR_NEGEDGE                 ((uint16_t)0x2000)
05599 #define  SDIO_CLKCR_HWFC_EN                 ((uint16_t)0x4000)
05601 /******************  Bit definition for SDIO_ARG register  ********************/
05602 #define  SDIO_ARG_CMDARG                    ((uint32_t)0xFFFFFFFF)
05604 /******************  Bit definition for SDIO_CMD register  ********************/
05605 #define  SDIO_CMD_CMDINDEX                  ((uint16_t)0x003F)
05607 #define  SDIO_CMD_WAITRESP                  ((uint16_t)0x00C0)
05608 #define  SDIO_CMD_WAITRESP_0                ((uint16_t)0x0040)
05609 #define  SDIO_CMD_WAITRESP_1                ((uint16_t)0x0080)
05611 #define  SDIO_CMD_WAITINT                   ((uint16_t)0x0100)
05612 #define  SDIO_CMD_WAITPEND                  ((uint16_t)0x0200)
```

```
05613 #define  SDIO_CMD_CPSMEN                    ((uint16_t)0x0400)
05614 #define  SDIO_CMD_SDIOSUSPEND               ((uint16_t)0x0800)
05615 #define  SDIO_CMD_ENCMDCOMPL                ((uint16_t)0x1000)
05616 #define  SDIO_CMD_NIEN                      ((uint16_t)0x2000)
05617 #define  SDIO_CMD_CEATACMD                  ((uint16_t)0x4000)
05619 /*****************  Bit definition for SDIO_RESPCMD register  *****************/
05620 #define  SDIO_RESPCMD_RESPCMD               ((uint8_t)0x3F)
05622 /******************  Bit definition for SDIO_RESP0 register  ******************/
05623 #define  SDIO_RESP0_CARDSTATUS0             ((uint32_t)0xFFFFFFFF)
05625 /******************  Bit definition for SDIO_RESP1 register  ******************/
05626 #define  SDIO_RESP1_CARDSTATUS1             ((uint32_t)0xFFFFFFFF)
05628 /******************  Bit definition for SDIO_RESP2 register  ******************/
05629 #define  SDIO_RESP2_CARDSTATUS2             ((uint32_t)0xFFFFFFFF)
05631 /******************  Bit definition for SDIO_RESP3 register  ******************/
05632 #define  SDIO_RESP3_CARDSTATUS3             ((uint32_t)0xFFFFFFFF)
05634 /******************  Bit definition for SDIO_RESP4 register  ******************/
05635 #define  SDIO_RESP4_CARDSTATUS4             ((uint32_t)0xFFFFFFFF)
05637 /******************  Bit definition for SDIO_DTIMER register  *****************/
05638 #define  SDIO_DTIMER_DATATIME               ((uint32_t)0xFFFFFFFF)
05640 /******************  Bit definition for SDIO_DLEN register  *******************/
05641 #define  SDIO_DLEN_DATALENGTH               ((uint32_t)0x01FFFFFF)
05643 /******************  Bit definition for SDIO_DCTRL register  ******************/
05644 #define  SDIO_DCTRL_DTEN                    ((uint16_t)0x0001)
05645 #define  SDIO_DCTRL_DTDIR                   ((uint16_t)0x0002)
05646 #define  SDIO_DCTRL_DTMODE                  ((uint16_t)0x0004)
05647 #define  SDIO_DCTRL_DMAEN                   ((uint16_t)0x0008)
05649 #define  SDIO_DCTRL_DBLOCKSIZE              ((uint16_t)0x00F0)
05650 #define  SDIO_DCTRL_DBLOCKSIZE_0            ((uint16_t)0x0010)
05651 #define  SDIO_DCTRL_DBLOCKSIZE_1            ((uint16_t)0x0020)
05652 #define  SDIO_DCTRL_DBLOCKSIZE_2            ((uint16_t)0x0040)
05653 #define  SDIO_DCTRL_DBLOCKSIZE_3            ((uint16_t)0x0080)
05655 #define  SDIO_DCTRL_RWSTART                 ((uint16_t)0x0100)
05656 #define  SDIO_DCTRL_RWSTOP                  ((uint16_t)0x0200)
05657 #define  SDIO_DCTRL_RWMOD                   ((uint16_t)0x0400)
05658 #define  SDIO_DCTRL_SDIOEN                  ((uint16_t)0x0800)
05660 /******************  Bit definition for SDIO_DCOUNT register  *****************/
05661 #define  SDIO_DCOUNT_DATACOUNT              ((uint32_t)0x01FFFFFF)
05663 /******************  Bit definition for SDIO_STA register  ********************/
05664 #define  SDIO_STA_CCRCFAIL                  ((uint32_t)0x00000001)
05665 #define  SDIO_STA_DCRCFAIL                  ((uint32_t)0x00000002)
05666 #define  SDIO_STA_CTIMEOUT                  ((uint32_t)0x00000004)
05667 #define  SDIO_STA_DTIMEOUT                  ((uint32_t)0x00000008)
05668 #define  SDIO_STA_TXUNDERR                  ((uint32_t)0x00000010)
05669 #define  SDIO_STA_RXOVERR                   ((uint32_t)0x00000020)
05670 #define  SDIO_STA_CMDREND                   ((uint32_t)0x00000040)
05671 #define  SDIO_STA_CMDSENT                   ((uint32_t)0x00000080)
05672 #define  SDIO_STA_DATAEND                   ((uint32_t)0x00000100)
05673 #define  SDIO_STA_STBITERR                  ((uint32_t)0x00000200)
05674 #define  SDIO_STA_DBCKEND                   ((uint32_t)0x00000400)
05675 #define  SDIO_STA_CMDACT                    ((uint32_t)0x00000800)
05676 #define  SDIO_STA_TXACT                     ((uint32_t)0x00001000)
05677 #define  SDIO_STA_RXACT                     ((uint32_t)0x00002000)
05678 #define  SDIO_STA_TXFIFOHE                  ((uint32_t)0x00004000)
05679 #define  SDIO_STA_RXFIFOHF                  ((uint32_t)0x00008000)
05680 #define  SDIO_STA_TXFIFOF                   ((uint32_t)0x00010000)
05681 #define  SDIO_STA_RXFIFOF                   ((uint32_t)0x00020000)
05682 #define  SDIO_STA_TXFIFOE                   ((uint32_t)0x00040000)
05683 #define  SDIO_STA_RXFIFOE                   ((uint32_t)0x00080000)
05684 #define  SDIO_STA_TXDAVL                    ((uint32_t)0x00100000)
05685 #define  SDIO_STA_RXDAVL                    ((uint32_t)0x00200000)
05686 #define  SDIO_STA_SDIOIT                    ((uint32_t)0x00400000)
05687 #define  SDIO_STA_CEATAEND                  ((uint32_t)0x00800000)
05689 /******************  Bit definition for SDIO_ICR register  ********************/
05690 #define  SDIO_ICR_CCRCFAILC                 ((uint32_t)0x00000001)
05691 #define  SDIO_ICR_DCRCFAILC                 ((uint32_t)0x00000002)
05692 #define  SDIO_ICR_CTIMEOUTC                 ((uint32_t)0x00000004)
05693 #define  SDIO_ICR_DTIMEOUTC                 ((uint32_t)0x00000008)
05694 #define  SDIO_ICR_TXUNDERRC                 ((uint32_t)0x00000010)
05695 #define  SDIO_ICR_RXOVERRC                  ((uint32_t)0x00000020)
05696 #define  SDIO_ICR_CMDRENDC                  ((uint32_t)0x00000040)
05697 #define  SDIO_ICR_CMDSENTC                  ((uint32_t)0x00000080)
05698 #define  SDIO_ICR_DATAENDC                  ((uint32_t)0x00000100)
05699 #define  SDIO_ICR_STBITERRC                 ((uint32_t)0x00000200)
05700 #define  SDIO_ICR_DBCKENDC                  ((uint32_t)0x00000400)
05701 #define  SDIO_ICR_SDIOITC                   ((uint32_t)0x00400000)
05702 #define  SDIO_ICR_CEATAENDC                 ((uint32_t)0x00800000)
05704 /******************  Bit definition for SDIO_MASK register  *******************/
05705 #define  SDIO_MASK_CCRCFAILIE               ((uint32_t)0x00000001)
05706 #define  SDIO_MASK_DCRCFAILIE               ((uint32_t)0x00000002)
05707 #define  SDIO_MASK_CTIMEOUTIE               ((uint32_t)0x00000004)
05708 #define  SDIO_MASK_DTIMEOUTIE               ((uint32_t)0x00000008)
05709 #define  SDIO_MASK_TXUNDERRIE               ((uint32_t)0x00000010)
05710 #define  SDIO_MASK_RXOVERRIE                ((uint32_t)0x00000020)
05711 #define  SDIO_MASK_CMDRENDIE                ((uint32_t)0x00000040)
05712 #define  SDIO_MASK_CMDSENTIE                ((uint32_t)0x00000080)
05713 #define  SDIO_MASK_DATAENDIE                ((uint32_t)0x00000100)
05714 #define  SDIO_MASK_STBITERRIE               ((uint32_t)0x00000200)
```

```
05715 #define  SDIO_MASK_DBCKENDIE                ((uint32_t)0x00000400)
05716 #define  SDIO_MASK_CMDACTIE                 ((uint32_t)0x00000800)
05717 #define  SDIO_MASK_TXACTIE                  ((uint32_t)0x00001000)
05718 #define  SDIO_MASK_RXACTIE                  ((uint32_t)0x00002000)
05719 #define  SDIO_MASK_TXFIFOHEIE               ((uint32_t)0x00004000)
05720 #define  SDIO_MASK_RXFIFOHFIE               ((uint32_t)0x00008000)
05721 #define  SDIO_MASK_TXFIFOFIE                ((uint32_t)0x00010000)
05722 #define  SDIO_MASK_RXFIFOFIE                ((uint32_t)0x00020000)
05723 #define  SDIO_MASK_TXFIFOEIE                ((uint32_t)0x00040000)
05724 #define  SDIO_MASK_RXFIFOEIE                ((uint32_t)0x00080000)
05725 #define  SDIO_MASK_TXDAVLIE                 ((uint32_t)0x00100000)
05726 #define  SDIO_MASK_RXDAVLIE                 ((uint32_t)0x00200000)
05727 #define  SDIO_MASK_SDIOITIE                 ((uint32_t)0x00400000)
05728 #define  SDIO_MASK_CEATAENDIE               ((uint32_t)0x00800000)
05730 /*****************  Bit definition for SDIO_FIFOCNT register  *****************/
05731 #define  SDIO_FIFOCNT_FIFOCOUNT             ((uint32_t)0x00FFFFFF)
05733 /*******************  Bit definition for SDIO_FIFO register  *******************/
05734 #define  SDIO_FIFO_FIFODATA                 ((uint32_t)0xFFFFFFFF)
05736 /******************************************************************************/
05737 /*                                                                            */
05738 /*                        Serial Peripheral Interface                         */
05739 /*                                                                            */
05740 /******************************************************************************/
05741 /********************  Bit definition for SPI_CR1 register  ********************/
05742 #define  SPI_CR1_CPHA                       ((uint16_t)0x0001)
05743 #define  SPI_CR1_CPOL                       ((uint16_t)0x0002)
05744 #define  SPI_CR1_MSTR                       ((uint16_t)0x0004)
05746 #define  SPI_CR1_BR                         ((uint16_t)0x0038)
05747 #define  SPI_CR1_BR_0                       ((uint16_t)0x0008)
05748 #define  SPI_CR1_BR_1                       ((uint16_t)0x0010)
05749 #define  SPI_CR1_BR_2                       ((uint16_t)0x0020)
05751 #define  SPI_CR1_SPE                        ((uint16_t)0x0040)
05752 #define  SPI_CR1_LSBFIRST                   ((uint16_t)0x0080)
05753 #define  SPI_CR1_SSI                        ((uint16_t)0x0100)
05754 #define  SPI_CR1_SSM                        ((uint16_t)0x0200)
05755 #define  SPI_CR1_RXONLY                     ((uint16_t)0x0400)
05756 #define  SPI_CR1_DFF                        ((uint16_t)0x0800)
05757 #define  SPI_CR1_CRCNEXT                    ((uint16_t)0x1000)
05758 #define  SPI_CR1_CRCEN                      ((uint16_t)0x2000)
05759 #define  SPI_CR1_BIDIOE                     ((uint16_t)0x4000)
05760 #define  SPI_CR1_BIDIMODE                   ((uint16_t)0x8000)
05762 /********************  Bit definition for SPI_CR2 register  ********************/
05763 #define  SPI_CR2_RXDMAEN                    ((uint8_t)0x01)
05764 #define  SPI_CR2_TXDMAEN                    ((uint8_t)0x02)
05765 #define  SPI_CR2_SSOE                       ((uint8_t)0x04)
05766 #define  SPI_CR2_ERRIE                      ((uint8_t)0x20)
05767 #define  SPI_CR2_RXNEIE                     ((uint8_t)0x40)
05768 #define  SPI_CR2_TXEIE                      ((uint8_t)0x80)
05770 /********************  Bit definition for SPI_SR register  ********************/
05771 #define  SPI_SR_RXNE                        ((uint8_t)0x01)
05772 #define  SPI_SR_TXE                         ((uint8_t)0x02)
05773 #define  SPI_SR_CHSIDE                      ((uint8_t)0x04)
05774 #define  SPI_SR_UDR                         ((uint8_t)0x08)
05775 #define  SPI_SR_CRCERR                      ((uint8_t)0x10)
05776 #define  SPI_SR_MODF                        ((uint8_t)0x20)
05777 #define  SPI_SR_OVR                         ((uint8_t)0x40)
05778 #define  SPI_SR_BSY                         ((uint8_t)0x80)
05780 /********************  Bit definition for SPI_DR register  ********************/
05781 #define  SPI_DR_DR                          ((uint16_t)0xFFFF)
05783 /******************  Bit definition for SPI_CRCPR register  *******************/
05784 #define  SPI_CRCPR_CRCPOLY                  ((uint16_t)0xFFFF)
05786 /******************  Bit definition for SPI_RXCRCR register  ******************/
05787 #define  SPI_RXCRCR_RXCRC                   ((uint16_t)0xFFFF)
05789 /******************  Bit definition for SPI_TXCRCR register  ******************/
05790 #define  SPI_TXCRCR_TXCRC                   ((uint16_t)0xFFFF)
05792 /******************  Bit definition for SPI_I2SCFGR register  *****************/
05793 #define  SPI_I2SCFGR_CHLEN                  ((uint16_t)0x0001)
05795 #define  SPI_I2SCFGR_DATLEN                 ((uint16_t)0x0006)
05796 #define  SPI_I2SCFGR_DATLEN_0               ((uint16_t)0x0002)
05797 #define  SPI_I2SCFGR_DATLEN_1               ((uint16_t)0x0004)
05799 #define  SPI_I2SCFGR_CKPOL                  ((uint16_t)0x0008)
05801 #define  SPI_I2SCFGR_I2SSTD                 ((uint16_t)0x0030)
05802 #define  SPI_I2SCFGR_I2SSTD_0               ((uint16_t)0x0010)
05803 #define  SPI_I2SCFGR_I2SSTD_1               ((uint16_t)0x0020)
05805 #define  SPI_I2SCFGR_PCMSYNC                ((uint16_t)0x0080)
05807 #define  SPI_I2SCFGR_I2SCFG                 ((uint16_t)0x0300)
05808 #define  SPI_I2SCFGR_I2SCFG_0               ((uint16_t)0x0100)
05809 #define  SPI_I2SCFGR_I2SCFG_1               ((uint16_t)0x0200)
05811 #define  SPI_I2SCFGR_I2SE                   ((uint16_t)0x0400)
05812 #define  SPI_I2SCFGR_I2SMOD                 ((uint16_t)0x0800)
05814 /******************  Bit definition for SPI_I2SPR register  *******************/
05815 #define  SPI_I2SPR_I2SDIV                   ((uint16_t)0x00FF)
05816 #define  SPI_I2SPR_ODD                      ((uint16_t)0x0100)
05817 #define  SPI_I2SPR_MCKOE                    ((uint16_t)0x0200)
05819 /******************************************************************************/
05820 /*                                                                            */
05821 /*                                 SYSCFG                                     */
```

```
05822 /*                                                                            */
05823 /******************************************************************************/
05824 /*****************  Bit definition for SYSCFG_MEMRMP register  ***************/
05825 #define SYSCFG_MEMRMP_MEM_MODE          ((uint32_t)0x00000003)
05826 #define SYSCFG_MEMRMP_MEM_MODE_0        ((uint32_t)0x00000001)
05827 #define SYSCFG_MEMRMP_MEM_MODE_1        ((uint32_t)0x00000002)
05828
05829 /*****************  Bit definition for SYSCFG_PMC register  *****************/
05830 #define SYSCFG_PMC_MII_RMII_SEL         ((uint32_t)0x00800000)
05831 /* Old MII_RMII_SEL bit definition, maintained for legacy purpose */
05832 #define SYSCFG_PMC_MII_RMII             SYSCFG_PMC_MII_RMII_SEL
05833
05834 /*****************  Bit definition for SYSCFG_EXTICR1 register  ***************/
05835 #define SYSCFG_EXTICR1_EXTI0            ((uint16_t)0x000F)
05836 #define SYSCFG_EXTICR1_EXTI1            ((uint16_t)0x00F0)
05837 #define SYSCFG_EXTICR1_EXTI2            ((uint16_t)0x0F00)
05838 #define SYSCFG_EXTICR1_EXTI3            ((uint16_t)0xF000)
05842 #define SYSCFG_EXTICR1_EXTI0_PA         ((uint16_t)0x0000)
05843 #define SYSCFG_EXTICR1_EXTI0_PB         ((uint16_t)0x0001)
05844 #define SYSCFG_EXTICR1_EXTI0_PC         ((uint16_t)0x0002)
05845 #define SYSCFG_EXTICR1_EXTI0_PD         ((uint16_t)0x0003)
05846 #define SYSCFG_EXTICR1_EXTI0_PE         ((uint16_t)0x0004)
05847 #define SYSCFG_EXTICR1_EXTI0_PF         ((uint16_t)0x0005)
05848 #define SYSCFG_EXTICR1_EXTI0_PG         ((uint16_t)0x0006)
05849 #define SYSCFG_EXTICR1_EXTI0_PH         ((uint16_t)0x0007)
05850 #define SYSCFG_EXTICR1_EXTI0_PI         ((uint16_t)0x0008)
05854 #define SYSCFG_EXTICR1_EXTI1_PA         ((uint16_t)0x0000)
05855 #define SYSCFG_EXTICR1_EXTI1_PB         ((uint16_t)0x0010)
05856 #define SYSCFG_EXTICR1_EXTI1_PC         ((uint16_t)0x0020)
05857 #define SYSCFG_EXTICR1_EXTI1_PD         ((uint16_t)0x0030)
05858 #define SYSCFG_EXTICR1_EXTI1_PE         ((uint16_t)0x0040)
05859 #define SYSCFG_EXTICR1_EXTI1_PF         ((uint16_t)0x0050)
05860 #define SYSCFG_EXTICR1_EXTI1_PG         ((uint16_t)0x0060)
05861 #define SYSCFG_EXTICR1_EXTI1_PH         ((uint16_t)0x0070)
05862 #define SYSCFG_EXTICR1_EXTI1_PI         ((uint16_t)0x0080)
05866 #define SYSCFG_EXTICR1_EXTI2_PA         ((uint16_t)0x0000)
05867 #define SYSCFG_EXTICR1_EXTI2_PB         ((uint16_t)0x0100)
05868 #define SYSCFG_EXTICR1_EXTI2_PC         ((uint16_t)0x0200)
05869 #define SYSCFG_EXTICR1_EXTI2_PD         ((uint16_t)0x0300)
05870 #define SYSCFG_EXTICR1_EXTI2_PE         ((uint16_t)0x0400)
05871 #define SYSCFG_EXTICR1_EXTI2_PF         ((uint16_t)0x0500)
05872 #define SYSCFG_EXTICR1_EXTI2_PG         ((uint16_t)0x0600)
05873 #define SYSCFG_EXTICR1_EXTI2_PH         ((uint16_t)0x0700)
05874 #define SYSCFG_EXTICR1_EXTI2_PI         ((uint16_t)0x0800)
05878 #define SYSCFG_EXTICR1_EXTI3_PA         ((uint16_t)0x0000)
05879 #define SYSCFG_EXTICR1_EXTI3_PB         ((uint16_t)0x1000)
05880 #define SYSCFG_EXTICR1_EXTI3_PC         ((uint16_t)0x2000)
05881 #define SYSCFG_EXTICR1_EXTI3_PD         ((uint16_t)0x3000)
05882 #define SYSCFG_EXTICR1_EXTI3_PE         ((uint16_t)0x4000)
05883 #define SYSCFG_EXTICR1_EXTI3_PF         ((uint16_t)0x5000)
05884 #define SYSCFG_EXTICR1_EXTI3_PG         ((uint16_t)0x6000)
05885 #define SYSCFG_EXTICR1_EXTI3_PH         ((uint16_t)0x7000)
05886 #define SYSCFG_EXTICR1_EXTI3_PI         ((uint16_t)0x8000)
05888 /*****************  Bit definition for SYSCFG_EXTICR2 register  ***************/
05889 #define SYSCFG_EXTICR2_EXTI4            ((uint16_t)0x000F)
05890 #define SYSCFG_EXTICR2_EXTI5            ((uint16_t)0x00F0)
05891 #define SYSCFG_EXTICR2_EXTI6            ((uint16_t)0x0F00)
05892 #define SYSCFG_EXTICR2_EXTI7            ((uint16_t)0xF000)
05896 #define SYSCFG_EXTICR2_EXTI4_PA         ((uint16_t)0x0000)
05897 #define SYSCFG_EXTICR2_EXTI4_PB         ((uint16_t)0x0001)
05898 #define SYSCFG_EXTICR2_EXTI4_PC         ((uint16_t)0x0002)
05899 #define SYSCFG_EXTICR2_EXTI4_PD         ((uint16_t)0x0003)
05900 #define SYSCFG_EXTICR2_EXTI4_PE         ((uint16_t)0x0004)
05901 #define SYSCFG_EXTICR2_EXTI4_PF         ((uint16_t)0x0005)
05902 #define SYSCFG_EXTICR2_EXTI4_PG         ((uint16_t)0x0006)
05903 #define SYSCFG_EXTICR2_EXTI4_PH         ((uint16_t)0x0007)
05904 #define SYSCFG_EXTICR2_EXTI4_PI         ((uint16_t)0x0008)
05908 #define SYSCFG_EXTICR2_EXTI5_PA         ((uint16_t)0x0000)
05909 #define SYSCFG_EXTICR2_EXTI5_PB         ((uint16_t)0x0010)
05910 #define SYSCFG_EXTICR2_EXTI5_PC         ((uint16_t)0x0020)
05911 #define SYSCFG_EXTICR2_EXTI5_PD         ((uint16_t)0x0030)
05912 #define SYSCFG_EXTICR2_EXTI5_PE         ((uint16_t)0x0040)
05913 #define SYSCFG_EXTICR2_EXTI5_PF         ((uint16_t)0x0050)
05914 #define SYSCFG_EXTICR2_EXTI5_PG         ((uint16_t)0x0060)
05915 #define SYSCFG_EXTICR2_EXTI5_PH         ((uint16_t)0x0070)
05916 #define SYSCFG_EXTICR2_EXTI5_PI         ((uint16_t)0x0080)
05920 #define SYSCFG_EXTICR2_EXTI6_PA         ((uint16_t)0x0000)
05921 #define SYSCFG_EXTICR2_EXTI6_PB         ((uint16_t)0x0100)
05922 #define SYSCFG_EXTICR2_EXTI6_PC         ((uint16_t)0x0200)
05923 #define SYSCFG_EXTICR2_EXTI6_PD         ((uint16_t)0x0300)
05924 #define SYSCFG_EXTICR2_EXTI6_PE         ((uint16_t)0x0400)
05925 #define SYSCFG_EXTICR2_EXTI6_PF         ((uint16_t)0x0500)
05926 #define SYSCFG_EXTICR2_EXTI6_PG         ((uint16_t)0x0600)
05927 #define SYSCFG_EXTICR2_EXTI6_PH         ((uint16_t)0x0700)
05928 #define SYSCFG_EXTICR2_EXTI6_PI         ((uint16_t)0x0800)
05932 #define SYSCFG_EXTICR2_EXTI7_PA         ((uint16_t)0x0000)
05933 #define SYSCFG_EXTICR2_EXTI7_PB         ((uint16_t)0x1000)
```

```
05934 #define SYSCFG_EXTICR2_EXTI7_PC         ((uint16_t)0x2000)
05935 #define SYSCFG_EXTICR2_EXTI7_PD         ((uint16_t)0x3000)
05936 #define SYSCFG_EXTICR2_EXTI7_PE         ((uint16_t)0x4000)
05937 #define SYSCFG_EXTICR2_EXTI7_PF         ((uint16_t)0x5000)
05938 #define SYSCFG_EXTICR2_EXTI7_PG         ((uint16_t)0x6000)
05939 #define SYSCFG_EXTICR2_EXTI7_PH         ((uint16_t)0x7000)
05940 #define SYSCFG_EXTICR2_EXTI7_PI         ((uint16_t)0x8000)
05942 /*****************  Bit definition for SYSCFG_EXTICR3 register  ***************/
05943 #define SYSCFG_EXTICR3_EXTI8            ((uint16_t)0x000F)
05944 #define SYSCFG_EXTICR3_EXTI9            ((uint16_t)0x00F0)
05945 #define SYSCFG_EXTICR3_EXTI10           ((uint16_t)0x0F00)
05946 #define SYSCFG_EXTICR3_EXTI11           ((uint16_t)0xF000)
05951 #define SYSCFG_EXTICR3_EXTI8_PA         ((uint16_t)0x0000)
05952 #define SYSCFG_EXTICR3_EXTI8_PB         ((uint16_t)0x0001)
05953 #define SYSCFG_EXTICR3_EXTI8_PC         ((uint16_t)0x0002)
05954 #define SYSCFG_EXTICR3_EXTI8_PD         ((uint16_t)0x0003)
05955 #define SYSCFG_EXTICR3_EXTI8_PE         ((uint16_t)0x0004)
05956 #define SYSCFG_EXTICR3_EXTI8_PF         ((uint16_t)0x0005)
05957 #define SYSCFG_EXTICR3_EXTI8_PG         ((uint16_t)0x0006)
05958 #define SYSCFG_EXTICR3_EXTI8_PH         ((uint16_t)0x0007)
05959 #define SYSCFG_EXTICR3_EXTI8_PI         ((uint16_t)0x0008)
05963 #define SYSCFG_EXTICR3_EXTI9_PA         ((uint16_t)0x0000)
05964 #define SYSCFG_EXTICR3_EXTI9_PB         ((uint16_t)0x0010)
05965 #define SYSCFG_EXTICR3_EXTI9_PC         ((uint16_t)0x0020)
05966 #define SYSCFG_EXTICR3_EXTI9_PD         ((uint16_t)0x0030)
05967 #define SYSCFG_EXTICR3_EXTI9_PE         ((uint16_t)0x0040)
05968 #define SYSCFG_EXTICR3_EXTI9_PF         ((uint16_t)0x0050)
05969 #define SYSCFG_EXTICR3_EXTI9_PG         ((uint16_t)0x0060)
05970 #define SYSCFG_EXTICR3_EXTI9_PH         ((uint16_t)0x0070)
05971 #define SYSCFG_EXTICR3_EXTI9_PI         ((uint16_t)0x0080)
05975 #define SYSCFG_EXTICR3_EXTI10_PA        ((uint16_t)0x0000)
05976 #define SYSCFG_EXTICR3_EXTI10_PB        ((uint16_t)0x0100)
05977 #define SYSCFG_EXTICR3_EXTI10_PC        ((uint16_t)0x0200)
05978 #define SYSCFG_EXTICR3_EXTI10_PD        ((uint16_t)0x0300)
05979 #define SYSCFG_EXTICR3_EXTI10_PE        ((uint16_t)0x0400)
05980 #define SYSCFG_EXTICR3_EXTI10_PF        ((uint16_t)0x0500)
05981 #define SYSCFG_EXTICR3_EXTI10_PG        ((uint16_t)0x0600)
05982 #define SYSCFG_EXTICR3_EXTI10_PH        ((uint16_t)0x0700)
05983 #define SYSCFG_EXTICR3_EXTI10_PI        ((uint16_t)0x0800)
05987 #define SYSCFG_EXTICR3_EXTI11_PA        ((uint16_t)0x0000)
05988 #define SYSCFG_EXTICR3_EXTI11_PB        ((uint16_t)0x1000)
05989 #define SYSCFG_EXTICR3_EXTI11_PC        ((uint16_t)0x2000)
05990 #define SYSCFG_EXTICR3_EXTI11_PD        ((uint16_t)0x3000)
05991 #define SYSCFG_EXTICR3_EXTI11_PE        ((uint16_t)0x4000)
05992 #define SYSCFG_EXTICR3_EXTI11_PF        ((uint16_t)0x5000)
05993 #define SYSCFG_EXTICR3_EXTI11_PG        ((uint16_t)0x6000)
05994 #define SYSCFG_EXTICR3_EXTI11_PH        ((uint16_t)0x7000)
05995 #define SYSCFG_EXTICR3_EXTI11_PI        ((uint16_t)0x8000)
05997 /*****************  Bit definition for SYSCFG_EXTICR4 register  ***************/
05998 #define SYSCFG_EXTICR4_EXTI12           ((uint16_t)0x000F)
05999 #define SYSCFG_EXTICR4_EXTI13           ((uint16_t)0x00F0)
06000 #define SYSCFG_EXTICR4_EXTI14           ((uint16_t)0x0F00)
06001 #define SYSCFG_EXTICR4_EXTI15           ((uint16_t)0xF000)
06005 #define SYSCFG_EXTICR4_EXTI12_PA        ((uint16_t)0x0000)
06006 #define SYSCFG_EXTICR4_EXTI12_PB        ((uint16_t)0x0001)
06007 #define SYSCFG_EXTICR4_EXTI12_PC        ((uint16_t)0x0002)
06008 #define SYSCFG_EXTICR4_EXTI12_PD        ((uint16_t)0x0003)
06009 #define SYSCFG_EXTICR4_EXTI12_PE        ((uint16_t)0x0004)
06010 #define SYSCFG_EXTICR4_EXTI12_PF        ((uint16_t)0x0005)
06011 #define SYSCFG_EXTICR4_EXTI12_PG        ((uint16_t)0x0006)
06012 #define SYSCFG_EXTICR3_EXTI12_PH        ((uint16_t)0x0007)
06016 #define SYSCFG_EXTICR4_EXTI13_PA        ((uint16_t)0x0000)
06017 #define SYSCFG_EXTICR4_EXTI13_PB        ((uint16_t)0x0010)
06018 #define SYSCFG_EXTICR4_EXTI13_PC        ((uint16_t)0x0020)
06019 #define SYSCFG_EXTICR4_EXTI13_PD        ((uint16_t)0x0030)
06020 #define SYSCFG_EXTICR4_EXTI13_PE        ((uint16_t)0x0040)
06021 #define SYSCFG_EXTICR4_EXTI13_PF        ((uint16_t)0x0050)
06022 #define SYSCFG_EXTICR4_EXTI13_PG        ((uint16_t)0x0060)
06023 #define SYSCFG_EXTICR3_EXTI13_PH        ((uint16_t)0x0070)
06027 #define SYSCFG_EXTICR4_EXTI14_PA        ((uint16_t)0x0000)
06028 #define SYSCFG_EXTICR4_EXTI14_PB        ((uint16_t)0x0100)
06029 #define SYSCFG_EXTICR4_EXTI14_PC        ((uint16_t)0x0200)
06030 #define SYSCFG_EXTICR4_EXTI14_PD        ((uint16_t)0x0300)
06031 #define SYSCFG_EXTICR4_EXTI14_PE        ((uint16_t)0x0400)
06032 #define SYSCFG_EXTICR4_EXTI14_PF        ((uint16_t)0x0500)
06033 #define SYSCFG_EXTICR4_EXTI14_PG        ((uint16_t)0x0600)
06034 #define SYSCFG_EXTICR3_EXTI14_PH        ((uint16_t)0x0700)
06038 #define SYSCFG_EXTICR4_EXTI15_PA        ((uint16_t)0x0000)
06039 #define SYSCFG_EXTICR4_EXTI15_PB        ((uint16_t)0x1000)
06040 #define SYSCFG_EXTICR4_EXTI15_PC        ((uint16_t)0x2000)
06041 #define SYSCFG_EXTICR4_EXTI15_PD        ((uint16_t)0x3000)
06042 #define SYSCFG_EXTICR4_EXTI15_PE        ((uint16_t)0x4000)
06043 #define SYSCFG_EXTICR4_EXTI15_PF        ((uint16_t)0x5000)
06044 #define SYSCFG_EXTICR4_EXTI15_PG        ((uint16_t)0x6000)
06045 #define SYSCFG_EXTICR3_EXTI15_PH        ((uint16_t)0x7000)
06047 /*****************  Bit definition for SYSCFG_CMPCR register  ***************/
06048 #define SYSCFG_CMPCR_CMP_PD             ((uint32_t)0x00000001)
```

```
06049 #define SYSCFG_CMPCR_READY              ((uint32_t)0x00000100)
06051 /******************************************************************************/
06052 /*                                                                            */
06053 /*                                 TIM                                        */
06054 /*                                                                            */
06055 /******************************************************************************/
06056 /*******************  Bit definition for TIM_CR1 register  ********************/
06057 #define  TIM_CR1_CEN                        ((uint16_t)0x0001)
06058 #define  TIM_CR1_UDIS                       ((uint16_t)0x0002)
06059 #define  TIM_CR1_URS                        ((uint16_t)0x0004)
06060 #define  TIM_CR1_OPM                        ((uint16_t)0x0008)
06061 #define  TIM_CR1_DIR                        ((uint16_t)0x0010)
06063 #define  TIM_CR1_CMS                        ((uint16_t)0x0060)
06064 #define  TIM_CR1_CMS_0                      ((uint16_t)0x0020)
06065 #define  TIM_CR1_CMS_1                      ((uint16_t)0x0040)
06067 #define  TIM_CR1_ARPE                       ((uint16_t)0x0080)
06069 #define  TIM_CR1_CKD                        ((uint16_t)0x0300)
06070 #define  TIM_CR1_CKD_0                      ((uint16_t)0x0100)
06071 #define  TIM_CR1_CKD_1                      ((uint16_t)0x0200)
06073 /*******************  Bit definition for TIM_CR2 register  ********************/
06074 #define  TIM_CR2_CCPC                       ((uint16_t)0x0001)
06075 #define  TIM_CR2_CCUS                       ((uint16_t)0x0004)
06076 #define  TIM_CR2_CCDS                       ((uint16_t)0x0008)
06078 #define  TIM_CR2_MMS                        ((uint16_t)0x0070)
06079 #define  TIM_CR2_MMS_0                      ((uint16_t)0x0010)
06080 #define  TIM_CR2_MMS_1                      ((uint16_t)0x0020)
06081 #define  TIM_CR2_MMS_2                      ((uint16_t)0x0040)
06083 #define  TIM_CR2_TI1S                       ((uint16_t)0x0080)
06084 #define  TIM_CR2_OIS1                       ((uint16_t)0x0100)
06085 #define  TIM_CR2_OIS1N                      ((uint16_t)0x0200)
06086 #define  TIM_CR2_OIS2                       ((uint16_t)0x0400)
06087 #define  TIM_CR2_OIS2N                      ((uint16_t)0x0800)
06088 #define  TIM_CR2_OIS3                       ((uint16_t)0x1000)
06089 #define  TIM_CR2_OIS3N                      ((uint16_t)0x2000)
06090 #define  TIM_CR2_OIS4                       ((uint16_t)0x4000)
06092 /*******************  Bit definition for TIM_SMCR register  *******************/
06093 #define  TIM_SMCR_SMS                       ((uint16_t)0x0007)
06094 #define  TIM_SMCR_SMS_0                     ((uint16_t)0x0001)
06095 #define  TIM_SMCR_SMS_1                     ((uint16_t)0x0002)
06096 #define  TIM_SMCR_SMS_2                     ((uint16_t)0x0004)
06098 #define  TIM_SMCR_TS                        ((uint16_t)0x0070)
06099 #define  TIM_SMCR_TS_0                      ((uint16_t)0x0010)
06100 #define  TIM_SMCR_TS_1                      ((uint16_t)0x0020)
06101 #define  TIM_SMCR_TS_2                      ((uint16_t)0x0040)
06103 #define  TIM_SMCR_MSM                       ((uint16_t)0x0080)
06105 #define  TIM_SMCR_ETF                       ((uint16_t)0x0F00)
06106 #define  TIM_SMCR_ETF_0                     ((uint16_t)0x0100)
06107 #define  TIM_SMCR_ETF_1                     ((uint16_t)0x0200)
06108 #define  TIM_SMCR_ETF_2                     ((uint16_t)0x0400)
06109 #define  TIM_SMCR_ETF_3                     ((uint16_t)0x0800)
06111 #define  TIM_SMCR_ETPS                      ((uint16_t)0x3000)
06112 #define  TIM_SMCR_ETPS_0                    ((uint16_t)0x1000)
06113 #define  TIM_SMCR_ETPS_1                    ((uint16_t)0x2000)
06115 #define  TIM_SMCR_ECE                       ((uint16_t)0x4000)
06116 #define  TIM_SMCR_ETP                       ((uint16_t)0x8000)
06118 /*******************  Bit definition for TIM_DIER register  *******************/
06119 #define  TIM_DIER_UIE                       ((uint16_t)0x0001)
06120 #define  TIM_DIER_CC1IE                     ((uint16_t)0x0002)
06121 #define  TIM_DIER_CC2IE                     ((uint16_t)0x0004)
06122 #define  TIM_DIER_CC3IE                     ((uint16_t)0x0008)
06123 #define  TIM_DIER_CC4IE                     ((uint16_t)0x0010)
06124 #define  TIM_DIER_COMIE                     ((uint16_t)0x0020)
06125 #define  TIM_DIER_TIE                       ((uint16_t)0x0040)
06126 #define  TIM_DIER_BIE                       ((uint16_t)0x0080)
06127 #define  TIM_DIER_UDE                       ((uint16_t)0x0100)
06128 #define  TIM_DIER_CC1DE                     ((uint16_t)0x0200)
06129 #define  TIM_DIER_CC2DE                     ((uint16_t)0x0400)
06130 #define  TIM_DIER_CC3DE                     ((uint16_t)0x0800)
06131 #define  TIM_DIER_CC4DE                     ((uint16_t)0x1000)
06132 #define  TIM_DIER_COMDE                     ((uint16_t)0x2000)
06133 #define  TIM_DIER_TDE                       ((uint16_t)0x4000)
06135 /********************  Bit definition for TIM_SR register  ********************/
06136 #define  TIM_SR_UIF                         ((uint16_t)0x0001)
06137 #define  TIM_SR_CC1IF                       ((uint16_t)0x0002)
06138 #define  TIM_SR_CC2IF                       ((uint16_t)0x0004)
06139 #define  TIM_SR_CC3IF                       ((uint16_t)0x0008)
06140 #define  TIM_SR_CC4IF                       ((uint16_t)0x0010)
06141 #define  TIM_SR_COMIF                       ((uint16_t)0x0020)
06142 #define  TIM_SR_TIF                         ((uint16_t)0x0040)
06143 #define  TIM_SR_BIF                         ((uint16_t)0x0080)
06144 #define  TIM_SR_CC1OF                       ((uint16_t)0x0200)
06145 #define  TIM_SR_CC2OF                       ((uint16_t)0x0400)
06146 #define  TIM_SR_CC3OF                       ((uint16_t)0x0800)
06147 #define  TIM_SR_CC4OF                       ((uint16_t)0x1000)
06149 /*******************  Bit definition for TIM_EGR register  ********************/
06150 #define  TIM_EGR_UG                         ((uint8_t)0x01)
06151 #define  TIM_EGR_CC1G                       ((uint8_t)0x02)
```

```
06152 #define  TIM_EGR_CC2G                        ((uint8_t)0x04)
06153 #define  TIM_EGR_CC3G                        ((uint8_t)0x08)
06154 #define  TIM_EGR_CC4G                        ((uint8_t)0x10)
06155 #define  TIM_EGR_COMG                        ((uint8_t)0x20)
06156 #define  TIM_EGR_TG                          ((uint8_t)0x40)
06157 #define  TIM_EGR_BG                          ((uint8_t)0x80)
06159 /******************  Bit definition for TIM_CCMR1 register  ******************/
06160 #define  TIM_CCMR1_CC1S                      ((uint16_t)0x0003)
06161 #define  TIM_CCMR1_CC1S_0                    ((uint16_t)0x0001)
06162 #define  TIM_CCMR1_CC1S_1                    ((uint16_t)0x0002)
06164 #define  TIM_CCMR1_OC1FE                     ((uint16_t)0x0004)
06165 #define  TIM_CCMR1_OC1PE                     ((uint16_t)0x0008)
06167 #define  TIM_CCMR1_OC1M                      ((uint16_t)0x0070)
06168 #define  TIM_CCMR1_OC1M_0                    ((uint16_t)0x0010)
06169 #define  TIM_CCMR1_OC1M_1                    ((uint16_t)0x0020)
06170 #define  TIM_CCMR1_OC1M_2                    ((uint16_t)0x0040)
06172 #define  TIM_CCMR1_OC1CE                     ((uint16_t)0x0080)
06174 #define  TIM_CCMR1_CC2S                      ((uint16_t)0x0300)
06175 #define  TIM_CCMR1_CC2S_0                    ((uint16_t)0x0100)
06176 #define  TIM_CCMR1_CC2S_1                    ((uint16_t)0x0200)
06178 #define  TIM_CCMR1_OC2FE                     ((uint16_t)0x0400)
06179 #define  TIM_CCMR1_OC2PE                     ((uint16_t)0x0800)
06181 #define  TIM_CCMR1_OC2M                      ((uint16_t)0x7000)
06182 #define  TIM_CCMR1_OC2M_0                    ((uint16_t)0x1000)
06183 #define  TIM_CCMR1_OC2M_1                    ((uint16_t)0x2000)
06184 #define  TIM_CCMR1_OC2M_2                    ((uint16_t)0x4000)
06186 #define  TIM_CCMR1_OC2CE                     ((uint16_t)0x8000)
06188 /*----------------------------------------------------------------------------*/
06189
06190 #define  TIM_CCMR1_IC1PSC                    ((uint16_t)0x000C)
06191 #define  TIM_CCMR1_IC1PSC_0                  ((uint16_t)0x0004)
06192 #define  TIM_CCMR1_IC1PSC_1                  ((uint16_t)0x0008)
06194 #define  TIM_CCMR1_IC1F                      ((uint16_t)0x00F0)
06195 #define  TIM_CCMR1_IC1F_0                    ((uint16_t)0x0010)
06196 #define  TIM_CCMR1_IC1F_1                    ((uint16_t)0x0020)
06197 #define  TIM_CCMR1_IC1F_2                    ((uint16_t)0x0040)
06198 #define  TIM_CCMR1_IC1F_3                    ((uint16_t)0x0080)
06200 #define  TIM_CCMR1_IC2PSC                    ((uint16_t)0x0C00)
06201 #define  TIM_CCMR1_IC2PSC_0                  ((uint16_t)0x0400)
06202 #define  TIM_CCMR1_IC2PSC_1                  ((uint16_t)0x0800)
06204 #define  TIM_CCMR1_IC2F                      ((uint16_t)0xF000)
06205 #define  TIM_CCMR1_IC2F_0                    ((uint16_t)0x1000)
06206 #define  TIM_CCMR1_IC2F_1                    ((uint16_t)0x2000)
06207 #define  TIM_CCMR1_IC2F_2                    ((uint16_t)0x4000)
06208 #define  TIM_CCMR1_IC2F_3                    ((uint16_t)0x8000)
06210 /******************  Bit definition for TIM_CCMR2 register  ******************/
06211 #define  TIM_CCMR2_CC3S                      ((uint16_t)0x0003)
06212 #define  TIM_CCMR2_CC3S_0                    ((uint16_t)0x0001)
06213 #define  TIM_CCMR2_CC3S_1                    ((uint16_t)0x0002)
06215 #define  TIM_CCMR2_OC3FE                     ((uint16_t)0x0004)
06216 #define  TIM_CCMR2_OC3PE                     ((uint16_t)0x0008)
06218 #define  TIM_CCMR2_OC3M                      ((uint16_t)0x0070)
06219 #define  TIM_CCMR2_OC3M_0                    ((uint16_t)0x0010)
06220 #define  TIM_CCMR2_OC3M_1                    ((uint16_t)0x0020)
06221 #define  TIM_CCMR2_OC3M_2                    ((uint16_t)0x0040)
06223 #define  TIM_CCMR2_OC3CE                     ((uint16_t)0x0080)
06225 #define  TIM_CCMR2_CC4S                      ((uint16_t)0x0300)
06226 #define  TIM_CCMR2_CC4S_0                    ((uint16_t)0x0100)
06227 #define  TIM_CCMR2_CC4S_1                    ((uint16_t)0x0200)
06229 #define  TIM_CCMR2_OC4FE                     ((uint16_t)0x0400)
06230 #define  TIM_CCMR2_OC4PE                     ((uint16_t)0x0800)
06232 #define  TIM_CCMR2_OC4M                      ((uint16_t)0x7000)
06233 #define  TIM_CCMR2_OC4M_0                    ((uint16_t)0x1000)
06234 #define  TIM_CCMR2_OC4M_1                    ((uint16_t)0x2000)
06235 #define  TIM_CCMR2_OC4M_2                    ((uint16_t)0x4000)
06237 #define  TIM_CCMR2_OC4CE                     ((uint16_t)0x8000)
06239 /*----------------------------------------------------------------------------*/
06240
06241 #define  TIM_CCMR2_IC3PSC                    ((uint16_t)0x000C)
06242 #define  TIM_CCMR2_IC3PSC_0                  ((uint16_t)0x0004)
06243 #define  TIM_CCMR2_IC3PSC_1                  ((uint16_t)0x0008)
06245 #define  TIM_CCMR2_IC3F                      ((uint16_t)0x00F0)
06246 #define  TIM_CCMR2_IC3F_0                    ((uint16_t)0x0010)
06247 #define  TIM_CCMR2_IC3F_1                    ((uint16_t)0x0020)
06248 #define  TIM_CCMR2_IC3F_2                    ((uint16_t)0x0040)
06249 #define  TIM_CCMR2_IC3F_3                    ((uint16_t)0x0080)
06251 #define  TIM_CCMR2_IC4PSC                    ((uint16_t)0x0C00)
06252 #define  TIM_CCMR2_IC4PSC_0                  ((uint16_t)0x0400)
06253 #define  TIM_CCMR2_IC4PSC_1                  ((uint16_t)0x0800)
06255 #define  TIM_CCMR2_IC4F                      ((uint16_t)0xF000)
06256 #define  TIM_CCMR2_IC4F_0                    ((uint16_t)0x1000)
06257 #define  TIM_CCMR2_IC4F_1                    ((uint16_t)0x2000)
06258 #define  TIM_CCMR2_IC4F_2                    ((uint16_t)0x4000)
06259 #define  TIM_CCMR2_IC4F_3                    ((uint16_t)0x8000)
06261 /******************  Bit definition for TIM_CCER register  ******************/
06262 #define  TIM_CCER_CC1E                       ((uint16_t)0x0001)
06263 #define  TIM_CCER_CC1P                       ((uint16_t)0x0002)
```

```
06264 #define  TIM_CCER_CC1NE                      ((uint16_t)0x0004)
06265 #define  TIM_CCER_CC1NP                      ((uint16_t)0x0008)
06266 #define  TIM_CCER_CC2E                       ((uint16_t)0x0010)
06267 #define  TIM_CCER_CC2P                       ((uint16_t)0x0020)
06268 #define  TIM_CCER_CC2NE                      ((uint16_t)0x0040)
06269 #define  TIM_CCER_CC2NP                      ((uint16_t)0x0080)
06270 #define  TIM_CCER_CC3E                       ((uint16_t)0x0100)
06271 #define  TIM_CCER_CC3P                       ((uint16_t)0x0200)
06272 #define  TIM_CCER_CC3NE                      ((uint16_t)0x0400)
06273 #define  TIM_CCER_CC3NP                      ((uint16_t)0x0800)
06274 #define  TIM_CCER_CC4E                       ((uint16_t)0x1000)
06275 #define  TIM_CCER_CC4P                       ((uint16_t)0x2000)
06276 #define  TIM_CCER_CC4NP                      ((uint16_t)0x8000)
06278 /*******************  Bit definition for TIM_CNT register  ********************/
06279 #define  TIM_CNT_CNT                         ((uint16_t)0xFFFF)
06281 /*******************  Bit definition for TIM_PSC register  ********************/
06282 #define  TIM_PSC_PSC                         ((uint16_t)0xFFFF)
06284 /*******************  Bit definition for TIM_ARR register  ********************/
06285 #define  TIM_ARR_ARR                         ((uint16_t)0xFFFF)
06287 /*******************  Bit definition for TIM_RCR register  ********************/
06288 #define  TIM_RCR_REP                         ((uint8_t)0xFF)
06290 /*******************  Bit definition for TIM_CCR1 register  *******************/
06291 #define  TIM_CCR1_CCR1                       ((uint16_t)0xFFFF)
06293 /*******************  Bit definition for TIM_CCR2 register  *******************/
06294 #define  TIM_CCR2_CCR2                       ((uint16_t)0xFFFF)
06296 /*******************  Bit definition for TIM_CCR3 register  *******************/
06297 #define  TIM_CCR3_CCR3                       ((uint16_t)0xFFFF)
06299 /*******************  Bit definition for TIM_CCR4 register  *******************/
06300 #define  TIM_CCR4_CCR4                       ((uint16_t)0xFFFF)
06302 /*******************  Bit definition for TIM_BDTR register  *******************/
06303 #define  TIM_BDTR_DTG                        ((uint16_t)0x00FF)
06304 #define  TIM_BDTR_DTG_0                      ((uint16_t)0x0001)
06305 #define  TIM_BDTR_DTG_1                      ((uint16_t)0x0002)
06306 #define  TIM_BDTR_DTG_2                      ((uint16_t)0x0004)
06307 #define  TIM_BDTR_DTG_3                      ((uint16_t)0x0008)
06308 #define  TIM_BDTR_DTG_4                      ((uint16_t)0x0010)
06309 #define  TIM_BDTR_DTG_5                      ((uint16_t)0x0020)
06310 #define  TIM_BDTR_DTG_6                      ((uint16_t)0x0040)
06311 #define  TIM_BDTR_DTG_7                      ((uint16_t)0x0080)
06313 #define  TIM_BDTR_LOCK                       ((uint16_t)0x0300)
06314 #define  TIM_BDTR_LOCK_0                     ((uint16_t)0x0100)
06315 #define  TIM_BDTR_LOCK_1                     ((uint16_t)0x0200)
06317 #define  TIM_BDTR_OSSI                       ((uint16_t)0x0400)
06318 #define  TIM_BDTR_OSSR                       ((uint16_t)0x0800)
06319 #define  TIM_BDTR_BKE                        ((uint16_t)0x1000)
06320 #define  TIM_BDTR_BKP                        ((uint16_t)0x2000)
06321 #define  TIM_BDTR_AOE                        ((uint16_t)0x4000)
06322 #define  TIM_BDTR_MOE                        ((uint16_t)0x8000)
06324 /*******************  Bit definition for TIM_DCR register  ********************/
06325 #define  TIM_DCR_DBA                         ((uint16_t)0x001F)
06326 #define  TIM_DCR_DBA_0                       ((uint16_t)0x0001)
06327 #define  TIM_DCR_DBA_1                       ((uint16_t)0x0002)
06328 #define  TIM_DCR_DBA_2                       ((uint16_t)0x0004)
06329 #define  TIM_DCR_DBA_3                       ((uint16_t)0x0008)
06330 #define  TIM_DCR_DBA_4                       ((uint16_t)0x0010)
06332 #define  TIM_DCR_DBL                         ((uint16_t)0x1F00)
06333 #define  TIM_DCR_DBL_0                       ((uint16_t)0x0100)
06334 #define  TIM_DCR_DBL_1                       ((uint16_t)0x0200)
06335 #define  TIM_DCR_DBL_2                       ((uint16_t)0x0400)
06336 #define  TIM_DCR_DBL_3                       ((uint16_t)0x0800)
06337 #define  TIM_DCR_DBL_4                       ((uint16_t)0x1000)
06339 /*******************  Bit definition for TIM_DMAR register  *******************/
06340 #define  TIM_DMAR_DMAB                       ((uint16_t)0xFFFF)
06342 /*******************  Bit definition for TIM_OR register  *********************/
06343 #define TIM_OR_TI4_RMP                       ((uint16_t)0x00C0)
06344 #define TIM_OR_TI4_RMP_0                     ((uint16_t)0x0040)
06345 #define TIM_OR_TI4_RMP_1                     ((uint16_t)0x0080)
06346 #define TIM_OR_ITR1_RMP                      ((uint16_t)0x0C00)
06347 #define TIM_OR_ITR1_RMP_0                    ((uint16_t)0x0400)
06348 #define TIM_OR_ITR1_RMP_1                    ((uint16_t)0x0800)
06351 /******************************************************************************/
06352 /*                                                                            */
06353 /*         Universal Synchronous Asynchronous Receiver Transmitter            */
06354 /*                                                                            */
06355 /******************************************************************************/
06356 /*******************  Bit definition for USART_SR register  *******************/
06357 #define  USART_SR_PE                         ((uint16_t)0x0001)
06358 #define  USART_SR_FE                         ((uint16_t)0x0002)
06359 #define  USART_SR_NE                         ((uint16_t)0x0004)
06360 #define  USART_SR_ORE                        ((uint16_t)0x0008)
06361 #define  USART_SR_IDLE                       ((uint16_t)0x0010)
06362 #define  USART_SR_RXNE                       ((uint16_t)0x0020)
06363 #define  USART_SR_TC                         ((uint16_t)0x0040)
06364 #define  USART_SR_TXE                        ((uint16_t)0x0080)
06365 #define  USART_SR_LBD                        ((uint16_t)0x0100)
06366 #define  USART_SR_CTS                        ((uint16_t)0x0200)
06368 /*******************  Bit definition for USART_DR register  *******************/
```

```
06369 #define  USART_DR_DR                         ((uint16_t)0x01FF)
06371 /******************  Bit definition for USART_BRR register  ******************/
06372 #define  USART_BRR_DIV_Fraction              ((uint16_t)0x000F)
06373 #define  USART_BRR_DIV_Mantissa              ((uint16_t)0xFFF0)
06375 /******************  Bit definition for USART_CR1 register  ******************/
06376 #define  USART_CR1_SBK                       ((uint16_t)0x0001)
06377 #define  USART_CR1_RWU                       ((uint16_t)0x0002)
06378 #define  USART_CR1_RE                        ((uint16_t)0x0004)
06379 #define  USART_CR1_TE                        ((uint16_t)0x0008)
06380 #define  USART_CR1_IDLEIE                    ((uint16_t)0x0010)
06381 #define  USART_CR1_RXNEIE                    ((uint16_t)0x0020)
06382 #define  USART_CR1_TCIE                      ((uint16_t)0x0040)
06383 #define  USART_CR1_TXEIE                     ((uint16_t)0x0080)
06384 #define  USART_CR1_PEIE                      ((uint16_t)0x0100)
06385 #define  USART_CR1_PS                        ((uint16_t)0x0200)
06386 #define  USART_CR1_PCE                       ((uint16_t)0x0400)
06387 #define  USART_CR1_WAKE                      ((uint16_t)0x0800)
06388 #define  USART_CR1_M                         ((uint16_t)0x1000)
06389 #define  USART_CR1_UE                        ((uint16_t)0x2000)
06390 #define  USART_CR1_OVER8                     ((uint16_t)0x8000)
06392 /******************  Bit definition for USART_CR2 register  ******************/
06393 #define  USART_CR2_ADD                       ((uint16_t)0x000F)
06394 #define  USART_CR2_LBDL                      ((uint16_t)0x0020)
06395 #define  USART_CR2_LBDIE                     ((uint16_t)0x0040)
06396 #define  USART_CR2_LBCL                      ((uint16_t)0x0100)
06397 #define  USART_CR2_CPHA                      ((uint16_t)0x0200)
06398 #define  USART_CR2_CPOL                      ((uint16_t)0x0400)
06399 #define  USART_CR2_CLKEN                     ((uint16_t)0x0800)
06401 #define  USART_CR2_STOP                      ((uint16_t)0x3000)
06402 #define  USART_CR2_STOP_0                    ((uint16_t)0x1000)
06403 #define  USART_CR2_STOP_1                    ((uint16_t)0x2000)
06405 #define  USART_CR2_LINEN                     ((uint16_t)0x4000)
06407 /******************  Bit definition for USART_CR3 register  ******************/
06408 #define  USART_CR3_EIE                       ((uint16_t)0x0001)
06409 #define  USART_CR3_IREN                      ((uint16_t)0x0002)
06410 #define  USART_CR3_IRLP                      ((uint16_t)0x0004)
06411 #define  USART_CR3_HDSEL                     ((uint16_t)0x0008)
06412 #define  USART_CR3_NACK                      ((uint16_t)0x0010)
06413 #define  USART_CR3_SCEN                      ((uint16_t)0x0020)
06414 #define  USART_CR3_DMAR                      ((uint16_t)0x0040)
06415 #define  USART_CR3_DMAT                      ((uint16_t)0x0080)
06416 #define  USART_CR3_RTSE                      ((uint16_t)0x0100)
06417 #define  USART_CR3_CTSE                      ((uint16_t)0x0200)
06418 #define  USART_CR3_CTSIE                     ((uint16_t)0x0400)
06419 #define  USART_CR3_ONEBIT                    ((uint16_t)0x0800)
06421 /******************  Bit definition for USART_GTPR register  ******************/
06422 #define  USART_GTPR_PSC                      ((uint16_t)0x00FF)
06423 #define  USART_GTPR_PSC_0                    ((uint16_t)0x0001)
06424 #define  USART_GTPR_PSC_1                    ((uint16_t)0x0002)
06425 #define  USART_GTPR_PSC_2                    ((uint16_t)0x0004)
06426 #define  USART_GTPR_PSC_3                    ((uint16_t)0x0008)
06427 #define  USART_GTPR_PSC_4                    ((uint16_t)0x0010)
06428 #define  USART_GTPR_PSC_5                    ((uint16_t)0x0020)
06429 #define  USART_GTPR_PSC_6                    ((uint16_t)0x0040)
06430 #define  USART_GTPR_PSC_7                    ((uint16_t)0x0080)
06432 #define  USART_GTPR_GT                       ((uint16_t)0xFF00)
06434 /*****************************************************************************/
06435 /*                                                                           */
06436 /*                          Window WATCHDOG                                  */
06437 /*                                                                           */
06438 /*****************************************************************************/
06439 /******************  Bit definition for WWDG_CR register  ********************/
06440 #define  WWDG_CR_T                           ((uint8_t)0x7F)
06441 #define  WWDG_CR_T0                          ((uint8_t)0x01)
06442 #define  WWDG_CR_T1                          ((uint8_t)0x02)
06443 #define  WWDG_CR_T2                          ((uint8_t)0x04)
06444 #define  WWDG_CR_T3                          ((uint8_t)0x08)
06445 #define  WWDG_CR_T4                          ((uint8_t)0x10)
06446 #define  WWDG_CR_T5                          ((uint8_t)0x20)
06447 #define  WWDG_CR_T6                          ((uint8_t)0x40)
06449 #define  WWDG_CR_WDGA                        ((uint8_t)0x80)
06451 /******************  Bit definition for WWDG_CFR register  *******************/
06452 #define  WWDG_CFR_W                          ((uint16_t)0x007F)
06453 #define  WWDG_CFR_W0                         ((uint16_t)0x0001)
06454 #define  WWDG_CFR_W1                         ((uint16_t)0x0002)
06455 #define  WWDG_CFR_W2                         ((uint16_t)0x0004)
06456 #define  WWDG_CFR_W3                         ((uint16_t)0x0008)
06457 #define  WWDG_CFR_W4                         ((uint16_t)0x0010)
06458 #define  WWDG_CFR_W5                         ((uint16_t)0x0020)
06459 #define  WWDG_CFR_W6                         ((uint16_t)0x0040)
06461 #define  WWDG_CFR_WDGTB                      ((uint16_t)0x0180)
06462 #define  WWDG_CFR_WDGTB0                     ((uint16_t)0x0080)
06463 #define  WWDG_CFR_WDGTB1                     ((uint16_t)0x0100)
06465 #define  WWDG_CFR_EWI                        ((uint16_t)0x0200)
06467 /******************  Bit definition for WWDG_SR register  ********************/
06468 #define  WWDG_SR_EWIF                        ((uint8_t)0x01)
06471 /*****************************************************************************/
```

```
06472 /*                                                                          */
06473 /*                             DBG                                          */
06474 /*                                                                          */
06475 /****************************************************************************/
06476 /********************  Bit definition for DBGMCU_IDCODE register  *************/
06477 #define  DBGMCU_IDCODE_DEV_ID                ((uint32_t)0x00000FFF)
06478 #define  DBGMCU_IDCODE_REV_ID                ((uint32_t)0xFFFF0000)
06479
06480 /********************  Bit definition for DBGMCU_CR register  *****************/
06481 #define  DBGMCU_CR_DBG_SLEEP                 ((uint32_t)0x00000001)
06482 #define  DBGMCU_CR_DBG_STOP                  ((uint32_t)0x00000002)
06483 #define  DBGMCU_CR_DBG_STANDBY               ((uint32_t)0x00000004)
06484 #define  DBGMCU_CR_TRACE_IOEN                ((uint32_t)0x00000020)
06485
06486 #define  DBGMCU_CR_TRACE_MODE                ((uint32_t)0x000000C0)
06487 #define  DBGMCU_CR_TRACE_MODE_0              ((uint32_t)0x00000040)
06488 #define  DBGMCU_CR_TRACE_MODE_1              ((uint32_t)0x00000080)
06490 /********************  Bit definition for DBGMCU_APB1_FZ register  ************/
06491 #define  DBGMCU_APB1_FZ_DBG_TIM2_STOP            ((uint32_t)0x00000001)
06492 #define  DBGMCU_APB1_FZ_DBG_TIM3_STOP            ((uint32_t)0x00000002)
06493 #define  DBGMCU_APB1_FZ_DBG_TIM4_STOP            ((uint32_t)0x00000004)
06494 #define  DBGMCU_APB1_FZ_DBG_TIM5_STOP            ((uint32_t)0x00000008)
06495 #define  DBGMCU_APB1_FZ_DBG_TIM6_STOP            ((uint32_t)0x00000010)
06496 #define  DBGMCU_APB1_FZ_DBG_TIM7_STOP            ((uint32_t)0x00000020)
06497 #define  DBGMCU_APB1_FZ_DBG_TIM12_STOP           ((uint32_t)0x00000040)
06498 #define  DBGMCU_APB1_FZ_DBG_TIM13_STOP           ((uint32_t)0x00000080)
06499 #define  DBGMCU_APB1_FZ_DBG_TIM14_STOP           ((uint32_t)0x00000100)
06500 #define  DBGMCU_APB1_FZ_DBG_RTC_STOP             ((uint32_t)0x00000400)
06501 #define  DBGMCU_APB1_FZ_DBG_WWDG_STOP            ((uint32_t)0x00000800)
06502 #define  DBGMCU_APB1_FZ_DBG_IWDG_STOP            ((uint32_t)0x00001000)
06503 #define  DBGMCU_APB1_FZ_DBG_I2C1_SMBUS_TIMEOUT   ((uint32_t)0x00200000)
06504 #define  DBGMCU_APB1_FZ_DBG_I2C2_SMBUS_TIMEOUT   ((uint32_t)0x00400000)
06505 #define  DBGMCU_APB1_FZ_DBG_I2C3_SMBUS_TIMEOUT   ((uint32_t)0x00800000)
06506 #define  DBGMCU_APB1_FZ_DBG_CAN1_STOP            ((uint32_t)0x02000000)
06507 #define  DBGMCU_APB1_FZ_DBG_CAN2_STOP            ((uint32_t)0x04000000)
06508 /* Old IWDGSTOP bit definition, maintained for legacy purpose */
06509 #define  DBGMCU_APB1_FZ_DBG_IWDEG_STOP           DBGMCU_APB1_FZ_DBG_IWDG_STOP
06510
06511 /********************  Bit definition for DBGMCU_APB2_FZ register  ************/
06512 #define  DBGMCU_APB1_FZ_DBG_TIM1_STOP        ((uint32_t)0x00000001)
06513 #define  DBGMCU_APB1_FZ_DBG_TIM8_STOP        ((uint32_t)0x00000002)
06514 #define  DBGMCU_APB1_FZ_DBG_TIM9_STOP        ((uint32_t)0x00010000)
06515 #define  DBGMCU_APB1_FZ_DBG_TIM10_STOP       ((uint32_t)0x00020000)
06516 #define  DBGMCU_APB1_FZ_DBG_TIM11_STOP       ((uint32_t)0x00040000)
06517
06518 /****************************************************************************/
06519 /*                                                                          */
06520 /*                 Ethernet MAC Registers bits definitions                  */
06521 /*                                                                          */
06522 /****************************************************************************/
06523 /* Bit definition for Ethernet MAC Control Register register */
06524 #define ETH_MACCR_WD      ((uint32_t)0x00800000)  /* Watchdog disable */
06525 #define ETH_MACCR_JD      ((uint32_t)0x00400000)  /* Jabber disable */
06526 #define ETH_MACCR_IFG     ((uint32_t)0x000E0000)  /* Inter-frame gap */
06527 #define ETH_MACCR_IFG_96Bit     ((uint32_t)0x00000000)  /* Minimum IFG between frames during
      transmission is 96Bit */
06528   #define ETH_MACCR_IFG_88Bit     ((uint32_t)0x00020000)  /* Minimum IFG between frames during
      transmission is 88Bit */
06529   #define ETH_MACCR_IFG_80Bit     ((uint32_t)0x00040000)  /* Minimum IFG between frames during
      transmission is 80Bit */
06530   #define ETH_MACCR_IFG_72Bit     ((uint32_t)0x00060000)  /* Minimum IFG between frames during
      transmission is 72Bit */
06531   #define ETH_MACCR_IFG_64Bit     ((uint32_t)0x00080000)  /* Minimum IFG between frames during
      transmission is 64Bit */
06532   #define ETH_MACCR_IFG_56Bit     ((uint32_t)0x000A0000)  /* Minimum IFG between frames during
      transmission is 56Bit */
06533   #define ETH_MACCR_IFG_48Bit     ((uint32_t)0x000C0000)  /* Minimum IFG between frames during
      transmission is 48Bit */
06534   #define ETH_MACCR_IFG_40Bit     ((uint32_t)0x000E0000)  /* Minimum IFG between frames during
      transmission is 40Bit */
06535 #define ETH_MACCR_CSD     ((uint32_t)0x00010000)  /* Carrier sense disable (during transmission) */
06536 #define ETH_MACCR_FES     ((uint32_t)0x00004000)  /* Fast ethernet speed */
06537 #define ETH_MACCR_ROD     ((uint32_t)0x00002000)  /* Receive own disable */
06538 #define ETH_MACCR_LM      ((uint32_t)0x00001000)  /* loopback mode */
06539 #define ETH_MACCR_DM      ((uint32_t)0x00000800)  /* Duplex mode */
06540 #define ETH_MACCR_IPCO    ((uint32_t)0x00000400)  /* IP Checksum offload */
06541 #define ETH_MACCR_RD      ((uint32_t)0x00000200)  /* Retry disable */
06542 #define ETH_MACCR_APCS    ((uint32_t)0x00000080)  /* Automatic Pad/CRC stripping */
06543 #define ETH_MACCR_BL      ((uint32_t)0x00000060)  /* Back-off limit: random integer number (r) of slot
      time delays before rescheduling
06544                                                     a transmission attempt during retries after a
      collision: 0 =< r <2^k */
06545   #define ETH_MACCR_BL_10    ((uint32_t)0x00000000)  /* k = min (n, 10) */
06546   #define ETH_MACCR_BL_8     ((uint32_t)0x00000020)  /* k = min (n, 8) */
06547   #define ETH_MACCR_BL_4     ((uint32_t)0x00000040)  /* k = min (n, 4) */
06548   #define ETH_MACCR_BL_1     ((uint32_t)0x00000060)  /* k = min (n, 1) */
06549 #define ETH_MACCR_DC      ((uint32_t)0x00000010)  /* Defferal check */
```

```
06550 #define ETH_MACCR_TE      ((uint32_t)0x00000008)  /* Transmitter enable */
06551 #define ETH_MACCR_RE      ((uint32_t)0x00000004)  /* Receiver enable */
06552
06553 /* Bit definition for Ethernet MAC Frame Filter Register */
06554 #define ETH_MACFFR_RA     ((uint32_t)0x80000000)  /* Receive all */
06555 #define ETH_MACFFR_HPF    ((uint32_t)0x00000400)  /* Hash or perfect filter */
06556 #define ETH_MACFFR_SAF    ((uint32_t)0x00000200)  /* Source address filter enable */
06557 #define ETH_MACFFR_SAIF   ((uint32_t)0x00000100)  /* SA inverse filtering */
06558 #define ETH_MACFFR_PCF    ((uint32_t)0x000000C0)  /* Pass control frames: 3 cases */
06559   #define ETH_MACFFR_PCF_BlockAll                ((uint32_t)0x00000040)  /* MAC filters all control
      frames from reaching the application */
06560   #define ETH_MACFFR_PCF_ForwardAll              ((uint32_t)0x00000080)  /* MAC forwards all control
      frames to application even if they fail the Address Filter */
06561   #define ETH_MACFFR_PCF_ForwardPassedAddrFilter ((uint32_t)0x000000C0)  /* MAC forwards control
      frames that pass the Address Filter. */
06562 #define ETH_MACFFR_BFD    ((uint32_t)0x00000020)  /* Broadcast frame disable */
06563 #define ETH_MACFFR_PAM    ((uint32_t)0x00000010)  /* Pass all mutlicast */
06564 #define ETH_MACFFR_DAIF   ((uint32_t)0x00000008)  /* DA Inverse filtering */
06565 #define ETH_MACFFR_HM     ((uint32_t)0x00000004)  /* Hash multicast */
06566 #define ETH_MACFFR_HU     ((uint32_t)0x00000002)  /* Hash unicast */
06567 #define ETH_MACFFR_PM     ((uint32_t)0x00000001)  /* Promiscuous mode */
06568
06569 /* Bit definition for Ethernet MAC Hash Table High Register */
06570 #define ETH_MACHTHR_HTH   ((uint32_t)0xFFFFFFFF)  /* Hash table high */
06571
06572 /* Bit definition for Ethernet MAC Hash Table Low Register */
06573 #define ETH_MACHTLR_HTL   ((uint32_t)0xFFFFFFFF)  /* Hash table low */
06574
06575 /* Bit definition for Ethernet MAC MII Address Register */
06576 #define ETH_MACMIIAR_PA   ((uint32_t)0x0000F800)  /* Physical layer address */
06577 #define ETH_MACMIIAR_MR   ((uint32_t)0x000007C0)  /* MII register in the selected PHY */
06578 #define ETH_MACMIIAR_CR   ((uint32_t)0x0000001C)  /* CR clock range: 6 cases */
06579   #define ETH_MACMIIAR_CR_Div42   ((uint32_t)0x00000000)  /* HCLK:60-100 MHz; MDC clock= HCLK/42 */
06580   #define ETH_MACMIIAR_CR_Div62   ((uint32_t)0x00000004)  /* HCLK:100-150 MHz; MDC clock= HCLK/62 */
06581   #define ETH_MACMIIAR_CR_Div16   ((uint32_t)0x00000008)  /* HCLK:20-35 MHz; MDC clock= HCLK/16 */
06582   #define ETH_MACMIIAR_CR_Div26   ((uint32_t)0x0000000C)  /* HCLK:35-60 MHz; MDC clock= HCLK/26 */
06583   #define ETH_MACMIIAR_CR_Div102  ((uint32_t)0x00000010)  /* HCLK:150-168 MHz; MDC clock= HCLK/102 */

06584 #define ETH_MACMIIAR_MW   ((uint32_t)0x00000002)  /* MII write */
06585 #define ETH_MACMIIAR_MB   ((uint32_t)0x00000001)  /* MII busy */
06586
06587 /* Bit definition for Ethernet MAC MII Data Register */
06588 #define ETH_MACMIIDR_MD   ((uint32_t)0x0000FFFF)  /* MII data: read/write data from/to PHY */
06589
06590 /* Bit definition for Ethernet MAC Flow Control Register */
06591 #define ETH_MACFCR_PT     ((uint32_t)0xFFFF0000)  /* Pause time */
06592 #define ETH_MACFCR_ZQPD   ((uint32_t)0x00000080)  /* Zero-quanta pause disable */
06593 #define ETH_MACFCR_PLT    ((uint32_t)0x00000030)  /* Pause low threshold: 4 cases */
06594   #define ETH_MACFCR_PLT_Minus4   ((uint32_t)0x00000000)  /* Pause time minus 4 slot times */
06595   #define ETH_MACFCR_PLT_Minus28  ((uint32_t)0x00000010)  /* Pause time minus 28 slot times */
06596   #define ETH_MACFCR_PLT_Minus144 ((uint32_t)0x00000020)  /* Pause time minus 144 slot times */
06597   #define ETH_MACFCR_PLT_Minus256 ((uint32_t)0x00000030)  /* Pause time minus 256 slot times */
06598 #define ETH_MACFCR_UPFD   ((uint32_t)0x00000008)  /* Unicast pause frame detect */
06599 #define ETH_MACFCR_RFCE   ((uint32_t)0x00000004)  /* Receive flow control enable */
06600 #define ETH_MACFCR_TFCE   ((uint32_t)0x00000002)  /* Transmit flow control enable */
06601 #define ETH_MACFCR_FCBBPA ((uint32_t)0x00000001)  /* Flow control busy/backpressure activate */
06602
06603 /* Bit definition for Ethernet MAC VLAN Tag Register */
06604 #define ETH_MACVLANTR_VLANTC ((uint32_t)0x00010000)  /* 12-bit VLAN tag comparison */
06605 #define ETH_MACVLANTR_VLANTI ((uint32_t)0x0000FFFF)  /* VLAN tag identifier (for receive frames) */
06606
06607 /* Bit definition for Ethernet MAC Remote Wake-UpFrame Filter Register */
06608 #define ETH_MACRWUFFR_D   ((uint32_t)0xFFFFFFFF)  /* Wake-up frame filter register data */
06609 /* Eight sequential Writes to this address (offset 0x28) will write all Wake-UpFrame Filter Registers.
06610    Eight sequential Reads from this address (offset 0x28) will read all Wake-UpFrame Filter Registers.
      */
06611 /* Wake-UpFrame Filter Reg0 : Filter 0 Byte Mask
06612    Wake-UpFrame Filter Reg1 : Filter 1 Byte Mask
06613    Wake-UpFrame Filter Reg2 : Filter 2 Byte Mask
06614    Wake-UpFrame Filter Reg3 : Filter 3 Byte Mask
06615    Wake-UpFrame Filter Reg4 : RSVD - Filter3 Command - RSVD - Filter2 Command -
06616                               RSVD - Filter1 Command - RSVD - Filter0 Command
06617    Wake-UpFrame Filter Re5 : Filter3 Offset - Filter2 Offset - Filter1 Offset - Filter0 Offset
06618    Wake-UpFrame Filter Re6 : Filter1 CRC16 - Filter0 CRC16
06619    Wake-UpFrame Filter Re7 : Filter3 CRC16 - Filter2 CRC16 */
06620
06621 /* Bit definition for Ethernet MAC PMT Control and Status Register */
06622 #define ETH_MACPMTCSR_WFFRPR ((uint32_t)0x80000000)  /* Wake-Up Frame Filter Register Pointer Reset */
06623 #define ETH_MACPMTCSR_GU     ((uint32_t)0x00000200)  /* Global Unicast */
06624 #define ETH_MACPMTCSR_WFR    ((uint32_t)0x00000040)  /* Wake-Up Frame Received */
06625 #define ETH_MACPMTCSR_MPR    ((uint32_t)0x00000020)  /* Magic Packet Received */
06626 #define ETH_MACPMTCSR_WFE    ((uint32_t)0x00000004)  /* Wake-Up Frame Enable */
06627 #define ETH_MACPMTCSR_MPE    ((uint32_t)0x00000002)  /* Magic Packet Enable */
06628 #define ETH_MACPMTCSR_PD     ((uint32_t)0x00000001)  /* Power Down */
06629
06630 /* Bit definition for Ethernet MAC Status Register */
06631 #define ETH_MACSR_TSTS      ((uint32_t)0x00000200)  /* Time stamp trigger status */
```

```
06632 #define ETH_MACSR_MMCTS     ((uint32_t)0x00000040)  /* MMC transmit status */
06633 #define ETH_MACSR_MMMCRS    ((uint32_t)0x00000020)  /* MMC receive status */
06634 #define ETH_MACSR_MMCS      ((uint32_t)0x00000010)  /* MMC status */
06635 #define ETH_MACSR_PMTS      ((uint32_t)0x00000008)  /* PMT status */
06636
06637 /* Bit definition for Ethernet MAC Interrupt Mask Register */
06638 #define ETH_MACIMR_TSTIM     ((uint32_t)0x00000200)  /* Time stamp trigger interrupt mask */
06639 #define ETH_MACIMR_PMTIM     ((uint32_t)0x00000008)  /* PMT interrupt mask */
06640
06641 /* Bit definition for Ethernet MAC Address0 High Register */
06642 #define ETH_MACA0HR_MACA0H   ((uint32_t)0x0000FFFF)  /* MAC address0 high */
06643
06644 /* Bit definition for Ethernet MAC Address0 Low Register */
06645 #define ETH_MACA0LR_MACA0L   ((uint32_t)0xFFFFFFFF)  /* MAC address0 low */
06646
06647 /* Bit definition for Ethernet MAC Address1 High Register */
06648 #define ETH_MACA1HR_AE       ((uint32_t)0x80000000)  /* Address enable */
06649 #define ETH_MACA1HR_SA       ((uint32_t)0x40000000)  /* Source address */
06650 #define ETH_MACA1HR_MBC      ((uint32_t)0x3F000000)  /* Mask byte control: bits to mask for comparison
      of the MAC Address bytes */
06651   #define ETH_MACA1HR_MBC_HBits15_8    ((uint32_t)0x20000000)  /* Mask MAC Address high reg bits
      [15:8] */
06652   #define ETH_MACA1HR_MBC_HBits7_0     ((uint32_t)0x10000000)  /* Mask MAC Address high reg bits [7:0]
      */
06653   #define ETH_MACA1HR_MBC_LBits31_24   ((uint32_t)0x08000000)  /* Mask MAC Address low reg bits
      [31:24] */
06654   #define ETH_MACA1HR_MBC_LBits23_16   ((uint32_t)0x04000000)  /* Mask MAC Address low reg bits
      [23:16] */
06655   #define ETH_MACA1HR_MBC_LBits15_8    ((uint32_t)0x02000000)  /* Mask MAC Address low reg bits [15:8]
      */
06656   #define ETH_MACA1HR_MBC_LBits7_0     ((uint32_t)0x01000000)  /* Mask MAC Address low reg bits [7:0]
      */
06657 #define ETH_MACA1HR_MACA1H   ((uint32_t)0x0000FFFF)  /* MAC address1 high */
06658
06659 /* Bit definition for Ethernet MAC Address1 Low Register */
06660 #define ETH_MACA1LR_MACA1L   ((uint32_t)0xFFFFFFFF)  /* MAC address1 low */
06661
06662 /* Bit definition for Ethernet MAC Address2 High Register */
06663 #define ETH_MACA2HR_AE       ((uint32_t)0x80000000)  /* Address enable */
06664 #define ETH_MACA2HR_SA       ((uint32_t)0x40000000)  /* Source address */
06665 #define ETH_MACA2HR_MBC      ((uint32_t)0x3F000000)  /* Mask byte control */
06666   #define ETH_MACA2HR_MBC_HBits15_8    ((uint32_t)0x20000000)  /* Mask MAC Address high reg bits
      [15:8] */
06667   #define ETH_MACA2HR_MBC_HBits7_0     ((uint32_t)0x10000000)  /* Mask MAC Address high reg bits [7:0]
      */
06668   #define ETH_MACA2HR_MBC_LBits31_24   ((uint32_t)0x08000000)  /* Mask MAC Address low reg bits
      [31:24] */
06669   #define ETH_MACA2HR_MBC_LBits23_16   ((uint32_t)0x04000000)  /* Mask MAC Address low reg bits
      [23:16] */
06670   #define ETH_MACA2HR_MBC_LBits15_8    ((uint32_t)0x02000000)  /* Mask MAC Address low reg bits [15:8]
      */
06671   #define ETH_MACA2HR_MBC_LBits7_0     ((uint32_t)0x01000000)  /* Mask MAC Address low reg bits [70]
      */
06672 #define ETH_MACA2HR_MACA2H   ((uint32_t)0x0000FFFF)  /* MAC address1 high */
06673
06674 /* Bit definition for Ethernet MAC Address2 Low Register */
06675 #define ETH_MACA2LR_MACA2L   ((uint32_t)0xFFFFFFFF)  /* MAC address2 low */
06676
06677 /* Bit definition for Ethernet MAC Address3 High Register */
06678 #define ETH_MACA3HR_AE       ((uint32_t)0x80000000)  /* Address enable */
06679 #define ETH_MACA3HR_SA       ((uint32_t)0x40000000)  /* Source address */
06680 #define ETH_MACA3HR_MBC      ((uint32_t)0x3F000000)  /* Mask byte control */
06681   #define ETH_MACA3HR_MBC_HBits15_8    ((uint32_t)0x20000000)  /* Mask MAC Address high reg bits
      [15:8] */
06682   #define ETH_MACA3HR_MBC_HBits7_0     ((uint32_t)0x10000000)  /* Mask MAC Address high reg bits [7:0]
      */
06683   #define ETH_MACA3HR_MBC_LBits31_24   ((uint32_t)0x08000000)  /* Mask MAC Address low reg bits
      [31:24] */
06684   #define ETH_MACA3HR_MBC_LBits23_16   ((uint32_t)0x04000000)  /* Mask MAC Address low reg bits
      [23:16] */
06685   #define ETH_MACA3HR_MBC_LBits15_8    ((uint32_t)0x02000000)  /* Mask MAC Address low reg bits [15:8]
      */
06686   #define ETH_MACA3HR_MBC_LBits7_0     ((uint32_t)0x01000000)  /* Mask MAC Address low reg bits [70]
      */
06687 #define ETH_MACA3HR_MACA3H   ((uint32_t)0x0000FFFF)  /* MAC address3 high */
06688
06689 /* Bit definition for Ethernet MAC Address3 Low Register */
06690 #define ETH_MACA3LR_MACA3L   ((uint32_t)0xFFFFFFFF)  /* MAC address3 low */
06691
06692 /******************************************************************************/
06693 /*                Ethernet MMC Registers bits definition                      */
06694 /******************************************************************************/
06695
06696 /* Bit definition for Ethernet MMC Contol Register */
06697 #define ETH_MMCCR_MCFHP      ((uint32_t)0x00000020)  /* MMC counter Full-Half preset */
06698 #define ETH_MMCCR_MCP        ((uint32_t)0x00000010)  /* MMC counter preset */
06699 #define ETH_MMCCR_MCF        ((uint32_t)0x00000008)  /* MMC Counter Freeze */
```

```
06700 #define ETH_MMCCR_ROR        ((uint32_t)0x00000004)  /* Reset on Read */
06701 #define ETH_MMCCR_CSR        ((uint32_t)0x00000002)  /* Counter Stop Rollover */
06702 #define ETH_MMCCR_CR         ((uint32_t)0x00000001)  /* Counters Reset */
06703
06704 /* Bit definition for Ethernet MMC Receive Interrupt Register */
06705 #define ETH_MMCRIR_RGUFS     ((uint32_t)0x00020000)  /* Set when Rx good unicast frames counter
      reaches half the maximum value */
06706 #define ETH_MMCRIR_RFAES     ((uint32_t)0x00000040)  /* Set when Rx alignment error counter reaches
      half the maximum value */
06707 #define ETH_MMCRIR_RFCES     ((uint32_t)0x00000020)  /* Set when Rx crc error counter reaches half the
      maximum value */
06708
06709 /* Bit definition for Ethernet MMC Transmit Interrupt Register */
06710 #define ETH_MMCTIR_TGFS      ((uint32_t)0x00200000)  /* Set when Tx good frame count counter reaches
      half the maximum value */
06711 #define ETH_MMCTIR_TGFMSCS   ((uint32_t)0x00008000)  /* Set when Tx good multi col counter reaches
      half the maximum value */
06712 #define ETH_MMCTIR_TGFSCS    ((uint32_t)0x00004000)  /* Set when Tx good single col counter reaches
      half the maximum value */
06713
06714 /* Bit definition for Ethernet MMC Receive Interrupt Mask Register */
06715 #define ETH_MMCRIMR_RGUFM    ((uint32_t)0x00020000)  /* Mask the interrupt when Rx good unicast frames
      counter reaches half the maximum value */
06716 #define ETH_MMCRIMR_RFAEM    ((uint32_t)0x00000040)  /* Mask the interrupt when when Rx alignment
      error counter reaches half the maximum value */
06717 #define ETH_MMCRIMR_RFCEM    ((uint32_t)0x00000020)  /* Mask the interrupt when Rx crc error counter
      reaches half the maximum value */
06718
06719 /* Bit definition for Ethernet MMC Transmit Interrupt Mask Register */
06720 #define ETH_MMCTIMR_TGFM     ((uint32_t)0x00200000)  /* Mask the interrupt when Tx good frame count
      counter reaches half the maximum value */
06721 #define ETH_MMCTIMR_TGFMSCM  ((uint32_t)0x00008000)  /* Mask the interrupt when Tx good multi col
      counter reaches half the maximum value */
06722 #define ETH_MMCTIMR_TGFSCM   ((uint32_t)0x00004000)  /* Mask the interrupt when Tx good single col
      counter reaches half the maximum value */
06723
06724 /* Bit definition for Ethernet MMC Transmitted Good Frames after Single Collision Counter Register */
06725 #define ETH_MMCTGFSCCR_TGFSCC     ((uint32_t)0xFFFFFFFF)  /* Number of successfully transmitted frames
      after a single collision in Half-duplex mode. */
06726
06727 /* Bit definition for Ethernet MMC Transmitted Good Frames after More than a Single Collision Counter
      Register */
06728 #define ETH_MMCTGFMSCCR_TGFMSCC   ((uint32_t)0xFFFFFFFF)  /* Number of successfully transmitted frames
      after more than a single collision in Half-duplex mode. */
06729
06730 /* Bit definition for Ethernet MMC Transmitted Good Frames Counter Register */
06731 #define ETH_MMCTGFCR_TGFC    ((uint32_t)0xFFFFFFFF)  /* Number of good frames transmitted. */
06732
06733 /* Bit definition for Ethernet MMC Received Frames with CRC Error Counter Register */
06734 #define ETH_MMCRFCECR_RFCEC  ((uint32_t)0xFFFFFFFF)  /* Number of frames received with CRC error. */
06735
06736 /* Bit definition for Ethernet MMC Received Frames with Alignement Error Counter Register */
06737 #define ETH_MMCRFAECR_RFAEC  ((uint32_t)0xFFFFFFFF)  /* Number of frames received with alignment
      (dribble) error */
06738
06739 /* Bit definition for Ethernet MMC Received Good Unicast Frames Counter Register */
06740 #define ETH_MMCRGUFCR_RGUFC  ((uint32_t)0xFFFFFFFF)  /* Number of good unicast frames received. */
06741
06742 /*****************************************************************************/
06743 /*               Ethernet PTP Registers bits definition                      */
06744 /*****************************************************************************/
06745
06746 /* Bit definition for Ethernet PTP Time Stamp Contol Register */
06747 #define ETH_PTPTSCR_TSCNT       ((uint32_t)0x00030000)  /* Time stamp clock node type */
06748 #define ETH_PTPTSSR_TSSMRME     ((uint32_t)0x00008000)  /* Time stamp snapshot for message relevant to
      master enable */
06749 #define ETH_PTPTSSR_TSSEME      ((uint32_t)0x00004000)  /* Time stamp snapshot for event message
      enable */
06750 #define ETH_PTPTSSR_TSSIPV4FE   ((uint32_t)0x00002000)  /* Time stamp snapshot for IPv4 frames enable
      */
06751 #define ETH_PTPTSSR_TSSIPV6FE   ((uint32_t)0x00001000)  /* Time stamp snapshot for IPv6 frames enable
      */
06752 #define ETH_PTPTSSR_TSSPTPOEFE  ((uint32_t)0x00000800)  /* Time stamp snapshot for PTP over ethernet
      frames enable */
06753 #define ETH_PTPTSSR_TSPTPPSV2E  ((uint32_t)0x00000400)  /* Time stamp PTP packet snooping for version2
      format enable */
06754 #define ETH_PTPTSSR_TSSSR       ((uint32_t)0x00000200)  /* Time stamp Sub-seconds rollover */
06755 #define ETH_PTPTSSR_TSSARFE     ((uint32_t)0x00000100)  /* Time stamp snapshot for all received frames
      enable */
06756
06757 #define ETH_PTPTSCR_TSARU    ((uint32_t)0x00000020)  /* Addend register update */
06758 #define ETH_PTPTSCR_TSITE    ((uint32_t)0x00000010)  /* Time stamp interrupt trigger enable */
06759 #define ETH_PTPTSCR_TSSTU    ((uint32_t)0x00000008)  /* Time stamp update */
06760 #define ETH_PTPTSCR_TSSTI    ((uint32_t)0x00000004)  /* Time stamp initialize */
06761 #define ETH_PTPTSCR_TSFCU    ((uint32_t)0x00000002)  /* Time stamp fine or coarse update */
06762 #define ETH_PTPTSCR_TSE      ((uint32_t)0x00000001)  /* Time stamp enable */
06763
```

```
06764 /* Bit definition for Ethernet PTP Sub-Second Increment Register */
06765 #define ETH_PTPSSIR_STSSI     ((uint32_t)0x000000FF)  /* System time Sub-second increment value */
06766
06767 /* Bit definition for Ethernet PTP Time Stamp High Register */
06768 #define ETH_PTPTSHR_STS       ((uint32_t)0xFFFFFFFF)  /* System Time second */
06769
06770 /* Bit definition for Ethernet PTP Time Stamp Low Register */
06771 #define ETH_PTPTSLR_STPNS     ((uint32_t)0x80000000)  /* System Time Positive or negative time */
06772 #define ETH_PTPTSLR_STSS      ((uint32_t)0x7FFFFFFF)  /* System Time sub-seconds */
06773
06774 /* Bit definition for Ethernet PTP Time Stamp High Update Register */
06775 #define ETH_PTPTSHUR_TSUS     ((uint32_t)0xFFFFFFFF)  /* Time stamp update seconds */
06776
06777 /* Bit definition for Ethernet PTP Time Stamp Low Update Register */
06778 #define ETH_PTPTSLUR_TSUPNS   ((uint32_t)0x80000000)  /* Time stamp update Positive or negative time */
06779 #define ETH_PTPTSLUR_TSUSS    ((uint32_t)0x7FFFFFFF)  /* Time stamp update sub-seconds */
06780
06781 /* Bit definition for Ethernet PTP Time Stamp Addend Register */
06782 #define ETH_PTPTSAR_TSA       ((uint32_t)0xFFFFFFFF)  /* Time stamp addend */
06783
06784 /* Bit definition for Ethernet PTP Target Time High Register */
06785 #define ETH_PTPTTHR_TTSH      ((uint32_t)0xFFFFFFFF)  /* Target time stamp high */
06786
06787 /* Bit definition for Ethernet PTP Target Time Low Register */
06788 #define ETH_PTPTTLR_TTSL      ((uint32_t)0xFFFFFFFF)  /* Target time stamp low */
06789
06790 /* Bit definition for Ethernet PTP Time Stamp Status Register */
06791 #define ETH_PTPTSSR_TSTTR     ((uint32_t)0x00000020)  /* Time stamp target time reached */
06792 #define ETH_PTPTSSR_TSSO      ((uint32_t)0x00000010)  /* Time stamp seconds overflow */
06793
06794 /*****************************************************************************/
06795 /*                  Ethernet DMA Registers bits definition                   */
06796 /*****************************************************************************/
06797
06798 /* Bit definition for Ethernet DMA Bus Mode Register */
06799 #define ETH_DMABMR_AAB        ((uint32_t)0x02000000)  /* Address-Aligned beats */
06800 #define ETH_DMABMR_FPM         ((uint32_t)0x01000000)  /* 4xPBL mode */
06801 #define ETH_DMABMR_USP        ((uint32_t)0x00800000)  /* Use separate PBL */
06802 #define ETH_DMABMR_RDP        ((uint32_t)0x007E0000)  /* RxDMA PBL */
06803   #define ETH_DMABMR_RDP_1Beat    ((uint32_t)0x00020000)  /* maximum number of beats to be transferred
      in one RxDMA transaction is 1 */
06804   #define ETH_DMABMR_RDP_2Beat    ((uint32_t)0x00040000)  /* maximum number of beats to be transferred
      in one RxDMA transaction is 2 */
06805   #define ETH_DMABMR_RDP_4Beat    ((uint32_t)0x00080000)  /* maximum number of beats to be transferred
      in one RxDMA transaction is 4 */
06806   #define ETH_DMABMR_RDP_8Beat    ((uint32_t)0x00100000)  /* maximum number of beats to be transferred
      in one RxDMA transaction is 8 */
06807   #define ETH_DMABMR_RDP_16Beat   ((uint32_t)0x00200000)  /* maximum number of beats to be transferred
      in one RxDMA transaction is 16 */
06808   #define ETH_DMABMR_RDP_32Beat   ((uint32_t)0x00400000)  /* maximum number of beats to be transferred
      in one RxDMA transaction is 32 */
06809   #define ETH_DMABMR_RDP_4xPBL_4Beat   ((uint32_t)0x01020000)  /* maximum number of beats to be
      transferred in one RxDMA transaction is 4 */
06810   #define ETH_DMABMR_RDP_4xPBL_8Beat   ((uint32_t)0x01040000)  /* maximum number of beats to be
      transferred in one RxDMA transaction is 8 */
06811   #define ETH_DMABMR_RDP_4xPBL_16Beat  ((uint32_t)0x01080000)  /* maximum number of beats to be
      transferred in one RxDMA transaction is 16 */
06812   #define ETH_DMABMR_RDP_4xPBL_32Beat  ((uint32_t)0x01100000)  /* maximum number of beats to be
      transferred in one RxDMA transaction is 32 */
06813   #define ETH_DMABMR_RDP_4xPBL_64Beat  ((uint32_t)0x01200000)  /* maximum number of beats to be
      transferred in one RxDMA transaction is 64 */
06814   #define ETH_DMABMR_RDP_4xPBL_128Beat ((uint32_t)0x01400000)  /* maximum number of beats to be
      transferred in one RxDMA transaction is 128 */
06815 #define ETH_DMABMR_FB         ((uint32_t)0x00010000)  /* Fixed Burst */
06816 #define ETH_DMABMR_RTPR       ((uint32_t)0x0000C000)  /* Rx Tx priority ratio */
06817   #define ETH_DMABMR_RTPR_1_1     ((uint32_t)0x00000000)  /* Rx Tx priority ratio */
06818   #define ETH_DMABMR_RTPR_2_1     ((uint32_t)0x00004000)  /* Rx Tx priority ratio */
06819   #define ETH_DMABMR_RTPR_3_1     ((uint32_t)0x00008000)  /* Rx Tx priority ratio */
06820   #define ETH_DMABMR_RTPR_4_1     ((uint32_t)0x0000C000)  /* Rx Tx priority ratio */
06821 #define ETH_DMABMR_PBL    ((uint32_t)0x00003F00)  /* Programmable burst length */
06822   #define ETH_DMABMR_PBL_1Beat    ((uint32_t)0x00000100)  /* maximum number of beats to be transferred
      in one TxDMA (or both) transaction is 1 */
06823   #define ETH_DMABMR_PBL_2Beat    ((uint32_t)0x00000200)  /* maximum number of beats to be transferred
      in one TxDMA (or both) transaction is 2 */
06824   #define ETH_DMABMR_PBL_4Beat    ((uint32_t)0x00000400)  /* maximum number of beats to be transferred
      in one TxDMA (or both) transaction is 4 */
06825   #define ETH_DMABMR_PBL_8Beat    ((uint32_t)0x00000800)  /* maximum number of beats to be transferred
      in one TxDMA (or both) transaction is 8 */
06826   #define ETH_DMABMR_PBL_16Beat   ((uint32_t)0x00001000)  /* maximum number of beats to be transferred
      in one TxDMA (or both) transaction is 16 */
06827   #define ETH_DMABMR_PBL_32Beat   ((uint32_t)0x00002000)  /* maximum number of beats to be transferred
      in one TxDMA (or both) transaction is 32 */
06828   #define ETH_DMABMR_PBL_4xPBL_4Beat   ((uint32_t)0x01000100)  /* maximum number of beats to be
      transferred in one TxDMA (or both) transaction is 4 */
06829   #define ETH_DMABMR_PBL_4xPBL_8Beat   ((uint32_t)0x01000200)  /* maximum number of beats to be
      transferred in one TxDMA (or both) transaction is 8 */
06830   #define ETH_DMABMR_PBL_4xPBL_16Beat  ((uint32_t)0x01000400)  /* maximum number of beats to be
```

```
      transferred in one TxDMA (or both) transaction is 16 */
06831   #define ETH_DMABMR_PBL_4xPBL_32Beat   ((uint32_t)0x01000800)  /* maximum number of beats to be
      transferred in one TxDMA (or both) transaction is 32 */
06832   #define ETH_DMABMR_PBL_4xPBL_64Beat   ((uint32_t)0x01001000)  /* maximum number of beats to be
      transferred in one TxDMA (or both) transaction is 64 */
06833   #define ETH_DMABMR_PBL_4xPBL_128Beat ((uint32_t)0x01002000)  /* maximum number of beats to be
      transferred in one TxDMA (or both) transaction is 128 */
06834 #define ETH_DMABMR_EDE       ((uint32_t)0x00000080)  /* Enhanced Descriptor Enable */
06835 #define ETH_DMABMR_DSL       ((uint32_t)0x0000007C)  /* Descriptor Skip Length */
06836 #define ETH_DMABMR_DA        ((uint32_t)0x00000002)  /* DMA arbitration scheme */
06837 #define ETH_DMABMR_SR        ((uint32_t)0x00000001)  /* Software reset */
06838
06839 /* Bit definition for Ethernet DMA Transmit Poll Demand Register */
06840 #define ETH_DMATPDR_TPD      ((uint32_t)0xFFFFFFFF)  /* Transmit poll demand */
06841
06842 /* Bit definition for Ethernet DMA Receive Poll Demand Register */
06843 #define ETH_DMARPDR_RPD      ((uint32_t)0xFFFFFFFF)  /* Receive poll demand  */
06844
06845 /* Bit definition for Ethernet DMA Receive Descriptor List Address Register */
06846 #define ETH_DMARDLAR_SRL     ((uint32_t)0xFFFFFFFF)  /* Start of receive list */
06847
06848 /* Bit definition for Ethernet DMA Transmit Descriptor List Address Register */
06849 #define ETH_DMATDLAR_STL     ((uint32_t)0xFFFFFFFF)  /* Start of transmit list */
06850
06851 /* Bit definition for Ethernet DMA Status Register */
06852 #define ETH_DMASR_TSTS       ((uint32_t)0x20000000)  /* Time-stamp trigger status */
06853 #define ETH_DMASR_PMTS       ((uint32_t)0x10000000)  /* PMT status */
06854 #define ETH_DMASR_MMCS       ((uint32_t)0x08000000)  /* MMC status */
06855 #define ETH_DMASR_EBS        ((uint32_t)0x03800000)  /* Error bits status */
06856   /* combination with EBS[2:0] for GetFlagStatus function */
06857   #define ETH_DMASR_EBS_DescAccess      ((uint32_t)0x02000000)  /* Error bits 0-data buffer, 1-desc.
      access */
06858   #define ETH_DMASR_EBS_ReadTransf      ((uint32_t)0x01000000)  /* Error bits 0-write trnsf, 1-read
      transfr */
06859   #define ETH_DMASR_EBS_DataTransfTx    ((uint32_t)0x00800000)  /* Error bits 0-Rx DMA, 1-Tx DMA */
06860 #define ETH_DMASR_TPS         ((uint32_t)0x00700000)  /* Transmit process state */
06861   #define ETH_DMASR_TPS_Stopped         ((uint32_t)0x00000000)  /* Stopped - Reset or Stop Tx Command
      issued  */
06862   #define ETH_DMASR_TPS_Fetching        ((uint32_t)0x00100000)  /* Running - fetching the Tx
      descriptor */
06863   #define ETH_DMASR_TPS_Waiting         ((uint32_t)0x00200000)  /* Running - waiting for status */
06864   #define ETH_DMASR_TPS_Reading         ((uint32_t)0x00300000)  /* Running - reading the data from
      host memory */
06865   #define ETH_DMASR_TPS_Suspended       ((uint32_t)0x00600000)  /* Suspended - Tx Descriptor
      unavailabe */
06866   #define ETH_DMASR_TPS_Closing         ((uint32_t)0x00700000)  /* Running - closing Rx descriptor */
06867 #define ETH_DMASR_RPS         ((uint32_t)0x000E0000)  /* Receive process state */
06868   #define ETH_DMASR_RPS_Stopped         ((uint32_t)0x00000000)  /* Stopped - Reset or Stop Rx Command
      issued */
06869   #define ETH_DMASR_RPS_Fetching        ((uint32_t)0x00020000)  /* Running - fetching the Rx
      descriptor */
06870   #define ETH_DMASR_RPS_Waiting         ((uint32_t)0x00060000)  /* Running - waiting for packet */
06871   #define ETH_DMASR_RPS_Suspended       ((uint32_t)0x00080000)  /* Suspended - Rx Descriptor
      unavailable */
06872   #define ETH_DMASR_RPS_Closing         ((uint32_t)0x000A0000)  /* Running - closing descriptor */
06873   #define ETH_DMASR_RPS_Queuing         ((uint32_t)0x000E0000)  /* Running - queuing the recieve frame
      into host memory */
06874 #define ETH_DMASR_NIS        ((uint32_t)0x00010000)  /* Normal interrupt summary */
06875 #define ETH_DMASR_AIS        ((uint32_t)0x00008000)  /* Abnormal interrupt summary */
06876 #define ETH_DMASR_ERS        ((uint32_t)0x00004000)  /* Early receive status */
06877 #define ETH_DMASR_FBES       ((uint32_t)0x00002000)  /* Fatal bus error status */
06878 #define ETH_DMASR_ETS        ((uint32_t)0x00000400)  /* Early transmit status */
06879 #define ETH_DMASR_RWTS       ((uint32_t)0x00000200)  /* Receive watchdog timeout status */
06880 #define ETH_DMASR_RPSS       ((uint32_t)0x00000100)  /* Receive process stopped status */
06881 #define ETH_DMASR_RBUS       ((uint32_t)0x00000080)  /* Receive buffer unavailable status */
06882 #define ETH_DMASR_RS         ((uint32_t)0x00000040)  /* Receive status */
06883 #define ETH_DMASR_TUS        ((uint32_t)0x00000020)  /* Transmit underflow status */
06884 #define ETH_DMASR_ROS        ((uint32_t)0x00000010)  /* Receive overflow status */
06885 #define ETH_DMASR_TJTS       ((uint32_t)0x00000008)  /* Transmit jabber timeout status */
06886 #define ETH_DMASR_TBUS       ((uint32_t)0x00000004)  /* Transmit buffer unavailable status */
06887 #define ETH_DMASR_TPSS       ((uint32_t)0x00000002)  /* Transmit process stopped status */
06888 #define ETH_DMASR_TS         ((uint32_t)0x00000001)  /* Transmit status */
06889
06890 /* Bit definition for Ethernet DMA Operation Mode Register */
06891 #define ETH_DMAOMR_DTCEFD    ((uint32_t)0x04000000)  /* Disable Dropping of TCP/IP checksum error
      frames */
06892 #define ETH_DMAOMR_RSF       ((uint32_t)0x02000000)  /* Receive store and forward */
06893 #define ETH_DMAOMR_DFRF      ((uint32_t)0x01000000)  /* Disable flushing of received frames */
06894 #define ETH_DMAOMR_TSF       ((uint32_t)0x00200000)  /* Transmit store and forward */
06895 #define ETH_DMAOMR_FTF       ((uint32_t)0x00100000)  /* Flush transmit FIFO */
06896 #define ETH_DMAOMR_TTC       ((uint32_t)0x0001C000)  /* Transmit threshold control */
06897   #define ETH_DMAOMR_TTC_64Bytes       ((uint32_t)0x00000000)  /* threshold level of the MTL Transmit
      FIFO is 64 Bytes */
06898   #define ETH_DMAOMR_TTC_128Bytes      ((uint32_t)0x00004000)  /* threshold level of the MTL Transmit
      FIFO is 128 Bytes */
06899   #define ETH_DMAOMR_TTC_192Bytes      ((uint32_t)0x00008000)  /* threshold level of the MTL Transmit
      FIFO is 192 Bytes */
```

```
06900   #define ETH_DMAOMR_TTC_256Bytes      ((uint32_t)0x0000C000)  /* threshold level of the MTL Transmit
      FIFO is 256 Bytes */
06901   #define ETH_DMAOMR_TTC_40Bytes       ((uint32_t)0x00010000)  /* threshold level of the MTL Transmit
      FIFO is 40 Bytes */
06902   #define ETH_DMAOMR_TTC_32Bytes       ((uint32_t)0x00014000)  /* threshold level of the MTL Transmit
      FIFO is 32 Bytes */
06903   #define ETH_DMAOMR_TTC_24Bytes       ((uint32_t)0x00018000)  /* threshold level of the MTL Transmit
      FIFO is 24 Bytes */
06904   #define ETH_DMAOMR_TTC_16Bytes       ((uint32_t)0x0001C000)  /* threshold level of the MTL Transmit
      FIFO is 16 Bytes */
06905 #define ETH_DMAOMR_ST        ((uint32_t)0x00002000)  /* Start/stop transmission command */
06906 #define ETH_DMAOMR_FEF       ((uint32_t)0x00000080)  /* Forward error frames */
06907 #define ETH_DMAOMR_FUGF      ((uint32_t)0x00000040)  /* Forward undersized good frames */
06908 #define ETH_DMAOMR_RTC       ((uint32_t)0x00000018)  /* receive threshold control */
06909   #define ETH_DMAOMR_RTC_64Bytes       ((uint32_t)0x00000000)  /* threshold level of the MTL Receive
      FIFO is 64 Bytes */
06910   #define ETH_DMAOMR_RTC_32Bytes       ((uint32_t)0x00000008)  /* threshold level of the MTL Receive
      FIFO is 32 Bytes */
06911   #define ETH_DMAOMR_RTC_96Bytes       ((uint32_t)0x00000010)  /* threshold level of the MTL Receive
      FIFO is 96 Bytes */
06912   #define ETH_DMAOMR_RTC_128Bytes      ((uint32_t)0x00000018)  /* threshold level of the MTL Receive
      FIFO is 128 Bytes */
06913 #define ETH_DMAOMR_OSF       ((uint32_t)0x00000004)  /* operate on second frame */
06914 #define ETH_DMAOMR_SR        ((uint32_t)0x00000002)  /* Start/stop receive */
06915
06916 /* Bit definition for Ethernet DMA Interrupt Enable Register */
06917 #define ETH_DMAIER_NISE      ((uint32_t)0x00010000)  /* Normal interrupt summary enable */
06918 #define ETH_DMAIER_AISE      ((uint32_t)0x00008000)  /* Abnormal interrupt summary enable */
06919 #define ETH_DMAIER_ERIE      ((uint32_t)0x00004000)  /* Early receive interrupt enable */
06920 #define ETH_DMAIER_FBEIE     ((uint32_t)0x00002000)  /* Fatal bus error interrupt enable */
06921 #define ETH_DMAIER_ETIE      ((uint32_t)0x00000400)  /* Early transmit interrupt enable */
06922 #define ETH_DMAIER_RWTIE     ((uint32_t)0x00000200)  /* Receive watchdog timeout interrupt enable */
06923 #define ETH_DMAIER_RPSIE     ((uint32_t)0x00000100)  /* Receive process stopped interrupt enable */
06924 #define ETH_DMAIER_RBUIE     ((uint32_t)0x00000080)  /* Receive buffer unavailable interrupt enable */
06925 #define ETH_DMAIER_RIE       ((uint32_t)0x00000040)  /* Receive interrupt enable */
06926 #define ETH_DMAIER_TUIE      ((uint32_t)0x00000020)  /* Transmit Underflow interrupt enable */
06927 #define ETH_DMAIER_ROIE      ((uint32_t)0x00000010)  /* Receive Overflow interrupt enable */
06928 #define ETH_DMAIER_TJTIE     ((uint32_t)0x00000008)  /* Transmit jabber timeout interrupt enable */
06929 #define ETH_DMAIER_TBUIE     ((uint32_t)0x00000004)  /* Transmit buffer unavailable interrupt enable
      */
06930 #define ETH_DMAIER_TPSIE     ((uint32_t)0x00000002)  /* Transmit process stopped interrupt enable */
06931 #define ETH_DMAIER_TIE       ((uint32_t)0x00000001)  /* Transmit interrupt enable */
06932
06933 /* Bit definition for Ethernet DMA Missed Frame and Buffer Overflow Counter Register */
06934 #define ETH_DMAMFBOCR_OFOC   ((uint32_t)0x10000000)  /* Overflow bit for FIFO overflow counter */
06935 #define ETH_DMAMFBOCR_MFA    ((uint32_t)0x0FFE0000)  /* Number of frames missed by the application */
06936 #define ETH_DMAMFBOCR_OMFC   ((uint32_t)0x00010000)  /* Overflow bit for missed frame counter */
06937 #define ETH_DMAMFBOCR_MFC    ((uint32_t)0x0000FFFF)  /* Number of frames missed by the controller */
06938
06939 /* Bit definition for Ethernet DMA Current Host Transmit Descriptor Register */
06940 #define ETH_DMACHTDR_HTDAP   ((uint32_t)0xFFFFFFFF)  /* Host transmit descriptor address pointer */
06941
06942 /* Bit definition for Ethernet DMA Current Host Receive Descriptor Register */
06943 #define ETH_DMACHRDR_HRDAP   ((uint32_t)0xFFFFFFFF)  /* Host receive descriptor address pointer */
06944
06945 /* Bit definition for Ethernet DMA Current Host Transmit Buffer Address Register */
06946 #define ETH_DMACHTBAR_HTBAP  ((uint32_t)0xFFFFFFFF)  /* Host transmit buffer address pointer */
06947
06948 /* Bit definition for Ethernet DMA Current Host Receive Buffer Address Register */
06949 #define ETH_DMACHRBAR_HRBAP  ((uint32_t)0xFFFFFFFF)  /* Host receive buffer address pointer */
06950
06959 #ifdef USE_STDPERIPH_DRIVER
06960   #include "stm32f4xx_conf.h"
06961 #endif /* USE_STDPERIPH_DRIVER */
06962
06967 #define SET_BIT(REG, BIT)     ((REG) |= (BIT))
06968
06969 #define CLEAR_BIT(REG, BIT)   ((REG) &= ~(BIT))
06970
06971 #define READ_BIT(REG, BIT)    ((REG) & (BIT))
06972
06973 #define CLEAR_REG(REG)        ((REG) = (0x0))
06974
06975 #define WRITE_REG(REG, VAL)   ((REG) = (VAL))
06976
06977 #define READ_REG(REG)         ((REG))
06978
06979 #define MODIFY_REG(REG, CLEARMASK, SETMASK)  WRITE_REG((REG), (((READ_REG(REG)) & (~(CLEARMASK))) |
      (SETMASK)))
06980
06985 #ifdef __cplusplus
06986 }
06987 #endif /* __cplusplus */
06988
06989 #endif /* __STM32F4xx_H */
06990
06999 /****************** (C) COPYRIGHT 2011 STMicroelectronics *****END OF FILE****/
```

7.25 stm32f4xx_conf.h

```
00001
00022 /* Define to prevent recursive inclusion -------------------------------------*/
00023 #ifndef __STM32F4xx_CONF_H
00024 #define __STM32F4xx_CONF_H
00025
00026 /* Includes ------------------------------------------------------------------*/
00027 /* Uncomment the line below to enable peripheral header file inclusion */
00028 /*#include "stm32f4xx_adc.h"
00029 #include "stm32f4xx_can.h"
00030 #include "stm32f4xx_crc.h"
00031 #include "stm32f4xx_cryp.h"
00032 #include "stm32f4xx_dac.h"
00033 #include "stm32f4xx_dbgmcu.h"
00034 #include "stm32f4xx_dcmi.h"
00035 #include "stm32f4xx_dma.h"
00036 #include "stm32f4xx_exti.h"
00037 #include "stm32f4xx_flash.h"
00038 #include "stm32f4xx_fsmc.h"
00039 #include "stm32f4xx_hash.h"
00040 #include "stm32f4xx_gpio.h"
00041 #include "stm32f4xx_i2c.h"
00042 #include "stm32f4xx_iwdg.h"
00043 #include "stm32f4xx_pwr.h"
00044 #include "stm32f4xx_rcc.h"
00045 #include "stm32f4xx_rng.h"
00046 #include "stm32f4xx_rtc.h"
00047 #include "stm32f4xx_sdio.h"
00048 #include "stm32f4xx_spi.h"
00049 #include "stm32f4xx_syscfg.h"
00050 #include "stm32f4xx_tim.h"
00051 #include "stm32f4xx_usart.h"
00052 #include "stm32f4xx_wwdg.h"
00053 #include "misc.h"*/
00054 /* High level functions for NVIC and SysTick (add-on to CMSIS functions) */
00055
00056 /* Exported types ------------------------------------------------------------*/
00057 /* Exported constants --------------------------------------------------------*/
00058
00059 /* If an external clock source is used, then the value of the following define
00060    should be set to the value of the external clock source, else, if no external
00061    clock is used, keep this define commented */
00062 /*#define I2S_EXTERNAL_CLOCK_VAL   12288000 */ /* Value of the external clock in Hz */
00063
00064
00065 /* Uncomment the line below to expanse the "assert_param" macro in the
00066    Standard Peripheral Library drivers code */
00067 /* #define USE_FULL_ASSERT    1 */
00068
00069 /* Exported macro ------------------------------------------------------------*/
00070 #ifdef  USE_FULL_ASSERT
00071
00080   #define assert_param(expr) ((expr) ? (void)0 : assert_failed((uint8_t *)__FILE__, __LINE__))
00081 /* Exported functions ------------------------------------------------------- */
00082   void assert_failed(uint8_t* file, uint32_t line);
00083 #else
00084   #define assert_param(expr) ((void)0)
00085 #endif /* USE_FULL_ASSERT */
00086
00087 #endif /* __STM32F4xx_CONF_H */
00088
00089 /******************* (C) COPYRIGHT 2011 STMicroelectronics *****END OF FILE****/
```

7.26 CUBE_IDE/VGA/Core/Inc/stm32f4xx_dma.h File Reference

This file contains all the functions prototypes for the DMA firmware library.

```
#include "stm32f4xx.h"
```

Data Structures

- struct DMA_InitTypeDef

  DMA Init structure definition.

Macros

- #define IS_DMA_ALL_PERIPH(PERIPH)
- #define IS_DMA_ALL_CONTROLLER(CONTROLLER)
- #define **DMA_Channel_0** ((uint32_t)0x00000000)
- #define **DMA_Channel_1** ((uint32_t)0x02000000)
- #define **DMA_Channel_2** ((uint32_t)0x04000000)
- #define **DMA_Channel_3** ((uint32_t)0x06000000)
- #define **DMA_Channel_4** ((uint32_t)0x08000000)
- #define **DMA_Channel_5** ((uint32_t)0x0A000000)
- #define **DMA_Channel_6** ((uint32_t)0x0C000000)
- #define **DMA_Channel_7** ((uint32_t)0x0E000000)
- #define IS_DMA_CHANNEL(CHANNEL)
- #define **DMA_DIR_PeripheralToMemory** ((uint32_t)0x00000000)
- #define **DMA_DIR_MemoryToPeripheral** ((uint32_t)0x00000040)
- #define **DMA_DIR_MemoryToMemory** ((uint32_t)0x00000080)
- #define IS_DMA_DIRECTION(DIRECTION)
- #define **IS_DMA_BUFFER_SIZE**(SIZE) (((SIZE) >= 0x1) && ((SIZE) < 0x10000))
- #define **DMA_PeripheralInc_Enable** ((uint32_t)0x00000200)
- #define **DMA_PeripheralInc_Disable** ((uint32_t)0x00000000)
- #define IS_DMA_PERIPHERAL_INC_STATE(STATE)
- #define **DMA_MemoryInc_Enable** ((uint32_t)0x00000400)
- #define **DMA_MemoryInc_Disable** ((uint32_t)0x00000000)
- #define IS_DMA_MEMORY_INC_STATE(STATE)
- #define **DMA_PeripheralDataSize_Byte** ((uint32_t)0x00000000)
- #define **DMA_PeripheralDataSize_HalfWord** ((uint32_t)0x00000800)
- #define **DMA_PeripheralDataSize_Word** ((uint32_t)0x00001000)
- #define IS_DMA_PERIPHERAL_DATA_SIZE(SIZE)
- #define **DMA_MemoryDataSize_Byte** ((uint32_t)0x00000000)
- #define **DMA_MemoryDataSize_HalfWord** ((uint32_t)0x00002000)
- #define **DMA_MemoryDataSize_Word** ((uint32_t)0x00004000)
- #define IS_DMA_MEMORY_DATA_SIZE(SIZE)
- #define **DMA_Mode_Normal** ((uint32_t)0x00000000)
- #define **DMA_Mode_Circular** ((uint32_t)0x00000100)
- #define IS_DMA_MODE(MODE)
- #define **DMA_Priority_Low** ((uint32_t)0x00000000)
- #define **DMA_Priority_Medium** ((uint32_t)0x00010000)
- #define **DMA_Priority_High** ((uint32_t)0x00020000)
- #define **DMA_Priority_VeryHigh** ((uint32_t)0x00030000)
- #define IS_DMA_PRIORITY(PRIORITY)
- #define **DMA_FIFOMode_Disable** ((uint32_t)0x00000000)
- #define **DMA_FIFOMode_Enable** ((uint32_t)0x00000004)
- #define IS_DMA_FIFO_MODE_STATE(STATE)
- #define **DMA_FIFOThreshold_1QuarterFull** ((uint32_t)0x00000000)
- #define **DMA_FIFOThreshold_HalfFull** ((uint32_t)0x00000001)
- #define **DMA_FIFOThreshold_3QuartersFull** ((uint32_t)0x00000002)
- #define **DMA_FIFOThreshold_Full** ((uint32_t)0x00000003)
- #define IS_DMA_FIFO_THRESHOLD(THRESHOLD)
- #define **DMA_MemoryBurst_Single** ((uint32_t)0x00000000)
- #define **DMA_MemoryBurst_INC4** ((uint32_t)0x00800000)
- #define **DMA_MemoryBurst_INC8** ((uint32_t)0x01000000)
- #define **DMA_MemoryBurst_INC16** ((uint32_t)0x01800000)
- #define IS_DMA_MEMORY_BURST(BURST)
- #define **DMA_PeripheralBurst_Single** ((uint32_t)0x00000000)
- #define **DMA_PeripheralBurst_INC4** ((uint32_t)0x00200000)

- #define **DMA_PeripheralBurst_INC8** ((uint32_t)0x00400000)
- #define **DMA_PeripheralBurst_INC16** ((uint32_t)0x00600000)
- #define IS_DMA_PERIPHERAL_BURST(BURST)
- #define **DMA_FIFOStatus_Less1QuarterFull** ((uint32_t)0x00000000 << 3)
- #define **DMA_FIFOStatus_1QuarterFull** ((uint32_t)0x00000001 << 3)
- #define **DMA_FIFOStatus_HalfFull** ((uint32_t)0x00000002 << 3)
- #define **DMA_FIFOStatus_3QuartersFull** ((uint32_t)0x00000003 << 3)
- #define **DMA_FIFOStatus_Empty** ((uint32_t)0x00000004 << 3)
- #define **DMA_FIFOStatus_Full** ((uint32_t)0x00000005 << 3)
- #define IS_DMA_FIFO_STATUS(STATUS)
- #define **DMA_FLAG_FEIF0** ((uint32_t)0x10800001)
- #define **DMA_FLAG_DMEIF0** ((uint32_t)0x10800004)
- #define **DMA_FLAG_TEIF0** ((uint32_t)0x10000008)
- #define **DMA_FLAG_HTIF0** ((uint32_t)0x10000010)
- #define **DMA_FLAG_TCIF0** ((uint32_t)0x10000020)
- #define **DMA_FLAG_FEIF1** ((uint32_t)0x10000040)
- #define **DMA_FLAG_DMEIF1** ((uint32_t)0x10000100)
- #define **DMA_FLAG_TEIF1** ((uint32_t)0x10000200)
- #define **DMA_FLAG_HTIF1** ((uint32_t)0x10000400)
- #define **DMA_FLAG_TCIF1** ((uint32_t)0x10000800)
- #define **DMA_FLAG_FEIF2** ((uint32_t)0x10010000)
- #define **DMA_FLAG_DMEIF2** ((uint32_t)0x10040000)
- #define **DMA_FLAG_TEIF2** ((uint32_t)0x10080000)
- #define **DMA_FLAG_HTIF2** ((uint32_t)0x10100000)
- #define **DMA_FLAG_TCIF2** ((uint32_t)0x10200000)
- #define **DMA_FLAG_FEIF3** ((uint32_t)0x10400000)
- #define **DMA_FLAG_DMEIF3** ((uint32_t)0x11000000)
- #define **DMA_FLAG_TEIF3** ((uint32_t)0x12000000)
- #define **DMA_FLAG_HTIF3** ((uint32_t)0x14000000)
- #define **DMA_FLAG_TCIF3** ((uint32_t)0x18000000)
- #define **DMA_FLAG_FEIF4** ((uint32_t)0x20000001)
- #define **DMA_FLAG_DMEIF4** ((uint32_t)0x20000004)
- #define **DMA_FLAG_TEIF4** ((uint32_t)0x20000008)
- #define **DMA_FLAG_HTIF4** ((uint32_t)0x20000010)
- #define **DMA_FLAG_TCIF4** ((uint32_t)0x20000020)
- #define **DMA_FLAG_FEIF5** ((uint32_t)0x20000040)
- #define **DMA_FLAG_DMEIF5** ((uint32_t)0x20000100)
- #define **DMA_FLAG_TEIF5** ((uint32_t)0x20000200)
- #define **DMA_FLAG_HTIF5** ((uint32_t)0x20000400)
- #define **DMA_FLAG_TCIF5** ((uint32_t)0x20000800)
- #define **DMA_FLAG_FEIF6** ((uint32_t)0x20010000)
- #define **DMA_FLAG_DMEIF6** ((uint32_t)0x20040000)
- #define **DMA_FLAG_TEIF6** ((uint32_t)0x20080000)
- #define **DMA_FLAG_HTIF6** ((uint32_t)0x20100000)
- #define **DMA_FLAG_TCIF6** ((uint32_t)0x20200000)
- #define **DMA_FLAG_FEIF7** ((uint32_t)0x20400000)
- #define **DMA_FLAG_DMEIF7** ((uint32_t)0x21000000)
- #define **DMA_FLAG_TEIF7** ((uint32_t)0x22000000)
- #define **DMA_FLAG_HTIF7** ((uint32_t)0x24000000)
- #define **DMA_FLAG_TCIF7** ((uint32_t)0x28000000)
- #define IS_DMA_CLEAR_FLAG(FLAG)
- #define IS_DMA_GET_FLAG(FLAG)
- #define **DMA_IT_TC** ((uint32_t)0x00000010)
- #define **DMA_IT_HT** ((uint32_t)0x00000008)
- #define **DMA_IT_TE** ((uint32_t)0x00000004)

- #define **DMA_IT_DME** ((uint32_t)0x00000002)
- #define **DMA_IT_FE** ((uint32_t)0x00000080)
- #define **IS_DMA_CONFIG_IT**(IT) ((((IT) & 0xFFFFFF61) == 0x00) && ((IT) != 0x00))
- #define **DMA_IT_FEIF0** ((uint32_t)0x90000001)
- #define **DMA_IT_DMEIF0** ((uint32_t)0x10001004)
- #define **DMA_IT_TEIF0** ((uint32_t)0x10002008)
- #define **DMA_IT_HTIF0** ((uint32_t)0x10004010)
- #define **DMA_IT_TCIF0** ((uint32_t)0x10008020)
- #define **DMA_IT_FEIF1** ((uint32_t)0x90000040)
- #define **DMA_IT_DMEIF1** ((uint32_t)0x10001100)
- #define **DMA_IT_TEIF1** ((uint32_t)0x10002200)
- #define **DMA_IT_HTIF1** ((uint32_t)0x10004400)
- #define **DMA_IT_TCIF1** ((uint32_t)0x10008800)
- #define **DMA_IT_FEIF2** ((uint32_t)0x90010000)
- #define **DMA_IT_DMEIF2** ((uint32_t)0x10041000)
- #define **DMA_IT_TEIF2** ((uint32_t)0x10082000)
- #define **DMA_IT_HTIF2** ((uint32_t)0x10104000)
- #define **DMA_IT_TCIF2** ((uint32_t)0x10208000)
- #define **DMA_IT_FEIF3** ((uint32_t)0x90400000)
- #define **DMA_IT_DMEIF3** ((uint32_t)0x11001000)
- #define **DMA_IT_TEIF3** ((uint32_t)0x12002000)
- #define **DMA_IT_HTIF3** ((uint32_t)0x14004000)
- #define **DMA_IT_TCIF3** ((uint32_t)0x18008000)
- #define **DMA_IT_FEIF4** ((uint32_t)0xA0000001)
- #define **DMA_IT_DMEIF4** ((uint32_t)0x20001004)
- #define **DMA_IT_TEIF4** ((uint32_t)0x20002008)
- #define **DMA_IT_HTIF4** ((uint32_t)0x20004010)
- #define **DMA_IT_TCIF4** ((uint32_t)0x20008020)
- #define **DMA_IT_FEIF5** ((uint32_t)0xA0000040)
- #define **DMA_IT_DMEIF5** ((uint32_t)0x20001100)
- #define **DMA_IT_TEIF5** ((uint32_t)0x20002200)
- #define **DMA_IT_HTIF5** ((uint32_t)0x20004400)
- #define **DMA_IT_TCIF5** ((uint32_t)0x20008800)
- #define **DMA_IT_FEIF6** ((uint32_t)0xA0010000)
- #define **DMA_IT_DMEIF6** ((uint32_t)0x20041000)
- #define **DMA_IT_TEIF6** ((uint32_t)0x20082000)
- #define **DMA_IT_HTIF6** ((uint32_t)0x20104000)
- #define **DMA_IT_TCIF6** ((uint32_t)0x20208000)
- #define **DMA_IT_FEIF7** ((uint32_t)0xA0400000)
- #define **DMA_IT_DMEIF7** ((uint32_t)0x21001000)
- #define **DMA_IT_TEIF7** ((uint32_t)0x22002000)
- #define **DMA_IT_HTIF7** ((uint32_t)0x24004000)
- #define **DMA_IT_TCIF7** ((uint32_t)0x28008000)
- #define IS_DMA_CLEAR_IT(IT)
- #define IS_DMA_GET_IT(IT)
- #define **DMA_PINCOS_Psize** ((uint32_t)0x00000000)
- #define **DMA_PINCOS_WordAligned** ((uint32_t)0x00008000)
- #define IS_DMA_PINCOS_SIZE(SIZE)
- #define **DMA_FlowCtrl_Memory** ((uint32_t)0x00000000)
- #define **DMA_FlowCtrl_Peripheral** ((uint32_t)0x00000020)
- #define IS_DMA_FLOW_CTRL(CTRL)
- #define **DMA_Memory_0** ((uint32_t)0x00000000)
- #define **DMA_Memory_1** ((uint32_t)0x00080000)
- #define **IS_DMA_CURRENT_MEM**(MEM) (((MEM) == DMA_Memory_0) || ((MEM) == DMA_Memory_1))

Functions

- void DMA_DeInit (DMA_Stream_TypeDef *DMAy_Streamx)

  Deinitialize the DMAy Streamx registers to their default reset values.
- void DMA_Init (DMA_Stream_TypeDef *DMAy_Streamx, DMA_InitTypeDef *DMA_InitStruct)

  Initializes the DMAy Streamx according to the specified parameters in the DMA_InitStruct structure.
- void DMA_StructInit (DMA_InitTypeDef *DMA_InitStruct)

  Fills each DMA_InitStruct member with its default value.
- void DMA_Cmd (DMA_Stream_TypeDef *DMAy_Streamx, FunctionalState NewState)

  Enables or disables the specified DMAy Streamx.
- void DMA_PeriphIncOffsetSizeConfig (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_Pincos)

  Configures, when the PINC (Peripheral Increment address mode) bit is set, if the peripheral address should be incremented with the data size (configured with PSIZE bits) or by a fixed offset equal to 4 (32-bit aligned addresses).
- void DMA_FlowControllerConfig (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_FlowCtrl)

  Configures, when the DMAy Streamx is disabled, the flow controller for the next transactions (Peripheral or Memory).
- void DMA_SetCurrDataCounter (DMA_Stream_TypeDef *DMAy_Streamx, uint16_t Counter)

  Writes the number of data units to be transferred on the DMAy Streamx.
- uint16_t DMA_GetCurrDataCounter (DMA_Stream_TypeDef *DMAy_Streamx)

  Returns the number of remaining data units in the current DMAy Streamx transfer.
- void DMA_DoubleBufferModeConfig (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t Memory1BaseAddr, uint32_t DMA_CurrentMemory)

  Configures, when the DMAy Streamx is disabled, the double buffer mode and the current memory target.
- void DMA_DoubleBufferModeCmd (DMA_Stream_TypeDef *DMAy_Streamx, FunctionalState NewState)

  Enables or disables the double buffer mode for the selected DMA stream.
- void DMA_MemoryTargetConfig (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t MemoryBaseAddr, uint32_t DMA_MemoryTarget)

  Configures the Memory address for the next buffer transfer in double buffer mode (for dynamic use). This function can be called when the DMA Stream is enabled and when the transfer is ongoing.
- uint32_t DMA_GetCurrentMemoryTarget (DMA_Stream_TypeDef *DMAy_Streamx)

  Returns the current memory target used by double buffer transfer.
- FunctionalState DMA_GetCmdStatus (DMA_Stream_TypeDef *DMAy_Streamx)

  Returns the status of EN bit for the specified DMAy Streamx.
- uint32_t DMA_GetFIFOStatus (DMA_Stream_TypeDef *DMAy_Streamx)

  Returns the current DMAy Streamx FIFO filled level.
- FlagStatus DMA_GetFlagStatus (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_FLAG)

  Checks whether the specified DMAy Streamx flag is set or not.
- void DMA_ClearFlag (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_FLAG)

  Clears the DMAy Streamx's pending flags.
- void DMA_ITConfig (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_IT, FunctionalState NewState)

  Enables or disables the specified DMAy Streamx interrupts.
- ITStatus DMA_GetITStatus (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_IT)

  Checks whether the specified DMAy Streamx interrupt has occurred or not.
- void DMA_ClearITPendingBit (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_IT)

  Clears the DMAy Streamx's interrupt pending bits.

7.26.1 Detailed Description

This file contains all the functions prototypes for the DMA firmware library.

Author

MCD Application Team

Version

V1.0.0

Date

30-September-2011

Attention

7.27 stm32f4xx_dma.h

Go to the documentation of this file.

```
00001
00023 /* Define to prevent recursive inclusion -------------------------------------*/
00024 #ifndef __STM32F4xx_DMA_H
00025 #define __STM32F4xx_DMA_H
00026
00027 #ifdef __cplusplus
00028  extern "C" {
00029 #endif
00030
00031 /* Includes ------------------------------------------------------------------*/
00032 #include "stm32f4xx.h"
00033
00042 /* Exported types ------------------------------------------------------------*/
00043
00048 typedef struct
00049 {
00050   uint32_t DMA_Channel;
00053   uint32_t DMA_PeripheralBaseAddr;
00055   uint32_t DMA_Memory0BaseAddr;
00059   uint32_t DMA_DIR;
00063   uint32_t DMA_BufferSize;
00067   uint32_t DMA_PeripheralInc;
00070   uint32_t DMA_MemoryInc;
00073   uint32_t DMA_PeripheralDataSize;
00076   uint32_t DMA_MemoryDataSize;
```

```
00079   uint32_t DMA_Mode;
00084   uint32_t DMA_Priority;
00087   uint32_t DMA_FIFOMode;
00092   uint32_t DMA_FIFOThreshold;
00095   uint32_t DMA_MemoryBurst;
00100   uint32_t DMA_PeripheralBurst;
00104 }DMA_InitTypeDef;
00105
00106 /* Exported constants --------------------------------------------------------*/
00107
00112 #define IS_DMA_ALL_PERIPH(PERIPH) (((PERIPH) == DMA1_Stream0) || \
00113                                    ((PERIPH) == DMA1_Stream1) || \
00114                                    ((PERIPH) == DMA1_Stream2) || \
00115                                    ((PERIPH) == DMA1_Stream3) || \
00116                                    ((PERIPH) == DMA1_Stream4) || \
00117                                    ((PERIPH) == DMA1_Stream5) || \
00118                                    ((PERIPH) == DMA1_Stream6) || \
00119                                    ((PERIPH) == DMA1_Stream7) || \
00120                                    ((PERIPH) == DMA2_Stream0) || \
00121                                    ((PERIPH) == DMA2_Stream1) || \
00122                                    ((PERIPH) == DMA2_Stream2) || \
00123                                    ((PERIPH) == DMA2_Stream3) || \
00124                                    ((PERIPH) == DMA2_Stream4) || \
00125                                    ((PERIPH) == DMA2_Stream5) || \
00126                                    ((PERIPH) == DMA2_Stream6) || \
00127                                    ((PERIPH) == DMA2_Stream7))
00128
00129 #define IS_DMA_ALL_CONTROLLER(CONTROLLER) (((CONTROLLER) == DMA1) || \
00130                                           ((CONTROLLER) == DMA2))
00131
00135 #define DMA_Channel_0                    ((uint32_t)0x00000000)
00136 #define DMA_Channel_1                    ((uint32_t)0x02000000)
00137 #define DMA_Channel_2                    ((uint32_t)0x04000000)
00138 #define DMA_Channel_3                    ((uint32_t)0x06000000)
00139 #define DMA_Channel_4                    ((uint32_t)0x08000000)
00140 #define DMA_Channel_5                    ((uint32_t)0x0A000000)
00141 #define DMA_Channel_6                    ((uint32_t)0x0C000000)
00142 #define DMA_Channel_7                    ((uint32_t)0x0E000000)
00143
00144 #define IS_DMA_CHANNEL(CHANNEL) (((CHANNEL) == DMA_Channel_0) || \
00145                                  ((CHANNEL) == DMA_Channel_1) || \
00146                                  ((CHANNEL) == DMA_Channel_2) || \
00147                                  ((CHANNEL) == DMA_Channel_3) || \
00148                                  ((CHANNEL) == DMA_Channel_4) || \
00149                                  ((CHANNEL) == DMA_Channel_5) || \
00150                                  ((CHANNEL) == DMA_Channel_6) || \
00151                                  ((CHANNEL) == DMA_Channel_7))
00160 #define DMA_DIR_PeripheralToMemory       ((uint32_t)0x00000000)
00161 #define DMA_DIR_MemoryToPeripheral       ((uint32_t)0x00000040)
00162 #define DMA_DIR_MemoryToMemory           ((uint32_t)0x00000080)
00163
00164 #define IS_DMA_DIRECTION(DIRECTION) (((DIRECTION) == DMA_DIR_PeripheralToMemory ) || \
00165                                      ((DIRECTION) == DMA_DIR_MemoryToPeripheral)  || \
00166                                      ((DIRECTION) == DMA_DIR_MemoryToMemory))
00175 #define IS_DMA_BUFFER_SIZE(SIZE) (((SIZE) >= 0x1) && ((SIZE) < 0x10000))
00184 #define DMA_PeripheralInc_Enable         ((uint32_t)0x00000200)
00185 #define DMA_PeripheralInc_Disable        ((uint32_t)0x00000000)
00186
00187 #define IS_DMA_PERIPHERAL_INC_STATE(STATE) (((STATE) == DMA_PeripheralInc_Enable) || \
00188                                            ((STATE) == DMA_PeripheralInc_Disable))
00197 #define DMA_MemoryInc_Enable             ((uint32_t)0x00000400)
00198 #define DMA_MemoryInc_Disable            ((uint32_t)0x00000000)
00199
00200 #define IS_DMA_MEMORY_INC_STATE(STATE) (((STATE) == DMA_MemoryInc_Enable) || \
00201                                         ((STATE) == DMA_MemoryInc_Disable))
00210 #define DMA_PeripheralDataSize_Byte      ((uint32_t)0x00000000)
00211 #define DMA_PeripheralDataSize_HalfWord  ((uint32_t)0x00000800)
00212 #define DMA_PeripheralDataSize_Word      ((uint32_t)0x00001000)
00213
00214 #define IS_DMA_PERIPHERAL_DATA_SIZE(SIZE) (((SIZE) == DMA_PeripheralDataSize_Byte)  || \
00215                                            ((SIZE) == DMA_PeripheralDataSize_HalfWord) || \
00216                                            ((SIZE) == DMA_PeripheralDataSize_Word))
00225 #define DMA_MemoryDataSize_Byte          ((uint32_t)0x00000000)
00226 #define DMA_MemoryDataSize_HalfWord      ((uint32_t)0x00002000)
00227 #define DMA_MemoryDataSize_Word          ((uint32_t)0x00004000)
00228
00229 #define IS_DMA_MEMORY_DATA_SIZE(SIZE) (((SIZE) == DMA_MemoryDataSize_Byte)  || \
00230                                        ((SIZE) == DMA_MemoryDataSize_HalfWord) || \
00231                                        ((SIZE) == DMA_MemoryDataSize_Word ))
00240 #define DMA_Mode_Normal                  ((uint32_t)0x00000000)
00241 #define DMA_Mode_Circular                ((uint32_t)0x00000100)
00242
00243 #define IS_DMA_MODE(MODE) (((MODE) == DMA_Mode_Normal ) || \
00244                            ((MODE) == DMA_Mode_Circular))
00253 #define DMA_Priority_Low                 ((uint32_t)0x00000000)
00254 #define DMA_Priority_Medium              ((uint32_t)0x00010000)
00255 #define DMA_Priority_High                ((uint32_t)0x00020000)
```

```
00256 #define DMA_Priority_VeryHigh             ((uint32_t)0x00030000)
00257
00258 #define IS_DMA_PRIORITY(PRIORITY) (((PRIORITY) == DMA_Priority_Low )   || \
00259                                    ((PRIORITY) == DMA_Priority_Medium) || \
00260                                    ((PRIORITY) == DMA_Priority_High)   || \
00261                                    ((PRIORITY) == DMA_Priority_VeryHigh))
00270 #define DMA_FIFOMode_Disable              ((uint32_t)0x00000000)
00271 #define DMA_FIFOMode_Enable               ((uint32_t)0x00000004)
00272
00273 #define IS_DMA_FIFO_MODE_STATE(STATE) (((STATE) == DMA_FIFOMode_Disable ) || \
00274                                        ((STATE) == DMA_FIFOMode_Enable))
00283 #define DMA_FIFOThreshold_1QuarterFull    ((uint32_t)0x00000000)
00284 #define DMA_FIFOThreshold_HalfFull        ((uint32_t)0x00000001)
00285 #define DMA_FIFOThreshold_3QuartersFull   ((uint32_t)0x00000002)
00286 #define DMA_FIFOThreshold_Full            ((uint32_t)0x00000003)
00287
00288 #define IS_DMA_FIFO_THRESHOLD(THRESHOLD) (((THRESHOLD) == DMA_FIFOThreshold_1QuarterFull ) || \
00289                                           ((THRESHOLD) == DMA_FIFOThreshold_HalfFull)      || \
00290                                           ((THRESHOLD) == DMA_FIFOThreshold_3QuartersFull) || \
00291                                           ((THRESHOLD) == DMA_FIFOThreshold_Full))
00300 #define DMA_MemoryBurst_Single            ((uint32_t)0x00000000)
00301 #define DMA_MemoryBurst_INC4              ((uint32_t)0x00800000)
00302 #define DMA_MemoryBurst_INC8              ((uint32_t)0x01000000)
00303 #define DMA_MemoryBurst_INC16             ((uint32_t)0x01800000)
00304
00305 #define IS_DMA_MEMORY_BURST(BURST) (((BURST) == DMA_MemoryBurst_Single) || \
00306                                     ((BURST) == DMA_MemoryBurst_INC4)  || \
00307                                     ((BURST) == DMA_MemoryBurst_INC8)  || \
00308                                     ((BURST) == DMA_MemoryBurst_INC16))
00317 #define DMA_PeripheralBurst_Single        ((uint32_t)0x00000000)
00318 #define DMA_PeripheralBurst_INC4          ((uint32_t)0x00200000)
00319 #define DMA_PeripheralBurst_INC8          ((uint32_t)0x00400000)
00320 #define DMA_PeripheralBurst_INC16         ((uint32_t)0x00600000)
00321
00322 #define IS_DMA_PERIPHERAL_BURST(BURST) (((BURST) == DMA_PeripheralBurst_Single) || \
00323                                         ((BURST) == DMA_PeripheralBurst_INC4)  || \
00324                                         ((BURST) == DMA_PeripheralBurst_INC8)  || \
00325                                         ((BURST) == DMA_PeripheralBurst_INC16))
00334 #define DMA_FIFOStatus_Less1QuarterFull   ((uint32_t)0x00000000 « 3)
00335 #define DMA_FIFOStatus_1QuarterFull       ((uint32_t)0x00000001 « 3)
00336 #define DMA_FIFOStatus_HalfFull           ((uint32_t)0x00000002 « 3)
00337 #define DMA_FIFOStatus_3QuartersFull      ((uint32_t)0x00000003 « 3)
00338 #define DMA_FIFOStatus_Empty              ((uint32_t)0x00000004 « 3)
00339 #define DMA_FIFOStatus_Full               ((uint32_t)0x00000005 « 3)
00340
00341 #define IS_DMA_FIFO_STATUS(STATUS) (((STATUS) == DMA_FIFOStatus_Less1QuarterFull ) || \
00342                                     ((STATUS) == DMA_FIFOStatus_HalfFull)          || \
00343                                     ((STATUS) == DMA_FIFOStatus_1QuarterFull)      || \
00344                                     ((STATUS) == DMA_FIFOStatus_3QuartersFull)     || \
00345                                     ((STATUS) == DMA_FIFOStatus_Full)              || \
00346                                     ((STATUS) == DMA_FIFOStatus_Empty))
00354 #define DMA_FLAG_FEIF0                    ((uint32_t)0x10800001)
00355 #define DMA_FLAG_DMEIF0                   ((uint32_t)0x10800004)
00356 #define DMA_FLAG_TEIF0                    ((uint32_t)0x10000008)
00357 #define DMA_FLAG_HTIF0                    ((uint32_t)0x10000010)
00358 #define DMA_FLAG_TCIF0                    ((uint32_t)0x10000020)
00359 #define DMA_FLAG_FEIF1                    ((uint32_t)0x10000040)
00360 #define DMA_FLAG_DMEIF1                   ((uint32_t)0x10000100)
00361 #define DMA_FLAG_TEIF1                    ((uint32_t)0x10000200)
00362 #define DMA_FLAG_HTIF1                    ((uint32_t)0x10000400)
00363 #define DMA_FLAG_TCIF1                    ((uint32_t)0x10000800)
00364 #define DMA_FLAG_FEIF2                    ((uint32_t)0x10010000)
00365 #define DMA_FLAG_DMEIF2                   ((uint32_t)0x10040000)
00366 #define DMA_FLAG_TEIF2                    ((uint32_t)0x10080000)
00367 #define DMA_FLAG_HTIF2                    ((uint32_t)0x10100000)
00368 #define DMA_FLAG_TCIF2                    ((uint32_t)0x10200000)
00369 #define DMA_FLAG_FEIF3                    ((uint32_t)0x10400000)
00370 #define DMA_FLAG_DMEIF3                   ((uint32_t)0x11000000)
00371 #define DMA_FLAG_TEIF3                    ((uint32_t)0x12000000)
00372 #define DMA_FLAG_HTIF3                    ((uint32_t)0x14000000)
00373 #define DMA_FLAG_TCIF3                    ((uint32_t)0x18000000)
00374 #define DMA_FLAG_FEIF4                    ((uint32_t)0x20000001)
00375 #define DMA_FLAG_DMEIF4                   ((uint32_t)0x20000004)
00376 #define DMA_FLAG_TEIF4                    ((uint32_t)0x20000008)
00377 #define DMA_FLAG_HTIF4                    ((uint32_t)0x20000010)
00378 #define DMA_FLAG_TCIF4                    ((uint32_t)0x20000020)
00379 #define DMA_FLAG_FEIF5                    ((uint32_t)0x20000040)
00380 #define DMA_FLAG_DMEIF5                   ((uint32_t)0x20000100)
00381 #define DMA_FLAG_TEIF5                    ((uint32_t)0x20000200)
00382 #define DMA_FLAG_HTIF5                    ((uint32_t)0x20000400)
00383 #define DMA_FLAG_TCIF5                    ((uint32_t)0x20000800)
00384 #define DMA_FLAG_FEIF6                    ((uint32_t)0x20010000)
00385 #define DMA_FLAG_DMEIF6                   ((uint32_t)0x20040000)
00386 #define DMA_FLAG_TEIF6                    ((uint32_t)0x20080000)
00387 #define DMA_FLAG_HTIF6                    ((uint32_t)0x20100000)
00388 #define DMA_FLAG_TCIF6                    ((uint32_t)0x20200000)
00389 #define DMA_FLAG_FEIF7                    ((uint32_t)0x20400000)
```

```
00390 #define DMA_FLAG_DMEIF7                  ((uint32_t)0x21000000)
00391 #define DMA_FLAG_TEIF7                   ((uint32_t)0x22000000)
00392 #define DMA_FLAG_HTIF7                   ((uint32_t)0x24000000)
00393 #define DMA_FLAG_TCIF7                   ((uint32_t)0x28000000)
00394
00395 #define IS_DMA_CLEAR_FLAG(FLAG) ((((FLAG) & 0x30000000) != 0x30000000) && (((FLAG) & 0x30000000) != 0)
      && \
00396                                  (((FLAG) & 0xC082F082) == 0x00) && ((FLAG) != 0x00))
00397
00398 #define IS_DMA_GET_FLAG(FLAG) (((FLAG) == DMA_FLAG_TCIF0)  || ((FLAG) == DMA_FLAG_HTIF0)  || \
00399                                ((FLAG) == DMA_FLAG_TEIF0)  || ((FLAG) == DMA_FLAG_DMEIF0) || \
00400                                ((FLAG) == DMA_FLAG_FEIF0)  || ((FLAG) == DMA_FLAG_TCIF1)  || \
00401                                ((FLAG) == DMA_FLAG_HTIF1)  || ((FLAG) == DMA_FLAG_TEIF1)  || \
00402                                ((FLAG) == DMA_FLAG_DMEIF1) || ((FLAG) == DMA_FLAG_FEIF1)  || \
00403                                ((FLAG) == DMA_FLAG_TCIF2)  || ((FLAG) == DMA_FLAG_HTIF2)  || \
00404                                ((FLAG) == DMA_FLAG_TEIF2)  || ((FLAG) == DMA_FLAG_DMEIF2) || \
00405                                ((FLAG) == DMA_FLAG_FEIF2)  || ((FLAG) == DMA_FLAG_TCIF3)  || \
00406                                ((FLAG) == DMA_FLAG_HTIF3)  || ((FLAG) == DMA_FLAG_TEIF3)  || \
00407                                ((FLAG) == DMA_FLAG_DMEIF3) || ((FLAG) == DMA_FLAG_FEIF3)  || \
00408                                ((FLAG) == DMA_FLAG_TCIF4)  || ((FLAG) == DMA_FLAG_HTIF4)  || \
00409                                ((FLAG) == DMA_FLAG_TEIF4)  || ((FLAG) == DMA_FLAG_DMEIF4) || \
00410                                ((FLAG) == DMA_FLAG_FEIF4)  || ((FLAG) == DMA_FLAG_TCIF5)  || \
00411                                ((FLAG) == DMA_FLAG_HTIF5)  || ((FLAG) == DMA_FLAG_TEIF5)  || \
00412                                ((FLAG) == DMA_FLAG_DMEIF5) || ((FLAG) == DMA_FLAG_FEIF5)  || \
00413                                ((FLAG) == DMA_FLAG_TCIF6)  || ((FLAG) == DMA_FLAG_HTIF6)  || \
00414                                ((FLAG) == DMA_FLAG_TEIF6)  || ((FLAG) == DMA_FLAG_DMEIF6) || \
00415                                ((FLAG) == DMA_FLAG_FEIF6)  || ((FLAG) == DMA_FLAG_TCIF7)  || \
00416                                ((FLAG) == DMA_FLAG_HTIF7)  || ((FLAG) == DMA_FLAG_TEIF7)  || \
00417                                ((FLAG) == DMA_FLAG_DMEIF7) || ((FLAG) == DMA_FLAG_FEIF7))
00426 #define DMA_IT_TC                        ((uint32_t)0x00000010)
00427 #define DMA_IT_HT                        ((uint32_t)0x00000008)
00428 #define DMA_IT_TE                        ((uint32_t)0x00000004)
00429 #define DMA_IT_DME                       ((uint32_t)0x00000002)
00430 #define DMA_IT_FE                        ((uint32_t)0x00000080)
00431
00432 #define IS_DMA_CONFIG_IT(IT) ((((IT) & 0xFFFFFF61) == 0x00) && ((IT) != 0x00))
00441 #define DMA_IT_FEIF0                     ((uint32_t)0x90000001)
00442 #define DMA_IT_DMEIF0                    ((uint32_t)0x10001004)
00443 #define DMA_IT_TEIF0                     ((uint32_t)0x10002008)
00444 #define DMA_IT_HTIF0                     ((uint32_t)0x10004010)
00445 #define DMA_IT_TCIF0                     ((uint32_t)0x10008020)
00446 #define DMA_IT_FEIF1                     ((uint32_t)0x90000040)
00447 #define DMA_IT_DMEIF1                    ((uint32_t)0x10001100)
00448 #define DMA_IT_TEIF1                     ((uint32_t)0x10002200)
00449 #define DMA_IT_HTIF1                     ((uint32_t)0x10004400)
00450 #define DMA_IT_TCIF1                     ((uint32_t)0x10008800)
00451 #define DMA_IT_FEIF2                     ((uint32_t)0x90010000)
00452 #define DMA_IT_DMEIF2                    ((uint32_t)0x10041000)
00453 #define DMA_IT_TEIF2                     ((uint32_t)0x10082000)
00454 #define DMA_IT_HTIF2                     ((uint32_t)0x10104000)
00455 #define DMA_IT_TCIF2                     ((uint32_t)0x10208000)
00456 #define DMA_IT_FEIF3                     ((uint32_t)0x90400000)
00457 #define DMA_IT_DMEIF3                    ((uint32_t)0x11001000)
00458 #define DMA_IT_TEIF3                     ((uint32_t)0x12002000)
00459 #define DMA_IT_HTIF3                     ((uint32_t)0x14004000)
00460 #define DMA_IT_TCIF3                     ((uint32_t)0x18008000)
00461 #define DMA_IT_FEIF4                     ((uint32_t)0xA0000001)
00462 #define DMA_IT_DMEIF4                    ((uint32_t)0x20001004)
00463 #define DMA_IT_TEIF4                     ((uint32_t)0x20002008)
00464 #define DMA_IT_HTIF4                     ((uint32_t)0x20004010)
00465 #define DMA_IT_TCIF4                     ((uint32_t)0x20008020)
00466 #define DMA_IT_FEIF5                     ((uint32_t)0xA0000040)
00467 #define DMA_IT_DMEIF5                    ((uint32_t)0x20001100)
00468 #define DMA_IT_TEIF5                     ((uint32_t)0x20002200)
00469 #define DMA_IT_HTIF5                     ((uint32_t)0x20004400)
00470 #define DMA_IT_TCIF5                     ((uint32_t)0x20008800)
00471 #define DMA_IT_FEIF6                     ((uint32_t)0xA0010000)
00472 #define DMA_IT_DMEIF6                    ((uint32_t)0x20041000)
00473 #define DMA_IT_TEIF6                     ((uint32_t)0x20082000)
00474 #define DMA_IT_HTIF6                     ((uint32_t)0x20104000)
00475 #define DMA_IT_TCIF6                     ((uint32_t)0x20208000)
00476 #define DMA_IT_FEIF7                     ((uint32_t)0xA0400000)
00477 #define DMA_IT_DMEIF7                    ((uint32_t)0x21001000)
00478 #define DMA_IT_TEIF7                     ((uint32_t)0x22002000)
00479 #define DMA_IT_HTIF7                     ((uint32_t)0x24004000)
00480 #define DMA_IT_TCIF7                     ((uint32_t)0x28008000)
00481
00482 #define IS_DMA_CLEAR_IT(IT) ((((IT) & 0x30000000) != 0x30000000) && \
00483                              (((IT) & 0x30000000) != 0) && ((IT) != 0x00) && \
00484                              (((IT) & 0x40820082) == 0x00))
00485
00486 #define IS_DMA_GET_IT(IT) (((IT) == DMA_IT_TCIF0) || ((IT) == DMA_IT_HTIF0)  || \
00487                            ((IT) == DMA_IT_TEIF0) || ((IT) == DMA_IT_DMEIF0) || \
00488                            ((IT) == DMA_IT_FEIF0) || ((IT) == DMA_IT_TCIF1)  || \
00489                            ((IT) == DMA_IT_HTIF1) || ((IT) == DMA_IT_TEIF1)  || \
00490                            ((IT) == DMA_IT_DMEIF1)|| ((IT) == DMA_IT_FEIF1)  || \
00491                            ((IT) == DMA_IT_TCIF2) || ((IT) == DMA_IT_HTIF2)  || \
```

```
00492                           ((IT) == DMA_IT_TEIF2) || ((IT) == DMA_IT_DMEIF2) || \
00493                           ((IT) == DMA_IT_FEIF2) || ((IT) == DMA_IT_TCIF3)  || \
00494                           ((IT) == DMA_IT_HTIF3) || ((IT) == DMA_IT_TEIF3)  || \
00495                           ((IT) == DMA_IT_DMEIF3)|| ((IT) == DMA_IT_FEIF3)  || \
00496                           ((IT) == DMA_IT_TCIF4) || ((IT) == DMA_IT_HTIF4)  || \
00497                           ((IT) == DMA_IT_TEIF4) || ((IT) == DMA_IT_DMEIF4) || \
00498                           ((IT) == DMA_IT_FEIF4) || ((IT) == DMA_IT_TCIF5)  || \
00499                           ((IT) == DMA_IT_HTIF5) || ((IT) == DMA_IT_TEIF5)  || \
00500                           ((IT) == DMA_IT_DMEIF5)|| ((IT) == DMA_IT_FEIF5)  || \
00501                           ((IT) == DMA_IT_TCIF6) || ((IT) == DMA_IT_HTIF6)  || \
00502                           ((IT) == DMA_IT_TEIF6) || ((IT) == DMA_IT_DMEIF6) || \
00503                           ((IT) == DMA_IT_FEIF6) || ((IT) == DMA_IT_TCIF7)  || \
00504                           ((IT) == DMA_IT_HTIF7) || ((IT) == DMA_IT_TEIF7)  || \
00505                           ((IT) == DMA_IT_DMEIF7)|| ((IT) == DMA_IT_FEIF7))
00514 #define DMA_PINCOS_Psize                ((uint32_t)0x00000000)
00515 #define DMA_PINCOS_WordAligned          ((uint32_t)0x00008000)
00516
00517 #define IS_DMA_PINCOS_SIZE(SIZE) (((SIZE) == DMA_PINCOS_Psize) || \
00518                                   ((SIZE) == DMA_PINCOS_WordAligned))
00527 #define DMA_FlowCtrl_Memory             ((uint32_t)0x00000000)
00528 #define DMA_FlowCtrl_Peripheral         ((uint32_t)0x00000020)
00529
00530 #define IS_DMA_FLOW_CTRL(CTRL) (((CTRL) == DMA_FlowCtrl_Memory) || \
00531                                 ((CTRL) == DMA_FlowCtrl_Peripheral))
00540 #define DMA_Memory_0                    ((uint32_t)0x00000000)
00541 #define DMA_Memory_1                    ((uint32_t)0x00080000)
00542
00543 #define IS_DMA_CURRENT_MEM(MEM) (((MEM) == DMA_Memory_0) || ((MEM) == DMA_Memory_1))
00552 /* Exported macro ------------------------------------------------------------*/
00553 /* Exported functions --------------------------------------------------------*/
00554
00555 /*  Function used to set the DMA configuration to the default reset state *****/
00556 void DMA_DeInit(DMA_Stream_TypeDef* DMAy_Streamx);
00557
00558 /* Initialization and Configuration functions *********************************/
00559 void DMA_Init(DMA_Stream_TypeDef* DMAy_Streamx, DMA_InitTypeDef* DMA_InitStruct);
00560 void DMA_StructInit(DMA_InitTypeDef* DMA_InitStruct);
00561 void DMA_Cmd(DMA_Stream_TypeDef* DMAy_Streamx, FunctionalState NewState);
00562
00563 /* Optional Configuration functions *******************************************/
00564 void DMA_PeriphIncOffsetSizeConfig(DMA_Stream_TypeDef* DMAy_Streamx, uint32_t DMA_Pincos);
00565 void DMA_FlowControllerConfig(DMA_Stream_TypeDef* DMAy_Streamx, uint32_t DMA_FlowCtrl);
00566
00567 /* Data Counter functions *****************************************************/
00568 void DMA_SetCurrDataCounter(DMA_Stream_TypeDef* DMAy_Streamx, uint16_t Counter);
00569 uint16_t DMA_GetCurrDataCounter(DMA_Stream_TypeDef* DMAy_Streamx);
00570
00571 /* Double Buffer mode functions ***********************************************/
00572 void DMA_DoubleBufferModeConfig(DMA_Stream_TypeDef* DMAy_Streamx, uint32_t Memory1BaseAddr,
00573                                 uint32_t DMA_CurrentMemory);
00574 void DMA_DoubleBufferModeCmd(DMA_Stream_TypeDef* DMAy_Streamx, FunctionalState NewState);
00575 void DMA_MemoryTargetConfig(DMA_Stream_TypeDef* DMAy_Streamx, uint32_t MemoryBaseAddr,
00576                             uint32_t DMA_MemoryTarget);
00577 uint32_t DMA_GetCurrentMemoryTarget(DMA_Stream_TypeDef* DMAy_Streamx);
00578
00579 /* Interrupts and flags management functions **********************************/
00580 FunctionalState DMA_GetCmdStatus(DMA_Stream_TypeDef* DMAy_Streamx);
00581 uint32_t DMA_GetFIFOStatus(DMA_Stream_TypeDef* DMAy_Streamx);
00582 FlagStatus DMA_GetFlagStatus(DMA_Stream_TypeDef* DMAy_Streamx, uint32_t DMA_FLAG);
00583 void DMA_ClearFlag(DMA_Stream_TypeDef* DMAy_Streamx, uint32_t DMA_FLAG);
00584 void DMA_ITConfig(DMA_Stream_TypeDef* DMAy_Streamx, uint32_t DMA_IT, FunctionalState NewState);
00585 ITStatus DMA_GetITStatus(DMA_Stream_TypeDef* DMAy_Streamx, uint32_t DMA_IT);
00586 void DMA_ClearITPendingBit(DMA_Stream_TypeDef* DMAy_Streamx, uint32_t DMA_IT);
00587
00588 #ifdef __cplusplus
00589 }
00590 #endif
00591
00592 #endif /*__STM32F4xx_DMA_H */
00593
00603 /******************* (C) COPYRIGHT 2011 STMicroelectronics *****END OF FILE****/
```

7.28 CUBE_IDE/VGA/Core/Inc/stm32f4xx_gpio.h File Reference

This file contains all the functions prototypes for the GPIO firmware library.

```
#include "stm32f4xx.h"
```

Data Structures

- struct GPIO_InitTypeDef

  GPIO Init structure definition

Macros

- #define IS_GPIO_ALL_PERIPH(PERIPH)
- #define IS_GPIO_MODE(MODE)
- #define **IS_GPIO_OTYPE**(OTYPE) (((OTYPE) == GPIO_OType_PP) || ((OTYPE) == GPIO_OType_OD))
- #define IS_GPIO_SPEED(SPEED)
- #define IS_GPIO_PUPD(PUPD)
- #define **IS_GPIO_BIT_ACTION**(ACTION) (((ACTION) == Bit_RESET) || ((ACTION) == Bit_SET))
- #define **GPIO_Pin_0** ((uint16_t)0x0001) /* Pin 0 selected */
- #define **GPIO_Pin_1** ((uint16_t)0x0002) /* Pin 1 selected */
- #define **GPIO_Pin_2** ((uint16_t)0x0004) /* Pin 2 selected */
- #define **GPIO_Pin_3** ((uint16_t)0x0008) /* Pin 3 selected */
- #define **GPIO_Pin_4** ((uint16_t)0x0010) /* Pin 4 selected */
- #define **GPIO_Pin_5** ((uint16_t)0x0020) /* Pin 5 selected */
- #define **GPIO_Pin_6** ((uint16_t)0x0040) /* Pin 6 selected */
- #define **GPIO_Pin_7** ((uint16_t)0x0080) /* Pin 7 selected */
- #define **GPIO_Pin_8** ((uint16_t)0x0100) /* Pin 8 selected */
- #define **GPIO_Pin_9** ((uint16_t)0x0200) /* Pin 9 selected */
- #define **GPIO_Pin_10** ((uint16_t)0x0400) /* Pin 10 selected */
- #define **GPIO_Pin_11** ((uint16_t)0x0800) /* Pin 11 selected */
- #define **GPIO_Pin_12** ((uint16_t)0x1000) /* Pin 12 selected */
- #define **GPIO_Pin_13** ((uint16_t)0x2000) /* Pin 13 selected */
- #define **GPIO_Pin_14** ((uint16_t)0x4000) /* Pin 14 selected */
- #define **GPIO_Pin_15** ((uint16_t)0x8000) /* Pin 15 selected */
- #define **GPIO_Pin_All** ((uint16_t)0xFFFF) /* All pins selected */
- #define **IS_GPIO_PIN**(PIN) ((((PIN) & (uint16_t)0x00) == 0x00) && ((PIN) != (uint16_t)0x00))
- #define IS_GET_GPIO_PIN(PIN)
- #define **GPIO_PinSource0** ((uint8_t)0x00)
- #define **GPIO_PinSource1** ((uint8_t)0x01)
- #define **GPIO_PinSource2** ((uint8_t)0x02)
- #define **GPIO_PinSource3** ((uint8_t)0x03)
- #define **GPIO_PinSource4** ((uint8_t)0x04)
- #define **GPIO_PinSource5** ((uint8_t)0x05)
- #define **GPIO_PinSource6** ((uint8_t)0x06)
- #define **GPIO_PinSource7** ((uint8_t)0x07)
- #define **GPIO_PinSource8** ((uint8_t)0x08)
- #define **GPIO_PinSource9** ((uint8_t)0x09)
- #define **GPIO_PinSource10** ((uint8_t)0x0A)
- #define **GPIO_PinSource11** ((uint8_t)0x0B)
- #define **GPIO_PinSource12** ((uint8_t)0x0C)
- #define **GPIO_PinSource13** ((uint8_t)0x0D)
- #define **GPIO_PinSource14** ((uint8_t)0x0E)
- #define **GPIO_PinSource15** ((uint8_t)0x0F)
- #define IS_GPIO_PIN_SOURCE(PINSOURCE)
- #define **GPIO_AF_RTC_50Hz** ((uint8_t)0x00) /* RTC_50Hz Alternate Function mapping */

  AF 0 selection

- #define **GPIO_AF_MCO** ((uint8_t)0x00) /* MCO (MCO1 and MCO2) Alternate Function mapping */

- #define **GPIO_AF_TAMPER** ((uint8_t)0x00) /* TAMPER (TAMPER_1 and TAMPER_2) Alternate Function mapping */
- #define **GPIO_AF_SWJ** ((uint8_t)0x00) /* SWJ (SWD and JTAG) Alternate Function mapping */
- #define **GPIO_AF_TRACE** ((uint8_t)0x00) /* TRACE Alternate Function mapping */
- #define **GPIO_AF_TIM1** ((uint8_t)0x01) /* TIM1 Alternate Function mapping */

  AF 1 selection

- #define **GPIO_AF_TIM2** ((uint8_t)0x01) /* TIM2 Alternate Function mapping */
- #define **GPIO_AF_TIM3** ((uint8_t)0x02) /* TIM3 Alternate Function mapping */

  AF 2 selection

- #define **GPIO_AF_TIM4** ((uint8_t)0x02) /* TIM4 Alternate Function mapping */
- #define **GPIO_AF_TIM5** ((uint8_t)0x02) /* TIM5 Alternate Function mapping */
- #define **GPIO_AF_TIM8** ((uint8_t)0x03) /* TIM8 Alternate Function mapping */

  AF 3 selection

- #define **GPIO_AF_TIM9** ((uint8_t)0x03) /* TIM9 Alternate Function mapping */
- #define **GPIO_AF_TIM10** ((uint8_t)0x03) /* TIM10 Alternate Function mapping */
- #define **GPIO_AF_TIM11** ((uint8_t)0x03) /* TIM11 Alternate Function mapping */
- #define **GPIO_AF_I2C1** ((uint8_t)0x04) /* I2C1 Alternate Function mapping */

  AF 4 selection

- #define **GPIO_AF_I2C2** ((uint8_t)0x04) /* I2C2 Alternate Function mapping */
- #define **GPIO_AF_I2C3** ((uint8_t)0x04) /* I2C3 Alternate Function mapping */
- #define **GPIO_AF_SPI1** ((uint8_t)0x05) /* SPI1 Alternate Function mapping */

  AF 5 selection

- #define **GPIO_AF_SPI2** ((uint8_t)0x05) /* SPI2/I2S2 Alternate Function mapping */
- #define **GPIO_AF_SPI3** ((uint8_t)0x06) /* SPI3/I2S3 Alternate Function mapping */

  AF 6 selection

- #define **GPIO_AF_USART1** ((uint8_t)0x07) /* USART1 Alternate Function mapping */

  AF 7 selection

- #define **GPIO_AF_USART2** ((uint8_t)0x07) /* USART2 Alternate Function mapping */
- #define **GPIO_AF_USART3** ((uint8_t)0x07) /* USART3 Alternate Function mapping */
- #define **GPIO_AF_I2S3ext** ((uint8_t)0x07) /* I2S3ext Alternate Function mapping */
- #define **GPIO_AF_UART4** ((uint8_t)0x08) /* UART4 Alternate Function mapping */

  AF 8 selection

- #define **GPIO_AF_UART5** ((uint8_t)0x08) /* UART5 Alternate Function mapping */
- #define **GPIO_AF_USART6** ((uint8_t)0x08) /* USART6 Alternate Function mapping */
- #define **GPIO_AF_CAN1** ((uint8_t)0x09) /* CAN1 Alternate Function mapping */

  AF 9 selection.
- #define **GPIO_AF_CAN2** ((uint8_t)0x09) /* CAN2 Alternate Function mapping */
- #define **GPIO_AF_TIM12** ((uint8_t)0x09) /* TIM12 Alternate Function mapping */
- #define **GPIO_AF_TIM13** ((uint8_t)0x09) /* TIM13 Alternate Function mapping */
- #define **GPIO_AF_TIM14** ((uint8_t)0x09) /* TIM14 Alternate Function mapping */
- #define **GPIO_AF_OTG_FS** ((uint8_t)0xA) /* OTG_FS Alternate Function mapping */

  AF 10 selection

- #define **GPIO_AF_OTG_HS** ((uint8_t)0xA) /* OTG_HS Alternate Function mapping */
- #define **GPIO_AF_ETH** ((uint8_t)0x0B) /* ETHERNET Alternate Function mapping */

  AF 11 selection

- #define **GPIO_AF_FSMC** ((uint8_t)0xC) /* FSMC Alternate Function mapping */

  AF 12 selection

- #define **GPIO_AF_OTG_HS_FS** ((uint8_t)0xC) /* OTG HS configured in FS, Alternate Function mapping */
- #define **GPIO_AF_SDIO** ((uint8_t)0xC) /* SDIO Alternate Function mapping */
- #define **GPIO_AF_DCMI** ((uint8_t)0x0D) /* DCMI Alternate Function mapping */

  AF 13 selection

- #define **GPIO_AF_EVENTOUT** ((uint8_t)0x0F) /* EVENTOUT Alternate Function mapping */

  AF 15 selection

- #define IS_GPIO_AF(AF)
- #define **GPIO_Mode_AIN** GPIO_Mode_AN
- #define **GPIO_AF_OTG1_FS** GPIO_AF_OTG_FS
- #define **GPIO_AF_OTG2_HS** GPIO_AF_OTG_HS
- #define **GPIO_AF_OTG2_FS** GPIO_AF_OTG_HS_FS

Enumerations

- enum GPIOMode_TypeDef { GPIO_Mode_IN = 0x00 , GPIO_Mode_OUT = 0x01 , GPIO_Mode_AF = 0x02 , GPIO_Mode_AN = 0x03 }

  GPIO Configuration Mode enumeration.
- enum GPIOOType_TypeDef { **GPIO_OType_PP** = 0x00 , **GPIO_OType_OD** = 0x01 }

  GPIO Output type enumeration.
- enum GPIOSpeed_TypeDef { GPIO_Speed_2MHz = 0x00 , GPIO_Speed_25MHz = 0x01 , GPIO_Speed_50MHz = 0x02 , GPIO_Speed_100MHz = 0x03 }

  GPIO Output Maximum frequency enumeration.
- enum GPIOPuPd_TypeDef { **GPIO_PuPd_NOPULL** = 0x00 , **GPIO_PuPd_UP** = 0x01 , **GPIO_PuPd_DOWN** = 0x02 }

  GPIO Configuration PullUp PullDown enumeration.
- enum BitAction { **Bit_RESET** = 0 , **Bit_SET** }

  GPIO Bit SET and Bit RESET enumeration.

Functions

- void GPIO_DeInit (GPIO_TypeDef *GPIOx)

  Deinitializes the GPIOx peripheral registers to their default reset values.
- void GPIO_Init (GPIO_TypeDef *GPIOx, GPIO_InitTypeDef *GPIO_InitStruct)

  Initializes the GPIOx peripheral according to the specified parameters in the GPIO_InitStruct.
- void GPIO_StructInit (GPIO_InitTypeDef *GPIO_InitStruct)

  Fills each GPIO_InitStruct member with its default value.
- void GPIO_PinLockConfig (GPIO_TypeDef *GPIOx, uint16_t GPIO_Pin)

  Locks GPIO Pins configuration registers.
- uint8_t GPIO_ReadInputDataBit (GPIO_TypeDef *GPIOx, uint16_t GPIO_Pin)

  Reads the specified input port pin.
- uint16_t GPIO_ReadInputData (GPIO_TypeDef *GPIOx)

  Reads the specified GPIO input data port.
- uint8_t GPIO_ReadOutputDataBit (GPIO_TypeDef *GPIOx, uint16_t GPIO_Pin)

  Reads the specified output data port bit.
- uint16_t GPIO_ReadOutputData (GPIO_TypeDef *GPIOx)

  Reads the specified GPIO output data port.

- void GPIO_SetBits (GPIO_TypeDef ∗GPIOx, uint16_t GPIO_Pin)

  Sets the selected data port bits.
- void GPIO_ResetBits (GPIO_TypeDef ∗GPIOx, uint16_t GPIO_Pin)

  Clears the selected data port bits.
- void GPIO_WriteBit (GPIO_TypeDef ∗GPIOx, uint16_t GPIO_Pin, BitAction BitVal)

  Sets or clears the selected data port bit.
- void GPIO_Write (GPIO_TypeDef ∗GPIOx, uint16_t PortVal)

  Writes data to the specified GPIO data port.
- void GPIO_ToggleBits (GPIO_TypeDef ∗GPIOx, uint16_t GPIO_Pin)

  Toggles the specified GPIO pins..
- void GPIO_PinAFConfig (GPIO_TypeDef ∗GPIOx, uint16_t GPIO_PinSource, uint8_t GPIO_AF)

  Changes the mapping of the specified pin.

7.28.1 Detailed Description

This file contains all the functions prototypes for the GPIO firmware library.

Author

MCD Application Team

Version

V1.0.0

Date

30-September-2011

Attention

7.29 stm32f4xx_gpio.h

Go to the documentation of this file.

```
00001
00023 /* Define to prevent recursive inclusion -------------------------------------*/
00024 #ifndef __STM32F4xx_GPIO_H
00025 #define __STM32F4xx_GPIO_H
00026
00027 #ifdef __cplusplus
00028  extern "C" {
00029 #endif
00030
00031 /* Includes ------------------------------------------------------------------*/
00032 #include "stm32f4xx.h"
00033
00042 /* Exported types ------------------------------------------------------------*/
00043
00044 #define IS_GPIO_ALL_PERIPH(PERIPH) (((PERIPH) == GPIOA) || \
00045                                     ((PERIPH) == GPIOB) || \
00046                                     ((PERIPH) == GPIOC) || \
00047                                     ((PERIPH) == GPIOD) || \
00048                                     ((PERIPH) == GPIOE) || \
00049                                     ((PERIPH) == GPIOF) || \
00050                                     ((PERIPH) == GPIOG) || \
00051                                     ((PERIPH) == GPIOH) || \
00052                                     ((PERIPH) == GPIOI))
00053
00057 typedef enum
00058 {
00059   GPIO_Mode_IN   = 0x00,
00060   GPIO_Mode_OUT  = 0x01,
00061   GPIO_Mode_AF   = 0x02,
00062   GPIO_Mode_AN   = 0x03
00063 }GPIOMode_TypeDef;
00064 #define IS_GPIO_MODE(MODE) (((MODE) == GPIO_Mode_IN)  || ((MODE) == GPIO_Mode_OUT) || \
00065                             ((MODE) == GPIO_Mode_AF)|| ((MODE) == GPIO_Mode_AN))
00066
00070 typedef enum
00071 {
00072   GPIO_OType_PP = 0x00,
00073   GPIO_OType_OD = 0x01
00074 }GPIOOType_TypeDef;
00075 #define IS_GPIO_OTYPE(OTYPE) (((OTYPE) == GPIO_OType_PP) || ((OTYPE) == GPIO_OType_OD))
00076
00077
00081 typedef enum
00082 {
00083   GPIO_Speed_2MHz   = 0x00,
00084   GPIO_Speed_25MHz  = 0x01,
00085   GPIO_Speed_50MHz  = 0x02,
00086   GPIO_Speed_100MHz = 0x03
00087 }GPIOSpeed_TypeDef;
00088 #define IS_GPIO_SPEED(SPEED) (((SPEED) == GPIO_Speed_2MHz) || ((SPEED) == GPIO_Speed_25MHz) || \
00089                               ((SPEED) == GPIO_Speed_50MHz)||  ((SPEED) == GPIO_Speed_100MHz))
00090
00094 typedef enum
00095 {
00096   GPIO_PuPd_NOPULL = 0x00,
00097   GPIO_PuPd_UP     = 0x01,
00098   GPIO_PuPd_DOWN   = 0x02
00099 }GPIOPuPd_TypeDef;
00100 #define IS_GPIO_PUPD(PUPD) (((PUPD) == GPIO_PuPd_NOPULL) || ((PUPD) == GPIO_PuPd_UP) || \
00101                             ((PUPD) == GPIO_PuPd_DOWN))
00102
00106 typedef enum
00107 {
00108   Bit_RESET = 0,
00109   Bit_SET
00110 }BitAction;
00111 #define IS_GPIO_BIT_ACTION(ACTION) (((ACTION) == Bit_RESET) || ((ACTION) == Bit_SET))
00112
00113
00117 typedef struct
00118 {
00119   uint32_t GPIO_Pin;
```

```
00122   GPIOMode_TypeDef GPIO_Mode;
00125   GPIOSpeed_TypeDef GPIO_Speed;
00128   GPIOOType_TypeDef GPIO_OType;
00131   GPIOPuPd_TypeDef GPIO_PuPd;
00133 }GPIO_InitTypeDef;
00134
00135 /* Exported constants --------------------------------------------------------*/
00136
00144 #define GPIO_Pin_0                 ((uint16_t)0x0001)  /* Pin 0 selected */
00145 #define GPIO_Pin_1                 ((uint16_t)0x0002)  /* Pin 1 selected */
00146 #define GPIO_Pin_2                 ((uint16_t)0x0004)  /* Pin 2 selected */
00147 #define GPIO_Pin_3                 ((uint16_t)0x0008)  /* Pin 3 selected */
00148 #define GPIO_Pin_4                 ((uint16_t)0x0010)  /* Pin 4 selected */
00149 #define GPIO_Pin_5                 ((uint16_t)0x0020)  /* Pin 5 selected */
00150 #define GPIO_Pin_6                 ((uint16_t)0x0040)  /* Pin 6 selected */
00151 #define GPIO_Pin_7                 ((uint16_t)0x0080)  /* Pin 7 selected */
00152 #define GPIO_Pin_8                 ((uint16_t)0x0100)  /* Pin 8 selected */
00153 #define GPIO_Pin_9                 ((uint16_t)0x0200)  /* Pin 9 selected */
00154 #define GPIO_Pin_10                ((uint16_t)0x0400)  /* Pin 10 selected */
00155 #define GPIO_Pin_11                ((uint16_t)0x0800)  /* Pin 11 selected */
00156 #define GPIO_Pin_12                ((uint16_t)0x1000)  /* Pin 12 selected */
00157 #define GPIO_Pin_13                ((uint16_t)0x2000)  /* Pin 13 selected */
00158 #define GPIO_Pin_14                ((uint16_t)0x4000)  /* Pin 14 selected */
00159 #define GPIO_Pin_15                ((uint16_t)0x8000)  /* Pin 15 selected */
00160 #define GPIO_Pin_All               ((uint16_t)0xFFFF)  /* All pins selected */
00161
00162 #define IS_GPIO_PIN(PIN) ((((PIN) & (uint16_t)0x00) == 0x00) && ((PIN) != (uint16_t)0x00))
00163 #define IS_GET_GPIO_PIN(PIN) (((PIN) == GPIO_Pin_0) || \
00164                               ((PIN) == GPIO_Pin_1) || \
00165                               ((PIN) == GPIO_Pin_2) || \
00166                               ((PIN) == GPIO_Pin_3) || \
00167                               ((PIN) == GPIO_Pin_4) || \
00168                               ((PIN) == GPIO_Pin_5) || \
00169                               ((PIN) == GPIO_Pin_6) || \
00170                               ((PIN) == GPIO_Pin_7) || \
00171                               ((PIN) == GPIO_Pin_8) || \
00172                               ((PIN) == GPIO_Pin_9) || \
00173                               ((PIN) == GPIO_Pin_10) || \
00174                               ((PIN) == GPIO_Pin_11) || \
00175                               ((PIN) == GPIO_Pin_12) || \
00176                               ((PIN) == GPIO_Pin_13) || \
00177                               ((PIN) == GPIO_Pin_14) || \
00178                               ((PIN) == GPIO_Pin_15))
00187 #define GPIO_PinSource0            ((uint8_t)0x00)
00188 #define GPIO_PinSource1            ((uint8_t)0x01)
00189 #define GPIO_PinSource2            ((uint8_t)0x02)
00190 #define GPIO_PinSource3            ((uint8_t)0x03)
00191 #define GPIO_PinSource4            ((uint8_t)0x04)
00192 #define GPIO_PinSource5            ((uint8_t)0x05)
00193 #define GPIO_PinSource6            ((uint8_t)0x06)
00194 #define GPIO_PinSource7            ((uint8_t)0x07)
00195 #define GPIO_PinSource8            ((uint8_t)0x08)
00196 #define GPIO_PinSource9            ((uint8_t)0x09)
00197 #define GPIO_PinSource10           ((uint8_t)0x0A)
00198 #define GPIO_PinSource11           ((uint8_t)0x0B)
00199 #define GPIO_PinSource12           ((uint8_t)0x0C)
00200 #define GPIO_PinSource13           ((uint8_t)0x0D)
00201 #define GPIO_PinSource14           ((uint8_t)0x0E)
00202 #define GPIO_PinSource15           ((uint8_t)0x0F)
00203
00204 #define IS_GPIO_PIN_SOURCE(PINSOURCE) (((PINSOURCE) == GPIO_PinSource0) || \
00205                                        ((PINSOURCE) == GPIO_PinSource1) || \
00206                                        ((PINSOURCE) == GPIO_PinSource2) || \
00207                                        ((PINSOURCE) == GPIO_PinSource3) || \
00208                                        ((PINSOURCE) == GPIO_PinSource4) || \
00209                                        ((PINSOURCE) == GPIO_PinSource5) || \
00210                                        ((PINSOURCE) == GPIO_PinSource6) || \
00211                                        ((PINSOURCE) == GPIO_PinSource7) || \
00212                                        ((PINSOURCE) == GPIO_PinSource8) || \
00213                                        ((PINSOURCE) == GPIO_PinSource9) || \
00214                                        ((PINSOURCE) == GPIO_PinSource10) || \
00215                                        ((PINSOURCE) == GPIO_PinSource11) || \
00216                                        ((PINSOURCE) == GPIO_PinSource12) || \
00217                                        ((PINSOURCE) == GPIO_PinSource13) || \
00218                                        ((PINSOURCE) == GPIO_PinSource14) || \
00219                                        ((PINSOURCE) == GPIO_PinSource15))
00230 #define GPIO_AF_RTC_50Hz      ((uint8_t)0x00)  /* RTC_50Hz Alternate Function mapping */
00231 #define GPIO_AF_MCO           ((uint8_t)0x00)  /* MCO (MCO1 and MCO2) Alternate Function mapping */
00232 #define GPIO_AF_TAMPER        ((uint8_t)0x00)  /* TAMPER (TAMPER_1 and TAMPER_2) Alternate Function
      mapping */
00233 #define GPIO_AF_SWJ           ((uint8_t)0x00)  /* SWJ (SWD and JTAG) Alternate Function mapping */
00234 #define GPIO_AF_TRACE         ((uint8_t)0x00)  /* TRACE Alternate Function mapping */
00235
00239 #define GPIO_AF_TIM1          ((uint8_t)0x01)  /* TIM1 Alternate Function mapping */
00240 #define GPIO_AF_TIM2          ((uint8_t)0x01)  /* TIM2 Alternate Function mapping */
00241
00245 #define GPIO_AF_TIM3          ((uint8_t)0x02)  /* TIM3 Alternate Function mapping */
```

```
00246 #define GPIO_AF_TIM4          ((uint8_t)0x02)  /* TIM4 Alternate Function mapping */
00247 #define GPIO_AF_TIM5          ((uint8_t)0x02)  /* TIM5 Alternate Function mapping */
00248
00252 #define GPIO_AF_TIM8          ((uint8_t)0x03)  /* TIM8 Alternate Function mapping */
00253 #define GPIO_AF_TIM9          ((uint8_t)0x03)  /* TIM9 Alternate Function mapping */
00254 #define GPIO_AF_TIM10         ((uint8_t)0x03)  /* TIM10 Alternate Function mapping */
00255 #define GPIO_AF_TIM11         ((uint8_t)0x03)  /* TIM11 Alternate Function mapping */
00256
00260 #define GPIO_AF_I2C1          ((uint8_t)0x04)  /* I2C1 Alternate Function mapping */
00261 #define GPIO_AF_I2C2          ((uint8_t)0x04)  /* I2C2 Alternate Function mapping */
00262 #define GPIO_AF_I2C3          ((uint8_t)0x04)  /* I2C3 Alternate Function mapping */
00263
00267 #define GPIO_AF_SPI1          ((uint8_t)0x05)  /* SPI1 Alternate Function mapping */
00268 #define GPIO_AF_SPI2          ((uint8_t)0x05)  /* SPI2/I2S2 Alternate Function mapping */
00269
00273 #define GPIO_AF_SPI3          ((uint8_t)0x06)  /* SPI3/I2S3 Alternate Function mapping */
00274
00278 #define GPIO_AF_USART1        ((uint8_t)0x07)  /* USART1 Alternate Function mapping */
00279 #define GPIO_AF_USART2        ((uint8_t)0x07)  /* USART2 Alternate Function mapping */
00280 #define GPIO_AF_USART3        ((uint8_t)0x07)  /* USART3 Alternate Function mapping */
00281 #define GPIO_AF_I2S3ext       ((uint8_t)0x07)  /* I2S3ext Alternate Function mapping */
00282
00286 #define GPIO_AF_UART4         ((uint8_t)0x08)  /* UART4 Alternate Function mapping */
00287 #define GPIO_AF_UART5         ((uint8_t)0x08)  /* UART5 Alternate Function mapping */
00288 #define GPIO_AF_USART6        ((uint8_t)0x08)  /* USART6 Alternate Function mapping */
00289
00293 #define GPIO_AF_CAN1          ((uint8_t)0x09)  /* CAN1 Alternate Function mapping */
00294 #define GPIO_AF_CAN2          ((uint8_t)0x09)  /* CAN2 Alternate Function mapping */
00295 #define GPIO_AF_TIM12         ((uint8_t)0x09)  /* TIM12 Alternate Function mapping */
00296 #define GPIO_AF_TIM13         ((uint8_t)0x09)  /* TIM13 Alternate Function mapping */
00297 #define GPIO_AF_TIM14         ((uint8_t)0x09)  /* TIM14 Alternate Function mapping */
00298
00302 #define GPIO_AF_OTG_FS         ((uint8_t)0xA)  /* OTG_FS Alternate Function mapping */
00303 #define GPIO_AF_OTG_HS         ((uint8_t)0xA)  /* OTG_HS Alternate Function mapping */
00304
00308 #define GPIO_AF_ETH             ((uint8_t)0x0B)  /* ETHERNET Alternate Function mapping */
00309
00313 #define GPIO_AF_FSMC            ((uint8_t)0xC)  /* FSMC Alternate Function mapping */
00314 #define GPIO_AF_OTG_HS_FS       ((uint8_t)0xC)  /* OTG HS configured in FS, Alternate Function mapping
      */
00315 #define GPIO_AF_SDIO            ((uint8_t)0xC)  /* SDIO Alternate Function mapping */
00316
00320 #define GPIO_AF_DCMI          ((uint8_t)0x0D)  /* DCMI Alternate Function mapping */
00321
00325 #define GPIO_AF_EVENTOUT      ((uint8_t)0x0F)  /* EVENTOUT Alternate Function mapping */
00326
00327 #define IS_GPIO_AF(AF)    (((AF) == GPIO_AF_RTC_50Hz)  || ((AF) == GPIO_AF_TIM14)  || \
00328                            ((AF) == GPIO_AF_MCO)       || ((AF) == GPIO_AF_TAMPER) || \
00329                            ((AF) == GPIO_AF_SWJ)       || ((AF) == GPIO_AF_TRACE)  || \
00330                            ((AF) == GPIO_AF_TIM1)      || ((AF) == GPIO_AF_TIM2)   || \
00331                            ((AF) == GPIO_AF_TIM3)      || ((AF) == GPIO_AF_TIM4)   || \
00332                            ((AF) == GPIO_AF_TIM5)      || ((AF) == GPIO_AF_TIM8)   || \
00333                            ((AF) == GPIO_AF_I2C1)      || ((AF) == GPIO_AF_I2C2)   || \
00334                            ((AF) == GPIO_AF_I2C3)      || ((AF) == GPIO_AF_SPI1)   || \
00335                            ((AF) == GPIO_AF_SPI2)      || ((AF) == GPIO_AF_TIM13)  || \
00336                            ((AF) == GPIO_AF_SPI3)      || ((AF) == GPIO_AF_TIM14)  || \
00337                            ((AF) == GPIO_AF_USART1)    || ((AF) == GPIO_AF_USART2) || \
00338                            ((AF) == GPIO_AF_USART3)    || ((AF) == GPIO_AF_UART4)  || \
00339                            ((AF) == GPIO_AF_UART5)     || ((AF) == GPIO_AF_USART6) || \
00340                            ((AF) == GPIO_AF_CAN1)      || ((AF) == GPIO_AF_CAN2)   || \
00341                            ((AF) == GPIO_AF_OTG_FS)    || ((AF) == GPIO_AF_OTG_HS) || \
00342                            ((AF) == GPIO_AF_ETH)       || ((AF) == GPIO_AF_FSMC)   || \
00343                            ((AF) == GPIO_AF_OTG_HS_FS) || ((AF) == GPIO_AF_SDIO)   || \
00344                            ((AF) == GPIO_AF_DCMI)      || ((AF) == GPIO_AF_EVENTOUT))
00353 #define GPIO_Mode_AIN           GPIO_Mode_AN
00354
00355 #define GPIO_AF_OTG1_FS         GPIO_AF_OTG_FS
00356 #define GPIO_AF_OTG2_HS         GPIO_AF_OTG_HS
00357 #define GPIO_AF_OTG2_FS         GPIO_AF_OTG_HS_FS
00358
00367 /* Exported macro ------------------------------------------------------------*/
00368 /* Exported functions --------------------------------------------------------*/
00369
00370 /*  Function used to set the GPIO configuration to the default reset state ****/
00371 void GPIO_DeInit(GPIO_TypeDef* GPIOx);
00372
00373 /* Initialization and Configuration functions *********************************/
00374 void GPIO_Init(GPIO_TypeDef* GPIOx, GPIO_InitTypeDef* GPIO_InitStruct);
00375 void GPIO_StructInit(GPIO_InitTypeDef* GPIO_InitStruct);
00376 void GPIO_PinLockConfig(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin);
00377
00378 /* GPIO Read and Write functions ***********************************************/
00379 uint8_t GPIO_ReadInputDataBit(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin);
00380 uint16_t GPIO_ReadInputData(GPIO_TypeDef* GPIOx);
00381 uint8_t GPIO_ReadOutputDataBit(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin);
00382 uint16_t GPIO_ReadOutputData(GPIO_TypeDef* GPIOx);
00383 void GPIO_SetBits(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin);
```

```
00384 void GPIO_ResetBits(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin);
00385 void GPIO_WriteBit(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin, BitAction BitVal);
00386 void GPIO_Write(GPIO_TypeDef* GPIOx, uint16_t PortVal);
00387 void GPIO_ToggleBits(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin);
00388
00389 /* GPIO Alternate functions configuration function ****************************/
00390 void GPIO_PinAFConfig(GPIO_TypeDef* GPIOx, uint16_t GPIO_PinSource, uint8_t GPIO_AF);
00391
00392 #ifdef __cplusplus
00393 }
00394 #endif
00395
00396 #endif /*__STM32F4xx_GPIO_H */
00397
00406 /******************* (C) COPYRIGHT 2011 STMicroelectronics *****END OF FILE****/
```

7.30 CUBE_IDE/VGA/Core/Inc/stm32f4xx_rcc.h File Reference

This file contains all the functions prototypes for the RCC firmware library.

```
#include "stm32f4xx.h"
```

Data Structures

- struct RCC_ClocksTypeDef

Macros

- #define **RCC_HSE_OFF** ((uint8_t)0x00)
- #define **RCC_HSE_ON** ((uint8_t)0x01)
- #define **RCC_HSE_Bypass** ((uint8_t)0x05)
- #define IS_RCC_HSE(HSE)
- #define **RCC_PLLSource_HSI** ((uint32_t)0x00000000)
- #define **RCC_PLLSource_HSE** ((uint32_t)0x00400000)
- #define IS_RCC_PLL_SOURCE(SOURCE)
- #define **IS_RCC_PLLM_VALUE**(VALUE) ((VALUE) <= 63)
- #define **IS_RCC_PLLN_VALUE**(VALUE) ((192 <= (VALUE)) && ((VALUE) <= 432))
- #define **IS_RCC_PLLP_VALUE**(VALUE) (((VALUE) == 2) || ((VALUE) == 4) || ((VALUE) == 6) || ((VALUE) == 8))
- #define **IS_RCC_PLLQ_VALUE**(VALUE) ((4 <= (VALUE)) && ((VALUE) <= 15))
- #define **IS_RCC_PLLI2SN_VALUE**(VALUE) ((192 <= (VALUE)) && ((VALUE) <= 432))
- #define **IS_RCC_PLLI2SR_VALUE**(VALUE) ((2 <= (VALUE)) && ((VALUE) <= 7))
- #define **RCC_SYSCLKSource_HSI** ((uint32_t)0x00000000)
- #define **RCC_SYSCLKSource_HSE** ((uint32_t)0x00000001)
- #define **RCC_SYSCLKSource_PLLCLK** ((uint32_t)0x00000002)
- #define IS_RCC_SYSCLK_SOURCE(SOURCE)
- #define **RCC_SYSCLK_Div1** ((uint32_t)0x00000000)
- #define **RCC_SYSCLK_Div2** ((uint32_t)0x00000080)
- #define **RCC_SYSCLK_Div4** ((uint32_t)0x00000090)
- #define **RCC_SYSCLK_Div8** ((uint32_t)0x000000A0)
- #define **RCC_SYSCLK_Div16** ((uint32_t)0x000000B0)
- #define **RCC_SYSCLK_Div64** ((uint32_t)0x000000C0)
- #define **RCC_SYSCLK_Div128** ((uint32_t)0x000000D0)
- #define **RCC_SYSCLK_Div256** ((uint32_t)0x000000E0)
- #define **RCC_SYSCLK_Div512** ((uint32_t)0x000000F0)

- #define IS_RCC_HCLK(HCLK)
- #define **RCC_HCLK_Div1** ((uint32_t)0x00000000)
- #define **RCC_HCLK_Div2** ((uint32_t)0x00001000)
- #define **RCC_HCLK_Div4** ((uint32_t)0x00001400)
- #define **RCC_HCLK_Div8** ((uint32_t)0x00001800)
- #define **RCC_HCLK_Div16** ((uint32_t)0x00001C00)
- #define IS_RCC_PCLK(PCLK)
- #define **RCC_IT_LSIRDY** ((uint8_t)0x01)
- #define **RCC_IT_LSERDY** ((uint8_t)0x02)
- #define **RCC_IT_HSIRDY** ((uint8_t)0x04)
- #define **RCC_IT_HSERDY** ((uint8_t)0x08)
- #define **RCC_IT_PLLRDY** ((uint8_t)0x10)
- #define **RCC_IT_PLLI2SRDY** ((uint8_t)0x20)
- #define **RCC_IT_CSS** ((uint8_t)0x80)
- #define **IS_RCC_IT**(IT) ((((IT) & (uint8_t)0xC0) == 0x00) && ((IT) != 0x00))
- #define IS_RCC_GET_IT(IT)
- #define **IS_RCC_CLEAR_IT**(IT) ((((IT) & (uint8_t)0x40) == 0x00) && ((IT) != 0x00))
- #define **RCC_LSE_OFF** ((uint8_t)0x00)
- #define **RCC_LSE_ON** ((uint8_t)0x01)
- #define **RCC_LSE_Bypass** ((uint8_t)0x04)
- #define IS_RCC_LSE(LSE)
- #define **RCC_RTCCLKSource_LSE** ((uint32_t)0x00000100)
- #define **RCC_RTCCLKSource_LSI** ((uint32_t)0x00000200)
- #define **RCC_RTCCLKSource_HSE_Div2** ((uint32_t)0x00020300)
- #define **RCC_RTCCLKSource_HSE_Div3** ((uint32_t)0x00030300)
- #define **RCC_RTCCLKSource_HSE_Div4** ((uint32_t)0x00040300)
- #define **RCC_RTCCLKSource_HSE_Div5** ((uint32_t)0x00050300)
- #define **RCC_RTCCLKSource_HSE_Div6** ((uint32_t)0x00060300)
- #define **RCC_RTCCLKSource_HSE_Div7** ((uint32_t)0x00070300)
- #define **RCC_RTCCLKSource_HSE_Div8** ((uint32_t)0x00080300)
- #define **RCC_RTCCLKSource_HSE_Div9** ((uint32_t)0x00090300)
- #define **RCC_RTCCLKSource_HSE_Div10** ((uint32_t)0x000A0300)
- #define **RCC_RTCCLKSource_HSE_Div11** ((uint32_t)0x000B0300)
- #define **RCC_RTCCLKSource_HSE_Div12** ((uint32_t)0x000C0300)
- #define **RCC_RTCCLKSource_HSE_Div13** ((uint32_t)0x000D0300)
- #define **RCC_RTCCLKSource_HSE_Div14** ((uint32_t)0x000E0300)
- #define **RCC_RTCCLKSource_HSE_Div15** ((uint32_t)0x000F0300)
- #define **RCC_RTCCLKSource_HSE_Div16** ((uint32_t)0x00100300)
- #define **RCC_RTCCLKSource_HSE_Div17** ((uint32_t)0x00110300)
- #define **RCC_RTCCLKSource_HSE_Div18** ((uint32_t)0x00120300)
- #define **RCC_RTCCLKSource_HSE_Div19** ((uint32_t)0x00130300)
- #define **RCC_RTCCLKSource_HSE_Div20** ((uint32_t)0x00140300)
- #define **RCC_RTCCLKSource_HSE_Div21** ((uint32_t)0x00150300)
- #define **RCC_RTCCLKSource_HSE_Div22** ((uint32_t)0x00160300)
- #define **RCC_RTCCLKSource_HSE_Div23** ((uint32_t)0x00170300)
- #define **RCC_RTCCLKSource_HSE_Div24** ((uint32_t)0x00180300)
- #define **RCC_RTCCLKSource_HSE_Div25** ((uint32_t)0x00190300)
- #define **RCC_RTCCLKSource_HSE_Div26** ((uint32_t)0x001A0300)
- #define **RCC_RTCCLKSource_HSE_Div27** ((uint32_t)0x001B0300)
- #define **RCC_RTCCLKSource_HSE_Div28** ((uint32_t)0x001C0300)
- #define **RCC_RTCCLKSource_HSE_Div29** ((uint32_t)0x001D0300)
- #define **RCC_RTCCLKSource_HSE_Div30** ((uint32_t)0x001E0300)
- #define **RCC_RTCCLKSource_HSE_Div31** ((uint32_t)0x001F0300)
- #define **IS_RCC_RTCCLK_SOURCE**(SOURCE)
- #define **RCC_I2S2CLKSource_PLLI2S** ((uint8_t)0x00)

- #define **RCC_I2S2CLKSource_Ext** ((uint8_t)0x01)
- #define **IS_RCC_I2SCLK_SOURCE**(SOURCE) (((SOURCE) == RCC_I2S2CLKSource_PLLI2S) || ((SOURCE) == RCC_I2S2CLKSource_Ext))
- #define **RCC_AHB1Periph_GPIOA** ((uint32_t)0x00000001)
- #define **RCC_AHB1Periph_GPIOB** ((uint32_t)0x00000002)
- #define **RCC_AHB1Periph_GPIOC** ((uint32_t)0x00000004)
- #define **RCC_AHB1Periph_GPIOD** ((uint32_t)0x00000008)
- #define **RCC_AHB1Periph_GPIOE** ((uint32_t)0x00000010)
- #define **RCC_AHB1Periph_GPIOF** ((uint32_t)0x00000020)
- #define **RCC_AHB1Periph_GPIOG** ((uint32_t)0x00000040)
- #define **RCC_AHB1Periph_GPIOH** ((uint32_t)0x00000080)
- #define **RCC_AHB1Periph_GPIOI** ((uint32_t)0x00000100)
- #define **RCC_AHB1Periph_CRC** ((uint32_t)0x00001000)
- #define **RCC_AHB1Periph_FLITF** ((uint32_t)0x00008000)
- #define **RCC_AHB1Periph_SRAM1** ((uint32_t)0x00010000)
- #define **RCC_AHB1Periph_SRAM2** ((uint32_t)0x00020000)
- #define **RCC_AHB1Periph_BKPSRAM** ((uint32_t)0x00040000)
- #define **RCC_AHB1Periph_CCMDATARAMEN** ((uint32_t)0x00100000)
- #define **RCC_AHB1Periph_DMA1** ((uint32_t)0x00200000)
- #define **RCC_AHB1Periph_DMA2** ((uint32_t)0x00400000)
- #define **RCC_AHB1Periph_ETH_MAC** ((uint32_t)0x02000000)
- #define **RCC_AHB1Periph_ETH_MAC_Tx** ((uint32_t)0x04000000)
- #define **RCC_AHB1Periph_ETH_MAC_Rx** ((uint32_t)0x08000000)
- #define **RCC_AHB1Periph_ETH_MAC_PTP** ((uint32_t)0x10000000)
- #define **RCC_AHB1Periph_OTG_HS** ((uint32_t)0x20000000)
- #define **RCC_AHB1Periph_OTG_HS_ULPI** ((uint32_t)0x40000000)
- #define **IS_RCC_AHB1_CLOCK_PERIPH**(PERIPH) ((((PERIPH) & 0x818BEE00) == 0x00) && ((PERIPH) != 0x00))
- #define **IS_RCC_AHB1_RESET_PERIPH**(PERIPH) ((((PERIPH) & 0xDD9FEE00) == 0x00) && ((PERIPH) != 0x00))
- #define **IS_RCC_AHB1_LPMODE_PERIPH**(PERIPH) ((((PERIPH) & 0x81986E00) == 0x00) && ((PERIPH) != 0x00))
- #define **RCC_AHB2Periph_DCMI** ((uint32_t)0x00000001)
- #define **RCC_AHB2Periph_CRYP** ((uint32_t)0x00000010)
- #define **RCC_AHB2Periph_HASH** ((uint32_t)0x00000020)
- #define **RCC_AHB2Periph_RNG** ((uint32_t)0x00000040)
- #define **RCC_AHB2Periph_OTG_FS** ((uint32_t)0x00000080)
- #define **IS_RCC_AHB2_PERIPH**(PERIPH) ((((PERIPH) & 0xFFFFFF0E) == 0x00) && ((PERIPH) != 0x00))
- #define **RCC_AHB3Periph_FSMC** ((uint32_t)0x00000001)
- #define **IS_RCC_AHB3_PERIPH**(PERIPH) ((((PERIPH) & 0xFFFFFFFE) == 0x00) && ((PERIPH) != 0x00))
- #define **RCC_APB1Periph_TIM2** ((uint32_t)0x00000001)
- #define **RCC_APB1Periph_TIM3** ((uint32_t)0x00000002)
- #define **RCC_APB1Periph_TIM4** ((uint32_t)0x00000004)
- #define **RCC_APB1Periph_TIM5** ((uint32_t)0x00000008)
- #define **RCC_APB1Periph_TIM6** ((uint32_t)0x00000010)
- #define **RCC_APB1Periph_TIM7** ((uint32_t)0x00000020)
- #define **RCC_APB1Periph_TIM12** ((uint32_t)0x00000040)
- #define **RCC_APB1Periph_TIM13** ((uint32_t)0x00000080)
- #define **RCC_APB1Periph_TIM14** ((uint32_t)0x00000100)
- #define **RCC_APB1Periph_WWDG** ((uint32_t)0x00000800)
- #define **RCC_APB1Periph_SPI2** ((uint32_t)0x00004000)
- #define **RCC_APB1Periph_SPI3** ((uint32_t)0x00008000)
- #define **RCC_APB1Periph_USART2** ((uint32_t)0x00020000)
- #define **RCC_APB1Periph_USART3** ((uint32_t)0x00040000)
- #define **RCC_APB1Periph_UART4** ((uint32_t)0x00080000)

- #define **RCC_APB1Periph_UART5** ((uint32_t)0x00100000)
- #define **RCC_APB1Periph_I2C1** ((uint32_t)0x00200000)
- #define **RCC_APB1Periph_I2C2** ((uint32_t)0x00400000)
- #define **RCC_APB1Periph_I2C3** ((uint32_t)0x00800000)
- #define **RCC_APB1Periph_CAN1** ((uint32_t)0x02000000)
- #define **RCC_APB1Periph_CAN2** ((uint32_t)0x04000000)
- #define **RCC_APB1Periph_PWR** ((uint32_t)0x10000000)
- #define **RCC_APB1Periph_DAC** ((uint32_t)0x20000000)
- #define **IS_RCC_APB1_PERIPH**(PERIPH) ((((PERIPH) & 0xC9013600) == 0x00) && ((PERIPH) != 0x00))
- #define **RCC_APB2Periph_TIM1** ((uint32_t)0x00000001)
- #define **RCC_APB2Periph_TIM8** ((uint32_t)0x00000002)
- #define **RCC_APB2Periph_USART1** ((uint32_t)0x00000010)
- #define **RCC_APB2Periph_USART6** ((uint32_t)0x00000020)
- #define **RCC_APB2Periph_ADC** ((uint32_t)0x00000100)
- #define **RCC_APB2Periph_ADC1** ((uint32_t)0x00000100)
- #define **RCC_APB2Periph_ADC2** ((uint32_t)0x00000200)
- #define **RCC_APB2Periph_ADC3** ((uint32_t)0x00000400)
- #define **RCC_APB2Periph_SDIO** ((uint32_t)0x00000800)
- #define **RCC_APB2Periph_SPI1** ((uint32_t)0x00001000)
- #define **RCC_APB2Periph_SYSCFG** ((uint32_t)0x00004000)
- #define **RCC_APB2Periph_TIM9** ((uint32_t)0x00010000)
- #define **RCC_APB2Periph_TIM10** ((uint32_t)0x00020000)
- #define **RCC_APB2Periph_TIM11** ((uint32_t)0x00040000)
- #define **IS_RCC_APB2_PERIPH**(PERIPH) ((((PERIPH) & 0xFFF8A0CC) == 0x00) && ((PERIPH) != 0x00))
- #define **IS_RCC_APB2_RESET_PERIPH**(PERIPH) ((((PERIPH) & 0xFFF8A6CC) == 0x00) && ((PERIPH) != 0x00))
- #define **RCC_MCO1Source_HSI** ((uint32_t)0x00000000)
- #define **RCC_MCO1Source_LSE** ((uint32_t)0x00200000)
- #define **RCC_MCO1Source_HSE** ((uint32_t)0x00400000)
- #define **RCC_MCO1Source_PLLCLK** ((uint32_t)0x00600000)
- #define **RCC_MCO1Div_1** ((uint32_t)0x00000000)
- #define **RCC_MCO1Div_2** ((uint32_t)0x04000000)
- #define **RCC_MCO1Div_3** ((uint32_t)0x05000000)
- #define **RCC_MCO1Div_4** ((uint32_t)0x06000000)
- #define **RCC_MCO1Div_5** ((uint32_t)0x07000000)
- #define IS_RCC_MCO1SOURCE(SOURCE)
- #define IS_RCC_MCO1DIV(DIV)
- #define **RCC_MCO2Source_SYSCLK** ((uint32_t)0x00000000)
- #define **RCC_MCO2Source_PLLI2SCLK** ((uint32_t)0x40000000)
- #define **RCC_MCO2Source_HSE** ((uint32_t)0x80000000)
- #define **RCC_MCO2Source_PLLCLK** ((uint32_t)0xC0000000)
- #define **RCC_MCO2Div_1** ((uint32_t)0x00000000)
- #define **RCC_MCO2Div_2** ((uint32_t)0x20000000)
- #define **RCC_MCO2Div_3** ((uint32_t)0x28000000)
- #define **RCC_MCO2Div_4** ((uint32_t)0x30000000)
- #define **RCC_MCO2Div_5** ((uint32_t)0x38000000)
- #define IS_RCC_MCO2SOURCE(SOURCE)
- #define IS_RCC_MCO2DIV(DIV)
- #define **RCC_FLAG_HSIRDY** ((uint8_t)0x21)
- #define **RCC_FLAG_HSERDY** ((uint8_t)0x31)
- #define **RCC_FLAG_PLLRDY** ((uint8_t)0x39)
- #define **RCC_FLAG_PLLI2SRDY** ((uint8_t)0x3B)
- #define **RCC_FLAG_LSERDY** ((uint8_t)0x41)
- #define **RCC_FLAG_LSIRDY** ((uint8_t)0x61)
- #define **RCC_FLAG_BORRST** ((uint8_t)0x79)

- #define **RCC_FLAG_PINRST** ((uint8_t)0x7A)
- #define **RCC_FLAG_PORRST** ((uint8_t)0x7B)
- #define **RCC_FLAG_SFTRST** ((uint8_t)0x7C)
- #define **RCC_FLAG_IWDGRST** ((uint8_t)0x7D)
- #define **RCC_FLAG_WWDGRST** ((uint8_t)0x7E)
- #define **RCC_FLAG_LPWRRST** ((uint8_t)0x7F)
- #define IS_RCC_FLAG(FLAG)
- #define **IS_RCC_CALIBRATION_VALUE**(VALUE) ((VALUE) <= 0x1F)

Functions

- void RCC_DeInit (void)

  Resets the RCC clock configuration to the default reset state.
- void RCC_HSEConfig (uint8_t RCC_HSE)

  Configures the External High Speed oscillator (HSE).
- ErrorStatus RCC_WaitForHSEStartUp (void)

  Waits for HSE start-up.
- void RCC_AdjustHSICalibrationValue (uint8_t HSICalibrationValue)

  Adjusts the Internal High Speed oscillator (HSI) calibration value.
- void RCC_HSICmd (FunctionalState NewState)

  Enables or disables the Internal High Speed oscillator (HSI).
- void RCC_LSEConfig (uint8_t RCC_LSE)

  Configures the External Low Speed oscillator (LSE).
- void RCC_LSICmd (FunctionalState NewState)

  Enables or disables the Internal Low Speed oscillator (LSI).
- void RCC_PLLConfig (uint32_t RCC_PLLSource, uint32_t PLLM, uint32_t PLLN, uint32_t PLLP, uint32_t PLLQ)

  Configures the main PLL clock source, multiplication and division factors.
- void RCC_PLLCmd (FunctionalState NewState)

  Enables or disables the main PLL.
- void RCC_PLLI2SConfig (uint32_t PLLI2SN, uint32_t PLLI2SR)

  Configures the PLLI2S clock multiplication and division factors.
- void RCC_PLLI2SCmd (FunctionalState NewState)

  Enables or disables the PLLI2S.
- void RCC_ClockSecuritySystemCmd (FunctionalState NewState)

  Enables or disables the Clock Security System.
- void RCC_MCO1Config (uint32_t RCC_MCO1Source, uint32_t RCC_MCO1Div)

  Selects the clock source to output on MCO1 pin(PA8).
- void RCC_MCO2Config (uint32_t RCC_MCO2Source, uint32_t RCC_MCO2Div)

  Selects the clock source to output on MCO2 pin(PC9).
- void RCC_SYSCLKConfig (uint32_t RCC_SYSCLKSource)

  Configures the system clock (SYSCLK).
- uint8_t RCC_GetSYSCLKSource (void)

  Returns the clock source used as system clock.
- void RCC_HCLKConfig (uint32_t RCC_SYSCLK)

  Configures the AHB clock (HCLK).
- void RCC_PCLK1Config (uint32_t RCC_HCLK)

  Configures the Low Speed APB clock (PCLK1).
- void RCC_PCLK2Config (uint32_t RCC_HCLK)

  Configures the High Speed APB clock (PCLK2).
- void RCC_GetClocksFreq (RCC_ClocksTypeDef *RCC_Clocks)

Returns the frequencies of different on chip clocks; SYSCLK, HCLK, PCLK1 and PCLK2.

- void RCC_RTCCLKConfig (uint32_t RCC_RTCCLKSource)

  Configures the RTC clock (RTCCLK).
- void RCC_RTCCLKCmd (FunctionalState NewState)

  Enables or disables the RTC clock.
- void RCC_BackupResetCmd (FunctionalState NewState)

  Forces or releases the Backup domain reset.
- void RCC_I2SCLKConfig (uint32_t RCC_I2SCLKSource)

  Configures the I2S clock source (I2SCLK).
- void RCC_AHB1PeriphClockCmd (uint32_t RCC_AHB1Periph, FunctionalState NewState)

  Enables or disables the AHB1 peripheral clock.
- void RCC_AHB2PeriphClockCmd (uint32_t RCC_AHB2Periph, FunctionalState NewState)

  Enables or disables the AHB2 peripheral clock.
- void RCC_AHB3PeriphClockCmd (uint32_t RCC_AHB3Periph, FunctionalState NewState)

  Enables or disables the AHB3 peripheral clock.
- void RCC_APB1PeriphClockCmd (uint32_t RCC_APB1Periph, FunctionalState NewState)

  Enables or disables the Low Speed APB (APB1) peripheral clock.
- void RCC_APB2PeriphClockCmd (uint32_t RCC_APB2Periph, FunctionalState NewState)

  Enables or disables the High Speed APB (APB2) peripheral clock.
- void RCC_AHB1PeriphResetCmd (uint32_t RCC_AHB1Periph, FunctionalState NewState)

  Forces or releases AHB1 peripheral reset.
- void RCC_AHB2PeriphResetCmd (uint32_t RCC_AHB2Periph, FunctionalState NewState)

  Forces or releases AHB2 peripheral reset.
- void RCC_AHB3PeriphResetCmd (uint32_t RCC_AHB3Periph, FunctionalState NewState)

  Forces or releases AHB3 peripheral reset.
- void RCC_APB1PeriphResetCmd (uint32_t RCC_APB1Periph, FunctionalState NewState)

  Forces or releases Low Speed APB (APB1) peripheral reset.
- void RCC_APB2PeriphResetCmd (uint32_t RCC_APB2Periph, FunctionalState NewState)

  Forces or releases High Speed APB (APB2) peripheral reset.
- void RCC_AHB1PeriphClockLPModeCmd (uint32_t RCC_AHB1Periph, FunctionalState NewState)

  Enables or disables the AHB1 peripheral clock during Low Power (Sleep) mode.
- void RCC_AHB2PeriphClockLPModeCmd (uint32_t RCC_AHB2Periph, FunctionalState NewState)

  Enables or disables the AHB2 peripheral clock during Low Power (Sleep) mode.
- void RCC_AHB3PeriphClockLPModeCmd (uint32_t RCC_AHB3Periph, FunctionalState NewState)

  Enables or disables the AHB3 peripheral clock during Low Power (Sleep) mode.
- void RCC_APB1PeriphClockLPModeCmd (uint32_t RCC_APB1Periph, FunctionalState NewState)

  Enables or disables the APB1 peripheral clock during Low Power (Sleep) mode.
- void RCC_APB2PeriphClockLPModeCmd (uint32_t RCC_APB2Periph, FunctionalState NewState)

  Enables or disables the APB2 peripheral clock during Low Power (Sleep) mode.
- void RCC_ITConfig (uint8_t RCC_IT, FunctionalState NewState)

  Enables or disables the specified RCC interrupts.
- FlagStatus RCC_GetFlagStatus (uint8_t RCC_FLAG)

  Checks whether the specified RCC flag is set or not.
- void RCC_ClearFlag (void)

  Clears the RCC reset flags. The reset flags are: RCC_FLAG_PINRST, RCC_FLAG_PORRST, RCC_FLAG_SFTRST, RCC_FLAG_IWDGRST, RCC_FLAG_WWDGRST, RCC_FLAG_LPWRRST.
- ITStatus RCC_GetITStatus (uint8_t RCC_IT)

  Checks whether the specified RCC interrupt has occurred or not.
- void RCC_ClearITPendingBit (uint8_t RCC_IT)

  Clears the RCC's interrupt pending bits.

7.30.1 Detailed Description

This file contains all the functions prototypes for the RCC firmware library.

Author

MCD Application Team

Version

V1.0.0

Date

30-September-2011

Attention

7.31 stm32f4xx_rcc.h

Go to the documentation of this file.

```
00001
00022 /* Define to prevent recursive inclusion -------------------------------------*/
00023 #ifndef __STM32F4xx_RCC_H
00024 #define __STM32F4xx_RCC_H
00025
00026 #ifdef __cplusplus
00027  extern "C" {
00028 #endif
00029
00030 /* Includes ------------------------------------------------------------------*/
00031 #include "stm32f4xx.h"
00032
00041 /* Exported types ------------------------------------------------------------*/
00042 typedef struct
00043 {
00044   uint32_t SYSCLK_Frequency;
00045   uint32_t HCLK_Frequency;
00046   uint32_t PCLK1_Frequency;
00047   uint32_t PCLK2_Frequency;
00048 }RCC_ClocksTypeDef;
00049
00050 /* Exported constants --------------------------------------------------------*/
00051
00059 #define RCC_HSE_OFF                      ((uint8_t)0x00)
00060 #define RCC_HSE_ON                       ((uint8_t)0x01)
```

```
00061 #define RCC_HSE_Bypass                   ((uint8_t)0x05)
00062 #define IS_RCC_HSE(HSE) (((HSE) == RCC_HSE_OFF) || ((HSE) == RCC_HSE_ON) || \
00063                          ((HSE) == RCC_HSE_Bypass))
00071 #define RCC_PLLSource_HSI                ((uint32_t)0x00000000)
00072 #define RCC_PLLSource_HSE                ((uint32_t)0x00400000)
00073 #define IS_RCC_PLL_SOURCE(SOURCE) (((SOURCE) == RCC_PLLSource_HSI) || \
00074                                    ((SOURCE) == RCC_PLLSource_HSE))
00075 #define IS_RCC_PLLM_VALUE(VALUE) ((VALUE) <= 63)
00076 #define IS_RCC_PLLN_VALUE(VALUE) ((192 <= (VALUE)) && ((VALUE) <= 432))
00077 #define IS_RCC_PLLP_VALUE(VALUE) (((VALUE) == 2) || ((VALUE) == 4) || ((VALUE) == 6) || ((VALUE) ==
      8))
00078 #define IS_RCC_PLLQ_VALUE(VALUE) ((4 <= (VALUE)) && ((VALUE) <= 15))
00079
00080 #define IS_RCC_PLLI2SN_VALUE(VALUE) ((192 <= (VALUE)) && ((VALUE) <= 432))
00081 #define IS_RCC_PLLI2SR_VALUE(VALUE) ((2 <= (VALUE)) && ((VALUE) <= 7))
00089 #define RCC_SYSCLKSource_HSI             ((uint32_t)0x00000000)
00090 #define RCC_SYSCLKSource_HSE             ((uint32_t)0x00000001)
00091 #define RCC_SYSCLKSource_PLLCLK          ((uint32_t)0x00000002)
00092 #define IS_RCC_SYSCLK_SOURCE(SOURCE) (((SOURCE) == RCC_SYSCLKSource_HSI) || \
00093                                       ((SOURCE) == RCC_SYSCLKSource_HSE) || \
00094                                       ((SOURCE) == RCC_SYSCLKSource_PLLCLK))
00102 #define RCC_SYSCLK_Div1                  ((uint32_t)0x00000000)
00103 #define RCC_SYSCLK_Div2                  ((uint32_t)0x00000080)
00104 #define RCC_SYSCLK_Div4                  ((uint32_t)0x00000090)
00105 #define RCC_SYSCLK_Div8                  ((uint32_t)0x000000A0)
00106 #define RCC_SYSCLK_Div16                 ((uint32_t)0x000000B0)
00107 #define RCC_SYSCLK_Div64                 ((uint32_t)0x000000C0)
00108 #define RCC_SYSCLK_Div128                ((uint32_t)0x000000D0)
00109 #define RCC_SYSCLK_Div256                ((uint32_t)0x000000E0)
00110 #define RCC_SYSCLK_Div512                ((uint32_t)0x000000F0)
00111 #define IS_RCC_HCLK(HCLK) (((HCLK) == RCC_SYSCLK_Div1) || ((HCLK) == RCC_SYSCLK_Div2) || \
00112                            ((HCLK) == RCC_SYSCLK_Div4) || ((HCLK) == RCC_SYSCLK_Div8) || \
00113                            ((HCLK) == RCC_SYSCLK_Div16) || ((HCLK) == RCC_SYSCLK_Div64) || \
00114                            ((HCLK) == RCC_SYSCLK_Div128) || ((HCLK) == RCC_SYSCLK_Div256) || \
00115                            ((HCLK) == RCC_SYSCLK_Div512))
00123 #define RCC_HCLK_Div1                    ((uint32_t)0x00000000)
00124 #define RCC_HCLK_Div2                    ((uint32_t)0x00001000)
00125 #define RCC_HCLK_Div4                    ((uint32_t)0x00001400)
00126 #define RCC_HCLK_Div8                    ((uint32_t)0x00001800)
00127 #define RCC_HCLK_Div16                   ((uint32_t)0x00001C00)
00128 #define IS_RCC_PCLK(PCLK) (((PCLK) == RCC_HCLK_Div1) || ((PCLK) == RCC_HCLK_Div2) || \
00129                            ((PCLK) == RCC_HCLK_Div4) || ((PCLK) == RCC_HCLK_Div8) || \
00130                            ((PCLK) == RCC_HCLK_Div16))
00138 #define RCC_IT_LSIRDY                    ((uint8_t)0x01)
00139 #define RCC_IT_LSERDY                    ((uint8_t)0x02)
00140 #define RCC_IT_HSIRDY                    ((uint8_t)0x04)
00141 #define RCC_IT_HSERDY                    ((uint8_t)0x08)
00142 #define RCC_IT_PLLRDY                    ((uint8_t)0x10)
00143 #define RCC_IT_PLLI2SRDY                 ((uint8_t)0x20)
00144 #define RCC_IT_CSS                       ((uint8_t)0x80)
00145 #define IS_RCC_IT(IT) ((((IT) & (uint8_t)0xC0) == 0x00) && ((IT) != 0x00))
00146 #define IS_RCC_GET_IT(IT) (((IT) == RCC_IT_LSIRDY) || ((IT) == RCC_IT_LSERDY) || \
00147                            ((IT) == RCC_IT_HSIRDY) || ((IT) == RCC_IT_HSERDY) || \
00148                            ((IT) == RCC_IT_PLLRDY) || ((IT) == RCC_IT_CSS) || \
00149                            ((IT) == RCC_IT_PLLI2SRDY))
00150 #define IS_RCC_CLEAR_IT(IT) ((((IT) & (uint8_t)0x40) == 0x00) && ((IT) != 0x00))
00158 #define RCC_LSE_OFF                      ((uint8_t)0x00)
00159 #define RCC_LSE_ON                       ((uint8_t)0x01)
00160 #define RCC_LSE_Bypass                   ((uint8_t)0x04)
00161 #define IS_RCC_LSE(LSE) (((LSE) == RCC_LSE_OFF) || ((LSE) == RCC_LSE_ON) || \
00162                          ((LSE) == RCC_LSE_Bypass))
00170 #define RCC_RTCCLKSource_LSE             ((uint32_t)0x00000100)
00171 #define RCC_RTCCLKSource_LSI             ((uint32_t)0x00000200)
00172 #define RCC_RTCCLKSource_HSE_Div2        ((uint32_t)0x00020300)
00173 #define RCC_RTCCLKSource_HSE_Div3        ((uint32_t)0x00030300)
00174 #define RCC_RTCCLKSource_HSE_Div4        ((uint32_t)0x00040300)
00175 #define RCC_RTCCLKSource_HSE_Div5        ((uint32_t)0x00050300)
00176 #define RCC_RTCCLKSource_HSE_Div6        ((uint32_t)0x00060300)
00177 #define RCC_RTCCLKSource_HSE_Div7        ((uint32_t)0x00070300)
00178 #define RCC_RTCCLKSource_HSE_Div8        ((uint32_t)0x00080300)
00179 #define RCC_RTCCLKSource_HSE_Div9        ((uint32_t)0x00090300)
00180 #define RCC_RTCCLKSource_HSE_Div10       ((uint32_t)0x000A0300)
00181 #define RCC_RTCCLKSource_HSE_Div11       ((uint32_t)0x000B0300)
00182 #define RCC_RTCCLKSource_HSE_Div12       ((uint32_t)0x000C0300)
00183 #define RCC_RTCCLKSource_HSE_Div13       ((uint32_t)0x000D0300)
00184 #define RCC_RTCCLKSource_HSE_Div14       ((uint32_t)0x000E0300)
00185 #define RCC_RTCCLKSource_HSE_Div15       ((uint32_t)0x000F0300)
00186 #define RCC_RTCCLKSource_HSE_Div16       ((uint32_t)0x00100300)
00187 #define RCC_RTCCLKSource_HSE_Div17       ((uint32_t)0x00110300)
00188 #define RCC_RTCCLKSource_HSE_Div18       ((uint32_t)0x00120300)
00189 #define RCC_RTCCLKSource_HSE_Div19       ((uint32_t)0x00130300)
00190 #define RCC_RTCCLKSource_HSE_Div20       ((uint32_t)0x00140300)
00191 #define RCC_RTCCLKSource_HSE_Div21       ((uint32_t)0x00150300)
00192 #define RCC_RTCCLKSource_HSE_Div22       ((uint32_t)0x00160300)
00193 #define RCC_RTCCLKSource_HSE_Div23       ((uint32_t)0x00170300)
00194 #define RCC_RTCCLKSource_HSE_Div24       ((uint32_t)0x00180300)
00195 #define RCC_RTCCLKSource_HSE_Div25       ((uint32_t)0x00190300)
```

```
00196 #define RCC_RTCCLKSource_HSE_Div26       ((uint32_t)0x001A0300)
00197 #define RCC_RTCCLKSource_HSE_Div27       ((uint32_t)0x001B0300)
00198 #define RCC_RTCCLKSource_HSE_Div28       ((uint32_t)0x001C0300)
00199 #define RCC_RTCCLKSource_HSE_Div29       ((uint32_t)0x001D0300)
00200 #define RCC_RTCCLKSource_HSE_Div30       ((uint32_t)0x001E0300)
00201 #define RCC_RTCCLKSource_HSE_Div31       ((uint32_t)0x001F0300)
00202 #define IS_RCC_RTCCLK_SOURCE(SOURCE) (((SOURCE) == RCC_RTCCLKSource_LSE) || \
00203                                       ((SOURCE) == RCC_RTCCLKSource_LSI) || \
00204                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div2) || \
00205                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div3) || \
00206                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div4) || \
00207                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div5) || \
00208                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div6) || \
00209                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div7) || \
00210                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div8) || \
00211                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div9) || \
00212                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div10) || \
00213                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div11) || \
00214                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div12) || \
00215                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div13) || \
00216                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div14) || \
00217                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div15) || \
00218                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div16) || \
00219                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div17) || \
00220                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div18) || \
00221                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div19) || \
00222                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div20) || \
00223                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div21) || \
00224                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div22) || \
00225                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div23) || \
00226                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div24) || \
00227                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div25) || \
00228                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div26) || \
00229                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div27) || \
00230                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div28) || \
00231                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div29) || \
00232                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div30) || \
00233                                       ((SOURCE) == RCC_RTCCLKSource_HSE_Div31))
00241 #define RCC_I2S2CLKSource_PLLI2S             ((uint8_t)0x00)
00242 #define RCC_I2S2CLKSource_Ext                ((uint8_t)0x01)
00243
00244 #define IS_RCC_I2SCLK_SOURCE(SOURCE) (((SOURCE) == RCC_I2S2CLKSource_PLLI2S) || ((SOURCE) ==
      RCC_I2S2CLKSource_Ext))
00252 #define RCC_AHB1Periph_GPIOA             ((uint32_t)0x00000001)
00253 #define RCC_AHB1Periph_GPIOB             ((uint32_t)0x00000002)
00254 #define RCC_AHB1Periph_GPIOC             ((uint32_t)0x00000004)
00255 #define RCC_AHB1Periph_GPIOD             ((uint32_t)0x00000008)
00256 #define RCC_AHB1Periph_GPIOE             ((uint32_t)0x00000010)
00257 #define RCC_AHB1Periph_GPIOF             ((uint32_t)0x00000020)
00258 #define RCC_AHB1Periph_GPIOG             ((uint32_t)0x00000040)
00259 #define RCC_AHB1Periph_GPIOH             ((uint32_t)0x00000080)
00260 #define RCC_AHB1Periph_GPIOI             ((uint32_t)0x00000100)
00261 #define RCC_AHB1Periph_CRC               ((uint32_t)0x00001000)
00262 #define RCC_AHB1Periph_FLITF             ((uint32_t)0x00008000)
00263 #define RCC_AHB1Periph_SRAM1             ((uint32_t)0x00010000)
00264 #define RCC_AHB1Periph_SRAM2             ((uint32_t)0x00020000)
00265 #define RCC_AHB1Periph_BKPSRAM           ((uint32_t)0x00040000)
00266 #define RCC_AHB1Periph_CCMDATARAMEN      ((uint32_t)0x00100000)
00267 #define RCC_AHB1Periph_DMA1              ((uint32_t)0x00200000)
00268 #define RCC_AHB1Periph_DMA2              ((uint32_t)0x00400000)
00269 #define RCC_AHB1Periph_ETH_MAC           ((uint32_t)0x02000000)
00270 #define RCC_AHB1Periph_ETH_MAC_Tx        ((uint32_t)0x04000000)
00271 #define RCC_AHB1Periph_ETH_MAC_Rx        ((uint32_t)0x08000000)
00272 #define RCC_AHB1Periph_ETH_MAC_PTP       ((uint32_t)0x10000000)
00273 #define RCC_AHB1Periph_OTG_HS            ((uint32_t)0x20000000)
00274 #define RCC_AHB1Periph_OTG_HS_ULPI       ((uint32_t)0x40000000)
00275 #define IS_RCC_AHB1_CLOCK_PERIPH(PERIPH) ((((PERIPH) & 0x818BEE00) == 0x00) && ((PERIPH) != 0x00))
00276 #define IS_RCC_AHB1_RESET_PERIPH(PERIPH) ((((PERIPH) & 0xDD9FEE00) == 0x00) && ((PERIPH) != 0x00))
00277 #define IS_RCC_AHB1_LPMODE_PERIPH(PERIPH) ((((PERIPH) & 0x81986E00) == 0x00) && ((PERIPH) != 0x00))
00285 #define RCC_AHB2Periph_DCMI              ((uint32_t)0x00000001)
00286 #define RCC_AHB2Periph_CRYP              ((uint32_t)0x00000010)
00287 #define RCC_AHB2Periph_HASH              ((uint32_t)0x00000020)
00288 #define RCC_AHB2Periph_RNG               ((uint32_t)0x00000040)
00289 #define RCC_AHB2Periph_OTG_FS            ((uint32_t)0x00000080)
00290 #define IS_RCC_AHB2_PERIPH(PERIPH) ((((PERIPH) & 0xFFFFFF0E) == 0x00) && ((PERIPH) != 0x00))
00298 #define RCC_AHB3Periph_FSMC              ((uint32_t)0x00000001)
00299 #define IS_RCC_AHB3_PERIPH(PERIPH) ((((PERIPH) & 0xFFFFFFFE) == 0x00) && ((PERIPH) != 0x00))
00307 #define RCC_APB1Periph_TIM2              ((uint32_t)0x00000001)
00308 #define RCC_APB1Periph_TIM3              ((uint32_t)0x00000002)
00309 #define RCC_APB1Periph_TIM4              ((uint32_t)0x00000004)
00310 #define RCC_APB1Periph_TIM5              ((uint32_t)0x00000008)
00311 #define RCC_APB1Periph_TIM6              ((uint32_t)0x00000010)
00312 #define RCC_APB1Periph_TIM7              ((uint32_t)0x00000020)
00313 #define RCC_APB1Periph_TIM12             ((uint32_t)0x00000040)
00314 #define RCC_APB1Periph_TIM13             ((uint32_t)0x00000080)
00315 #define RCC_APB1Periph_TIM14             ((uint32_t)0x00000100)
00316 #define RCC_APB1Periph_WWDG              ((uint32_t)0x00000800)
```

```
00317 #define RCC_APB1Periph_SPI2              ((uint32_t)0x00004000)
00318 #define RCC_APB1Periph_SPI3              ((uint32_t)0x00008000)
00319 #define RCC_APB1Periph_USART2            ((uint32_t)0x00020000)
00320 #define RCC_APB1Periph_USART3            ((uint32_t)0x00040000)
00321 #define RCC_APB1Periph_UART4             ((uint32_t)0x00080000)
00322 #define RCC_APB1Periph_UART5             ((uint32_t)0x00100000)
00323 #define RCC_APB1Periph_I2C1              ((uint32_t)0x00200000)
00324 #define RCC_APB1Periph_I2C2              ((uint32_t)0x00400000)
00325 #define RCC_APB1Periph_I2C3              ((uint32_t)0x00800000)
00326 #define RCC_APB1Periph_CAN1              ((uint32_t)0x02000000)
00327 #define RCC_APB1Periph_CAN2              ((uint32_t)0x04000000)
00328 #define RCC_APB1Periph_PWR               ((uint32_t)0x10000000)
00329 #define RCC_APB1Periph_DAC               ((uint32_t)0x20000000)
00330 #define IS_RCC_APB1_PERIPH(PERIPH) ((((PERIPH) & 0xC9013600) == 0x00) && ((PERIPH) != 0x00))
00338 #define RCC_APB2Periph_TIM1              ((uint32_t)0x00000001)
00339 #define RCC_APB2Periph_TIM8              ((uint32_t)0x00000002)
00340 #define RCC_APB2Periph_USART1            ((uint32_t)0x00000010)
00341 #define RCC_APB2Periph_USART6            ((uint32_t)0x00000020)
00342 #define RCC_APB2Periph_ADC               ((uint32_t)0x00000100)
00343 #define RCC_APB2Periph_ADC1              ((uint32_t)0x00000100)
00344 #define RCC_APB2Periph_ADC2              ((uint32_t)0x00000200)
00345 #define RCC_APB2Periph_ADC3              ((uint32_t)0x00000400)
00346 #define RCC_APB2Periph_SDIO              ((uint32_t)0x00000800)
00347 #define RCC_APB2Periph_SPI1              ((uint32_t)0x00001000)
00348 #define RCC_APB2Periph_SYSCFG            ((uint32_t)0x00004000)
00349 #define RCC_APB2Periph_TIM9              ((uint32_t)0x00010000)
00350 #define RCC_APB2Periph_TIM10             ((uint32_t)0x00020000)
00351 #define RCC_APB2Periph_TIM11             ((uint32_t)0x00040000)
00352 #define IS_RCC_APB2_PERIPH(PERIPH) ((((PERIPH) & 0xFFF8A0CC) == 0x00) && ((PERIPH) != 0x00))
00353 #define IS_RCC_APB2_RESET_PERIPH(PERIPH) ((((PERIPH) & 0xFFF8A6CC) == 0x00) && ((PERIPH) != 0x00))
00361 #define RCC_MCO1Source_HSI               ((uint32_t)0x00000000)
00362 #define RCC_MCO1Source_LSE               ((uint32_t)0x00200000)
00363 #define RCC_MCO1Source_HSE               ((uint32_t)0x00400000)
00364 #define RCC_MCO1Source_PLLCLK            ((uint32_t)0x00600000)
00365 #define RCC_MCO1Div_1                    ((uint32_t)0x00000000)
00366 #define RCC_MCO1Div_2                    ((uint32_t)0x04000000)
00367 #define RCC_MCO1Div_3                    ((uint32_t)0x05000000)
00368 #define RCC_MCO1Div_4                    ((uint32_t)0x06000000)
00369 #define RCC_MCO1Div_5                    ((uint32_t)0x07000000)
00370 #define IS_RCC_MCO1SOURCE(SOURCE) (((SOURCE) == RCC_MCO1Source_HSI) || ((SOURCE) ==
      RCC_MCO1Source_LSE) || \
00371                                    ((SOURCE) == RCC_MCO1Source_HSE) || ((SOURCE) ==
      RCC_MCO1Source_PLLCLK))
00372
00373 #define IS_RCC_MCO1DIV(DIV) (((DIV) == RCC_MCO1Div_1) || ((DIV) == RCC_MCO1Div_2) || \
00374                              ((DIV) == RCC_MCO1Div_3) || ((DIV) == RCC_MCO1Div_4) || \
00375                              ((DIV) == RCC_MCO1Div_5))
00383 #define RCC_MCO2Source_SYSCLK            ((uint32_t)0x00000000)
00384 #define RCC_MCO2Source_PLLI2SCLK         ((uint32_t)0x40000000)
00385 #define RCC_MCO2Source_HSE               ((uint32_t)0x80000000)
00386 #define RCC_MCO2Source_PLLCLK            ((uint32_t)0xC0000000)
00387 #define RCC_MCO2Div_1                    ((uint32_t)0x00000000)
00388 #define RCC_MCO2Div_2                    ((uint32_t)0x20000000)
00389 #define RCC_MCO2Div_3                    ((uint32_t)0x28000000)
00390 #define RCC_MCO2Div_4                    ((uint32_t)0x30000000)
00391 #define RCC_MCO2Div_5                    ((uint32_t)0x38000000)
00392 #define IS_RCC_MCO2SOURCE(SOURCE) (((SOURCE) == RCC_MCO2Source_SYSCLK) || ((SOURCE) ==
      RCC_MCO2Source_PLLI2SCLK)|| \
00393                                    ((SOURCE) == RCC_MCO2Source_HSE) || ((SOURCE) ==
      RCC_MCO2Source_PLLCLK))
00394
00395 #define IS_RCC_MCO2DIV(DIV) (((DIV) == RCC_MCO2Div_1) || ((DIV) == RCC_MCO2Div_2) || \
00396                              ((DIV) == RCC_MCO2Div_3) || ((DIV) == RCC_MCO2Div_4) || \
00397                              ((DIV) == RCC_MCO2Div_5))
00405 #define RCC_FLAG_HSIRDY                  ((uint8_t)0x21)
00406 #define RCC_FLAG_HSERDY                  ((uint8_t)0x31)
00407 #define RCC_FLAG_PLLRDY                  ((uint8_t)0x39)
00408 #define RCC_FLAG_PLLI2SRDY               ((uint8_t)0x3B)
00409 #define RCC_FLAG_LSERDY                  ((uint8_t)0x41)
00410 #define RCC_FLAG_LSIRDY                  ((uint8_t)0x61)
00411 #define RCC_FLAG_BORRST                  ((uint8_t)0x79)
00412 #define RCC_FLAG_PINRST                  ((uint8_t)0x7A)
00413 #define RCC_FLAG_PORRST                  ((uint8_t)0x7B)
00414 #define RCC_FLAG_SFTRST                  ((uint8_t)0x7C)
00415 #define RCC_FLAG_IWDGRST                 ((uint8_t)0x7D)
00416 #define RCC_FLAG_WWDGRST                 ((uint8_t)0x7E)
00417 #define RCC_FLAG_LPWRRST                 ((uint8_t)0x7F)
00418 #define IS_RCC_FLAG(FLAG) (((FLAG) == RCC_FLAG_HSIRDY) || ((FLAG) == RCC_FLAG_HSERDY) || \
00419                            ((FLAG) == RCC_FLAG_PLLRDY) || ((FLAG) == RCC_FLAG_LSERDY) || \
00420                            ((FLAG) == RCC_FLAG_LSIRDY) || ((FLAG) == RCC_FLAG_BORRST) || \
00421                            ((FLAG) == RCC_FLAG_PINRST) || ((FLAG) == RCC_FLAG_PORRST) || \
00422                            ((FLAG) == RCC_FLAG_SFTRST) || ((FLAG) == RCC_FLAG_IWDGRST)|| \
00423                            ((FLAG) == RCC_FLAG_WWDGRST)|| ((FLAG) == RCC_FLAG_LPWRRST)|| \
00424                            ((FLAG) == RCC_FLAG_PLLI2SRDY))
00425 #define IS_RCC_CALIBRATION_VALUE(VALUE) ((VALUE) <= 0x1F)
00434 /* Exported macro ------------------------------------------------------------*/
00435 /* Exported functions --------------------------------------------------------*/
```

```
00436
00437 /* Function used to set the RCC clock configuration to the default reset state */
00438 void RCC_DeInit(void);
00439
00440 /* Internal/external clocks, PLL, CSS and MCO configuration functions *********/
00441 void RCC_HSEConfig(uint8_t RCC_HSE);
00442 ErrorStatus RCC_WaitForHSEStartUp(void);
00443 void RCC_AdjustHSICalibrationValue(uint8_t HSICalibrationValue);
00444 void RCC_HSICmd(FunctionalState NewState);
00445 void RCC_LSEConfig(uint8_t RCC_LSE);
00446 void RCC_LSICmd(FunctionalState NewState);
00447
00448 void RCC_PLLConfig(uint32_t RCC_PLLSource, uint32_t PLLM, uint32_t PLLN, uint32_t PLLP, uint32_t
      PLLQ);
00449 void RCC_PLLCmd(FunctionalState NewState);
00450 void RCC_PLLI2SConfig(uint32_t PLLI2SN, uint32_t PLLI2SR);
00451 void RCC_PLLI2SCmd(FunctionalState NewState);
00452
00453 void RCC_ClockSecuritySystemCmd(FunctionalState NewState);
00454 void RCC_MCO1Config(uint32_t RCC_MCO1Source, uint32_t RCC_MCO1Div);
00455 void RCC_MCO2Config(uint32_t RCC_MCO2Source, uint32_t RCC_MCO2Div);
00456
00457 /* System, AHB and APB busses clocks configuration functions ******************/
00458 void RCC_SYSCLKConfig(uint32_t RCC_SYSCLKSource);
00459 uint8_t RCC_GetSYSCLKSource(void);
00460 void RCC_HCLKConfig(uint32_t RCC_SYSCLK);
00461 void RCC_PCLK1Config(uint32_t RCC_HCLK);
00462 void RCC_PCLK2Config(uint32_t RCC_HCLK);
00463 void RCC_GetClocksFreq(RCC_ClocksTypeDef* RCC_Clocks);
00464
00465 /* Peripheral clocks configuration functions ************************************/
00466 void RCC_RTCCLKConfig(uint32_t RCC_RTCCLKSource);
00467 void RCC_RTCCLKCmd(FunctionalState NewState);
00468 void RCC_BackupResetCmd(FunctionalState NewState);
00469 void RCC_I2SCLKConfig(uint32_t RCC_I2SCLKSource);
00470
00471 void RCC_AHB1PeriphClockCmd(uint32_t RCC_AHB1Periph, FunctionalState NewState);
00472 void RCC_AHB2PeriphClockCmd(uint32_t RCC_AHB2Periph, FunctionalState NewState);
00473 void RCC_AHB3PeriphClockCmd(uint32_t RCC_AHB3Periph, FunctionalState NewState);
00474 void RCC_APB1PeriphClockCmd(uint32_t RCC_APB1Periph, FunctionalState NewState);
00475 void RCC_APB2PeriphClockCmd(uint32_t RCC_APB2Periph, FunctionalState NewState);
00476
00477 void RCC_AHB1PeriphResetCmd(uint32_t RCC_AHB1Periph, FunctionalState NewState);
00478 void RCC_AHB2PeriphResetCmd(uint32_t RCC_AHB2Periph, FunctionalState NewState);
00479 void RCC_AHB3PeriphResetCmd(uint32_t RCC_AHB3Periph, FunctionalState NewState);
00480 void RCC_APB1PeriphResetCmd(uint32_t RCC_APB1Periph, FunctionalState NewState);
00481 void RCC_APB2PeriphResetCmd(uint32_t RCC_APB2Periph, FunctionalState NewState);
00482
00483 void RCC_AHB1PeriphClockLPModeCmd(uint32_t RCC_AHB1Periph, FunctionalState NewState);
00484 void RCC_AHB2PeriphClockLPModeCmd(uint32_t RCC_AHB2Periph, FunctionalState NewState);
00485 void RCC_AHB3PeriphClockLPModeCmd(uint32_t RCC_AHB3Periph, FunctionalState NewState);
00486 void RCC_APB1PeriphClockLPModeCmd(uint32_t RCC_APB1Periph, FunctionalState NewState);
00487 void RCC_APB2PeriphClockLPModeCmd(uint32_t RCC_APB2Periph, FunctionalState NewState);
00488
00489 /* Interrupts and flags management functions **********************************/
00490 void RCC_ITConfig(uint8_t RCC_IT, FunctionalState NewState);
00491 FlagStatus RCC_GetFlagStatus(uint8_t RCC_FLAG);
00492 void RCC_ClearFlag(void);
00493 ITStatus RCC_GetITStatus(uint8_t RCC_IT);
00494 void RCC_ClearITPendingBit(uint8_t RCC_IT);
00495
00496 #ifdef __cplusplus
00497 }
00498 #endif
00499
00500 #endif /* __STM32F4xx_RCC_H */
00501
00510 /****************** (C) COPYRIGHT 2011 STMicroelectronics *****END OF FILE****/
```

7.32 CUBE_IDE/VGA/Core/Inc/stm32f4xx_tim.h File Reference

This file contains all the functions prototypes for the TIM firmware library.

```
#include "stm32f4xx.h"
```

Data Structures

- struct TIM_TimeBaseInitTypeDef

  TIM Time Base Init structure definition

- struct TIM_OCInitTypeDef

  TIM Output Compare Init structure definition

- struct TIM_ICInitTypeDef

  TIM Input Capture Init structure definition

- struct TIM_BDTRInitTypeDef

  BDTR structure definition.

Macros

- #define IS_TIM_ALL_PERIPH(PERIPH)
- #define IS_TIM_LIST1_PERIPH(PERIPH)
- #define IS_TIM_LIST2_PERIPH(PERIPH)
- #define IS_TIM_LIST3_PERIPH(PERIPH)
- #define IS_TIM_LIST4_PERIPH(PERIPH)
- #define IS_TIM_LIST5_PERIPH(PERIPH)
- #define IS_TIM_LIST6_PERIPH(TIMx)
- #define **TIM_OCMode_Timing** ((uint16_t)0x0000)
- #define **TIM_OCMode_Active** ((uint16_t)0x0010)
- #define **TIM_OCMode_Inactive** ((uint16_t)0x0020)
- #define **TIM_OCMode_Toggle** ((uint16_t)0x0030)
- #define **TIM_OCMode_PWM1** ((uint16_t)0x0060)
- #define **TIM_OCMode_PWM2** ((uint16_t)0x0070)
- #define IS_TIM_OC_MODE(MODE)
- #define IS_TIM_OCM(MODE)
- #define **TIM_OPMode_Single** ((uint16_t)0x0008)
- #define **TIM_OPMode_Repetitive** ((uint16_t)0x0000)
- #define IS_TIM_OPM_MODE(MODE)
- #define **TIM_Channel_1** ((uint16_t)0x0000)
- #define **TIM_Channel_2** ((uint16_t)0x0004)
- #define **TIM_Channel_3** ((uint16_t)0x0008)
- #define **TIM_Channel_4** ((uint16_t)0x000C)
- #define IS_TIM_CHANNEL(CHANNEL)
- #define IS_TIM_PWMI_CHANNEL(CHANNEL)
- #define IS_TIM_COMPLEMENTARY_CHANNEL(CHANNEL)
- #define **TIM_CKD_DIV1** ((uint16_t)0x0000)
- #define **TIM_CKD_DIV2** ((uint16_t)0x0100)
- #define **TIM_CKD_DIV4** ((uint16_t)0x0200)
- #define IS_TIM_CKD_DIV(DIV)
- #define **TIM_CounterMode_Up** ((uint16_t)0x0000)
- #define **TIM_CounterMode_Down** ((uint16_t)0x0010)
- #define **TIM_CounterMode_CenterAligned1** ((uint16_t)0x0020)
- #define **TIM_CounterMode_CenterAligned2** ((uint16_t)0x0040)
- #define **TIM_CounterMode_CenterAligned3** ((uint16_t)0x0060)
- #define IS_TIM_COUNTER_MODE(MODE)
- #define **TIM_OCPolarity_High** ((uint16_t)0x0000)
- #define **TIM_OCPolarity_Low** ((uint16_t)0x0002)

- #define IS_TIM_OC_POLARITY(POLARITY)
- #define **TIM_OCNPolarity_High** ((uint16_t)0x0000)
- #define **TIM_OCNPolarity_Low** ((uint16_t)0x0008)
- #define IS_TIM_OCN_POLARITY(POLARITY)
- #define **TIM_OutputState_Disable** ((uint16_t)0x0000)
- #define **TIM_OutputState_Enable** ((uint16_t)0x0001)
- #define IS_TIM_OUTPUT_STATE(STATE)
- #define **TIM_OutputNState_Disable** ((uint16_t)0x0000)
- #define **TIM_OutputNState_Enable** ((uint16_t)0x0004)
- #define IS_TIM_OUTPUTN_STATE(STATE)
- #define **TIM_CCx_Enable** ((uint16_t)0x0001)
- #define **TIM_CCx_Disable** ((uint16_t)0x0000)
- #define IS_TIM_CCX(CCX)
- #define **TIM_CCxN_Enable** ((uint16_t)0x0004)
- #define **TIM_CCxN_Disable** ((uint16_t)0x0000)
- #define IS_TIM_CCXN(CCXN)
- #define **TIM_Break_Enable** ((uint16_t)0x1000)
- #define **TIM_Break_Disable** ((uint16_t)0x0000)
- #define IS_TIM_BREAK_STATE(STATE)
- #define **TIM_BreakPolarity_Low** ((uint16_t)0x0000)
- #define **TIM_BreakPolarity_High** ((uint16_t)0x2000)
- #define IS_TIM_BREAK_POLARITY(POLARITY)
- #define **TIM_AutomaticOutput_Enable** ((uint16_t)0x4000)
- #define **TIM_AutomaticOutput_Disable** ((uint16_t)0x0000)
- #define IS_TIM_AUTOMATIC_OUTPUT_STATE(STATE)
- #define **TIM_LOCKLevel_OFF** ((uint16_t)0x0000)
- #define **TIM_LOCKLevel_1** ((uint16_t)0x0100)
- #define **TIM_LOCKLevel_2** ((uint16_t)0x0200)
- #define **TIM_LOCKLevel_3** ((uint16_t)0x0300)
- #define IS_TIM_LOCK_LEVEL(LEVEL)
- #define **TIM_OSSIState_Enable** ((uint16_t)0x0400)
- #define **TIM_OSSIState_Disable** ((uint16_t)0x0000)
- #define IS_TIM_OSSI_STATE(STATE)
- #define **TIM_OSSRState_Enable** ((uint16_t)0x0800)
- #define **TIM_OSSRState_Disable** ((uint16_t)0x0000)
- #define IS_TIM_OSSR_STATE(STATE)
- #define **TIM_OCIdleState_Set** ((uint16_t)0x0100)
- #define **TIM_OCIdleState_Reset** ((uint16_t)0x0000)
- #define IS_TIM_OCIDLE_STATE(STATE)
- #define **TIM_OCNIdleState_Set** ((uint16_t)0x0200)
- #define **TIM_OCNIdleState_Reset** ((uint16_t)0x0000)
- #define IS_TIM_OCNIDLE_STATE(STATE)
- #define **TIM_ICPolarity_Rising** ((uint16_t)0x0000)
- #define **TIM_ICPolarity_Falling** ((uint16_t)0x0002)
- #define **TIM_ICPolarity_BothEdge** ((uint16_t)0x000A)
- #define IS_TIM_IC_POLARITY(POLARITY)
- #define TIM_ICSelection_DirectTI ((uint16_t)0x0001)
- #define TIM_ICSelection_IndirectTI ((uint16_t)0x0002)
- #define TIM_ICSelection_TRC ((uint16_t)0x0003)
- #define IS_TIM_IC_SELECTION(SELECTION)
- #define TIM_ICPSC_DIV1 ((uint16_t)0x0000)
- #define TIM_ICPSC_DIV2 ((uint16_t)0x0004)
- #define TIM_ICPSC_DIV4 ((uint16_t)0x0008)
- #define TIM_ICPSC_DIV8 ((uint16_t)0x000C)
- #define IS_TIM_IC_PRESCALER(PRESCALER)

- #define **TIM_IT_Update** ((uint16_t)0x0001)
- #define **TIM_IT_CC1** ((uint16_t)0x0002)
- #define **TIM_IT_CC2** ((uint16_t)0x0004)
- #define **TIM_IT_CC3** ((uint16_t)0x0008)
- #define **TIM_IT_CC4** ((uint16_t)0x0010)
- #define **TIM_IT_COM** ((uint16_t)0x0020)
- #define **TIM_IT_Trigger** ((uint16_t)0x0040)
- #define **TIM_IT_Break** ((uint16_t)0x0080)
- #define **IS_TIM_IT**(IT) ((((IT) & (uint16_t)0xFF00) == 0x0000) && ((IT) != 0x0000))
- #define IS_TIM_GET_IT(IT)
- #define **TIM_DMABase_CR1** ((uint16_t)0x0000)
- #define **TIM_DMABase_CR2** ((uint16_t)0x0001)
- #define **TIM_DMABase_SMCR** ((uint16_t)0x0002)
- #define **TIM_DMABase_DIER** ((uint16_t)0x0003)
- #define **TIM_DMABase_SR** ((uint16_t)0x0004)
- #define **TIM_DMABase_EGR** ((uint16_t)0x0005)
- #define **TIM_DMABase_CCMR1** ((uint16_t)0x0006)
- #define **TIM_DMABase_CCMR2** ((uint16_t)0x0007)
- #define **TIM_DMABase_CCER** ((uint16_t)0x0008)
- #define **TIM_DMABase_CNT** ((uint16_t)0x0009)
- #define **TIM_DMABase_PSC** ((uint16_t)0x000A)
- #define **TIM_DMABase_ARR** ((uint16_t)0x000B)
- #define **TIM_DMABase_RCR** ((uint16_t)0x000C)
- #define **TIM_DMABase_CCR1** ((uint16_t)0x000D)
- #define **TIM_DMABase_CCR2** ((uint16_t)0x000E)
- #define **TIM_DMABase_CCR3** ((uint16_t)0x000F)
- #define **TIM_DMABase_CCR4** ((uint16_t)0x0010)
- #define **TIM_DMABase_BDTR** ((uint16_t)0x0011)
- #define **TIM_DMABase_DCR** ((uint16_t)0x0012)
- #define **TIM_DMABase_OR** ((uint16_t)0x0013)
- #define IS_TIM_DMA_BASE(BASE)
- #define **TIM_DMABurstLength_1Transfer** ((uint16_t)0x0000)
- #define **TIM_DMABurstLength_2Transfers** ((uint16_t)0x0100)
- #define **TIM_DMABurstLength_3Transfers** ((uint16_t)0x0200)
- #define **TIM_DMABurstLength_4Transfers** ((uint16_t)0x0300)
- #define **TIM_DMABurstLength_5Transfers** ((uint16_t)0x0400)
- #define **TIM_DMABurstLength_6Transfers** ((uint16_t)0x0500)
- #define **TIM_DMABurstLength_7Transfers** ((uint16_t)0x0600)
- #define **TIM_DMABurstLength_8Transfers** ((uint16_t)0x0700)
- #define **TIM_DMABurstLength_9Transfers** ((uint16_t)0x0800)
- #define **TIM_DMABurstLength_10Transfers** ((uint16_t)0x0900)
- #define **TIM_DMABurstLength_11Transfers** ((uint16_t)0x0A00)
- #define **TIM_DMABurstLength_12Transfers** ((uint16_t)0x0B00)
- #define **TIM_DMABurstLength_13Transfers** ((uint16_t)0x0C00)
- #define **TIM_DMABurstLength_14Transfers** ((uint16_t)0x0D00)
- #define **TIM_DMABurstLength_15Transfers** ((uint16_t)0x0E00)
- #define **TIM_DMABurstLength_16Transfers** ((uint16_t)0x0F00)
- #define **TIM_DMABurstLength_17Transfers** ((uint16_t)0x1000)
- #define **TIM_DMABurstLength_18Transfers** ((uint16_t)0x1100)
- #define IS_TIM_DMA_LENGTH(LENGTH)
- #define **TIM_DMA_Update** ((uint16_t)0x0100)
- #define **TIM_DMA_CC1** ((uint16_t)0x0200)
- #define **TIM_DMA_CC2** ((uint16_t)0x0400)
- #define **TIM_DMA_CC3** ((uint16_t)0x0800)
- #define **TIM_DMA_CC4** ((uint16_t)0x1000)

- #define **TIM_DMA_COM** ((uint16_t)0x2000)
- #define **TIM_DMA_Trigger** ((uint16_t)0x4000)
- #define **IS_TIM_DMA_SOURCE**(SOURCE) ((((SOURCE) & (uint16_t)0x80FF) == 0x0000) && ((SOURCE) != 0x0000))
- #define **TIM_ExtTRGPSC_OFF** ((uint16_t)0x0000)
- #define **TIM_ExtTRGPSC_DIV2** ((uint16_t)0x1000)
- #define **TIM_ExtTRGPSC_DIV4** ((uint16_t)0x2000)
- #define **TIM_ExtTRGPSC_DIV8** ((uint16_t)0x3000)
- #define IS_TIM_EXT_PRESCALER(PRESCALER)
- #define **TIM_TS_ITR0** ((uint16_t)0x0000)
- #define **TIM_TS_ITR1** ((uint16_t)0x0010)
- #define **TIM_TS_ITR2** ((uint16_t)0x0020)
- #define **TIM_TS_ITR3** ((uint16_t)0x0030)
- #define **TIM_TS_TI1F_ED** ((uint16_t)0x0040)
- #define **TIM_TS_TI1FP1** ((uint16_t)0x0050)
- #define **TIM_TS_TI2FP2** ((uint16_t)0x0060)
- #define **TIM_TS_ETRF** ((uint16_t)0x0070)
- #define IS_TIM_TRIGGER_SELECTION(SELECTION)
- #define IS_TIM_INTERNAL_TRIGGER_SELECTION(SELECTION)
- #define **TIM_TIxExternalCLK1Source_TI1** ((uint16_t)0x0050)
- #define **TIM_TIxExternalCLK1Source_TI2** ((uint16_t)0x0060)
- #define **TIM_TIxExternalCLK1Source_TI1ED** ((uint16_t)0x0040)
- #define **TIM_ExtTRGPolarity_Inverted** ((uint16_t)0x8000)
- #define **TIM_ExtTRGPolarity_NonInverted** ((uint16_t)0x0000)
- #define IS_TIM_EXT_POLARITY(POLARITY)
- #define **TIM_PSCReloadMode_Update** ((uint16_t)0x0000)
- #define **TIM_PSCReloadMode_Immediate** ((uint16_t)0x0001)
- #define IS_TIM_PRESCALER_RELOAD(RELOAD)
- #define **TIM_ForcedAction_Active** ((uint16_t)0x0050)
- #define **TIM_ForcedAction_InActive** ((uint16_t)0x0040)
- #define IS_TIM_FORCED_ACTION(ACTION)
- #define **TIM_EncoderMode_TI1** ((uint16_t)0x0001)
- #define **TIM_EncoderMode_TI2** ((uint16_t)0x0002)
- #define **TIM_EncoderMode_TI12** ((uint16_t)0x0003)
- #define IS_TIM_ENCODER_MODE(MODE)
- #define **TIM_EventSource_Update** ((uint16_t)0x0001)
- #define **TIM_EventSource_CC1** ((uint16_t)0x0002)
- #define **TIM_EventSource_CC2** ((uint16_t)0x0004)
- #define **TIM_EventSource_CC3** ((uint16_t)0x0008)
- #define **TIM_EventSource_CC4** ((uint16_t)0x0010)
- #define **TIM_EventSource_COM** ((uint16_t)0x0020)
- #define **TIM_EventSource_Trigger** ((uint16_t)0x0040)
- #define **TIM_EventSource_Break** ((uint16_t)0x0080)
- #define **IS_TIM_EVENT_SOURCE**(SOURCE) ((((SOURCE) & (uint16_t)0xFF00) == 0x0000) && ((SOURCE) != 0x0000))
- #define TIM_UpdateSource_Global ((uint16_t)0x0000)
- #define TIM_UpdateSource_Regular ((uint16_t)0x0001)
- #define IS_TIM_UPDATE_SOURCE(SOURCE)
- #define **TIM_OCPreload_Enable** ((uint16_t)0x0008)
- #define **TIM_OCPreload_Disable** ((uint16_t)0x0000)
- #define IS_TIM_OCPRELOAD_STATE(STATE)
- #define **TIM_OCFast_Enable** ((uint16_t)0x0004)
- #define **TIM_OCFast_Disable** ((uint16_t)0x0000)
- #define IS_TIM_OCFAST_STATE(STATE)
- #define **TIM_OCClear_Enable** ((uint16_t)0x0080)

- #define **TIM_OCClear_Disable** ((uint16_t)0x0000)
- #define IS_TIM_OCCLEAR_STATE(STATE)
- #define **TIM_TRGOSource_Reset** ((uint16_t)0x0000)
- #define **TIM_TRGOSource_Enable** ((uint16_t)0x0010)
- #define **TIM_TRGOSource_Update** ((uint16_t)0x0020)
- #define **TIM_TRGOSource_OC1** ((uint16_t)0x0030)
- #define **TIM_TRGOSource_OC1Ref** ((uint16_t)0x0040)
- #define **TIM_TRGOSource_OC2Ref** ((uint16_t)0x0050)
- #define **TIM_TRGOSource_OC3Ref** ((uint16_t)0x0060)
- #define **TIM_TRGOSource_OC4Ref** ((uint16_t)0x0070)
- #define IS_TIM_TRGO_SOURCE(SOURCE)
- #define **TIM_SlaveMode_Reset** ((uint16_t)0x0004)
- #define **TIM_SlaveMode_Gated** ((uint16_t)0x0005)
- #define **TIM_SlaveMode_Trigger** ((uint16_t)0x0006)
- #define **TIM_SlaveMode_External1** ((uint16_t)0x0007)
- #define IS_TIM_SLAVE_MODE(MODE)
- #define **TIM_MasterSlaveMode_Enable** ((uint16_t)0x0080)
- #define **TIM_MasterSlaveMode_Disable** ((uint16_t)0x0000)
- #define IS_TIM_MSM_STATE(STATE)
- #define **TIM2_TIM8_TRGO** ((uint16_t)0x0000)
- #define **TIM2_ETH_PTP** ((uint16_t)0x0400)
- #define **TIM2_USBFS_SOF** ((uint16_t)0x0800)
- #define **TIM2_USBHS_SOF** ((uint16_t)0x0C00)
- #define **TIM5_GPIO** ((uint16_t)0x0000)
- #define **TIM5_LSI** ((uint16_t)0x0040)
- #define **TIM5_LSE** ((uint16_t)0x0080)
- #define **TIM5_RTC** ((uint16_t)0x00C0)
- #define **TIM11_GPIO** ((uint16_t)0x0000)
- #define **TIM11_HSE** ((uint16_t)0x0002)
- #define IS_TIM_REMAP(TIM_REMAP)
- #define **TIM_FLAG_Update** ((uint16_t)0x0001)
- #define **TIM_FLAG_CC1** ((uint16_t)0x0002)
- #define **TIM_FLAG_CC2** ((uint16_t)0x0004)
- #define **TIM_FLAG_CC3** ((uint16_t)0x0008)
- #define **TIM_FLAG_CC4** ((uint16_t)0x0010)
- #define **TIM_FLAG_COM** ((uint16_t)0x0020)
- #define **TIM_FLAG_Trigger** ((uint16_t)0x0040)
- #define **TIM_FLAG_Break** ((uint16_t)0x0080)
- #define **TIM_FLAG_CC1OF** ((uint16_t)0x0200)
- #define **TIM_FLAG_CC2OF** ((uint16_t)0x0400)
- #define **TIM_FLAG_CC3OF** ((uint16_t)0x0800)
- #define **TIM_FLAG_CC4OF** ((uint16_t)0x1000)
- #define IS_TIM_GET_FLAG(FLAG)
- #define **IS_TIM_IC_FILTER**(ICFILTER) ((ICFILTER) <= 0xF)
- #define **IS_TIM_EXT_FILTER**(EXTFILTER) ((EXTFILTER) <= 0xF)
- #define **TIM_DMABurstLength_1Byte** TIM_DMABurstLength_1Transfer
- #define **TIM_DMABurstLength_2Bytes** TIM_DMABurstLength_2Transfers
- #define **TIM_DMABurstLength_3Bytes** TIM_DMABurstLength_3Transfers
- #define **TIM_DMABurstLength_4Bytes** TIM_DMABurstLength_4Transfers
- #define **TIM_DMABurstLength_5Bytes** TIM_DMABurstLength_5Transfers
- #define **TIM_DMABurstLength_6Bytes** TIM_DMABurstLength_6Transfers
- #define **TIM_DMABurstLength_7Bytes** TIM_DMABurstLength_7Transfers
- #define **TIM_DMABurstLength_8Bytes** TIM_DMABurstLength_8Transfers
- #define **TIM_DMABurstLength_9Bytes** TIM_DMABurstLength_9Transfers
- #define **TIM_DMABurstLength_10Bytes** TIM_DMABurstLength_10Transfers

- #define **TIM_DMABurstLength_11Bytes** TIM_DMABurstLength_11Transfers
- #define **TIM_DMABurstLength_12Bytes** TIM_DMABurstLength_12Transfers
- #define **TIM_DMABurstLength_13Bytes** TIM_DMABurstLength_13Transfers
- #define **TIM_DMABurstLength_14Bytes** TIM_DMABurstLength_14Transfers
- #define **TIM_DMABurstLength_15Bytes** TIM_DMABurstLength_15Transfers
- #define **TIM_DMABurstLength_16Bytes** TIM_DMABurstLength_16Transfers
- #define **TIM_DMABurstLength_17Bytes** TIM_DMABurstLength_17Transfers
- #define **TIM_DMABurstLength_18Bytes** TIM_DMABurstLength_18Transfers

Functions

- void TIM_DeInit (TIM_TypeDef *TIMx)

  Deinitializes the TIMx peripheral registers to their default reset values.
- void TIM_TimeBaseInit (TIM_TypeDef *TIMx, TIM_TimeBaseInitTypeDef *TIM_TimeBaseInitStruct)

  Initializes the TIMx Time Base Unit peripheral according to the specified parameters in the TIM_TimeBaseInitStruct.
- void TIM_TimeBaseStructInit (TIM_TimeBaseInitTypeDef *TIM_TimeBaseInitStruct)

  Fills each TIM_TimeBaseInitStruct member with its default value.
- void TIM_PrescalerConfig (TIM_TypeDef *TIMx, uint16_t Prescaler, uint16_t TIM_PSCReloadMode)

  Configures the TIMx Prescaler.
- void TIM_CounterModeConfig (TIM_TypeDef *TIMx, uint16_t TIM_CounterMode)

  Specifies the TIMx Counter Mode to be used.
- void TIM_SetCounter (TIM_TypeDef *TIMx, uint32_t Counter)

  Sets the TIMx Counter Register value.
- void TIM_SetAutoreload (TIM_TypeDef *TIMx, uint32_t Autoreload)

  Sets the TIMx Autoreload Register value.
- uint32_t TIM_GetCounter (TIM_TypeDef *TIMx)

  Gets the TIMx Counter value.
- uint16_t TIM_GetPrescaler (TIM_TypeDef *TIMx)

  Gets the TIMx Prescaler value.
- void TIM_UpdateDisableConfig (TIM_TypeDef *TIMx, FunctionalState NewState)

  Enables or Disables the TIMx Update event.
- void TIM_UpdateRequestConfig (TIM_TypeDef *TIMx, uint16_t TIM_UpdateSource)

  Configures the TIMx Update Request Interrupt source.
- void TIM_ARRPreloadConfig (TIM_TypeDef *TIMx, FunctionalState NewState)

  Enables or disables TIMx peripheral Preload register on ARR.
- void TIM_SelectOnePulseMode (TIM_TypeDef *TIMx, uint16_t TIM_OPMode)

  Selects the TIMx's One Pulse Mode.
- void TIM_SetClockDivision (TIM_TypeDef *TIMx, uint16_t TIM_CKD)

  Sets the TIMx Clock Division value.
- void TIM_Cmd (TIM_TypeDef *TIMx, FunctionalState NewState)

  Enables or disables the specified TIM peripheral.
- void TIM_OC1Init (TIM_TypeDef *TIMx, TIM_OCInitTypeDef *TIM_OCInitStruct)

  Initializes the TIMx Channel1 according to the specified parameters in the TIM_OCInitStruct.
- void TIM_OC2Init (TIM_TypeDef *TIMx, TIM_OCInitTypeDef *TIM_OCInitStruct)

  Initializes the TIMx Channel2 according to the specified parameters in the TIM_OCInitStruct.
- void TIM_OC3Init (TIM_TypeDef *TIMx, TIM_OCInitTypeDef *TIM_OCInitStruct)

  Initializes the TIMx Channel3 according to the specified parameters in the TIM_OCInitStruct.
- void TIM_OC4Init (TIM_TypeDef *TIMx, TIM_OCInitTypeDef *TIM_OCInitStruct)

  Initializes the TIMx Channel4 according to the specified parameters in the TIM_OCInitStruct.
- void TIM_OCStructInit (TIM_OCInitTypeDef *TIM_OCInitStruct)

Fills each TIM_OCInitStruct member with its default value.

- void TIM_SelectOCxM (TIM_TypeDef *TIMx, uint16_t TIM_Channel, uint16_t TIM_OCMode)

  Selects the TIM Output Compare Mode.
- void TIM_SetCompare1 (TIM_TypeDef *TIMx, uint32_t Compare1)

  Sets the TIMx Capture Compare1 Register value.
- void TIM_SetCompare2 (TIM_TypeDef *TIMx, uint32_t Compare2)

  Sets the TIMx Capture Compare2 Register value.
- void TIM_SetCompare3 (TIM_TypeDef *TIMx, uint32_t Compare3)

  Sets the TIMx Capture Compare3 Register value.
- void TIM_SetCompare4 (TIM_TypeDef *TIMx, uint32_t Compare4)

  Sets the TIMx Capture Compare4 Register value.
- void TIM_ForcedOC1Config (TIM_TypeDef *TIMx, uint16_t TIM_ForcedAction)

  Forces the TIMx output 1 waveform to active or inactive level.
- void TIM_ForcedOC2Config (TIM_TypeDef *TIMx, uint16_t TIM_ForcedAction)

  Forces the TIMx output 2 waveform to active or inactive level.
- void TIM_ForcedOC3Config (TIM_TypeDef *TIMx, uint16_t TIM_ForcedAction)

  Forces the TIMx output 3 waveform to active or inactive level.
- void TIM_ForcedOC4Config (TIM_TypeDef *TIMx, uint16_t TIM_ForcedAction)

  Forces the TIMx output 4 waveform to active or inactive level.
- void TIM_OC1PreloadConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPreload)

  Enables or disables the TIMx peripheral Preload register on CCR1.
- void TIM_OC2PreloadConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPreload)

  Enables or disables the TIMx peripheral Preload register on CCR2.
- void TIM_OC3PreloadConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPreload)

  Enables or disables the TIMx peripheral Preload register on CCR3.
- void TIM_OC4PreloadConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPreload)

  Enables or disables the TIMx peripheral Preload register on CCR4.
- void TIM_OC1FastConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCFast)

  Configures the TIMx Output Compare 1 Fast feature.
- void TIM_OC2FastConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCFast)

  Configures the TIMx Output Compare 2 Fast feature.
- void TIM_OC3FastConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCFast)

  Configures the TIMx Output Compare 3 Fast feature.
- void TIM_OC4FastConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCFast)

  Configures the TIMx Output Compare 4 Fast feature.
- void TIM_ClearOC1Ref (TIM_TypeDef *TIMx, uint16_t TIM_OCClear)

  Clears or safeguards the OCREF1 signal on an external event.
- void TIM_ClearOC2Ref (TIM_TypeDef *TIMx, uint16_t TIM_OCClear)

  Clears or safeguards the OCREF2 signal on an external event.
- void TIM_ClearOC3Ref (TIM_TypeDef *TIMx, uint16_t TIM_OCClear)

  Clears or safeguards the OCREF3 signal on an external event.
- void TIM_ClearOC4Ref (TIM_TypeDef *TIMx, uint16_t TIM_OCClear)

  Clears or safeguards the OCREF4 signal on an external event.
- void TIM_OC1PolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPolarity)

  Configures the TIMx channel 1 polarity.
- void TIM_OC1NPolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCNPolarity)

  Configures the TIMx Channel 1N polarity.
- void TIM_OC2PolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPolarity)

  Configures the TIMx channel 2 polarity.
- void TIM_OC2NPolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCNPolarity)

  Configures the TIMx Channel 2N polarity.

- void TIM_OC3PolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPolarity)

  Configures the TIMx channel 3 polarity.
- void TIM_OC3NPolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCNPolarity)

  Configures the TIMx Channel 3N polarity.
- void TIM_OC4PolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPolarity)

  Configures the TIMx channel 4 polarity.
- void TIM_CCxCmd (TIM_TypeDef *TIMx, uint16_t TIM_Channel, uint16_t TIM_CCx)

  Enables or disables the TIM Capture Compare Channel x.
- void TIM_CCxNCmd (TIM_TypeDef *TIMx, uint16_t TIM_Channel, uint16_t TIM_CCxN)

  Enables or disables the TIM Capture Compare Channel xN.
- void TIM_ICInit (TIM_TypeDef *TIMx, TIM_ICInitTypeDef *TIM_ICInitStruct)

  Initializes the TIM peripheral according to the specified parameters in the TIM_ICInitStruct.
- void TIM_ICStructInit (TIM_ICInitTypeDef *TIM_ICInitStruct)

  Fills each TIM_ICInitStruct member with its default value.
- void TIM_PWMIConfig (TIM_TypeDef *TIMx, TIM_ICInitTypeDef *TIM_ICInitStruct)

  Configures the TIM peripheral according to the specified parameters in the TIM_ICInitStruct to measure an external PWM signal.
- uint32_t TIM_GetCapture1 (TIM_TypeDef *TIMx)

  Gets the TIMx Input Capture 1 value.
- uint32_t TIM_GetCapture2 (TIM_TypeDef *TIMx)

  Gets the TIMx Input Capture 2 value.
- uint32_t TIM_GetCapture3 (TIM_TypeDef *TIMx)

  Gets the TIMx Input Capture 3 value.
- uint32_t TIM_GetCapture4 (TIM_TypeDef *TIMx)

  Gets the TIMx Input Capture 4 value.
- void TIM_SetIC1Prescaler (TIM_TypeDef *TIMx, uint16_t TIM_ICPSC)

  Sets the TIMx Input Capture 1 prescaler.
- void TIM_SetIC2Prescaler (TIM_TypeDef *TIMx, uint16_t TIM_ICPSC)

  Sets the TIMx Input Capture 2 prescaler.
- void TIM_SetIC3Prescaler (TIM_TypeDef *TIMx, uint16_t TIM_ICPSC)

  Sets the TIMx Input Capture 3 prescaler.
- void TIM_SetIC4Prescaler (TIM_TypeDef *TIMx, uint16_t TIM_ICPSC)

  Sets the TIMx Input Capture 4 prescaler.
- void TIM_BDTRConfig (TIM_TypeDef *TIMx, TIM_BDTRInitTypeDef *TIM_BDTRInitStruct)

  Configures the Break feature, dead time, Lock level, OSSI/OSSR State and the AOE(automatic output enable).
- void TIM_BDTRStructInit (TIM_BDTRInitTypeDef *TIM_BDTRInitStruct)

  Fills each TIM_BDTRInitStruct member with its default value.
- void TIM_CtrlPWMOutputs (TIM_TypeDef *TIMx, FunctionalState NewState)

  Enables or disables the TIM peripheral Main Outputs.
- void TIM_SelectCOM (TIM_TypeDef *TIMx, FunctionalState NewState)

  Selects the TIM peripheral Commutation event.
- void TIM_CCPreloadControl (TIM_TypeDef *TIMx, FunctionalState NewState)

  Sets or Resets the TIM peripheral Capture Compare Preload Control bit.
- void TIM_ITConfig (TIM_TypeDef *TIMx, uint16_t TIM_IT, FunctionalState NewState)

  Enables or disables the specified TIM interrupts.
- void TIM_GenerateEvent (TIM_TypeDef *TIMx, uint16_t TIM_EventSource)

  Configures the TIMx event to be generate by software.
- FlagStatus TIM_GetFlagStatus (TIM_TypeDef *TIMx, uint16_t TIM_FLAG)

  Checks whether the specified TIM flag is set or not.
- void TIM_ClearFlag (TIM_TypeDef *TIMx, uint16_t TIM_FLAG)

  Clears the TIMx's pending flags.

- ITStatus TIM_GetITStatus (TIM_TypeDef *TIMx, uint16_t TIM_IT)

  Checks whether the TIM interrupt has occurred or not.
- void TIM_ClearITPendingBit (TIM_TypeDef *TIMx, uint16_t TIM_IT)

  Clears the TIMx's interrupt pending bits.
- void TIM_DMAConfig (TIM_TypeDef *TIMx, uint16_t TIM_DMABase, uint16_t TIM_DMABurstLength)

  Configures the TIMx's DMA interface.
- void TIM_DMACmd (TIM_TypeDef *TIMx, uint16_t TIM_DMASource, FunctionalState NewState)

  Enables or disables the TIMx's DMA Requests.
- void TIM_SelectCCDMA (TIM_TypeDef *TIMx, FunctionalState NewState)

  Selects the TIMx peripheral Capture Compare DMA source.
- void TIM_InternalClockConfig (TIM_TypeDef *TIMx)

  Configures the TIMx internal Clock.
- void TIM_ITRxExternalClockConfig (TIM_TypeDef *TIMx, uint16_t TIM_InputTriggerSource)

  Configures the TIMx Internal Trigger as External Clock.
- void TIM_TIxExternalClockConfig (TIM_TypeDef *TIMx, uint16_t TIM_TIxExternalCLKSource, uint16_t TIM_ICPolarity, uint16_t ICFilter)

  Configures the TIMx Trigger as External Clock.
- void TIM_ETRClockMode1Config (TIM_TypeDef *TIMx, uint16_t TIM_ExtTRGPrescaler, uint16_t TIM_ExtTRGPolarity, uint16_t ExtTRGFilter)

  Configures the External clock Mode1.
- void TIM_ETRClockMode2Config (TIM_TypeDef *TIMx, uint16_t TIM_ExtTRGPrescaler, uint16_t TIM_ExtTRGPolarity, uint16_t ExtTRGFilter)

  Configures the External clock Mode2.
- void TIM_SelectInputTrigger (TIM_TypeDef *TIMx, uint16_t TIM_InputTriggerSource)

  Selects the Input Trigger source.
- void TIM_SelectOutputTrigger (TIM_TypeDef *TIMx, uint16_t TIM_TRGOSource)

  Selects the TIMx Trigger Output Mode.
- void TIM_SelectSlaveMode (TIM_TypeDef *TIMx, uint16_t TIM_SlaveMode)

  Selects the TIMx Slave Mode.
- void TIM_SelectMasterSlaveMode (TIM_TypeDef *TIMx, uint16_t TIM_MasterSlaveMode)

  Sets or Resets the TIMx Master/Slave Mode.
- void TIM_ETRConfig (TIM_TypeDef *TIMx, uint16_t TIM_ExtTRGPrescaler, uint16_t TIM_ExtTRGPolarity, uint16_t ExtTRGFilter)

  Configures the TIMx External Trigger (ETR).
- void TIM_EncoderInterfaceConfig (TIM_TypeDef *TIMx, uint16_t TIM_EncoderMode, uint16_t TIM_IC1Polarity, uint16_t TIM_IC2Polarity)

  Configures the TIMx Encoder Interface.
- void TIM_SelectHallSensor (TIM_TypeDef *TIMx, FunctionalState NewState)

  Enables or disables the TIMx's Hall sensor interface.
- void TIM_RemapConfig (TIM_TypeDef *TIMx, uint16_t TIM_Remap)

  Configures the TIM2, TIM5 and TIM11 Remapping input capabilities.

7.32.1 Detailed Description

This file contains all the functions prototypes for the TIM firmware library.

Author

MCD Application Team

Version

V1.0.0

Date

30-September-2011

Attention

7.33 stm32f4xx_tim.h

Go to the documentation of this file.

```
00001
00023 /* Define to prevent recursive inclusion -------------------------------------*/
00024 #ifndef __STM32F4xx_TIM_H
00025 #define __STM32F4xx_TIM_H
00026
00027 #ifdef __cplusplus
00028  extern "C" {
00029 #endif
00030
00031 /* Includes ------------------------------------------------------------------*/
00032 #include "stm32f4xx.h"
00033
00042 /* Exported types ------------------------------------------------------------*/
00043
00049 typedef struct
00050 {
00051   uint16_t TIM_Prescaler;
00054   uint16_t TIM_CounterMode;
00057   uint32_t TIM_Period;
00061   uint16_t TIM_ClockDivision;
00064   uint8_t TIM_RepetitionCounter;
00072 } TIM_TimeBaseInitTypeDef;
00073
00078 typedef struct
00079 {
00080   uint16_t TIM_OCMode;
00083   uint16_t TIM_OutputState;
00086   uint16_t TIM_OutputNState;
00090   uint32_t TIM_Pulse;
00093   uint16_t TIM_OCPolarity;
00096   uint16_t TIM_OCNPolarity;
00100   uint16_t TIM_OCIdleState;
00104   uint16_t TIM_OCNIdleState;
00107 } TIM_OCInitTypeDef;
00108
00113 typedef struct
00114 {
00115
00116   uint16_t TIM_Channel;
00119   uint16_t TIM_ICPolarity;
```

```
00122   uint16_t TIM_ICSelection;
00125   uint16_t TIM_ICPrescaler;
00128   uint16_t TIM_ICFilter;
00130 } TIM_ICInitTypeDef;
00131
00137 typedef struct
00138 {
00139
00140   uint16_t TIM_OSSRState;
00143   uint16_t TIM_OSSIState;
00146   uint16_t TIM_LOCKLevel;
00149   uint16_t TIM_DeadTime;
00153   uint16_t TIM_Break;
00156   uint16_t TIM_BreakPolarity;
00159   uint16_t TIM_AutomaticOutput;
00161 } TIM_BDTRInitTypeDef;
00162
00163 /* Exported constants --------------------------------------------------------*/
00164
00169 #define IS_TIM_ALL_PERIPH(PERIPH) (((PERIPH) == TIM1) || \
00170                                    ((PERIPH) == TIM2) || \
00171                                    ((PERIPH) == TIM3) || \
00172                                    ((PERIPH) == TIM4) || \
00173                                    ((PERIPH) == TIM5) || \
00174                                    ((PERIPH) == TIM6) || \
00175                                    ((PERIPH) == TIM7) || \
00176                                    ((PERIPH) == TIM8) || \
00177                                    ((PERIPH) == TIM9) || \
00178                                    ((PERIPH) == TIM10) || \
00179                                    ((PERIPH) == TIM11) || \
00180                                    ((PERIPH) == TIM12) || \
00181                                    ((PERIPH) == TIM13) || \
00182                                    ((PERIPH) == TIM14)))
00183 /* LIST1: TIM1, TIM2, TIM3, TIM4, TIM5, TIM8, TIM9, TIM10, TIM11, TIM12, TIM13 and TIM14 */

00184 #define IS_TIM_LIST1_PERIPH(PERIPH) (((PERIPH) == TIM1) || \
00185                                      ((PERIPH) == TIM2) || \
00186                                      ((PERIPH) == TIM3) || \
00187                                      ((PERIPH) == TIM4) || \
00188                                      ((PERIPH) == TIM5) || \
00189                                      ((PERIPH) == TIM8) || \
00190                                      ((PERIPH) == TIM9) || \
00191                                      ((PERIPH) == TIM10) || \
00192                                      ((PERIPH) == TIM11) || \
00193                                      ((PERIPH) == TIM12) || \
00194                                      ((PERIPH) == TIM13) || \
00195                                      ((PERIPH) == TIM14))
00196
00197 /* LIST2: TIM1, TIM2, TIM3, TIM4, TIM5, TIM8, TIM9 and TIM12 */
00198 #define IS_TIM_LIST2_PERIPH(PERIPH) (((PERIPH) == TIM1) || \
00199                                      ((PERIPH) == TIM2) || \
00200                                      ((PERIPH) == TIM3) || \
00201                                      ((PERIPH) == TIM4) || \
00202                                      ((PERIPH) == TIM5) || \
00203                                      ((PERIPH) == TIM8) || \
00204                                      ((PERIPH) == TIM9) || \
00205                                      ((PERIPH) == TIM12))
00206 /* LIST3: TIM1, TIM2, TIM3, TIM4, TIM5 and TIM8 */
00207 #define IS_TIM_LIST3_PERIPH(PERIPH) (((PERIPH) == TIM1) || \
00208                                      ((PERIPH) == TIM2) || \
00209                                      ((PERIPH) == TIM3) || \
00210                                      ((PERIPH) == TIM4) || \
00211                                      ((PERIPH) == TIM5) || \
00212                                      ((PERIPH) == TIM8))
00213 /* LIST4: TIM1 and TIM8 */
00214 #define IS_TIM_LIST4_PERIPH(PERIPH) (((PERIPH) == TIM1) || \
00215                                      ((PERIPH) == TIM8))
00216 /* LIST5: TIM1, TIM2, TIM3, TIM4, TIM5, TIM6, TIM7 and TIM8 */
00217 #define IS_TIM_LIST5_PERIPH(PERIPH) (((PERIPH) == TIM1) || \
00218                                      ((PERIPH) == TIM2) || \
00219                                      ((PERIPH) == TIM3) || \
00220                                      ((PERIPH) == TIM4) || \
00221                                      ((PERIPH) == TIM5) || \
00222                                      ((PERIPH) == TIM6) || \
00223                                      ((PERIPH) == TIM7) || \
00224                                      ((PERIPH) == TIM8))
00225 /* LIST6: TIM2, TIM5 and TIM11 */
00226 #define IS_TIM_LIST6_PERIPH(TIMx)(((TIMx) == TIM2) || \
00227                                   ((TIMx) == TIM5) || \
00228                                   ((TIMx) == TIM11))
00229
00234 #define TIM_OCMode_Timing                  ((uint16_t)0x0000)
00235 #define TIM_OCMode_Active                  ((uint16_t)0x0010)
00236 #define TIM_OCMode_Inactive                ((uint16_t)0x0020)
00237 #define TIM_OCMode_Toggle                  ((uint16_t)0x0030)
00238 #define TIM_OCMode_PWM1                    ((uint16_t)0x0060)
00239 #define TIM_OCMode_PWM2                    ((uint16_t)0x0070)
```

```
00240 #define IS_TIM_OC_MODE(MODE) (((MODE) == TIM_OCMode_Timing) || \
00241                               ((MODE) == TIM_OCMode_Active) || \
00242                               ((MODE) == TIM_OCMode_Inactive) || \
00243                               ((MODE) == TIM_OCMode_Toggle)|| \
00244                               ((MODE) == TIM_OCMode_PWM1) || \
00245                               ((MODE) == TIM_OCMode_PWM2))
00246 #define IS_TIM_OCM(MODE) (((MODE) == TIM_OCMode_Timing) || \
00247                           ((MODE) == TIM_OCMode_Active) || \
00248                           ((MODE) == TIM_OCMode_Inactive) || \
00249                           ((MODE) == TIM_OCMode_Toggle)|| \
00250                           ((MODE) == TIM_OCMode_PWM1) || \
00251                           ((MODE) == TIM_OCMode_PWM2) ||    \
00252                           ((MODE) == TIM_ForcedAction_Active) || \
00253                           ((MODE) == TIM_ForcedAction_InActive))
00262 #define TIM_OPMode_Single                  ((uint16_t)0x0008)
00263 #define TIM_OPMode_Repetitive              ((uint16_t)0x0000)
00264 #define IS_TIM_OPM_MODE(MODE) (((MODE) == TIM_OPMode_Single) || \
00265                                ((MODE) == TIM_OPMode_Repetitive))
00274 #define TIM_Channel_1                      ((uint16_t)0x0000)
00275 #define TIM_Channel_2                      ((uint16_t)0x0004)
00276 #define TIM_Channel_3                      ((uint16_t)0x0008)
00277 #define TIM_Channel_4                      ((uint16_t)0x000C)
00278
00279 #define IS_TIM_CHANNEL(CHANNEL) (((CHANNEL) == TIM_Channel_1) || \
00280                                  ((CHANNEL) == TIM_Channel_2) || \
00281                                  ((CHANNEL) == TIM_Channel_3) || \
00282                                  ((CHANNEL) == TIM_Channel_4))
00283
00284 #define IS_TIM_PWMI_CHANNEL(CHANNEL) (((CHANNEL) == TIM_Channel_1) || \
00285                                       ((CHANNEL) == TIM_Channel_2))
00286 #define IS_TIM_COMPLEMENTARY_CHANNEL(CHANNEL) (((CHANNEL) == TIM_Channel_1) || \
00287                                                ((CHANNEL) == TIM_Channel_2) || \
00288                                                ((CHANNEL) == TIM_Channel_3))
00297 #define TIM_CKD_DIV1                       ((uint16_t)0x0000)
00298 #define TIM_CKD_DIV2                       ((uint16_t)0x0100)
00299 #define TIM_CKD_DIV4                       ((uint16_t)0x0200)
00300 #define IS_TIM_CKD_DIV(DIV) (((DIV) == TIM_CKD_DIV1) || \
00301                              ((DIV) == TIM_CKD_DIV2) || \
00302                              ((DIV) == TIM_CKD_DIV4))
00311 #define TIM_CounterMode_Up                 ((uint16_t)0x0000)
00312 #define TIM_CounterMode_Down               ((uint16_t)0x0010)
00313 #define TIM_CounterMode_CenterAligned1     ((uint16_t)0x0020)
00314 #define TIM_CounterMode_CenterAligned2     ((uint16_t)0x0040)
00315 #define TIM_CounterMode_CenterAligned3     ((uint16_t)0x0060)
00316 #define IS_TIM_COUNTER_MODE(MODE) (((MODE) == TIM_CounterMode_Up) ||  \
00317                                    ((MODE) == TIM_CounterMode_Down) || \
00318                                    ((MODE) == TIM_CounterMode_CenterAligned1) || \
00319                                    ((MODE) == TIM_CounterMode_CenterAligned2) || \
00320                                    ((MODE) == TIM_CounterMode_CenterAligned3))
00329 #define TIM_OCPolarity_High                ((uint16_t)0x0000)
00330 #define TIM_OCPolarity_Low                 ((uint16_t)0x0002)
00331 #define IS_TIM_OC_POLARITY(POLARITY) (((POLARITY) == TIM_OCPolarity_High) || \
00332                                       ((POLARITY) == TIM_OCPolarity_Low))
00341 #define TIM_OCNPolarity_High               ((uint16_t)0x0000)
00342 #define TIM_OCNPolarity_Low                ((uint16_t)0x0008)
00343 #define IS_TIM_OCN_POLARITY(POLARITY) (((POLARITY) == TIM_OCNPolarity_High) || \
00344                                        ((POLARITY) == TIM_OCNPolarity_Low))
00353 #define TIM_OutputState_Disable            ((uint16_t)0x0000)
00354 #define TIM_OutputState_Enable             ((uint16_t)0x0001)
00355 #define IS_TIM_OUTPUT_STATE(STATE) (((STATE) == TIM_OutputState_Disable) || \
00356                                     ((STATE) == TIM_OutputState_Enable))
00365 #define TIM_OutputNState_Disable           ((uint16_t)0x0000)
00366 #define TIM_OutputNState_Enable            ((uint16_t)0x0004)
00367 #define IS_TIM_OUTPUTN_STATE(STATE) (((STATE) == TIM_OutputNState_Disable) || \
00368                                      ((STATE) == TIM_OutputNState_Enable))
00377 #define TIM_CCx_Enable                      ((uint16_t)0x0001)
00378 #define TIM_CCx_Disable                     ((uint16_t)0x0000)
00379 #define IS_TIM_CCX(CCX) (((CCX) == TIM_CCx_Enable) || \
00380                          ((CCX) == TIM_CCx_Disable))
00389 #define TIM_CCxN_Enable                     ((uint16_t)0x0004)
00390 #define TIM_CCxN_Disable                    ((uint16_t)0x0000)
00391 #define IS_TIM_CCXN(CCXN) (((CCXN) == TIM_CCxN_Enable) || \
00392                            ((CCXN) == TIM_CCxN_Disable))
00401 #define TIM_Break_Enable                   ((uint16_t)0x1000)
00402 #define TIM_Break_Disable                  ((uint16_t)0x0000)
00403 #define IS_TIM_BREAK_STATE(STATE) (((STATE) == TIM_Break_Enable) || \
00404                                    ((STATE) == TIM_Break_Disable))
00413 #define TIM_BreakPolarity_Low              ((uint16_t)0x0000)
00414 #define TIM_BreakPolarity_High             ((uint16_t)0x2000)
00415 #define IS_TIM_BREAK_POLARITY(POLARITY) (((POLARITY) == TIM_BreakPolarity_Low) || \
00416                                          ((POLARITY) == TIM_BreakPolarity_High))
00425 #define TIM_AutomaticOutput_Enable         ((uint16_t)0x4000)
00426 #define TIM_AutomaticOutput_Disable        ((uint16_t)0x0000)
00427 #define IS_TIM_AUTOMATIC_OUTPUT_STATE(STATE) (((STATE) == TIM_AutomaticOutput_Enable) || \
00428                                               ((STATE) == TIM_AutomaticOutput_Disable))
00437 #define TIM_LOCKLevel_OFF                  ((uint16_t)0x0000)
00438 #define TIM_LOCKLevel_1                    ((uint16_t)0x0100)
```

```
00439 #define TIM_LOCKLevel_2                    ((uint16_t)0x0200)
00440 #define TIM_LOCKLevel_3                    ((uint16_t)0x0300)
00441 #define IS_TIM_LOCK_LEVEL(LEVEL) (((LEVEL) == TIM_LOCKLevel_OFF) || \
00442                                   ((LEVEL) == TIM_LOCKLevel_1) || \
00443                                   ((LEVEL) == TIM_LOCKLevel_2) || \
00444                                   ((LEVEL) == TIM_LOCKLevel_3))
00453 #define TIM_OSSIState_Enable               ((uint16_t)0x0400)
00454 #define TIM_OSSIState_Disable              ((uint16_t)0x0000)
00455 #define IS_TIM_OSSI_STATE(STATE) (((STATE) == TIM_OSSIState_Enable) || \
00456                                   ((STATE) == TIM_OSSIState_Disable))
00465 #define TIM_OSSRState_Enable               ((uint16_t)0x0800)
00466 #define TIM_OSSRState_Disable              ((uint16_t)0x0000)
00467 #define IS_TIM_OSSR_STATE(STATE) (((STATE) == TIM_OSSRState_Enable) || \
00468                                   ((STATE) == TIM_OSSRState_Disable))
00477 #define TIM_OCIdleState_Set                ((uint16_t)0x0100)
00478 #define TIM_OCIdleState_Reset              ((uint16_t)0x0000)
00479 #define IS_TIM_OCIDLE_STATE(STATE) (((STATE) == TIM_OCIdleState_Set) || \
00480                                     ((STATE) == TIM_OCIdleState_Reset))
00489 #define TIM_OCNIdleState_Set               ((uint16_t)0x0200)
00490 #define TIM_OCNIdleState_Reset             ((uint16_t)0x0000)
00491 #define IS_TIM_OCNIDLE_STATE(STATE) (((STATE) == TIM_OCNIdleState_Set) || \
00492                                      ((STATE) == TIM_OCNIdleState_Reset))
00501 #define  TIM_ICPolarity_Rising             ((uint16_t)0x0000)
00502 #define  TIM_ICPolarity_Falling            ((uint16_t)0x0002)
00503 #define  TIM_ICPolarity_BothEdge           ((uint16_t)0x000A)
00504 #define IS_TIM_IC_POLARITY(POLARITY) (((POLARITY) == TIM_ICPolarity_Rising) || \
00505                                       ((POLARITY) == TIM_ICPolarity_Falling)|| \
00506                                       ((POLARITY) == TIM_ICPolarity_BothEdge))
00515 #define TIM_ICSelection_DirectTI           ((uint16_t)0x0001)
00517 #define TIM_ICSelection_IndirectTI         ((uint16_t)0x0002)
00519 #define TIM_ICSelection_TRC                ((uint16_t)0x0003)
00520 #define IS_TIM_IC_SELECTION(SELECTION) (((SELECTION) == TIM_ICSelection_DirectTI) || \
00521                                         ((SELECTION) == TIM_ICSelection_IndirectTI) || \
00522                                         ((SELECTION) == TIM_ICSelection_TRC))
00531 #define TIM_ICPSC_DIV1                     ((uint16_t)0x0000)
00532 #define TIM_ICPSC_DIV2                     ((uint16_t)0x0004)
00533 #define TIM_ICPSC_DIV4                     ((uint16_t)0x0008)
00534 #define TIM_ICPSC_DIV8                     ((uint16_t)0x000C)
00535 #define IS_TIM_IC_PRESCALER(PRESCALER) (((PRESCALER) == TIM_ICPSC_DIV1) || \
00536                                         ((PRESCALER) == TIM_ICPSC_DIV2) || \
00537                                         ((PRESCALER) == TIM_ICPSC_DIV4) || \
00538                                         ((PRESCALER) == TIM_ICPSC_DIV8))
00547 #define TIM_IT_Update                      ((uint16_t)0x0001)
00548 #define TIM_IT_CC1                         ((uint16_t)0x0002)
00549 #define TIM_IT_CC2                         ((uint16_t)0x0004)
00550 #define TIM_IT_CC3                         ((uint16_t)0x0008)
00551 #define TIM_IT_CC4                         ((uint16_t)0x0010)
00552 #define TIM_IT_COM                         ((uint16_t)0x0020)
00553 #define TIM_IT_Trigger                     ((uint16_t)0x0040)
00554 #define TIM_IT_Break                       ((uint16_t)0x0080)
00555 #define IS_TIM_IT(IT) ((((IT) & (uint16_t)0xFF00) == 0x0000) && ((IT) != 0x0000))
00556
00557 #define IS_TIM_GET_IT(IT) (((IT) == TIM_IT_Update) || \
00558                            ((IT) == TIM_IT_CC1) || \
00559                            ((IT) == TIM_IT_CC2) || \
00560                            ((IT) == TIM_IT_CC3) || \
00561                            ((IT) == TIM_IT_CC4) || \
00562                            ((IT) == TIM_IT_COM) || \
00563                            ((IT) == TIM_IT_Trigger) || \
00564                            ((IT) == TIM_IT_Break))
00573 #define TIM_DMABase_CR1                    ((uint16_t)0x0000)
00574 #define TIM_DMABase_CR2                    ((uint16_t)0x0001)
00575 #define TIM_DMABase_SMCR                   ((uint16_t)0x0002)
00576 #define TIM_DMABase_DIER                   ((uint16_t)0x0003)
00577 #define TIM_DMABase_SR                     ((uint16_t)0x0004)
00578 #define TIM_DMABase_EGR                    ((uint16_t)0x0005)
00579 #define TIM_DMABase_CCMR1                  ((uint16_t)0x0006)
00580 #define TIM_DMABase_CCMR2                  ((uint16_t)0x0007)
00581 #define TIM_DMABase_CCER                   ((uint16_t)0x0008)
00582 #define TIM_DMABase_CNT                    ((uint16_t)0x0009)
00583 #define TIM_DMABase_PSC                    ((uint16_t)0x000A)
00584 #define TIM_DMABase_ARR                    ((uint16_t)0x000B)
00585 #define TIM_DMABase_RCR                    ((uint16_t)0x000C)
00586 #define TIM_DMABase_CCR1                   ((uint16_t)0x000D)
00587 #define TIM_DMABase_CCR2                   ((uint16_t)0x000E)
00588 #define TIM_DMABase_CCR3                   ((uint16_t)0x000F)
00589 #define TIM_DMABase_CCR4                   ((uint16_t)0x0010)
00590 #define TIM_DMABase_BDTR                   ((uint16_t)0x0011)
00591 #define TIM_DMABase_DCR                    ((uint16_t)0x0012)
00592 #define TIM_DMABase_OR                     ((uint16_t)0x0013)
00593 #define IS_TIM_DMA_BASE(BASE) (((BASE) == TIM_DMABase_CR1) || \
00594                                ((BASE) == TIM_DMABase_CR2) || \
00595                                ((BASE) == TIM_DMABase_SMCR) || \
00596                                ((BASE) == TIM_DMABase_DIER) || \
00597                                ((BASE) == TIM_DMABase_SR) || \
00598                                ((BASE) == TIM_DMABase_EGR) || \
00599                                ((BASE) == TIM_DMABase_CCMR1) || \
```

```
00600                                ((BASE) == TIM_DMABase_CCMR2) || \
00601                                ((BASE) == TIM_DMABase_CCER) || \
00602                                ((BASE) == TIM_DMABase_CNT) || \
00603                                ((BASE) == TIM_DMABase_PSC) || \
00604                                ((BASE) == TIM_DMABase_ARR) || \
00605                                ((BASE) == TIM_DMABase_RCR) || \
00606                                ((BASE) == TIM_DMABase_CCR1) || \
00607                                ((BASE) == TIM_DMABase_CCR2) || \
00608                                ((BASE) == TIM_DMABase_CCR3) || \
00609                                ((BASE) == TIM_DMABase_CCR4) || \
00610                                ((BASE) == TIM_DMABase_BDTR) || \
00611                                ((BASE) == TIM_DMABase_DCR) || \
00612                                ((BASE) == TIM_DMABase_OR))
00621 #define TIM_DMABurstLength_1Transfer           ((uint16_t)0x0000)
00622 #define TIM_DMABurstLength_2Transfers          ((uint16_t)0x0100)
00623 #define TIM_DMABurstLength_3Transfers          ((uint16_t)0x0200)
00624 #define TIM_DMABurstLength_4Transfers          ((uint16_t)0x0300)
00625 #define TIM_DMABurstLength_5Transfers          ((uint16_t)0x0400)
00626 #define TIM_DMABurstLength_6Transfers          ((uint16_t)0x0500)
00627 #define TIM_DMABurstLength_7Transfers          ((uint16_t)0x0600)
00628 #define TIM_DMABurstLength_8Transfers          ((uint16_t)0x0700)
00629 #define TIM_DMABurstLength_9Transfers          ((uint16_t)0x0800)
00630 #define TIM_DMABurstLength_10Transfers         ((uint16_t)0x0900)
00631 #define TIM_DMABurstLength_11Transfers         ((uint16_t)0x0A00)
00632 #define TIM_DMABurstLength_12Transfers         ((uint16_t)0x0B00)
00633 #define TIM_DMABurstLength_13Transfers         ((uint16_t)0x0C00)
00634 #define TIM_DMABurstLength_14Transfers         ((uint16_t)0x0D00)
00635 #define TIM_DMABurstLength_15Transfers         ((uint16_t)0x0E00)
00636 #define TIM_DMABurstLength_16Transfers         ((uint16_t)0x0F00)
00637 #define TIM_DMABurstLength_17Transfers         ((uint16_t)0x1000)
00638 #define TIM_DMABurstLength_18Transfers         ((uint16_t)0x1100)
00639 #define IS_TIM_DMA_LENGTH(LENGTH) (((LENGTH) == TIM_DMABurstLength_1Transfer) || \
00640                                    ((LENGTH) == TIM_DMABurstLength_2Transfers) || \
00641                                    ((LENGTH) == TIM_DMABurstLength_3Transfers) || \
00642                                    ((LENGTH) == TIM_DMABurstLength_4Transfers) || \
00643                                    ((LENGTH) == TIM_DMABurstLength_5Transfers) || \
00644                                    ((LENGTH) == TIM_DMABurstLength_6Transfers) || \
00645                                    ((LENGTH) == TIM_DMABurstLength_7Transfers) || \
00646                                    ((LENGTH) == TIM_DMABurstLength_8Transfers) || \
00647                                    ((LENGTH) == TIM_DMABurstLength_9Transfers) || \
00648                                    ((LENGTH) == TIM_DMABurstLength_10Transfers) || \
00649                                    ((LENGTH) == TIM_DMABurstLength_11Transfers) || \
00650                                    ((LENGTH) == TIM_DMABurstLength_12Transfers) || \
00651                                    ((LENGTH) == TIM_DMABurstLength_13Transfers) || \
00652                                    ((LENGTH) == TIM_DMABurstLength_14Transfers) || \
00653                                    ((LENGTH) == TIM_DMABurstLength_15Transfers) || \
00654                                    ((LENGTH) == TIM_DMABurstLength_16Transfers) || \
00655                                    ((LENGTH) == TIM_DMABurstLength_17Transfers) || \
00656                                    ((LENGTH) == TIM_DMABurstLength_18Transfers))
00665 #define TIM_DMA_Update                  ((uint16_t)0x0100)
00666 #define TIM_DMA_CC1                     ((uint16_t)0x0200)
00667 #define TIM_DMA_CC2                     ((uint16_t)0x0400)
00668 #define TIM_DMA_CC3                     ((uint16_t)0x0800)
00669 #define TIM_DMA_CC4                     ((uint16_t)0x1000)
00670 #define TIM_DMA_COM                     ((uint16_t)0x2000)
00671 #define TIM_DMA_Trigger                 ((uint16_t)0x4000)
00672 #define IS_TIM_DMA_SOURCE(SOURCE) ((((SOURCE) & (uint16_t)0x80FF) == 0x0000) && ((SOURCE) != 0x0000))
00673
00682 #define TIM_ExtTRGPSC_OFF               ((uint16_t)0x0000)
00683 #define TIM_ExtTRGPSC_DIV2              ((uint16_t)0x1000)
00684 #define TIM_ExtTRGPSC_DIV4              ((uint16_t)0x2000)
00685 #define TIM_ExtTRGPSC_DIV8              ((uint16_t)0x3000)
00686 #define IS_TIM_EXT_PRESCALER(PRESCALER) (((PRESCALER) == TIM_ExtTRGPSC_OFF) || \
00687                                          ((PRESCALER) == TIM_ExtTRGPSC_DIV2) || \
00688                                          ((PRESCALER) == TIM_ExtTRGPSC_DIV4) || \
00689                                          ((PRESCALER) == TIM_ExtTRGPSC_DIV8))
00698 #define TIM_TS_ITR0                     ((uint16_t)0x0000)
00699 #define TIM_TS_ITR1                     ((uint16_t)0x0010)
00700 #define TIM_TS_ITR2                     ((uint16_t)0x0020)
00701 #define TIM_TS_ITR3                     ((uint16_t)0x0030)
00702 #define TIM_TS_TI1F_ED                  ((uint16_t)0x0040)
00703 #define TIM_TS_TI1FP1                   ((uint16_t)0x0050)
00704 #define TIM_TS_TI2FP2                   ((uint16_t)0x0060)
00705 #define TIM_TS_ETRF                     ((uint16_t)0x0070)
00706 #define IS_TIM_TRIGGER_SELECTION(SELECTION) (((SELECTION) == TIM_TS_ITR0) || \
00707                                             ((SELECTION) == TIM_TS_ITR1) || \
00708                                             ((SELECTION) == TIM_TS_ITR2) || \
00709                                             ((SELECTION) == TIM_TS_ITR3) || \
00710                                             ((SELECTION) == TIM_TS_TI1F_ED) || \
00711                                             ((SELECTION) == TIM_TS_TI1FP1) || \
00712                                             ((SELECTION) == TIM_TS_TI2FP2) || \
00713                                             ((SELECTION) == TIM_TS_ETRF))
00714 #define IS_TIM_INTERNAL_TRIGGER_SELECTION(SELECTION) (((SELECTION) == TIM_TS_ITR0) || \
00715                                                       ((SELECTION) == TIM_TS_ITR1) || \
00716                                                       ((SELECTION) == TIM_TS_ITR2) || \
00717                                                       ((SELECTION) == TIM_TS_ITR3))
00726 #define TIM_TIxExternalCLK1Source_TI1      ((uint16_t)0x0050)
```

```
00727 #define TIM_TIxExternalCLK1Source_TI2      ((uint16_t)0x0060)
00728 #define TIM_TIxExternalCLK1Source_TI1ED    ((uint16_t)0x0040)
00729
00737 #define TIM_ExtTRGPolarity_Inverted        ((uint16_t)0x8000)
00738 #define TIM_ExtTRGPolarity_NonInverted     ((uint16_t)0x0000)
00739 #define IS_TIM_EXT_POLARITY(POLARITY) (((POLARITY) == TIM_ExtTRGPolarity_Inverted) || \
00740                                        ((POLARITY) == TIM_ExtTRGPolarity_NonInverted))
00749 #define TIM_PSCReloadMode_Update           ((uint16_t)0x0000)
00750 #define TIM_PSCReloadMode_Immediate        ((uint16_t)0x0001)
00751 #define IS_TIM_PRESCALER_RELOAD(RELOAD) (((RELOAD) == TIM_PSCReloadMode_Update) || \
00752                                          ((RELOAD) == TIM_PSCReloadMode_Immediate))
00761 #define TIM_ForcedAction_Active            ((uint16_t)0x0050)
00762 #define TIM_ForcedAction_InActive          ((uint16_t)0x0040)
00763 #define IS_TIM_FORCED_ACTION(ACTION) (((ACTION) == TIM_ForcedAction_Active) || \
00764                                       ((ACTION) == TIM_ForcedAction_InActive))
00773 #define TIM_EncoderMode_TI1                ((uint16_t)0x0001)
00774 #define TIM_EncoderMode_TI2                ((uint16_t)0x0002)
00775 #define TIM_EncoderMode_TI12               ((uint16_t)0x0003)
00776 #define IS_TIM_ENCODER_MODE(MODE) (((MODE) == TIM_EncoderMode_TI1) || \
00777                                    ((MODE) == TIM_EncoderMode_TI2) || \
00778                                    ((MODE) == TIM_EncoderMode_TI12))
00788 #define TIM_EventSource_Update             ((uint16_t)0x0001)
00789 #define TIM_EventSource_CC1                ((uint16_t)0x0002)
00790 #define TIM_EventSource_CC2                ((uint16_t)0x0004)
00791 #define TIM_EventSource_CC3                ((uint16_t)0x0008)
00792 #define TIM_EventSource_CC4                ((uint16_t)0x0010)
00793 #define TIM_EventSource_COM                ((uint16_t)0x0020)
00794 #define TIM_EventSource_Trigger            ((uint16_t)0x0040)
00795 #define TIM_EventSource_Break              ((uint16_t)0x0080)
00796 #define IS_TIM_EVENT_SOURCE(SOURCE) ((((SOURCE) & (uint16_t)0xFF00) == 0x0000) && ((SOURCE) !=
      0x0000))
00797
00806 #define TIM_UpdateSource_Global            ((uint16_t)0x0000)
00809 #define TIM_UpdateSource_Regular           ((uint16_t)0x0001)
00810 #define IS_TIM_UPDATE_SOURCE(SOURCE) (((SOURCE) == TIM_UpdateSource_Global) || \
00811                                       ((SOURCE) == TIM_UpdateSource_Regular))
00820 #define TIM_OCPreload_Enable               ((uint16_t)0x0008)
00821 #define TIM_OCPreload_Disable              ((uint16_t)0x0000)
00822 #define IS_TIM_OCPRELOAD_STATE(STATE) (((STATE) == TIM_OCPreload_Enable) || \
00823                                        ((STATE) == TIM_OCPreload_Disable))
00832 #define TIM_OCFast_Enable                  ((uint16_t)0x0004)
00833 #define TIM_OCFast_Disable                 ((uint16_t)0x0000)
00834 #define IS_TIM_OCFAST_STATE(STATE) (((STATE) == TIM_OCFast_Enable) || \
00835                                     ((STATE) == TIM_OCFast_Disable))
00836
00845 #define TIM_OCClear_Enable                 ((uint16_t)0x0080)
00846 #define TIM_OCClear_Disable                ((uint16_t)0x0000)
00847 #define IS_TIM_OCCLEAR_STATE(STATE) (((STATE) == TIM_OCClear_Enable) || \
00848                                      ((STATE) == TIM_OCClear_Disable))
00857 #define TIM_TRGOSource_Reset               ((uint16_t)0x0000)
00858 #define TIM_TRGOSource_Enable              ((uint16_t)0x0010)
00859 #define TIM_TRGOSource_Update              ((uint16_t)0x0020)
00860 #define TIM_TRGOSource_OC1                 ((uint16_t)0x0030)
00861 #define TIM_TRGOSource_OC1Ref              ((uint16_t)0x0040)
00862 #define TIM_TRGOSource_OC2Ref              ((uint16_t)0x0050)
00863 #define TIM_TRGOSource_OC3Ref              ((uint16_t)0x0060)
00864 #define TIM_TRGOSource_OC4Ref              ((uint16_t)0x0070)
00865 #define IS_TIM_TRGO_SOURCE(SOURCE) (((SOURCE) == TIM_TRGOSource_Reset) || \
00866                                     ((SOURCE) == TIM_TRGOSource_Enable) || \
00867                                     ((SOURCE) == TIM_TRGOSource_Update) || \
00868                                     ((SOURCE) == TIM_TRGOSource_OC1) || \
00869                                     ((SOURCE) == TIM_TRGOSource_OC1Ref) || \
00870                                     ((SOURCE) == TIM_TRGOSource_OC2Ref) || \
00871                                     ((SOURCE) == TIM_TRGOSource_OC3Ref) || \
00872                                     ((SOURCE) == TIM_TRGOSource_OC4Ref))
00881 #define TIM_SlaveMode_Reset                ((uint16_t)0x0004)
00882 #define TIM_SlaveMode_Gated                ((uint16_t)0x0005)
00883 #define TIM_SlaveMode_Trigger              ((uint16_t)0x0006)
00884 #define TIM_SlaveMode_External1            ((uint16_t)0x0007)
00885 #define IS_TIM_SLAVE_MODE(MODE) (((MODE) == TIM_SlaveMode_Reset) || \
00886                                  ((MODE) == TIM_SlaveMode_Gated) || \
00887                                  ((MODE) == TIM_SlaveMode_Trigger) || \
00888                                  ((MODE) == TIM_SlaveMode_External1))
00897 #define TIM_MasterSlaveMode_Enable         ((uint16_t)0x0080)
00898 #define TIM_MasterSlaveMode_Disable        ((uint16_t)0x0000)
00899 #define IS_TIM_MSM_STATE(STATE) (((STATE) == TIM_MasterSlaveMode_Enable) || \
00900                                  ((STATE) == TIM_MasterSlaveMode_Disable))
00908 #define TIM2_TIM8_TRGO                     ((uint16_t)0x0000)
00909 #define TIM2_ETH_PTP                       ((uint16_t)0x0400)
00910 #define TIM2_USBFS_SOF                     ((uint16_t)0x0800)
00911 #define TIM2_USBHS_SOF                     ((uint16_t)0x0C00)
00912
00913 #define TIM5_GPIO                          ((uint16_t)0x0000)
00914 #define TIM5_LSI                           ((uint16_t)0x0040)
00915 #define TIM5_LSE                           ((uint16_t)0x0080)
00916 #define TIM5_RTC                           ((uint16_t)0x00C0)
00917
```

```
00918 #define TIM11_GPIO                         ((uint16_t)0x0000)
00919 #define TIM11_HSE                          ((uint16_t)0x0002)
00920
00921 #define IS_TIM_REMAP(TIM_REMAP)  (((TIM_REMAP) == TIM2_TIM8_TRGO)||\
00922                                   ((TIM_REMAP) == TIM2_ETH_PTP)||\
00923                                   ((TIM_REMAP) == TIM2_USBFS_SOF)||\
00924                                   ((TIM_REMAP) == TIM2_USBHS_SOF)||\
00925                                   ((TIM_REMAP) == TIM5_GPIO)||\
00926                                   ((TIM_REMAP) == TIM5_LSI)||\
00927                                   ((TIM_REMAP) == TIM5_LSE)||\
00928                                   ((TIM_REMAP) == TIM5_RTC)||\
00929                                   ((TIM_REMAP) == TIM11_GPIO)||\
00930                                   ((TIM_REMAP) == TIM11_HSE))
00931
00939 #define TIM_FLAG_Update                    ((uint16_t)0x0001)
00940 #define TIM_FLAG_CC1                       ((uint16_t)0x0002)
00941 #define TIM_FLAG_CC2                       ((uint16_t)0x0004)
00942 #define TIM_FLAG_CC3                       ((uint16_t)0x0008)
00943 #define TIM_FLAG_CC4                       ((uint16_t)0x0010)
00944 #define TIM_FLAG_COM                       ((uint16_t)0x0020)
00945 #define TIM_FLAG_Trigger                   ((uint16_t)0x0040)
00946 #define TIM_FLAG_Break                     ((uint16_t)0x0080)
00947 #define TIM_FLAG_CC1OF                     ((uint16_t)0x0200)
00948 #define TIM_FLAG_CC2OF                     ((uint16_t)0x0400)
00949 #define TIM_FLAG_CC3OF                     ((uint16_t)0x0800)
00950 #define TIM_FLAG_CC4OF                     ((uint16_t)0x1000)
00951 #define IS_TIM_GET_FLAG(FLAG) (((FLAG) == TIM_FLAG_Update) || \
00952                                ((FLAG) == TIM_FLAG_CC1) || \
00953                                ((FLAG) == TIM_FLAG_CC2) || \
00954                                ((FLAG) == TIM_FLAG_CC3) || \
00955                                ((FLAG) == TIM_FLAG_CC4) || \
00956                                ((FLAG) == TIM_FLAG_COM) || \
00957                                ((FLAG) == TIM_FLAG_Trigger) || \
00958                                ((FLAG) == TIM_FLAG_Break) || \
00959                                ((FLAG) == TIM_FLAG_CC1OF) || \
00960                                ((FLAG) == TIM_FLAG_CC2OF) || \
00961                                ((FLAG) == TIM_FLAG_CC3OF) || \
00962                                ((FLAG) == TIM_FLAG_CC4OF))
00963
00972 #define IS_TIM_IC_FILTER(ICFILTER) ((ICFILTER) <= 0xF)
00981 #define IS_TIM_EXT_FILTER(EXTFILTER) ((EXTFILTER) <= 0xF)
00990 #define TIM_DMABurstLength_1Byte           TIM_DMABurstLength_1Transfer
00991 #define TIM_DMABurstLength_2Bytes          TIM_DMABurstLength_2Transfers
00992 #define TIM_DMABurstLength_3Bytes          TIM_DMABurstLength_3Transfers
00993 #define TIM_DMABurstLength_4Bytes          TIM_DMABurstLength_4Transfers
00994 #define TIM_DMABurstLength_5Bytes          TIM_DMABurstLength_5Transfers
00995 #define TIM_DMABurstLength_6Bytes          TIM_DMABurstLength_6Transfers
00996 #define TIM_DMABurstLength_7Bytes          TIM_DMABurstLength_7Transfers
00997 #define TIM_DMABurstLength_8Bytes          TIM_DMABurstLength_8Transfers
00998 #define TIM_DMABurstLength_9Bytes          TIM_DMABurstLength_9Transfers
00999 #define TIM_DMABurstLength_10Bytes         TIM_DMABurstLength_10Transfers
01000 #define TIM_DMABurstLength_11Bytes         TIM_DMABurstLength_11Transfers
01001 #define TIM_DMABurstLength_12Bytes         TIM_DMABurstLength_12Transfers
01002 #define TIM_DMABurstLength_13Bytes         TIM_DMABurstLength_13Transfers
01003 #define TIM_DMABurstLength_14Bytes         TIM_DMABurstLength_14Transfers
01004 #define TIM_DMABurstLength_15Bytes         TIM_DMABurstLength_15Transfers
01005 #define TIM_DMABurstLength_16Bytes         TIM_DMABurstLength_16Transfers
01006 #define TIM_DMABurstLength_17Bytes         TIM_DMABurstLength_17Transfers
01007 #define TIM_DMABurstLength_18Bytes         TIM_DMABurstLength_18Transfers
01016 /* Exported macro ------------------------------------------------------------*/
01017 /* Exported functions --------------------------------------------------------*/
01018
01019 /* TimeBase management ******************************************************/
01020 void TIM_DeInit(TIM_TypeDef* TIMx);
01021 void TIM_TimeBaseInit(TIM_TypeDef* TIMx, TIM_TimeBaseInitTypeDef* TIM_TimeBaseInitStruct);
01022 void TIM_TimeBaseStructInit(TIM_TimeBaseInitTypeDef* TIM_TimeBaseInitStruct);
01023 void TIM_PrescalerConfig(TIM_TypeDef* TIMx, uint16_t Prescaler, uint16_t TIM_PSCReloadMode);
01024 void TIM_CounterModeConfig(TIM_TypeDef* TIMx, uint16_t TIM_CounterMode);
01025 void TIM_SetCounter(TIM_TypeDef* TIMx, uint32_t Counter);
01026 void TIM_SetAutoreload(TIM_TypeDef* TIMx, uint32_t Autoreload);
01027 uint32_t TIM_GetCounter(TIM_TypeDef* TIMx);
01028 uint16_t TIM_GetPrescaler(TIM_TypeDef* TIMx);
01029 void TIM_UpdateDisableConfig(TIM_TypeDef* TIMx, FunctionalState NewState);
01030 void TIM_UpdateRequestConfig(TIM_TypeDef* TIMx, uint16_t TIM_UpdateSource);
01031 void TIM_ARRPreloadConfig(TIM_TypeDef* TIMx, FunctionalState NewState);
01032 void TIM_SelectOnePulseMode(TIM_TypeDef* TIMx, uint16_t TIM_OPMode);
01033 void TIM_SetClockDivision(TIM_TypeDef* TIMx, uint16_t TIM_CKD);
01034 void TIM_Cmd(TIM_TypeDef* TIMx, FunctionalState NewState);
01035
01036 /* Output Compare management ************************************************/
01037 void TIM_OC1Init(TIM_TypeDef* TIMx, TIM_OCInitTypeDef* TIM_OCInitStruct);
01038 void TIM_OC2Init(TIM_TypeDef* TIMx, TIM_OCInitTypeDef* TIM_OCInitStruct);
01039 void TIM_OC3Init(TIM_TypeDef* TIMx, TIM_OCInitTypeDef* TIM_OCInitStruct);
01040 void TIM_OC4Init(TIM_TypeDef* TIMx, TIM_OCInitTypeDef* TIM_OCInitStruct);
01041 void TIM_OCStructInit(TIM_OCInitTypeDef* TIM_OCInitStruct);
01042 void TIM_SelectOCxM(TIM_TypeDef* TIMx, uint16_t TIM_Channel, uint16_t TIM_OCMode);
01043 void TIM_SetCompare1(TIM_TypeDef* TIMx, uint32_t Compare1);
```

```
01044 void TIM_SetCompare2(TIM_TypeDef* TIMx, uint32_t Compare2);
01045 void TIM_SetCompare3(TIM_TypeDef* TIMx, uint32_t Compare3);
01046 void TIM_SetCompare4(TIM_TypeDef* TIMx, uint32_t Compare4);
01047 void TIM_ForcedOC1Config(TIM_TypeDef* TIMx, uint16_t TIM_ForcedAction);
01048 void TIM_ForcedOC2Config(TIM_TypeDef* TIMx, uint16_t TIM_ForcedAction);
01049 void TIM_ForcedOC3Config(TIM_TypeDef* TIMx, uint16_t TIM_ForcedAction);
01050 void TIM_ForcedOC4Config(TIM_TypeDef* TIMx, uint16_t TIM_ForcedAction);
01051 void TIM_OC1PreloadConfig(TIM_TypeDef* TIMx, uint16_t TIM_OCPreload);
01052 void TIM_OC2PreloadConfig(TIM_TypeDef* TIMx, uint16_t TIM_OCPreload);
01053 void TIM_OC3PreloadConfig(TIM_TypeDef* TIMx, uint16_t TIM_OCPreload);
01054 void TIM_OC4PreloadConfig(TIM_TypeDef* TIMx, uint16_t TIM_OCPreload);
01055 void TIM_OC1FastConfig(TIM_TypeDef* TIMx, uint16_t TIM_OCFast);
01056 void TIM_OC2FastConfig(TIM_TypeDef* TIMx, uint16_t TIM_OCFast);
01057 void TIM_OC3FastConfig(TIM_TypeDef* TIMx, uint16_t TIM_OCFast);
01058 void TIM_OC4FastConfig(TIM_TypeDef* TIMx, uint16_t TIM_OCFast);
01059 void TIM_ClearOC1Ref(TIM_TypeDef* TIMx, uint16_t TIM_OCClear);
01060 void TIM_ClearOC2Ref(TIM_TypeDef* TIMx, uint16_t TIM_OCClear);
01061 void TIM_ClearOC3Ref(TIM_TypeDef* TIMx, uint16_t TIM_OCClear);
01062 void TIM_ClearOC4Ref(TIM_TypeDef* TIMx, uint16_t TIM_OCClear);
01063 void TIM_OC1PolarityConfig(TIM_TypeDef* TIMx, uint16_t TIM_OCPolarity);
01064 void TIM_OC1NPolarityConfig(TIM_TypeDef* TIMx, uint16_t TIM_OCNPolarity);
01065 void TIM_OC2PolarityConfig(TIM_TypeDef* TIMx, uint16_t TIM_OCPolarity);
01066 void TIM_OC2NPolarityConfig(TIM_TypeDef* TIMx, uint16_t TIM_OCNPolarity);
01067 void TIM_OC3PolarityConfig(TIM_TypeDef* TIMx, uint16_t TIM_OCPolarity);
01068 void TIM_OC3NPolarityConfig(TIM_TypeDef* TIMx, uint16_t TIM_OCNPolarity);
01069 void TIM_OC4PolarityConfig(TIM_TypeDef* TIMx, uint16_t TIM_OCPolarity);
01070 void TIM_CCxCmd(TIM_TypeDef* TIMx, uint16_t TIM_Channel, uint16_t TIM_CCx);
01071 void TIM_CCxNCmd(TIM_TypeDef* TIMx, uint16_t TIM_Channel, uint16_t TIM_CCxN);
01072
01073 /* Input Capture management ******************************************************/
01074 void TIM_ICInit(TIM_TypeDef* TIMx, TIM_ICInitTypeDef* TIM_ICInitStruct);
01075 void TIM_ICStructInit(TIM_ICInitTypeDef* TIM_ICInitStruct);
01076 void TIM_PWMIConfig(TIM_TypeDef* TIMx, TIM_ICInitTypeDef* TIM_ICInitStruct);
01077 uint32_t TIM_GetCapture1(TIM_TypeDef* TIMx);
01078 uint32_t TIM_GetCapture2(TIM_TypeDef* TIMx);
01079 uint32_t TIM_GetCapture3(TIM_TypeDef* TIMx);
01080 uint32_t TIM_GetCapture4(TIM_TypeDef* TIMx);
01081 void TIM_SetIC1Prescaler(TIM_TypeDef* TIMx, uint16_t TIM_ICPSC);
01082 void TIM_SetIC2Prescaler(TIM_TypeDef* TIMx, uint16_t TIM_ICPSC);
01083 void TIM_SetIC3Prescaler(TIM_TypeDef* TIMx, uint16_t TIM_ICPSC);
01084 void TIM_SetIC4Prescaler(TIM_TypeDef* TIMx, uint16_t TIM_ICPSC);
01085
01086 /* Advanced-control timers (TIM1 and TIM8) specific features *****************/
01087 void TIM_BDTRConfig(TIM_TypeDef* TIMx, TIM_BDTRInitTypeDef *TIM_BDTRInitStruct);
01088 void TIM_BDTRStructInit(TIM_BDTRInitTypeDef* TIM_BDTRInitStruct);
01089 void TIM_CtrlPWMOutputs(TIM_TypeDef* TIMx, FunctionalState NewState);
01090 void TIM_SelectCOM(TIM_TypeDef* TIMx, FunctionalState NewState);
01091 void TIM_CCPreloadControl(TIM_TypeDef* TIMx, FunctionalState NewState);
01092
01093 /* Interrupts, DMA and flags management ******************************************/
01094 void TIM_ITConfig(TIM_TypeDef* TIMx, uint16_t TIM_IT, FunctionalState NewState);
01095 void TIM_GenerateEvent(TIM_TypeDef* TIMx, uint16_t TIM_EventSource);
01096 FlagStatus TIM_GetFlagStatus(TIM_TypeDef* TIMx, uint16_t TIM_FLAG);
01097 void TIM_ClearFlag(TIM_TypeDef* TIMx, uint16_t TIM_FLAG);
01098 ITStatus TIM_GetITStatus(TIM_TypeDef* TIMx, uint16_t TIM_IT);
01099 void TIM_ClearITPendingBit(TIM_TypeDef* TIMx, uint16_t TIM_IT);
01100 void TIM_DMAConfig(TIM_TypeDef* TIMx, uint16_t TIM_DMABase, uint16_t TIM_DMABurstLength);
01101 void TIM_DMACmd(TIM_TypeDef* TIMx, uint16_t TIM_DMASource, FunctionalState NewState);
01102 void TIM_SelectCCDMA(TIM_TypeDef* TIMx, FunctionalState NewState);
01103
01104 /* Clocks management *************************************************************/
01105 void TIM_InternalClockConfig(TIM_TypeDef* TIMx);
01106 void TIM_ITRxExternalClockConfig(TIM_TypeDef* TIMx, uint16_t TIM_InputTriggerSource);
01107 void TIM_TIxExternalClockConfig(TIM_TypeDef* TIMx, uint16_t TIM_TIxExternalCLKSource,
01108                                 uint16_t TIM_ICPolarity, uint16_t ICFilter);
01109 void TIM_ETRClockMode1Config(TIM_TypeDef* TIMx, uint16_t TIM_ExtTRGPrescaler, uint16_t
      TIM_ExtTRGPolarity,
01110                              uint16_t ExtTRGFilter);
01111 void TIM_ETRClockMode2Config(TIM_TypeDef* TIMx, uint16_t TIM_ExtTRGPrescaler,
01112                              uint16_t TIM_ExtTRGPolarity, uint16_t ExtTRGFilter);
01113
01114 /* Synchronization management ****************************************************/
01115 void TIM_SelectInputTrigger(TIM_TypeDef* TIMx, uint16_t TIM_InputTriggerSource);
01116 void TIM_SelectOutputTrigger(TIM_TypeDef* TIMx, uint16_t TIM_TRGOSource);
01117 void TIM_SelectSlaveMode(TIM_TypeDef* TIMx, uint16_t TIM_SlaveMode);
01118 void TIM_SelectMasterSlaveMode(TIM_TypeDef* TIMx, uint16_t TIM_MasterSlaveMode);
01119 void TIM_ETRConfig(TIM_TypeDef* TIMx, uint16_t TIM_ExtTRGPrescaler, uint16_t TIM_ExtTRGPolarity,
01120                    uint16_t ExtTRGFilter);
01121
01122 /* Specific interface management *************************************************/
01123 void TIM_EncoderInterfaceConfig(TIM_TypeDef* TIMx, uint16_t TIM_EncoderMode,
01124                                 uint16_t TIM_IC1Polarity, uint16_t TIM_IC2Polarity);
01125 void TIM_SelectHallSensor(TIM_TypeDef* TIMx, FunctionalState NewState);
01126
01127 /* Specific remapping management *************************************************/
01128 void TIM_RemapConfig(TIM_TypeDef* TIMx, uint16_t TIM_Remap);
01129
```

```
01130 #ifdef __cplusplus
01131 }
01132 #endif
01133
01134 #endif /*__STM32F4xx_TIM_H */
01135
01144 /****************** (C) COPYRIGHT 2011 STMicroelectronics *****END OF FILE****/
```

7.34 CUBE_IDE/VGA/Core/Inc/system_stm32f4xx.h File Reference

CMSIS Cortex-M4 Device System Source File for STM32F4xx devices.

Functions

- void SystemInit (void)

  Setup the microcontroller system Initialize the Embedded Flash Interface, the PLL and update the SystemFrequency variable.
- void SystemCoreClockUpdate (void)

  Update SystemCoreClock variable according to Clock Register Values. The SystemCoreClock variable contains the core clock (HCLK), it can be used by the user application to setup the SysTick timer or configure other parameters.

Variables

- uint32_t SystemCoreClock

7.34.1 Detailed Description

CMSIS Cortex-M4 Device System Source File for STM32F4xx devices.

Author

MCD Application Team

Version

V1.0.0

Date

30-September-2011

Attention

7.35 system_stm32f4xx.h

Go to the documentation of this file.

```
00001
00033 #ifndef __SYSTEM_STM32F4XX_H
00034 #define __SYSTEM_STM32F4XX_H
00035
00036 #ifdef __cplusplus
00037  extern "C" {
00038 #endif
00039
00053 extern uint32_t SystemCoreClock;
00080 extern void SystemInit(void);
00081 extern void SystemCoreClockUpdate(void);
00086 #ifdef __cplusplus
00087 }
00088 #endif
00089
00090 #endif /*__SYSTEM_STM32F4XX_H */
00091
00099 /****************** (C) COPYRIGHT 2011 STMicroelectronics *****END OF FILE****/
```

7.36 CUBE_IDE/VGA/Core/Inc/UART_communication.h File Reference

: This file contains the functions which are used in the source file 'UART_communication.c'.

```
#include <stdio.h>
#include <string.h>
#include "main.h"
#include "error.h"
```

Data Structures

- struct UART_Communication

Macros

- #define **WAIT_FOR_DATA** 4000000
- #define **STORAGE** 100
- #define **NO_DATA** 69

Typedefs

- typedef struct UART_Communication **UART**
- typedef struct UART_Communication * **PUART**

Functions

- void UART2_config (void)

  Function to enable all the necessary registers and operations to use UART.
- void UART_sendChar (uint8_t c)

  Function that sends char data to the connected terminal.
- void UART_sendString (char *string)

  Function that sends a string to the connected terminal.
- uint8_t UART_getChar (void)

  Function that receives data from the connected terminal.
- UART UART_receiver (void)

  Function that read a single line and stores the data.
- void UART_errorHandling (int err)

  Function that handles the occuring errors to the terminal.

7.36.1 Detailed Description

: This file contains the functions which are used in the source file 'UART_communication.c'.

Authors

: Skip Wijtman

Date

: 3-5-2023

Version

: 1.1

7.36.2 Function Documentation

7.36.2.1 UART2_config()

```
void UART2_config (
            void  )
```

Function to enable all the necessary registers and operations to use UART.

Parameters

Nothing	

Returns

Nothing

7.36.2.2 UART_errorHandling()

```
void UART_errorHandling (
            int err )
```

Function that handles the occuring errors to the terminal.

Parameters

err	error that is supposed to be shown

Returns

nothing

7.36.2.3 UART_getChar()

```
uint8_t UART_getChar (
            void  )
```

Function that receives data from the connected terminal.

Parameters

Nothing	

Returns

Received data

7.36.2.4 UART_receiver()

```
UART UART_receiver (
            void  )
```

Function that read a single line and stores the data.

Parameters

Nothing	

Returns

struct variable

7.36.2.5 UART_sendChar()

```
void UART_sendChar (
            uint8_t c )
```

Function that sends char data to the connected terminal.

Parameters

Char	data do be send

Returns

Nothing

7.36.2.6 UART_sendString()

```
void UART_sendString (
            char * string )
```

Function that sends a string to the connected terminal.

Parameters

String	do be send

Returns

Nothing

7.37 UART_communication.h

Go to the documentation of this file.

```
00001 /*****************************************************/
00012 #ifndef INC_UART_COMMUNICATION_H_
```

```
00013 #define INC_UART_COMMUNICATION_H_
00014
00015 //#include <library-header>
00016 #include <stdio.h>
00017 #include <string.h>
00018
00019 //#include other "user-header"
00020 #include "main.h"
00021 #include "error.h"
00022
00023 //#define-statements
00024 #define WAIT_FOR_DATA   4000000
00025 #define STORAGE         100
00026 #define NO_DATA         69
00027
00028 //external vars
00029
00030
00031 //struct definition
00032 typedef struct UART_Communication
00033 {
00034     uint8_t receive[STORAGE];
00035     uint8_t buffer[STORAGE];
00036     int     err_code;
00037 }UART, *PUART;
00038
00039 //user functies
00040 void UART2_config(void);
00041 void UART_sendChar(uint8_t c);
00042 void UART_sendString(char *string);
00043 uint8_t UART_getChar(void);
00044 UART UART_receiver(void);
00045 void UART_errorHandling(int err);
00046
00047 #endif /* INC_UART_COMMUNICATION_H_ */
```

7.38 CUBE_IDE/VGA/Core/Src/API_functions.c File Reference

All API functions are created in this file.

```
#include "API_functions.h"
#include "bitmap_arrows.h"
#include "bitmap_smileys.h"
```

Functions

- int API_draw_line (int x_1, int y_1, int x_2, int y_2, int color, int weight)

  This function gives the user the ability to draw lines on a VGA screen.
- int API_clearscreen (int color)

  This function gives the user the ability to clear the screen to a certain color.
- int API_draw_rectangle (int x_1, int y_1, int width, int height, int color, int filled)

  This function gives the user the ability to draw a rectangle on a VGA screen.
- int API_draw_bitmap (int bm_nr, int x_lup, int y_lup)

  This function gives the user the ability to put a bitmap on the VGA screen.

7.38.1 Detailed Description

All API functions are created in this file.

Authors

Naomi Born

Date

17-05-2023

Version

1.0

7.38.2 Function Documentation

7.38.2.1 API_clearscreen()

```
int API_clearscreen (
            int color )
```

This function gives the user the ability to clear the screen to a certain color.

Parameters

color	Color of the screen

Returns

Error code if error occurs

7.38.2.2 API_draw_bitmap()

```
int API_draw_bitmap (
            int bm_nr,
            int x_lup,
            int y_lup )
```

This function gives the user the ability to put a bitmap on the VGA screen.

Parameters

x_lup	X coordinate of x left up
y_lup	Y coordinate of y left up
bm_nr	Number of the bitmap from 1 to 10

Returns

Error code if error or no error occurs

7.38.2.3 API_draw_line()

```
int API_draw_line (
            int x_1,
            int y_1,
            int x_2,
            int y_2,
            int color,
            int weight )
```

This function gives the user the ability to draw lines on a VGA screen.

Parameters

x_1	Starting point coördinate of x
x_2	Ending point coördinate of x
y_1	Starting point coördinate of y
y_2	Ending point coördinate of y
color	Color of the line
weight	Width of the line

Returns

Error code if error or no error occurs

7.38.2.4 API_draw_rectangle()

```
int API_draw_rectangle (
            int x_1,
            int y_1,
            int width,
            int height,
            int color,
            int filled )
```

This function gives the user the ability to draw a rectangle on a VGA screen.

Parameters

x_1	Starting point coördinate of x (upper left)
y_1	Starting point coördinate of y (upper left)
width	Width of the rectangle max. 320
height	Height of the rectangle max. 240
color	Color of the rectangle/borders
filled	1 is filled rectangle 0 is just borders

Returns

Error code if error occurs

7.39 CUBE_IDE/VGA/Core/Src/logic_layer.c File Reference

Here commands will be processed and executed.

```
#include "logic_layer.h"
```

Functions

- int parse_cmd (UART data)

  Function that parses the command of the received script.
- int draw_options (char cmd, UART data)

  Function that determines with the command which function should be used, Here the script also get parsed further and decodes ASCII to useful data.
- PARSE parse_data (PARSE parsing, UART data, int LEN, int var_counter, int num_checker, int let_checker, int num_counter)

  This function is used for parsing data and converts ASCII to decimals.
- int number_converter (char ASCII)

  Function that converts ASCII numbers to decimals.
- PARSE color_assign (UART data, int i, PARSE parsing)

  Function that reads received script text and when possible converts this to the correspondiong color.

Variables

- char **cmd** [MAX_CMD_LEN] = {0}
- char compare_cmd [4][MAX_CMD_LEN]
- char compare_col [16][MAX_COL_LEN]

7.39.1 Detailed Description

Here commands will be processed and executed.

Authors

Skip Wijtman

Date

24-5-2023

Version

1.2

7.39.2 Function Documentation

7.39.2.1 color_assign()

```
PARSE color_assign (
            UART data,
            int i,
            PARSE parsing )
```

Function that reads received script text and when possible converts this to the correspondiong color.

Parameters

data	is a struct variable with the received script
i	is the variable for the loop iterator
parsing	is a struct variable that stores all needed data for functions

Returns

parsing is a struct variable with data info which are retrieved from the script

7.39.2.2 draw_options()

```
int draw_options (
            char cmd,
            UART data )
```

Function that determines with the command which function should be used, Here the script also get parsed further and decodes ASCII to useful data.

Here a switch-case is used to determine the received command, Cases 0 to 4 are numbered as followed:

1. lijn
2. rechthoek
3. clearscherm
4. tekst
5. bitmap

Parameters

cmd	is a interger of the found command
data	is a struct variable with the received script

Returns

Error code

7.39.2.3 number_converter()

```
int number_converter (
            char ASCII )
```

Function that converts ASCII numbers to decimals.

Parameters

ASCII	is character of a ASCII number

Returns

decimal number

7.39.2.4 parse_cmd()

```
int parse_cmd (
            UART data )
```

Function that parses the command of the received script.

Parameters

data	is a struct variable with the received script

Returns

Error code or the current index of found

7.39.2.5 parse_data()

```
PARSE parse_data (
            PARSE parsing,
            UART data,
            int LEN,
            int var_counter,
            int num_checker,
```

```
            int let_checker,
            int num_counter )
```

This function is used for parsing data and converts ASCII to decimals.

Parameters

parsing	is a struct variable that stores all needed data for functions
data	is a struct variable with the received script
LEN	is the length of the command + the comma to skip these is the parser
var_counter	is an index of an array to store the data in from the script
num_checker	is a variable to confirm a ASCII number is found and ensures that no junk color values are stored in the data array
let_checker	is a variable to confirm a letter is found and ensures that no junk number values are stored in the data array
num_counter	is a variable which counts the amount of ASCII numbers between two commas

Returns

Error code

7.39.3 Variable Documentation

7.39.3.1 compare_cmd

```
char compare_cmd[4][MAX_CMD_LEN]
```

Initial value:

```
= { {"lijn"},
                                        {"rechthoek"},
                                        {"clearscherm"},
                                        {"bitmap"}}
```

7.39.3.2 compare_col

```
char compare_col[16][MAX_COL_LEN]
```

Initial value:

```
= { {"zwart"},
                                        {"blauw"},
                                        {"lichtblauw"},
                                        {"groen"},
                                        {"lichtgroen"},
                                        {"cyaan"},
                                        {"lichtcyaan"},
                                        {"rood"},
                                        {"lichtrood"},
                                        {"magenta"},
                                        {"lichtmagenta"},
                                        {"bruin"},
                                        {"geel"},
                                        {"grijs"},
                                        {"roze"},
                                        {"wit"}}
```

7.40 CUBE_IDE/VGA/Core/Src/main.c File Reference

The file that gets executed and is used for operating a screen via VGA with external functions.

```
#include <math.h>
#include "main.h"
#include "stm32_ub_vga_screen.h"
#include "UART_communication.h"
#include "API_functions.h"
#include "logic_layer.h"
#include "error.h"
```

Functions

- int main (void)

  Program start.

Variables

- uint8_t **data**
- char **save** [100]
- char **save2** [100]
- uint8_t **arr** [100]
- uint8_t **i** =0
- int **x** = 0
- unsigned char **j**
- int **val** = 0

7.40.1 Detailed Description

The file that gets executed and is used for operating a screen via VGA with external functions.

Extra infromation:

1. CPU : STM32F4
2. IDE : CooCox CoIDE 1.7.x
3. Module : CMSIS_BOOT, M4_CMSIS_CORE
4. Function : VGA_core DMA LIB 320x240, 8bit color

Authors

UB, J.F van der Bent, Skip Wijtman

Date

3-5-2023

Version

1.0 (Updates with every SWD branch)

7.40.2 Function Documentation

7.40.2.1 main()

```
int main (
            void  )
```

Program start.

Parameters

Nothing	

Returns

integer val

7.41 CUBE_IDE/VGA/Core/Src/misc.c File Reference

This file provides all the miscellaneous firmware functions (add-on to CMSIS functions).

```
#include "misc.h"
#include "stm32f4xx_conf.h"
```

Macros

- #define **AIRCR_VECTKEY_MASK** ((uint32_t)0x05FA0000)

Functions

- void NVIC_PriorityGroupConfig (uint32_t NVIC_PriorityGroup)

  Configures the priority grouping: pre-emption priority and subpriority.
- void NVIC_Init (NVIC_InitTypeDef *NVIC_InitStruct)

  Initializes the NVIC peripheral according to the specified parameters in the NVIC_InitStruct.
- void NVIC_SetVectorTable (uint32_t NVIC_VectTab, uint32_t Offset)

  Sets the vector table location and Offset.
- void NVIC_SystemLPConfig (uint8_t LowPowerMode, FunctionalState NewState)

  Selects the condition for the system to enter low power mode.
- void SysTick_CLKSourceConfig (uint32_t SysTick_CLKSource)

  Configures the SysTick clock source.

7.41.1 Detailed Description

This file provides all the miscellaneous firmware functions (add-on to CMSIS functions).

Author

MCD Application Team

Version

V1.0.0

Date

30-September-2011

```
*
*          ===================================================================
*                        How to configure Interrupts using driver
*          ===================================================================
*
*            This section provide functions allowing to configure the NVIC interrupts (IRQ).
*            The Cortex-M4 exceptions are managed by CMSIS functions.
*
*            1. Configure the NVIC Priority Grouping using NVIC_PriorityGroupConfig()
*                function according to the following table.

*  The table below gives the allowed values of the pre-emption priority and subpriority according
*  to the Priority Grouping configuration performed by NVIC_PriorityGroupConfig function
*    ========================================================================================================
*      NVIC_PriorityGroup   | NVIC_IRQChannelPreemptionPriority | NVIC_IRQChannelSubPriority  |       Descript
*    ========================================================================================================
*     NVIC_PriorityGroup_0  |                0                 |            0-15             | 0 bits for pre
*                           |                                  |                             | 4 bits for sub
*    --------------------------------------------------------------------------------------------------------
*     NVIC_PriorityGroup_1  |                0-1               |            0-7              | 1 bits for pre
*                           |                                  |                             | 3 bits for sub
*    --------------------------------------------------------------------------------------------------------
*     NVIC_PriorityGroup_2  |                0-3               |            0-3              | 2 bits for pre
*                           |                                  |                             | 2 bits for sub
*    --------------------------------------------------------------------------------------------------------
*     NVIC_PriorityGroup_3  |                0-7               |            0-1              | 3 bits for pre
*                           |                                  |                             | 1 bits for sub
*    --------------------------------------------------------------------------------------------------------
*     NVIC_PriorityGroup_4  |                0-15              |            0                | 4 bits for pre
*                           |                                  |                             | 0 bits for sub
*    ========================================================================================================
*
*            2. Enable and Configure the priority of the selected IRQ Channels using NVIC_Init()
*
* @note  When the NVIC_PriorityGroup_0 is selected, IRQ pre-emption is no more possible.
*        The pending IRQ priority will be managed only by the subpriority.
*
* @note  IRQ priority order (sorted by highest to lowest priority):
*         - Lowest pre-emption priority
*         - Lowest subpriority
*         - Lowest hardware priority (IRQ number)
*
*
```

Attention

THE PRESENT FIRMWARE WHICH IS FOR GUIDANCE ONLY AIMS AT PROVIDING CUSTOMERS WITH CODING INFORMATION REGARDING THEIR PRODUCTS IN ORDER FOR THEM TO SAVE TIME. AS A RESULT, STMICROELECTRONICS SHALL NOT BE HELD LIABLE FOR ANY DIRECT, INDIRECT OR CONSEQUENTIAL DAMAGES WITH RESPECT TO ANY CLAIMS ARISING FROM THE CONTENT OF SUCH FIRMWARE AND/OR THE USE MADE BY CUSTOMERS OF THE CODING INFORMATION CONTAINED HEREIN IN CONNECTION WITH THEIR PRODUCTS.

7.42 CUBE_IDE/VGA/Core/Src/stm32f4xx_dma.c File Reference

This file provides firmware functions to manage the following functionalities of the Direct Memory Access controller (DMA):

```
#include "stm32f4xx_dma.h"
#include "stm32f4xx_rcc.h"
#include "stm32f4xx_conf.h"
```

Macros

- #define TRANSFER_IT_ENABLE_MASK
- #define DMA_Stream0_IT_MASK
- #define **DMA_Stream1_IT_MASK** (uint32_t)(DMA_Stream0_IT_MASK << 6)
- #define **DMA_Stream2_IT_MASK** (uint32_t)(DMA_Stream0_IT_MASK << 16)
- #define **DMA_Stream3_IT_MASK** (uint32_t)(DMA_Stream0_IT_MASK << 22)
- #define **DMA_Stream4_IT_MASK** (uint32_t)(DMA_Stream0_IT_MASK | (uint32_t)0x20000000)
- #define **DMA_Stream5_IT_MASK** (uint32_t)(DMA_Stream1_IT_MASK | (uint32_t)0x20000000)
- #define **DMA_Stream6_IT_MASK** (uint32_t)(DMA_Stream2_IT_MASK | (uint32_t)0x20000000)
- #define **DMA_Stream7_IT_MASK** (uint32_t)(DMA_Stream3_IT_MASK | (uint32_t)0x20000000)
- #define **TRANSFER_IT_MASK** (uint32_t)0x0F3C0F3C
- #define **HIGH_ISR_MASK** (uint32_t)0x20000000
- #define **RESERVED_MASK** (uint32_t)0x0F7D0F7D

Functions

- void DMA_DeInit (DMA_Stream_TypeDef *DMAy_Streamx)

  Deinitialize the DMAy Streamx registers to their default reset values.
- void DMA_Init (DMA_Stream_TypeDef *DMAy_Streamx, DMA_InitTypeDef *DMA_InitStruct)

  Initializes the DMAy Streamx according to the specified parameters in the DMA_InitStruct structure.
- void DMA_StructInit (DMA_InitTypeDef *DMA_InitStruct)

  Fills each DMA_InitStruct member with its default value.
- void DMA_Cmd (DMA_Stream_TypeDef *DMAy_Streamx, FunctionalState NewState)

  Enables or disables the specified DMAy Streamx.
- void DMA_PeriphIncOffsetSizeConfig (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_Pincos)

  Configures, when the PINC (Peripheral Increment address mode) bit is set, if the peripheral address should be incremented with the data size (configured with PSIZE bits) or by a fixed offset equal to 4 (32-bit aligned addresses).
- void DMA_FlowControllerConfig (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_FlowCtrl)

  Configures, when the DMAy Streamx is disabled, the flow controller for the next transactions (Peripheral or Memory).
- void DMA_SetCurrDataCounter (DMA_Stream_TypeDef *DMAy_Streamx, uint16_t Counter)

  Writes the number of data units to be transferred on the DMAy Streamx.
- uint16_t DMA_GetCurrDataCounter (DMA_Stream_TypeDef *DMAy_Streamx)

  Returns the number of remaining data units in the current DMAy Streamx transfer.
- void DMA_DoubleBufferModeConfig (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t Memory1BaseAddr, uint32_t DMA_CurrentMemory)

  Configures, when the DMAy Streamx is disabled, the double buffer mode and the current memory target.
- void DMA_DoubleBufferModeCmd (DMA_Stream_TypeDef *DMAy_Streamx, FunctionalState NewState)

  Enables or disables the double buffer mode for the selected DMA stream.
- void DMA_MemoryTargetConfig (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t MemoryBaseAddr, uint32_t DMA_MemoryTarget)

  Configures the Memory address for the next buffer transfer in double buffer mode (for dynamic use). This function can be called when the DMA Stream is enabled and when the transfer is ongoing.
- uint32_t DMA_GetCurrentMemoryTarget (DMA_Stream_TypeDef *DMAy_Streamx)

  Returns the current memory target used by double buffer transfer.
- FunctionalState DMA_GetCmdStatus (DMA_Stream_TypeDef *DMAy_Streamx)

  Returns the status of EN bit for the specified DMAy Streamx.
- uint32_t DMA_GetFIFOStatus (DMA_Stream_TypeDef *DMAy_Streamx)

  Returns the current DMAy Streamx FIFO filled level.
- FlagStatus DMA_GetFlagStatus (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_FLAG)

  Checks whether the specified DMAy Streamx flag is set or not.
- void DMA_ClearFlag (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_FLAG)

  Clears the DMAy Streamx's pending flags.
- void DMA_ITConfig (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_IT, FunctionalState NewState)

  Enables or disables the specified DMAy Streamx interrupts.
- ITStatus DMA_GetITStatus (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_IT)

  Checks whether the specified DMAy Streamx interrupt has occurred or not.
- void DMA_ClearITPendingBit (DMA_Stream_TypeDef *DMAy_Streamx, uint32_t DMA_IT)

  Clears the DMAy Streamx's interrupt pending bits.

7.42.1 Detailed Description

This file provides firmware functions to manage the following functionalities of the Direct Memory Access controller (DMA):

Author

MCD Application Team

Version

V1.0.0

Date

30-September-2011

- Initialization and Configuration
- Data Counter
- Double Buffer mode configuration and command
- Interrupts and flags management

```
*
*          ===================================================================
*                              How to use this driver
*          ===================================================================
*          1. Enable The DMA controller clock using RCC_AHB1PeriphResetCmd(RCC_AHB1Periph_DMA1, ENABLE)
*             function for DMA1 or using RCC_AHB1PeriphResetCmd(RCC_AHB1Periph_DMA2, ENABLE)
*             function for DMA2.
*
*          2. Enable and configure the peripheral to be connected to the DMA Stream
*             (except for internal SRAM / FLASH memories: no initialization is
*             necessary).
*
*          3. For a given Stream, program the required configuration through following parameters:
*             Source and Destination addresses, Transfer Direction, Transfer size, Source and Destination
*             data formats, Circular or Normal mode, Stream Priority level, Source and Destination
*             Incrementation mode, FIFO mode and its Threshold (if needed), Burst mode for Source and/or
*             Destination (if needed) using the DMA_Init() function.
*             To avoid filling un-nesecessary fields, you can call DMA_StructInit() function
*             to initialize a given structure with default values (reset values), the modify
*             only necessary fields (ie. Source and Destination addresses, Transfer size and Data Formats).
*
*          4. Enable the NVIC and the corresponding interrupt(s) using the function
*             DMA_ITConfig() if you need to use DMA interrupts.
*
*          5. Optionally, if the Circular mode is enabled, you can use the Double buffer mode by configuring
*             the second Memory address and the first Memory to be used through the function
*             DMA_DoubleBufferModeConfig(). Then enable the Double buffer mode through the function
*             DMA_DoubleBufferModeCmd(). These operations must be done before step 6.
*
*          6. Enable the DMA stream using the DMA_Cmd() function.
*
*          7. Activate the needed Stream Request using PPP_DMACmd() function for
*             any PPP peripheral except internal SRAM and FLASH (ie. SPI, USART ...)
*             The function allowing this operation is provided in each PPP peripheral
*             driver (ie. SPI_DMACmd for SPI peripheral).
*             Once the Stream is enabled, it is not possible to modify its configuration
*             unless the stream is stopped and disabled.
*             After enabling the Stream, it is advised to monitor the EN bit status using
*             the function DMA_GetCmdStatus(). In case of configuration errors or bus errors
*             this bit will remain reset and all transfers on this Stream will remain on hold.
```

```
*
*          8. Optionally, you can configure the number of data to be transferred
*             when the Stream is disabled (ie. after each Transfer Complete event
*             or when a Transfer Error occurs) using the function DMA_SetCurrDataCounter().
*             And you can get the number of remaining data to be transferred using
*             the function DMA_GetCurrDataCounter() at run time (when the DMA Stream is
*             enabled and running).
*
*          9. To control DMA events you can use one of the following
*              two methods:
*               a- Check on DMA Stream flags using the function DMA_GetFlagStatus().
*               b- Use DMA interrupts through the function DMA_ITConfig() at initialization
*                  phase and DMA_GetITStatus() function into interrupt routines in
*                  communication phase.
*             After checking on a flag you should clear it using DMA_ClearFlag()
*             function. And after checking on an interrupt event you should
*             clear it using DMA_ClearITPendingBit() function.
*
*          10. Optionally, if Circular mode and Double Buffer mode are enabled, you can modify
*              the Memory Addresses using the function DMA_MemoryTargetConfig(). Make sure that
*              the Memory Address to be modified is not the one currently in use by DMA Stream.
*              This condition can be monitored using the function DMA_GetCurrentMemoryTarget().
*
*          11. Optionally, Pause-Resume operations may be performed:
*              The DMA_Cmd() function may be used to perform Pause-Resume operation. When a
*              transfer is ongoing, calling this function to disable the Stream will cause the
*              transfer to be paused. All configuration registers and the number of remaining
*              data will be preserved. When calling again this function to re-enable the Stream,
*              the transfer will be resumed from the point where it was paused.
*
* @note   Memory-to-Memory transfer is possible by setting the address of the memory into
*         the Peripheral registers. In this mode, Circular mode and Double Buffer mode
*         are not allowed.
*
* @note   The FIFO is used mainly to reduce bus usage and to allow data packing/unpacking: it is
*         possible to set different Data Sizes for the Peripheral and the Memory (ie. you can set
*         Half-Word data size for the peripheral to access its data register and set Word data size
*         for the Memory to gain in access time. Each two Half-words will be packed and written in
*         a single access to a Word in the Memory).
*
* @note  When FIFO is disabled, it is not allowed to configure different Data Sizes for Source
*        and Destination. In this case the Peripheral Data Size will be applied to both Source
*        and Destination.
*
*
```

Attention

7.43 CUBE_IDE/VGA/Core/Src/stm32f4xx_gpio.c File Reference

This file provides firmware functions to manage the following functionalities of the GPIO peripheral:

```
#include "stm32f4xx_gpio.h"
#include "stm32f4xx_rcc.h"
#include "stm32f4xx_conf.h"
```

Functions

- void GPIO_DeInit (GPIO_TypeDef ∗GPIOx)

  Deinitializes the GPIOx peripheral registers to their default reset values.
- void GPIO_Init (GPIO_TypeDef ∗GPIOx, GPIO_InitTypeDef ∗GPIO_InitStruct)

  Initializes the GPIOx peripheral according to the specified parameters in the GPIO_InitStruct.
- void GPIO_StructInit (GPIO_InitTypeDef ∗GPIO_InitStruct)

  Fills each GPIO_InitStruct member with its default value.
- void GPIO_PinLockConfig (GPIO_TypeDef ∗GPIOx, uint16_t GPIO_Pin)

  Locks GPIO Pins configuration registers.
- uint8_t GPIO_ReadInputDataBit (GPIO_TypeDef ∗GPIOx, uint16_t GPIO_Pin)

  Reads the specified input port pin.
- uint16_t GPIO_ReadInputData (GPIO_TypeDef ∗GPIOx)

  Reads the specified GPIO input data port.
- uint8_t GPIO_ReadOutputDataBit (GPIO_TypeDef ∗GPIOx, uint16_t GPIO_Pin)

  Reads the specified output data port bit.
- uint16_t GPIO_ReadOutputData (GPIO_TypeDef ∗GPIOx)

  Reads the specified GPIO output data port.
- void GPIO_SetBits (GPIO_TypeDef ∗GPIOx, uint16_t GPIO_Pin)

  Sets the selected data port bits.
- void GPIO_ResetBits (GPIO_TypeDef ∗GPIOx, uint16_t GPIO_Pin)

  Clears the selected data port bits.
- void GPIO_WriteBit (GPIO_TypeDef ∗GPIOx, uint16_t GPIO_Pin, BitAction BitVal)

  Sets or clears the selected data port bit.
- void GPIO_Write (GPIO_TypeDef ∗GPIOx, uint16_t PortVal)

  Writes data to the specified GPIO data port.
- void GPIO_ToggleBits (GPIO_TypeDef ∗GPIOx, uint16_t GPIO_Pin)

  Toggles the specified GPIO pins..
- void GPIO_PinAFConfig (GPIO_TypeDef ∗GPIOx, uint16_t GPIO_PinSource, uint8_t GPIO_AF)

  Changes the mapping of the specified pin.

7.43.1 Detailed Description

This file provides firmware functions to manage the following functionalities of the GPIO peripheral:

Author

MCD Application Team

Version

V1.0.0

Date

30-September-2011

- Initialization and Configuration
- GPIO Read and Write
- GPIO Alternate functions configuration

```
*
*          ===================================================================
*                              How to use this driver
*          ===================================================================
*           1. Enable the GPIO AHB clock using the following function
*                RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOx, ENABLE);
*
*           2. Configure the GPIO pin(s) using GPIO_Init()
*              Four possible configuration are available for each pin:
*                - Input: Floating, Pull-up, Pull-down.
*                - Output: Push-Pull (Pull-up, Pull-down or no Pull)
*                          Open Drain (Pull-up, Pull-down or no Pull).
*                  In output mode, the speed is configurable: 2 MHz, 25 MHz,
*                  50 MHz or 100 MHz.
*                - Alternate Function: Push-Pull (Pull-up, Pull-down or no Pull)
*                                      Open Drain (Pull-up, Pull-down or no Pull).
*                - Analog: required mode when a pin is to be used as ADC channel
*                          or DAC output.
*
*          3- Peripherals alternate function:
*              - For ADC and DAC, configure the desired pin in analog mode using
*                 GPIO_InitStruct->GPIO_Mode = GPIO_Mode_AN;
*              - For other peripherals (TIM, USART...):
*                 - Connect the pin to the desired peripherals' Alternate
*                   Function (AF) using GPIO_PinAFConfig() function
*                 - Configure the desired pin in alternate function mode using
*                   GPIO_InitStruct->GPIO_Mode = GPIO_Mode_AF
*                 - Select the type, pull-up/pull-down and output speed via
*                   GPIO_PuPd, GPIO_OType and GPIO_Speed members
*                 - Call GPIO_Init() function
*
*          4. To get the level of a pin configured in input mode use GPIO_ReadInputDataBit()
*
*          5. To set/reset the level of a pin configured in output mode use
*             GPIO_SetBits()/GPIO_ResetBits()
*
*          6. During and just after reset, the alternate functions are not
*             active and the GPIO pins are configured in input floating mode
*             (except JTAG pins).
*
*          7. The LSE oscillator pins OSC32_IN and OSC32_OUT can be used as
*             general-purpose (PC14 and PC15, respectively) when the LSE
*             oscillator is off. The LSE has priority over the GPIO function.
*
*          8. The HSE oscillator pins OSC_IN/OSC_OUT can be used as
*             general-purpose PH0 and PH1, respectively, when the HSE
*             oscillator is off. The HSE has priority over the GPIO function.
*
*
```

Attention

7.44 CUBE_IDE/VGA/Core/Src/stm32f4xx_rcc.c File Reference

This file provides firmware functions to manage the following functionalities of the Reset and clock control (RCC) peripheral:

```
#include "stm32f4xx_rcc.h"
#include "stm32f4xx_conf.h"
```

Macros

- #define **RCC_OFFSET** (RCC_BASE - PERIPH_BASE)
- #define **CR_OFFSET** (RCC_OFFSET + 0x00)
- #define **HSION_BitNumber** 0x00
- #define **CR_HSION_BB** (PERIPH_BB_BASE + (CR_OFFSET ∗ 32) + (HSION_BitNumber ∗ 4))
- #define **CSSON_BitNumber** 0x13
- #define **CR_CSSON_BB** (PERIPH_BB_BASE + (CR_OFFSET ∗ 32) + (CSSON_BitNumber ∗ 4))
- #define **PLLON_BitNumber** 0x18
- #define **CR_PLLON_BB** (PERIPH_BB_BASE + (CR_OFFSET ∗ 32) + (PLLON_BitNumber ∗ 4))
- #define **PLLI2SON_BitNumber** 0x1A
- #define **CR_PLLI2SON_BB** (PERIPH_BB_BASE + (CR_OFFSET ∗ 32) + (PLLI2SON_BitNumber ∗ 4))
- #define **CFGR_OFFSET** (RCC_OFFSET + 0x08)
- #define **I2SSRC_BitNumber** 0x17
- #define **CFGR_I2SSRC_BB** (PERIPH_BB_BASE + (CFGR_OFFSET ∗ 32) + (I2SSRC_BitNumber ∗ 4))
- #define **BDCR_OFFSET** (RCC_OFFSET + 0x70)
- #define **RTCEN_BitNumber** 0x0F
- #define **BDCR_RTCEN_BB** (PERIPH_BB_BASE + (BDCR_OFFSET ∗ 32) + (RTCEN_BitNumber ∗ 4))
- #define **BDRST_BitNumber** 0x10
- #define **BDCR_BDRST_BB** (PERIPH_BB_BASE + (BDCR_OFFSET ∗ 32) + (BDRST_BitNumber ∗ 4))
- #define **CSR_OFFSET** (RCC_OFFSET + 0x74)
- #define **LSION_BitNumber** 0x00
- #define **CSR_LSION_BB** (PERIPH_BB_BASE + (CSR_OFFSET ∗ 32) + (LSION_BitNumber ∗ 4))
- #define **CFGR_MCO2_RESET_MASK** ((uint32_t)0x07FFFFFF)
- #define **CFGR_MCO1_RESET_MASK** ((uint32_t)0xF89FFFFF)
- #define **FLAG_MASK** ((uint8_t)0x1F)
- #define **CR_BYTE3_ADDRESS** ((uint32_t)0x40023802)
- #define **CIR_BYTE2_ADDRESS** ((uint32_t)(RCC_BASE + 0x0C + 0x01))
- #define **CIR_BYTE3_ADDRESS** ((uint32_t)(RCC_BASE + 0x0C + 0x02))
- #define **BDCR_ADDRESS** (PERIPH_BASE + BDCR_OFFSET)

Functions

- void RCC_DeInit (void)

  Resets the RCC clock configuration to the default reset state.
- void RCC_HSEConfig (uint8_t RCC_HSE)

  Configures the External High Speed oscillator (HSE).
- ErrorStatus RCC_WaitForHSEStartUp (void)

  Waits for HSE start-up.
- void RCC_AdjustHSICalibrationValue (uint8_t HSICalibrationValue)

  Adjusts the Internal High Speed oscillator (HSI) calibration value.
- void RCC_HSICmd (FunctionalState NewState)

  Enables or disables the Internal High Speed oscillator (HSI).
- void RCC_LSEConfig (uint8_t RCC_LSE)

  Configures the External Low Speed oscillator (LSE).
- void RCC_LSICmd (FunctionalState NewState)

  Enables or disables the Internal Low Speed oscillator (LSI).
- void RCC_PLLConfig (uint32_t RCC_PLLSource, uint32_t PLLM, uint32_t PLLN, uint32_t PLLP, uint32_t PLLQ)

  Configures the main PLL clock source, multiplication and division factors.
- void RCC_PLLCmd (FunctionalState NewState)

  Enables or disables the main PLL.
- void RCC_PLLI2SConfig (uint32_t PLLI2SN, uint32_t PLLI2SR)

  Configures the PLLI2S clock multiplication and division factors.
- void RCC_PLLI2SCmd (FunctionalState NewState)

  Enables or disables the PLLI2S.
- void RCC_ClockSecuritySystemCmd (FunctionalState NewState)

  Enables or disables the Clock Security System.
- void RCC_MCO1Config (uint32_t RCC_MCO1Source, uint32_t RCC_MCO1Div)

  Selects the clock source to output on MCO1 pin(PA8).
- void RCC_MCO2Config (uint32_t RCC_MCO2Source, uint32_t RCC_MCO2Div)

  Selects the clock source to output on MCO2 pin(PC9).
- void RCC_SYSCLKConfig (uint32_t RCC_SYSCLKSource)

  Configures the system clock (SYSCLK).
- uint8_t RCC_GetSYSCLKSource (void)

  Returns the clock source used as system clock.
- void RCC_HCLKConfig (uint32_t RCC_SYSCLK)

  Configures the AHB clock (HCLK).
- void RCC_PCLK1Config (uint32_t RCC_HCLK)

  Configures the Low Speed APB clock (PCLK1).
- void RCC_PCLK2Config (uint32_t RCC_HCLK)

  Configures the High Speed APB clock (PCLK2).
- void RCC_GetClocksFreq (RCC_ClocksTypeDef *RCC_Clocks)

  Returns the frequencies of different on chip clocks; SYSCLK, HCLK, PCLK1 and PCLK2.
- void RCC_RTCCLKConfig (uint32_t RCC_RTCCLKSource)

  Configures the RTC clock (RTCCLK).
- void RCC_RTCCLKCmd (FunctionalState NewState)

  Enables or disables the RTC clock.
- void RCC_BackupResetCmd (FunctionalState NewState)

  Forces or releases the Backup domain reset.
- void RCC_I2SCLKConfig (uint32_t RCC_I2SCLKSource)

Configures the I2S clock source (I2SCLK).

- void RCC_AHB1PeriphClockCmd (uint32_t RCC_AHB1Periph, FunctionalState NewState)

  Enables or disables the AHB1 peripheral clock.
- void RCC_AHB2PeriphClockCmd (uint32_t RCC_AHB2Periph, FunctionalState NewState)

  Enables or disables the AHB2 peripheral clock.
- void RCC_AHB3PeriphClockCmd (uint32_t RCC_AHB3Periph, FunctionalState NewState)

  Enables or disables the AHB3 peripheral clock.
- void RCC_APB1PeriphClockCmd (uint32_t RCC_APB1Periph, FunctionalState NewState)

  Enables or disables the Low Speed APB (APB1) peripheral clock.
- void RCC_APB2PeriphClockCmd (uint32_t RCC_APB2Periph, FunctionalState NewState)

  Enables or disables the High Speed APB (APB2) peripheral clock.
- void RCC_AHB1PeriphResetCmd (uint32_t RCC_AHB1Periph, FunctionalState NewState)

  Forces or releases AHB1 peripheral reset.
- void RCC_AHB2PeriphResetCmd (uint32_t RCC_AHB2Periph, FunctionalState NewState)

  Forces or releases AHB2 peripheral reset.
- void RCC_AHB3PeriphResetCmd (uint32_t RCC_AHB3Periph, FunctionalState NewState)

  Forces or releases AHB3 peripheral reset.
- void RCC_APB1PeriphResetCmd (uint32_t RCC_APB1Periph, FunctionalState NewState)

  Forces or releases Low Speed APB (APB1) peripheral reset.
- void RCC_APB2PeriphResetCmd (uint32_t RCC_APB2Periph, FunctionalState NewState)

  Forces or releases High Speed APB (APB2) peripheral reset.
- void RCC_AHB1PeriphClockLPModeCmd (uint32_t RCC_AHB1Periph, FunctionalState NewState)

  Enables or disables the AHB1 peripheral clock during Low Power (Sleep) mode.
- void RCC_AHB2PeriphClockLPModeCmd (uint32_t RCC_AHB2Periph, FunctionalState NewState)

  Enables or disables the AHB2 peripheral clock during Low Power (Sleep) mode.
- void RCC_AHB3PeriphClockLPModeCmd (uint32_t RCC_AHB3Periph, FunctionalState NewState)

  Enables or disables the AHB3 peripheral clock during Low Power (Sleep) mode.
- void RCC_APB1PeriphClockLPModeCmd (uint32_t RCC_APB1Periph, FunctionalState NewState)

  Enables or disables the APB1 peripheral clock during Low Power (Sleep) mode.
- void RCC_APB2PeriphClockLPModeCmd (uint32_t RCC_APB2Periph, FunctionalState NewState)

  Enables or disables the APB2 peripheral clock during Low Power (Sleep) mode.
- void RCC_ITConfig (uint8_t RCC_IT, FunctionalState NewState)

  Enables or disables the specified RCC interrupts.
- FlagStatus RCC_GetFlagStatus (uint8_t RCC_FLAG)

  Checks whether the specified RCC flag is set or not.
- void RCC_ClearFlag (void)

  Clears the RCC reset flags. The reset flags are: RCC_FLAG_PINRST, RCC_FLAG_PORRST, RCC_FLAG_SFTRST, RCC_FLAG_IWDGRST, RCC_FLAG_WWDGRST, RCC_FLAG_LPWRRST.
- ITStatus RCC_GetITStatus (uint8_t RCC_IT)

  Checks whether the specified RCC interrupt has occurred or not.
- void RCC_ClearITPendingBit (uint8_t RCC_IT)

  Clears the RCC's interrupt pending bits.

7.44.1 Detailed Description

This file provides firmware functions to manage the following functionalities of the Reset and clock control (RCC) peripheral:

Author

MCD Application Team

Version

V1.0.0

Date

30-September-2011

- Internal/external clocks, PLL, CSS and MCO configuration
- System, AHB and APB busses clocks configuration
- Peripheral clocks configuration
- Interrupts and flags management

```
*
*          ===================================================================
*                              RCC specific features
*          ===================================================================
*
*          After reset the device is running from Internal High Speed oscillator
*          (HSI 16MHz) with Flash 0 wait state, Flash prefetch buffer, D-Cache
*          and I-Cache are disabled, and all peripherals are off except internal
*          SRAM, Flash and JTAG.
*           - There is no prescaler on High speed (AHB) and Low speed (APB) busses;
*             all peripherals mapped on these busses are running at HSI speed.
*             - The clock for all peripherals is switched off, except the SRAM and FLASH.
*           - All GPIOs are in input floating state, except the JTAG pins which
*             are assigned to be used for debug purpose.
*
*          Once the device started from reset, the user application has to:
*           - Configure the clock source to be used to drive the System clock
*             (if the application needs higher frequency/performance)
*           - Configure the System clock frequency and Flash settings
*           - Configure the AHB and APB busses prescalers
*           - Enable the clock for the peripheral(s) to be used
*           - Configure the clock source(s) for peripherals which clocks are not
*             derived from the System clock (I2S, RTC, ADC, USB OTG FS/SDIO/RNG)
*
*
```

Attention

7.45 CUBE_IDE/VGA/Core/Src/stm32f4xx_tim.c File Reference

This file provides firmware functions to manage the following functionalities of the TIM peripheral:

```
#include "stm32f4xx_tim.h"
#include "stm32f4xx_rcc.h"
#include "stm32f4xx_conf.h"
```

Macros

- #define **SMCR_ETR_MASK** ((uint16_t)0x00FF)
- #define **CCMR_OFFSET** ((uint16_t)0x0018)
- #define **CCER_CCE_SET** ((uint16_t)0x0001)
- #define **CCER_CCNE_SET** ((uint16_t)0x0004)
- #define **CCMR_OC13M_MASK** ((uint16_t)0xFF8F)
- #define **CCMR_OC24M_MASK** ((uint16_t)0x8FFF)

Functions

- void TIM_DeInit (TIM_TypeDef ∗TIMx)

  Deinitializes the TIMx peripheral registers to their default reset values.
- void TIM_TimeBaseInit (TIM_TypeDef ∗TIMx, TIM_TimeBaseInitTypeDef ∗TIM_TimeBaseInitStruct)

  Initializes the TIMx Time Base Unit peripheral according to the specified parameters in the TIM_TimeBaseInitStruct.
- void TIM_TimeBaseStructInit (TIM_TimeBaseInitTypeDef ∗TIM_TimeBaseInitStruct)

  Fills each TIM_TimeBaseInitStruct member with its default value.
- void TIM_PrescalerConfig (TIM_TypeDef ∗TIMx, uint16_t Prescaler, uint16_t TIM_PSCReloadMode)

  Configures the TIMx Prescaler.
- void TIM_CounterModeConfig (TIM_TypeDef ∗TIMx, uint16_t TIM_CounterMode)

  Specifies the TIMx Counter Mode to be used.
- void TIM_SetCounter (TIM_TypeDef ∗TIMx, uint32_t Counter)

  Sets the TIMx Counter Register value.
- void TIM_SetAutoreload (TIM_TypeDef ∗TIMx, uint32_t Autoreload)

  Sets the TIMx Autoreload Register value.
- uint32_t TIM_GetCounter (TIM_TypeDef ∗TIMx)

  Gets the TIMx Counter value.
- uint16_t TIM_GetPrescaler (TIM_TypeDef ∗TIMx)

  Gets the TIMx Prescaler value.
- void TIM_UpdateDisableConfig (TIM_TypeDef ∗TIMx, FunctionalState NewState)

  Enables or Disables the TIMx Update event.
- void TIM_UpdateRequestConfig (TIM_TypeDef ∗TIMx, uint16_t TIM_UpdateSource)

  Configures the TIMx Update Request Interrupt source.
- void TIM_ARRPreloadConfig (TIM_TypeDef ∗TIMx, FunctionalState NewState)

  Enables or disables TIMx peripheral Preload register on ARR.
- void TIM_SelectOnePulseMode (TIM_TypeDef ∗TIMx, uint16_t TIM_OPMode)

  Selects the TIMx's One Pulse Mode.
- void TIM_SetClockDivision (TIM_TypeDef ∗TIMx, uint16_t TIM_CKD)

  Sets the TIMx Clock Division value.
- void TIM_Cmd (TIM_TypeDef ∗TIMx, FunctionalState NewState)

  Enables or disables the specified TIM peripheral.
- void TIM_OC1Init (TIM_TypeDef ∗TIMx, TIM_OCInitTypeDef ∗TIM_OCInitStruct)

  Initializes the TIMx Channel1 according to the specified parameters in the TIM_OCInitStruct.
- void TIM_OC2Init (TIM_TypeDef ∗TIMx, TIM_OCInitTypeDef ∗TIM_OCInitStruct)

  Initializes the TIMx Channel2 according to the specified parameters in the TIM_OCInitStruct.
- void TIM_OC3Init (TIM_TypeDef ∗TIMx, TIM_OCInitTypeDef ∗TIM_OCInitStruct)

  Initializes the TIMx Channel3 according to the specified parameters in the TIM_OCInitStruct.
- void TIM_OC4Init (TIM_TypeDef ∗TIMx, TIM_OCInitTypeDef ∗TIM_OCInitStruct)

  Initializes the TIMx Channel4 according to the specified parameters in the TIM_OCInitStruct.
- void TIM_OCStructInit (TIM_OCInitTypeDef ∗TIM_OCInitStruct)

  Fills each TIM_OCInitStruct member with its default value.

- void TIM_SelectOCxM (TIM_TypeDef *TIMx, uint16_t TIM_Channel, uint16_t TIM_OCMode)

  Selects the TIM Output Compare Mode.
- void TIM_SetCompare1 (TIM_TypeDef *TIMx, uint32_t Compare1)

  Sets the TIMx Capture Compare1 Register value.
- void TIM_SetCompare2 (TIM_TypeDef *TIMx, uint32_t Compare2)

  Sets the TIMx Capture Compare2 Register value.
- void TIM_SetCompare3 (TIM_TypeDef *TIMx, uint32_t Compare3)

  Sets the TIMx Capture Compare3 Register value.
- void TIM_SetCompare4 (TIM_TypeDef *TIMx, uint32_t Compare4)

  Sets the TIMx Capture Compare4 Register value.
- void TIM_ForcedOC1Config (TIM_TypeDef *TIMx, uint16_t TIM_ForcedAction)

  Forces the TIMx output 1 waveform to active or inactive level.
- void TIM_ForcedOC2Config (TIM_TypeDef *TIMx, uint16_t TIM_ForcedAction)

  Forces the TIMx output 2 waveform to active or inactive level.
- void TIM_ForcedOC3Config (TIM_TypeDef *TIMx, uint16_t TIM_ForcedAction)

  Forces the TIMx output 3 waveform to active or inactive level.
- void TIM_ForcedOC4Config (TIM_TypeDef *TIMx, uint16_t TIM_ForcedAction)

  Forces the TIMx output 4 waveform to active or inactive level.
- void TIM_OC1PreloadConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPreload)

  Enables or disables the TIMx peripheral Preload register on CCR1.
- void TIM_OC2PreloadConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPreload)

  Enables or disables the TIMx peripheral Preload register on CCR2.
- void TIM_OC3PreloadConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPreload)

  Enables or disables the TIMx peripheral Preload register on CCR3.
- void TIM_OC4PreloadConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPreload)

  Enables or disables the TIMx peripheral Preload register on CCR4.
- void TIM_OC1FastConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCFast)

  Configures the TIMx Output Compare 1 Fast feature.
- void TIM_OC2FastConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCFast)

  Configures the TIMx Output Compare 2 Fast feature.
- void TIM_OC3FastConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCFast)

  Configures the TIMx Output Compare 3 Fast feature.
- void TIM_OC4FastConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCFast)

  Configures the TIMx Output Compare 4 Fast feature.
- void TIM_ClearOC1Ref (TIM_TypeDef *TIMx, uint16_t TIM_OCClear)

  Clears or safeguards the OCREF1 signal on an external event.
- void TIM_ClearOC2Ref (TIM_TypeDef *TIMx, uint16_t TIM_OCClear)

  Clears or safeguards the OCREF2 signal on an external event.
- void TIM_ClearOC3Ref (TIM_TypeDef *TIMx, uint16_t TIM_OCClear)

  Clears or safeguards the OCREF3 signal on an external event.
- void TIM_ClearOC4Ref (TIM_TypeDef *TIMx, uint16_t TIM_OCClear)

  Clears or safeguards the OCREF4 signal on an external event.
- void TIM_OC1PolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPolarity)

  Configures the TIMx channel 1 polarity.
- void TIM_OC1NPolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCNPolarity)

  Configures the TIMx Channel 1N polarity.
- void TIM_OC2PolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPolarity)

  Configures the TIMx channel 2 polarity.
- void TIM_OC2NPolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCNPolarity)

  Configures the TIMx Channel 2N polarity.
- void TIM_OC3PolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPolarity)

Configures the TIMx channel 3 polarity.

- void TIM_OC3NPolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCNPolarity)

  Configures the TIMx Channel 3N polarity.
- void TIM_OC4PolarityConfig (TIM_TypeDef *TIMx, uint16_t TIM_OCPolarity)

  Configures the TIMx channel 4 polarity.
- void TIM_CCxCmd (TIM_TypeDef *TIMx, uint16_t TIM_Channel, uint16_t TIM_CCx)

  Enables or disables the TIM Capture Compare Channel x.
- void TIM_CCxNCmd (TIM_TypeDef *TIMx, uint16_t TIM_Channel, uint16_t TIM_CCxN)

  Enables or disables the TIM Capture Compare Channel xN.
- void TIM_ICInit (TIM_TypeDef *TIMx, TIM_ICInitTypeDef *TIM_ICInitStruct)

  Initializes the TIM peripheral according to the specified parameters in the TIM_ICInitStruct.
- void TIM_ICStructInit (TIM_ICInitTypeDef *TIM_ICInitStruct)

  Fills each TIM_ICInitStruct member with its default value.
- void TIM_PWMIConfig (TIM_TypeDef *TIMx, TIM_ICInitTypeDef *TIM_ICInitStruct)

  Configures the TIM peripheral according to the specified parameters in the TIM_ICInitStruct to measure an external PWM signal.
- uint32_t TIM_GetCapture1 (TIM_TypeDef *TIMx)

  Gets the TIMx Input Capture 1 value.
- uint32_t TIM_GetCapture2 (TIM_TypeDef *TIMx)

  Gets the TIMx Input Capture 2 value.
- uint32_t TIM_GetCapture3 (TIM_TypeDef *TIMx)

  Gets the TIMx Input Capture 3 value.
- uint32_t TIM_GetCapture4 (TIM_TypeDef *TIMx)

  Gets the TIMx Input Capture 4 value.
- void TIM_SetIC1Prescaler (TIM_TypeDef *TIMx, uint16_t TIM_ICPSC)

  Sets the TIMx Input Capture 1 prescaler.
- void TIM_SetIC2Prescaler (TIM_TypeDef *TIMx, uint16_t TIM_ICPSC)

  Sets the TIMx Input Capture 2 prescaler.
- void TIM_SetIC3Prescaler (TIM_TypeDef *TIMx, uint16_t TIM_ICPSC)

  Sets the TIMx Input Capture 3 prescaler.
- void TIM_SetIC4Prescaler (TIM_TypeDef *TIMx, uint16_t TIM_ICPSC)

  Sets the TIMx Input Capture 4 prescaler.
- void TIM_BDTRConfig (TIM_TypeDef *TIMx, TIM_BDTRInitTypeDef *TIM_BDTRInitStruct)

  Configures the Break feature, dead time, Lock level, OSSI/OSSR State and the AOE(automatic output enable).
- void TIM_BDTRStructInit (TIM_BDTRInitTypeDef *TIM_BDTRInitStruct)

  Fills each TIM_BDTRInitStruct member with its default value.
- void TIM_CtrlPWMOutputs (TIM_TypeDef *TIMx, FunctionalState NewState)

  Enables or disables the TIM peripheral Main Outputs.
- void TIM_SelectCOM (TIM_TypeDef *TIMx, FunctionalState NewState)

  Selects the TIM peripheral Commutation event.
- void TIM_CCPreloadControl (TIM_TypeDef *TIMx, FunctionalState NewState)

  Sets or Resets the TIM peripheral Capture Compare Preload Control bit.
- void TIM_ITConfig (TIM_TypeDef *TIMx, uint16_t TIM_IT, FunctionalState NewState)

  Enables or disables the specified TIM interrupts.
- void TIM_GenerateEvent (TIM_TypeDef *TIMx, uint16_t TIM_EventSource)

  Configures the TIMx event to be generate by software.
- FlagStatus TIM_GetFlagStatus (TIM_TypeDef *TIMx, uint16_t TIM_FLAG)

  Checks whether the specified TIM flag is set or not.
- void TIM_ClearFlag (TIM_TypeDef *TIMx, uint16_t TIM_FLAG)

  Clears the TIMx's pending flags.
- ITStatus TIM_GetITStatus (TIM_TypeDef *TIMx, uint16_t TIM_IT)

Checks whether the TIM interrupt has occurred or not.

- void TIM_ClearITPendingBit (TIM_TypeDef *TIMx, uint16_t TIM_IT)

  Clears the TIMx's interrupt pending bits.
- void TIM_DMAConfig (TIM_TypeDef *TIMx, uint16_t TIM_DMABase, uint16_t TIM_DMABurstLength)

  Configures the TIMx's DMA interface.
- void TIM_DMACmd (TIM_TypeDef *TIMx, uint16_t TIM_DMASource, FunctionalState NewState)

  Enables or disables the TIMx's DMA Requests.
- void TIM_SelectCCDMA (TIM_TypeDef *TIMx, FunctionalState NewState)

  Selects the TIMx peripheral Capture Compare DMA source.
- void TIM_InternalClockConfig (TIM_TypeDef *TIMx)

  Configures the TIMx internal Clock.
- void TIM_ITRxExternalClockConfig (TIM_TypeDef *TIMx, uint16_t TIM_InputTriggerSource)

  Configures the TIMx Internal Trigger as External Clock.
- void TIM_TIxExternalClockConfig (TIM_TypeDef *TIMx, uint16_t TIM_TIxExternalCLKSource, uint16_t TIM_ICPolarity, uint16_t ICFilter)

  Configures the TIMx Trigger as External Clock.
- void TIM_ETRClockMode1Config (TIM_TypeDef *TIMx, uint16_t TIM_ExtTRGPrescaler, uint16_t TIM_ExtTRGPolarity, uint16_t ExtTRGFilter)

  Configures the External clock Mode1.
- void TIM_ETRClockMode2Config (TIM_TypeDef *TIMx, uint16_t TIM_ExtTRGPrescaler, uint16_t TIM_ExtTRGPolarity, uint16_t ExtTRGFilter)

  Configures the External clock Mode2.
- void TIM_SelectInputTrigger (TIM_TypeDef *TIMx, uint16_t TIM_InputTriggerSource)

  Selects the Input Trigger source.
- void TIM_SelectOutputTrigger (TIM_TypeDef *TIMx, uint16_t TIM_TRGOSource)

  Selects the TIMx Trigger Output Mode.
- void TIM_SelectSlaveMode (TIM_TypeDef *TIMx, uint16_t TIM_SlaveMode)

  Selects the TIMx Slave Mode.
- void TIM_SelectMasterSlaveMode (TIM_TypeDef *TIMx, uint16_t TIM_MasterSlaveMode)

  Sets or Resets the TIMx Master/Slave Mode.
- void TIM_ETRConfig (TIM_TypeDef *TIMx, uint16_t TIM_ExtTRGPrescaler, uint16_t TIM_ExtTRGPolarity, uint16_t ExtTRGFilter)

  Configures the TIMx External Trigger (ETR).
- void TIM_EncoderInterfaceConfig (TIM_TypeDef *TIMx, uint16_t TIM_EncoderMode, uint16_t TIM_IC1Polarity, uint16_t TIM_IC2Polarity)

  Configures the TIMx Encoder Interface.
- void TIM_SelectHallSensor (TIM_TypeDef *TIMx, FunctionalState NewState)

  Enables or disables the TIMx's Hall sensor interface.
- void TIM_RemapConfig (TIM_TypeDef *TIMx, uint16_t TIM_Remap)

  Configures the TIM2, TIM5 and TIM11 Remapping input capabilities.

7.45.1 Detailed Description

This file provides firmware functions to manage the following functionalities of the TIM peripheral:

Author

MCD Application Team

Version

V1.0.0

Date

30-September-2011

- TimeBase management
- Output Compare management
- Input Capture management
- Advanced-control timers (TIM1 and TIM8) specific features

- Interrupts, DMA and flags management
- Clocks management
- Synchronization management
- Specific interface management
- Specific remapping management

```
*
*          ===================================================================
*                              How to use this driver
*          ===================================================================
*          This driver provides functions to configure and program the TIM
*          of all STM32F4xx devices.
*          These functions are split in 9 groups:
*
*          1. TIM TimeBase management: this group includes all needed functions
*             to configure the TM Timebase unit:
*                   - Set/Get Prescaler
*                   - Set/Get Autoreload
*                   - Counter modes configuration
*                   - Set Clock division
*                   - Select the One Pulse mode
*                   - Update Request Configuration
*                   - Update Disable Configuration
*                   - Auto-Preload Configuration
*                   - Enable/Disable the counter
*
*          2. TIM Output Compare management: this group includes all needed
*             functions to configure the Capture/Compare unit used in Output
*             compare mode:
*                   - Configure each channel, independently, in Output Compare mode
*                   - Select the output compare modes
*                   - Select the Polarities of each channel
*                   - Set/Get the Capture/Compare register values
*                   - Select the Output Compare Fast mode
*                   - Select the Output Compare Forced mode
*                   - Output Compare-Preload Configuration
*                   - Clear Output Compare Reference
*                   - Select the OCREF Clear signal
*                   - Enable/Disable the Capture/Compare Channels
*
*          3. TIM Input Capture management: this group includes all needed
*             functions to configure the Capture/Compare unit used in
*             Input Capture mode:
*                   - Configure each channel in input capture mode
*                   - Configure Channel1/2 in PWM Input mode
*                   - Set the Input Capture Prescaler
*                   - Get the Capture/Compare values
*
*          4. Advanced-control timers (TIM1 and TIM8) specific features
*                   - Configures the Break input, dead time, Lock level, the OSSI,
*                     the OSSR State and the AOE(automatic output enable)
```

```
*                   - Enable/Disable the TIM peripheral Main Outputs
*                   - Select the Commutation event
*                   - Set/Reset the Capture Compare Preload Control bit
*
*          5. TIM interrupts, DMA and flags management
*                   - Enable/Disable interrupt sources
*                   - Get flags status
*                   - Clear flags/ Pending bits
*                   - Enable/Disable DMA requests
*                   - Configure DMA burst mode
*                   - Select CaptureCompare DMA request
*
*          6. TIM clocks management: this group includes all needed functions
*             to configure the clock controller unit:
*                   - Select internal/External clock
*                   - Select the external clock mode: ETR(Mode1/Mode2), TIx or ITRx
*
*          7. TIM synchronization management: this group includes all needed
*             functions to configure the Synchronization unit:
*                   - Select Input Trigger
*                   - Select Output Trigger
*                   - Select Master Slave Mode
*                   - ETR Configuration when used as external trigger
*
*          8. TIM specific interface management, this group includes all
*             needed functions to use the specific TIM interface:
*                   - Encoder Interface Configuration
*                   - Select Hall Sensor
*
*          9. TIM specific remapping management includes the Remapping
*             configuration of specific timers
*
*
```

Attention

7.46 CUBE_IDE/VGA/Core/Src/system_stm32f4xx.c File Reference

CMSIS Cortex-M4 Device Peripheral Access Layer System Source File. This file contains the system clock configuration for STM32F4xx devices, and is generated by the clock configuration tool stm32f4xx_Clock_Configuration↩_V1.0.0.xls.

```
#include "stm32f4xx.h"
```

Macros

- #define VECT_TAB_OFFSET 0x00
- #define **PLL_M** 8
- #define **PLL_N** 336
- #define **PLL_P** 2
- #define **PLL_Q** 7

Functions

- void SystemInit (void)

  Setup the microcontroller system Initialize the Embedded Flash Interface, the PLL and update the SystemFrequency variable.
- void SystemCoreClockUpdate (void)

  Update SystemCoreClock variable according to Clock Register Values. The SystemCoreClock variable contains the core clock (HCLK), it can be used by the user application to setup the SysTick timer or configure other parameters.

Variables

- uint32_t SystemCoreClock = 168000000
- __I uint8_t **AHBPrescTable** [16] = {0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 3, 4, 6, 7, 8, 9}

7.46.1 Detailed Description

CMSIS Cortex-M4 Device Peripheral Access Layer System Source File. This file contains the system clock configuration for STM32F4xx devices, and is generated by the clock configuration tool stm32f4xx_Clock_Configuration↩_V1.0.0.xls.

Author

MCD Application Team

Version

V1.0.0

Date

19-September-2011

1. This file provides two functions and one global variable to be called from user application:
   - SystemInit(): Setups the system clock (System clock source, PLL Multiplier and Divider factors, AHB/↩APBx prescalers and Flash settings), depending on the configuration made in the clock xls tool. This function is called at startup just after reset and before branch to main program. This call is made inside the "startup_stm32f4xx.s" file.
   - SystemCoreClock variable: Contains the core clock (HCLK), it can be used by the user application to setup the SysTick timer or configure other parameters.
   - SystemCoreClockUpdate(): Updates the variable SystemCoreClock and must be called whenever the core clock is changed during program execution.
2. After each device reset the HSI (16 MHz) is used as system clock source. Then SystemInit() function is called, in "startup_stm32f4xx.s" file, to configure the system clock before to branch to main program.
3. If the system clock source selected by user fails to startup, the SystemInit() function will do nothing and HSI still used as system clock source. User can add some code to deal with this issue inside the SetSysClock() function.
4. The default value of HSE crystal is set to 8 MHz, refer to "HSE_VALUE" define in "stm32f4xx.h" file. When HSE is used as system clock source, directly or through PLL, and you are using different crystal you have to adapt the HSE value to your own configuration.

7.46.2 5. This file configures the system clock as follows:

==========

7.46.2.1 Supported STM32F4xx device revision | Rev A

7.46.2.2 System Clock source | PLL (HSE)

7.46.2.3 SYSCLK(Hz) | 168000000

7.46.2.4 HCLK(Hz) | 168000000

7.46.2.5 AHB Prescaler | 1

7.46.2.6 APB1 Prescaler | 4

7.46.2.7 APB2 Prescaler | 2

7.46.2.8 HSE Frequency(Hz) | 8000000

7.46.2.9 PLL_M | 8

7.46.2.10 PLL_N | 336

7.46.2.11 PLL_P | 2

7.46.2.12 PLL_Q | 7

7.46.2.13 PLLI2S_N | NA

7.46.2.14 PLLI2S_R | NA

7.46.2.15 I2S input clock | NA

7.46.2.16 VDD(V) | 3.3

7.46.2.17 High Performance mode | Enabled

7.46.2.18 Flash Latency(WS) | 5

7.46.2.19 Prefetch Buffer | OFF

7.46.2.20 Instruction cache | ON

7.46.2.21 Data cache | ON

Require 48MHz for USB OTG FS, | Enabled

7.46.2.22 SDIO and RNG clock |

==========

Attention

THE PRESENT FIRMWARE WHICH IS FOR GUIDANCE ONLY AIMS AT PROVIDING CUSTOMERS WITH CODING INFORMATION REGARDING THEIR PRODUCTS IN ORDER FOR THEM TO SAVE TIME. AS A RESULT, STMICROELECTRONICS SHALL NOT BE HELD LIABLE FOR ANY DIRECT, INDIRECT OR CONSEQUENTIAL DAMAGES WITH RESPECT TO ANY CLAIMS ARISING FROM THE CONTENT OF SUCH FIRMWARE AND/OR THE USE MADE BY CUSTOMERS OF THE CODING INFORMATION CONTAINED HEREIN IN CONNECTION WITH THEIR PRODUCTS.

7.47 CUBE_IDE/VGA/Core/Src/UART_communication.c File Reference

In this file UART is used to make communication possible between the STM32F4 board and a computer. With this code messages can be received and sent. This code is made with the assistance of an online guide from 'Controllerstech'.

```
#include "UART_communication.h"
```

Functions

- void UART2_config (void)

  Function to enable all the necessary registers and operations to use UART.
- void UART_sendChar (uint8_t c)

  Function that sends char data to the connected terminal.
- void UART_sendString (char ∗string)

  Function that sends a string to the connected terminal.
- uint8_t UART_getChar (void)

  Function that receives data from the connected terminal.
- UART UART_receiver (void)

  Function that read a single line and stores the data.
- void UART_errorHandling (int err)

  Function that handles the occuring errors to the terminal.

7.47.1 Detailed Description

In this file UART is used to make communication possible between the STM32F4 board and a computer. With this code messages can be received and sent. This code is made with the assistance of an online guide from 'Controllerstech'.

Authors

Skip Wijtman

Date

3-5-2023

Version

1.1

7.47.2 Function Documentation

7.47.2.1 UART2_config()

```
void UART2_config (
             void  )
```

Function to enable all the necessary registers and operations to use UART.

Parameters

Nothing	

Returns

Nothing

7.47.2.2 UART_errorHandling()

```
void UART_errorHandling (
             int err )
```

Function that handles the occuring errors to the terminal.

Parameters

err	error that is supposed to be shown

Returns

nothing

7.47.2.3 UART_getChar()

```
uint8_t UART_getChar (
            void  )
```

Function that receives data from the connected terminal.

Parameters

Nothing	

Returns

Received data

7.47.2.4 UART_receiver()

```
UART UART_receiver (
            void  )
```

Function that read a single line and stores the data.

Parameters

Nothing	

Returns

struct variable

7.47.2.5 UART_sendChar()

```
void UART_sendChar (
            uint8_t c )
```

Function that sends char data to the connected terminal.

Parameters

Char	data do be send

Returns

Nothing

7.47.2.6 UART_sendString()

```
void UART_sendString (
            char * string )
```

Function that sends a string to the connected terminal.

Parameters

String	do be send

Returns

Nothing