

# Контроллеры ARMatura

© Dmitry Ponyatov <dponyatov@gmail.com>, SSAU ASCL

1 марта 2013 г.

## Оглавление

<b>Оглавление</b>	<b>1</b>
<b>I Введение</b>	<b>6</b>
<b>II Железо</b>	<b>8</b>
1 STM32VLDISCOVERY /STM32F100RBT6/	9
2 STM32F4DISCOVERY /SRM32F407VGT6/	11
3 ARMatura /STM32F417IGT/	13
4 PION /STM32F100C4T6B/	14
<b>III Первые шаги</b>	<b>15</b>
5 Установка пакета ПО STlink	16
6 Установка Keil MDK-ARM	22
7 Первый проект: blink	29
7.1 Настройки проекта в Keil . . . . .	30
7.2 Структура файлов . . . . .	43
7.3 blink.c . . . . .	44
8 Hell Of World	45

<b>IV Средства разработки</b>	<b>46</b>
<b>9 Keil MDK-ARM</b>	<b>47</b>
<b>10 Компиляторы</b>	<b>48</b>
10.1 GCC . . . . .	48
10.2 KeilCC . . . . .	48
10.3 IAR . . . . .	48
<b>11 IDE</b>	<b>49</b>
11.1 Eclipse . . . . .	49
11.2 Code::Blocks . . . . .	49
11.3 gVim . . . . .	49
11.4 Keil uVision . . . . .	49
11.5 IAR . . . . .	49
<b>12 Программаторы</b>	<b>50</b>
12.1 STlink . . . . .	50
12.2 Serial Boot . . . . .	50
<b>13 Отладчики</b>	<b>51</b>
13.1 JTAG . . . . .	51
13.2 STM32 SWD . . . . .	51
13.3 GDB . . . . .	51
<b>V Основы языка C<sup>+</sup></b>	<b>52</b>
<b>14 Синтаксис</b>	<b>53</b>
<b>15 Типы данных</b>	<b>54</b>
<b>16 Стандартная библиотека libc</b>	<b>55</b>
<b>VI Отладка</b>	<b>56</b>
<b>17 JTAG</b>	<b>57</b>
<b>18 GDB</b>	<b>58</b>
<b>19 OpenOCD</b>	<b>59</b>
<b>VICMSIS</b>	<b>60</b>
<b>20 Startup: файлы стартового кода</b>	<b>61</b>
<b>21 Стандартная библиотека STM32</b>	<b>74</b>
21.1 stm32f4xx.h . . . . .	75
<b>22 USB client/host</b>	<b>280</b>

<b>VII Ядро Cortex-Mx</b>	<b>281</b>
<b>23 Режимы ARM и Thumb</b>	<b>282</b>
<b>24 DMA</b>	<b>283</b>
<b>25 DSP /Cortex-M3/</b>	<b>284</b>
<b>26 FPU /Cortex-M4F/</b>	<b>285</b>
<b>IX Интерфейсы</b>	<b>286</b>
<b>27 USB</b>	<b>287</b>
<b>28 UART</b>	<b>288</b>
<b>29 SPI</b>	<b>289</b>
<b>30 I2C</b>	<b>290</b>
<b>31 CAN</b>	<b>291</b>
<b>X Операционные системы OCPB</b>	<b>292</b>
<b>32 Keil RTX</b>	<b>293</b>
<b>33 FreeRTOS</b>	<b>294</b>
<b>34 eCos</b>	<b>295</b>
<b>35 Linux</b>	<b>296</b>
<b>XI Стек TCP/IP</b>	<b>297</b>
<b>36 Ethernet</b>	<b>298</b>
<b>37 PPP</b>	<b>299</b>
<b>XII Типовые применения</b>	<b>300</b>
<b>38 GPS</b>	<b>301</b>
38.1 Протокол NMEA 0183 . . . . .	301
38.2 \$GPRMC — рекомендуемый минимум навигационных данных . . . . .	302
38.3 Системы координат (датум) . . . . .	303
38.4 Методы преобразования систем координат . . . . .	304
38.5 Геоид и эллипсоиды . . . . .	305
38.6 Tistar15 . . . . .	306
38.7 WISMO228 . . . . .	306
<b>39 GSM</b>	<b>307</b>

39.1 WISMO228 . . . . .	307
<b>40 шина Dallas 1Wire</b>	<b>308</b>
40.1 RTC . . . . .	308
40.2 Датчики температуры DS18x20 . . . . .	308
<b>XIIИстраиваемый Linux</b>	<b>309</b>
<b>XIIIQA</b>	<b>310</b>
<b>XIVПриложения</b>	<b>312</b>
<b>41 Сводная таблица процессоров</b>	<b>313</b>
41.1 STM32F10x . . . . .	313

# Listings

7.1	blink.c	44
20.1	keil_startup_stm32f4xx.s	61
21.1	stm32f4xx.h	75

# Часть I

## Введение

Эта книга – набор методичек по разработке ПО для встраиваемых систем, написанных для Института космического приборостроения СГАУ.

Для применения в реальных проектах научной аппаратуры была разработана линейка унифицированных модулей:

1. ARMatura — модуль на мощном микропроцессоре STM32F417IGT: 1M Flash, 192K SRAM, TQFP176, DSP, FPU,.. [41](#)

предназначен для использования в качестве центрального процессора цифровой системы: обработка данных, сложные алгоритмы управления, ЦОС, вычисления, реализация протоколов передачи данных по интерфейсам USB, Ethernet, RS232/UART, SPI, I2C, CAN,..

2. PION [4](#) — модуль на самом простом и дешевом STM32F100C4T6B: 128K Flash, 8K SRAM, UART, SPI [41.1](#)

периферийный модуль для стыковки с аналоговыми датчиками и исполнительными устройствами, предварительная ЦОС обработка, передача данных на ARMatura-модули для дальнейшей обработки данных.

также модуль применим в качестве самостоятельного простого интерфейса при замене на чип STM32F103 с портом USB или установки внешних интерфейсных микросхем FT232RL (USB Serial), CP1202, MC1551 (CAN).

3. BACKPLANE — коммутационная плата межмодульного интерфейса
4. POWER — модуль импульсного источника питания
5. STEPPER — модуль управления двухфазным шаговым двигателем
6. WISMO — несущая плата для GPS/GSM модуля WISMO 228
7. QVGA — несущая плата для TFT touch-панели

В качестве базового микроконтроллера были выбраны чипы семейства STM32Fxxx с ядрами Cortex-M3, Cortex-M4F (ARM) как самые дешевые, и имеющие хорошую поддержку в виде отладочных плат линейки Discovery.

В общем, линейка модулей ARMatura может рассматриваться в качестве замены устаревшей линейки периферийных контроллеров Arduino на базе МК AVR8.

Проект размещен в репозитории <https://github.com/ponyatov/ARMatura.git> и предоставляется на условиях OpenHardware licence (за исключением прошивок и схем по тематике ИКП СГАУ).

Контакты разработчиков:

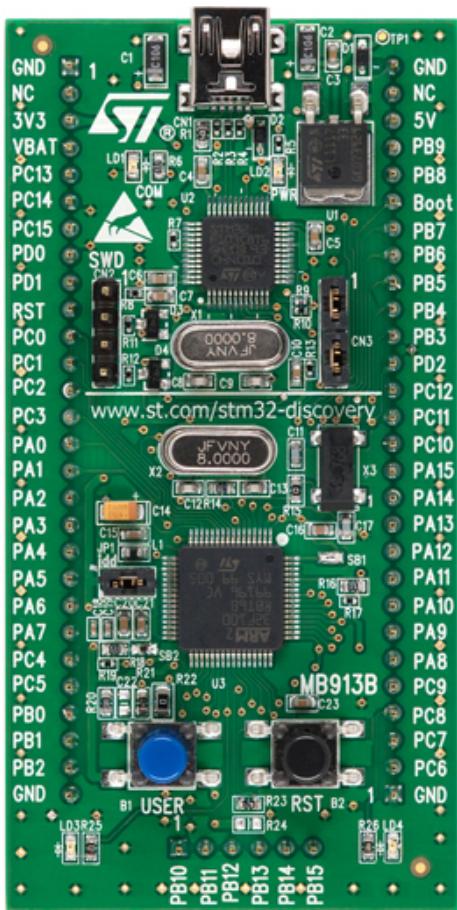
- ИКП СГАУ <[semkin@ssau.ru](mailto:semkin@ssau.ru)>
- Дмитрий Понятов <[dponyatov@gmail.com](mailto:dponyatov@gmail.com)>

## Часть II

### Железо

# Глава 1

## STM32VLDISCOVERY /STM32F100RBT6/



- Микроконтроллер STM32F100RB [41](#), 128 KB Flash, 8 KB RAM in 64-pin LQFP
- Встроенный ST-Link с возможностью использования в режиме внешнего программатора (только с SWD коннектором)
- Разработана для питания как от USB, так и от внешнего источника 3.3 или 5 вольт
- Может обеспечить питание 3 и 5 вольт для внешних устройств
- Два пользовательских светодиода (зеленый и синий)
- Пользовательская кнопка USER

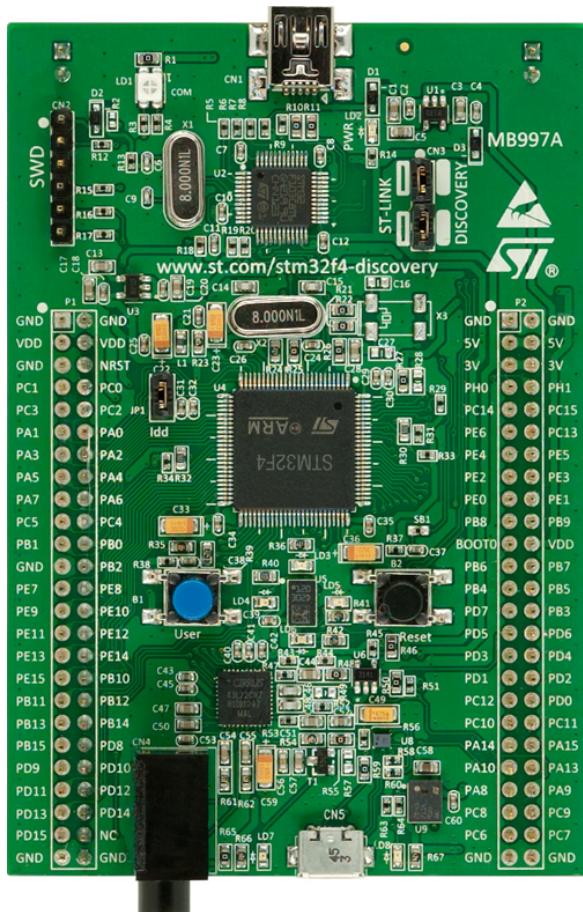
- Кнопка сброса RESET
- Контактные гребенки для всех выводов QFP64 для быстрого подключения и монтажа прототипа

Цена в розницу: 750 р.

<http://www.voltmaster.ru/cgi-bin/qwery.pl?id=127000573571&group=7000046>

## Глава 2

# STM32F4DISCOVERY /SRM32F407VGT6/



- микроконтроллер STM32F407VGT6 41 на базе 32-битного ядра Cortex-M4F, 1 MB Flash, 192 KB RAM в корпусе LQFP100
- встроенный ST-LINK/V2 с возможностью использования в режиме внешнего программатора (только с SWD коннектором)
- Разработана для питания как от USB, так и от внешнего источника 5 вольт
- Может обеспечить питание 3 и 5 вольт для внешних устройств
- LIS302DL, ST MEMS датчик движения, 3-осевой цифровой гироскоп
- MP45DT02, ST MEMS аудиодатчик, всенаправленный цифровой микрофон

- CS43L22, аудиоЖАП со встроенным аудиоусилителем класса D
- Восемь LEDs: LD1 (red/green) for USB communicationLD2 (red) for 3.3 V power onFour user LEDs, LD3 (orange), LD4 (green), LD5 (red) and LD6 (blue)2 USB OTG LEDs LD7 (green) VBus and LD8 (red) over-current
- Пользовательская кнопка USER
- Кнопка сброса RESET
- USB OTG FS with micro-AB connector
- Контактные гребенки для всех выводов LQFP100 ля быстрого подключения и мон-тажа прототипа

Цена в розницу: 1140 р.

<http://www.voltmaster.ru/cgi-bin/qwery.pl?id=127000854271&group=7000046>

## Глава 3

### ARMatura /STM32F417IGT/

## Глава 4

# PION /STM32F100C4T6B/

Модуль PION предназначен для мелких задач управления, первичной обработки данных,стыковки с устройствами измерения и исполнительными устройствами, т.е. для тех задач, для которых ранее использовались микроконтроллеры Atmel AVR8.

процессор	STM32F100C4T6B	41.1
ROM		16K
RAM		4K
шина	AUTObus	
интерфейсы	UART	1
	SPI	1
	AIЦП	10x12b
	ЦАП	2x12b
буфер	Parallel Flash	64K

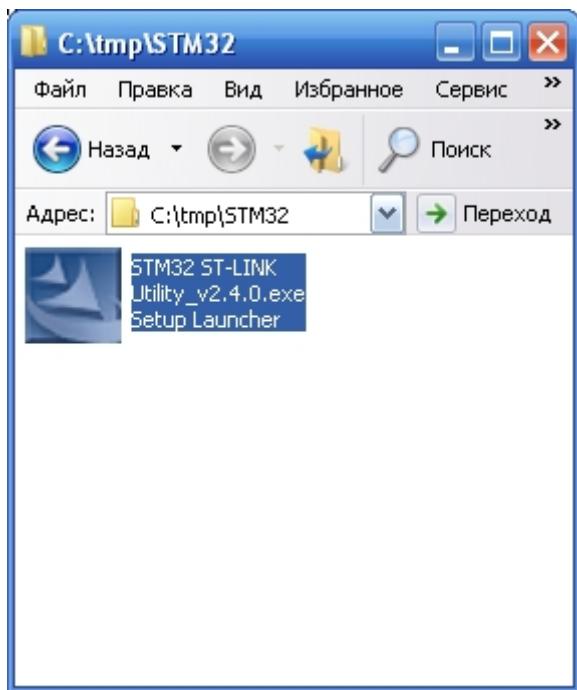
# **Часть III**

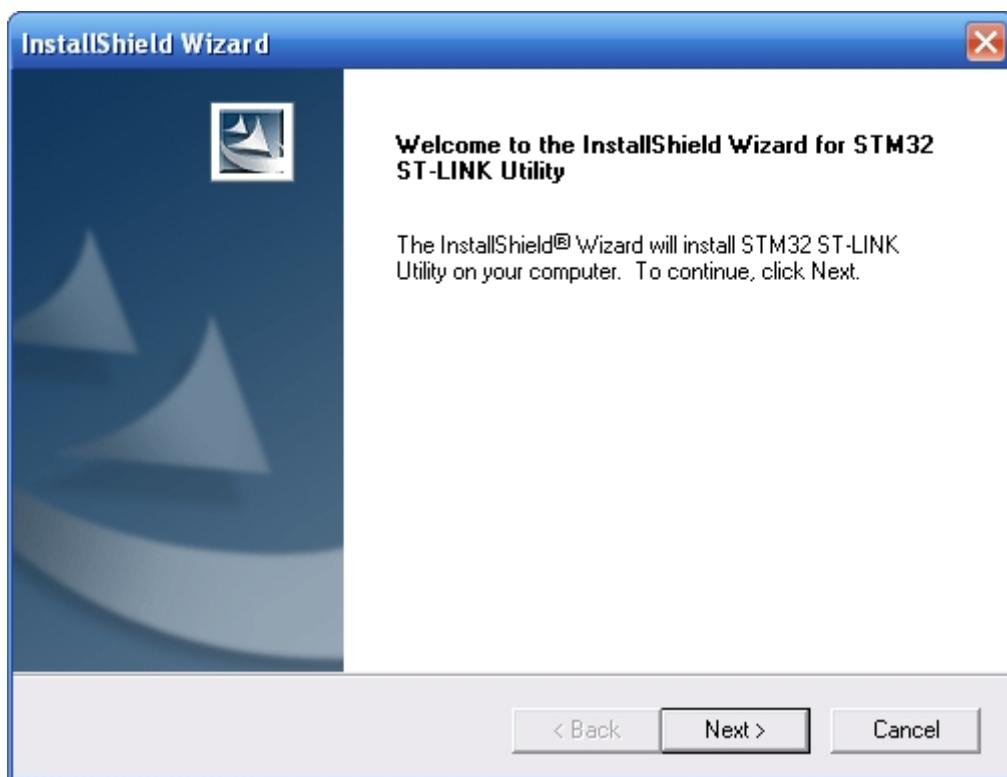
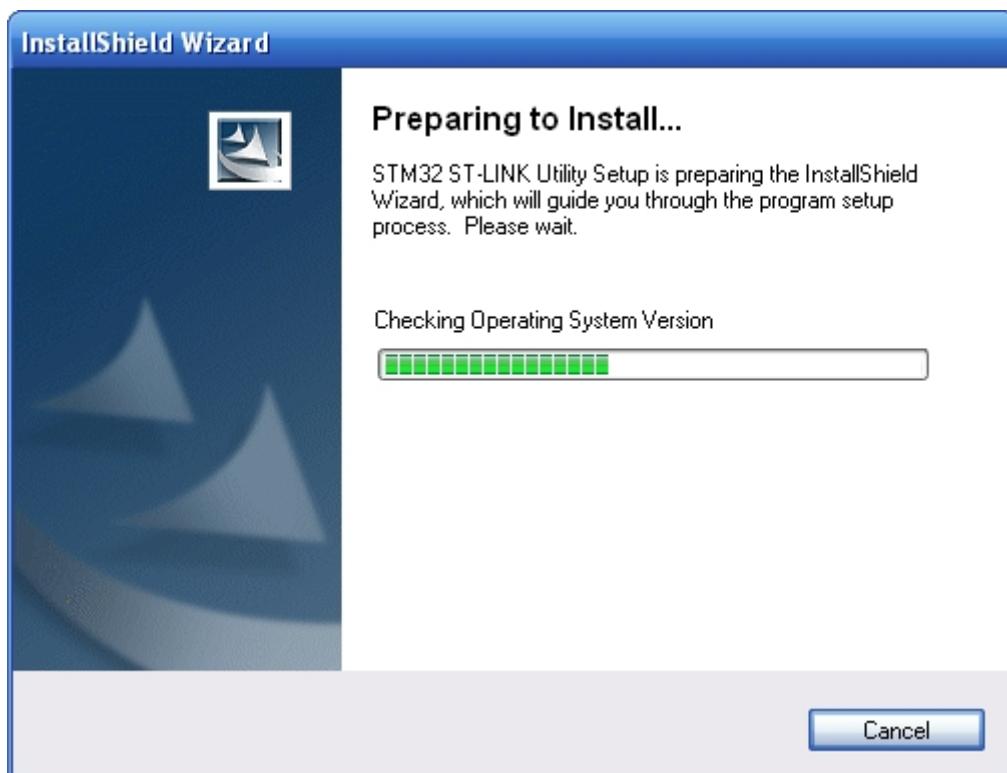
## **Первые шаги**

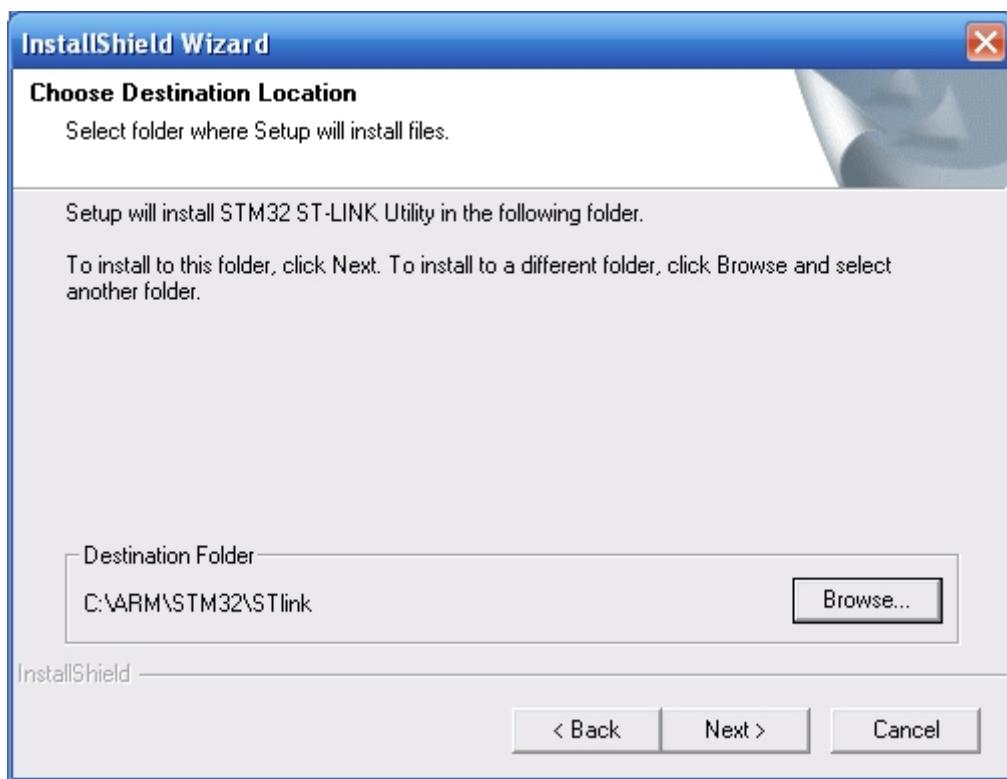
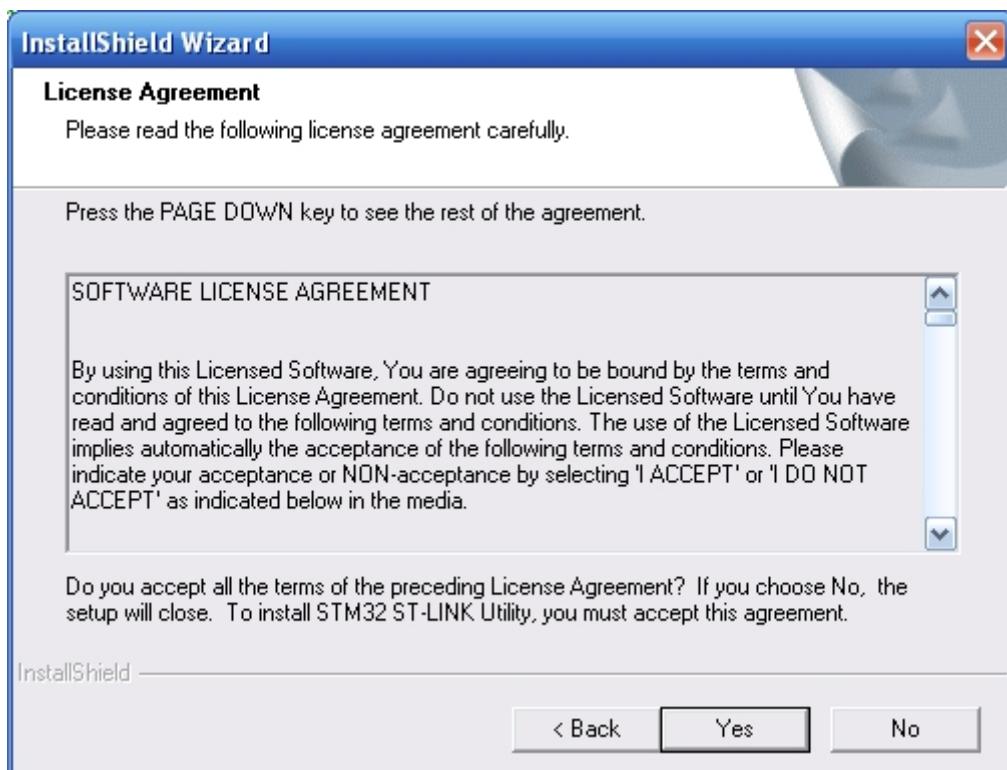
## Глава 5

### Установка пакета ПО STlink

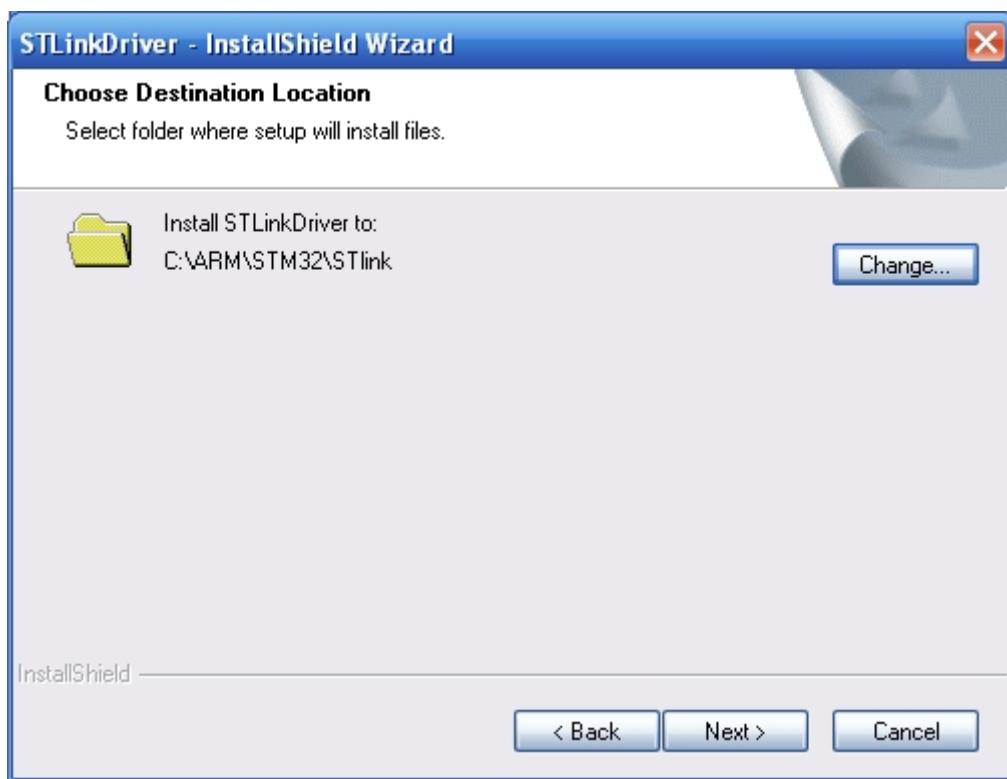
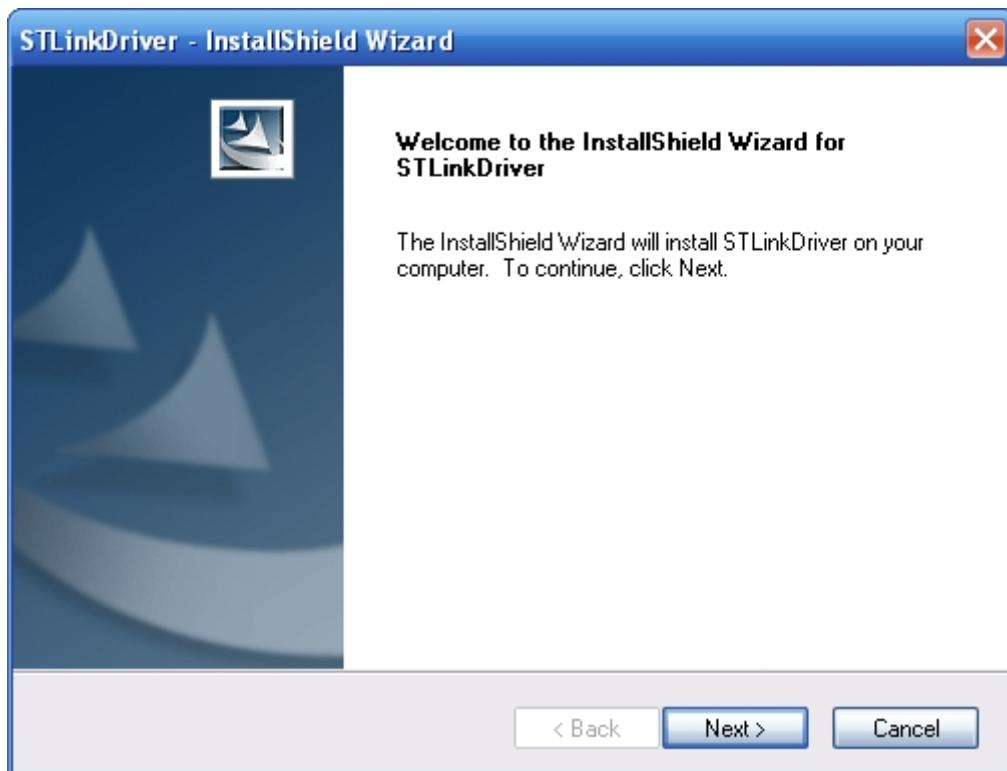
Для работы с чипами STM32 используя программаторы STlink, встроенные на демоплаты Discovery необходимо установить пакет STSW-LINK004 (STM32 ST-link Utility и драйвер), скачав его с <http://www.st.com/web/en/catalog/tools/PF258168>

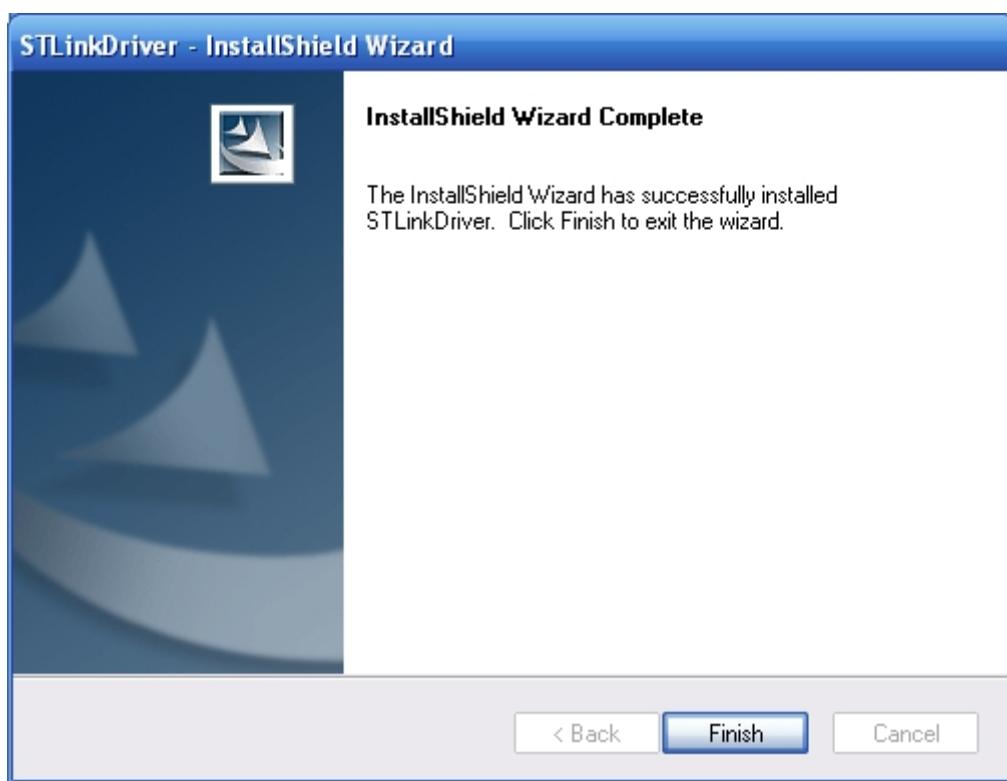
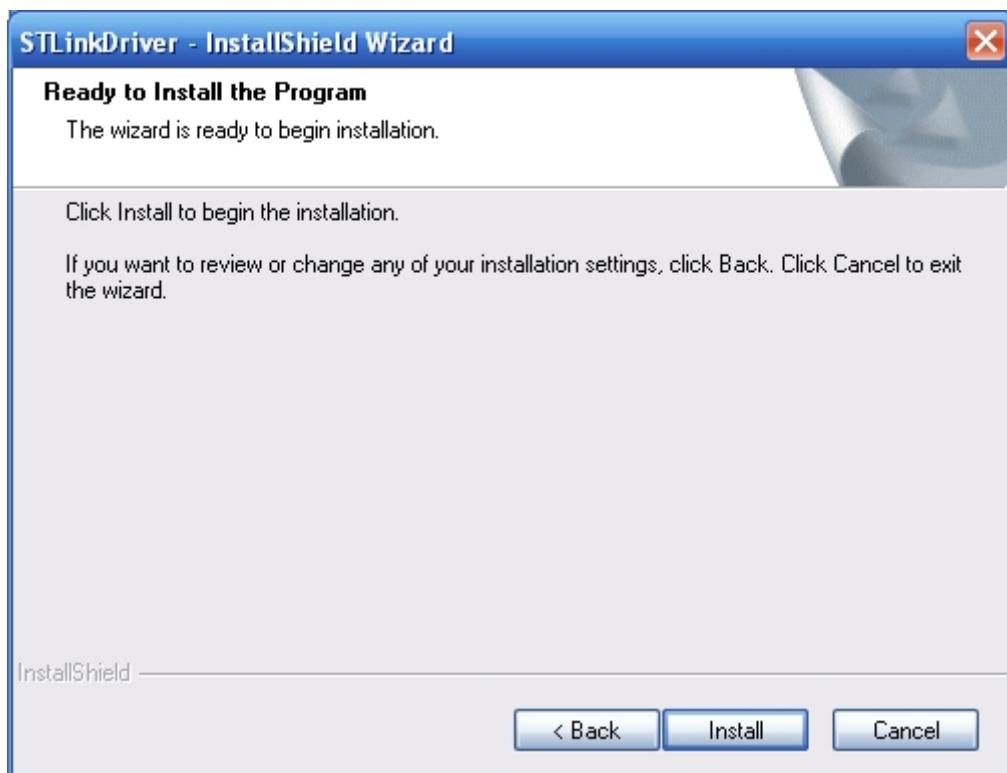


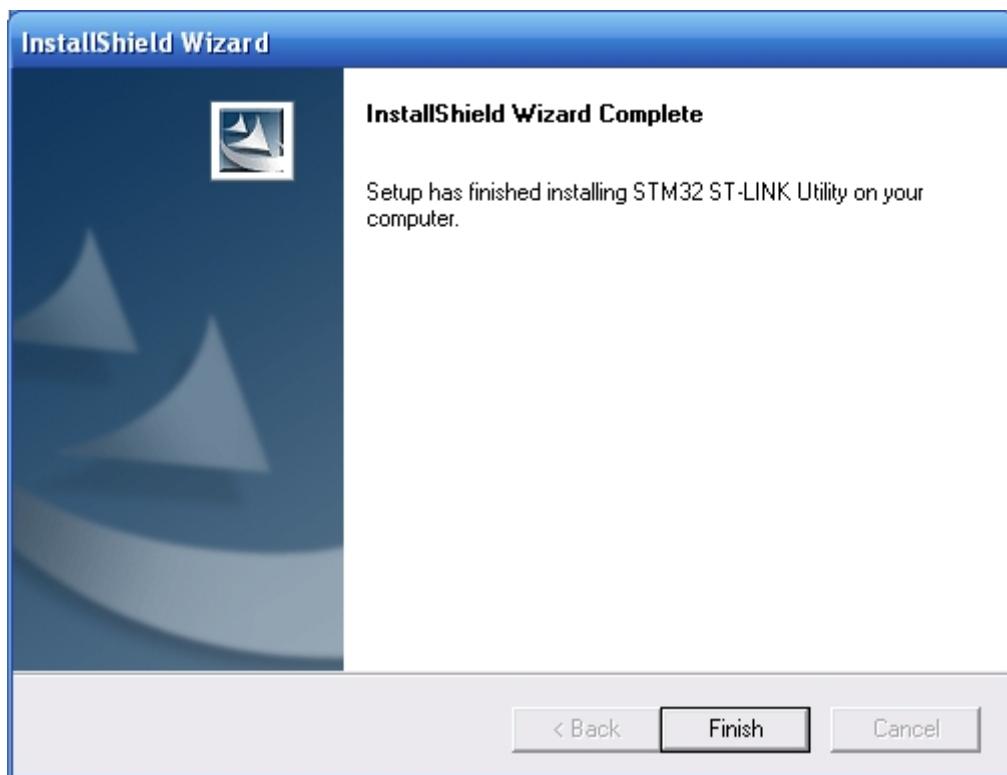




Пакет драйвера входит в пакет STSW-LINK004, и запускается автоматически







# Глава 6

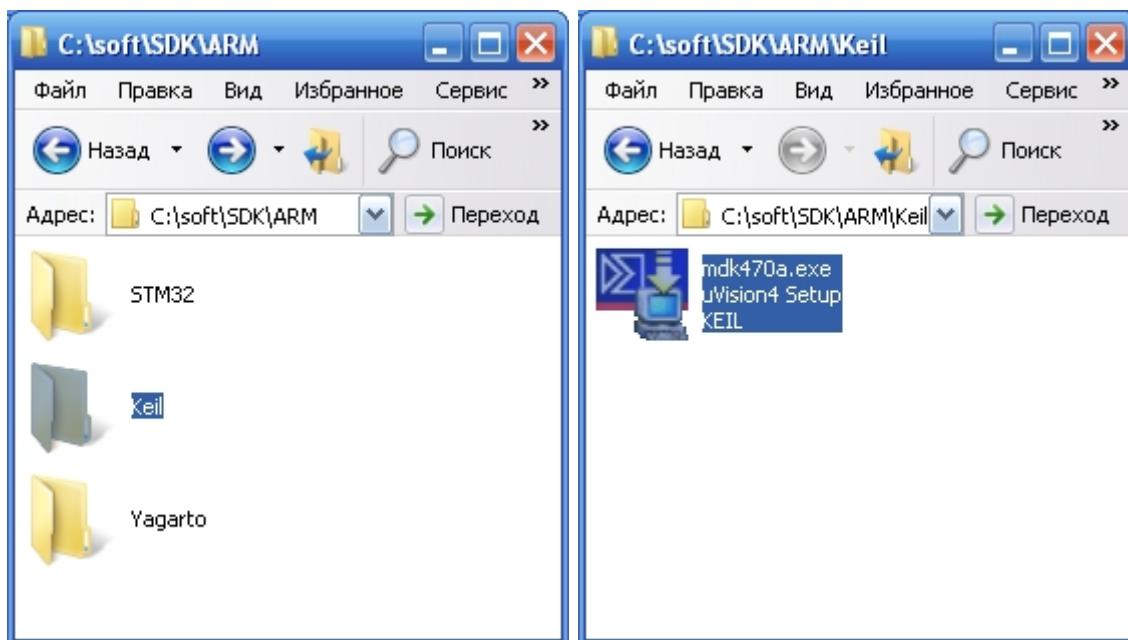
## Установка Keil MDK-ARM

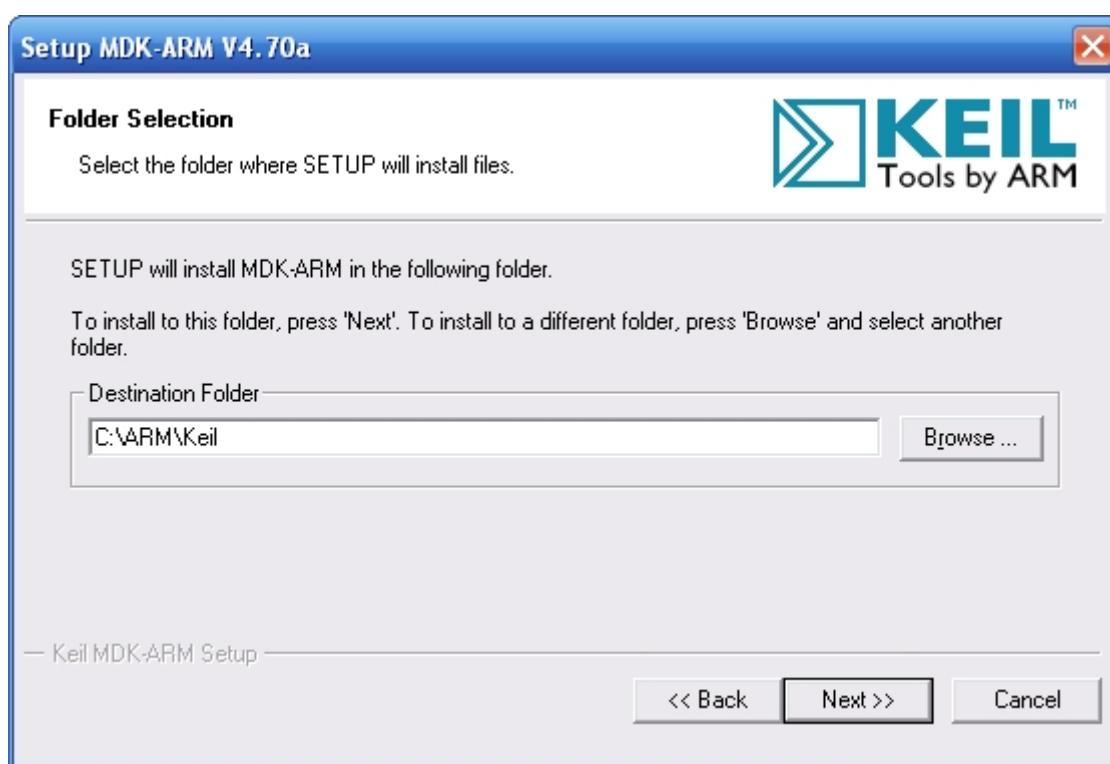
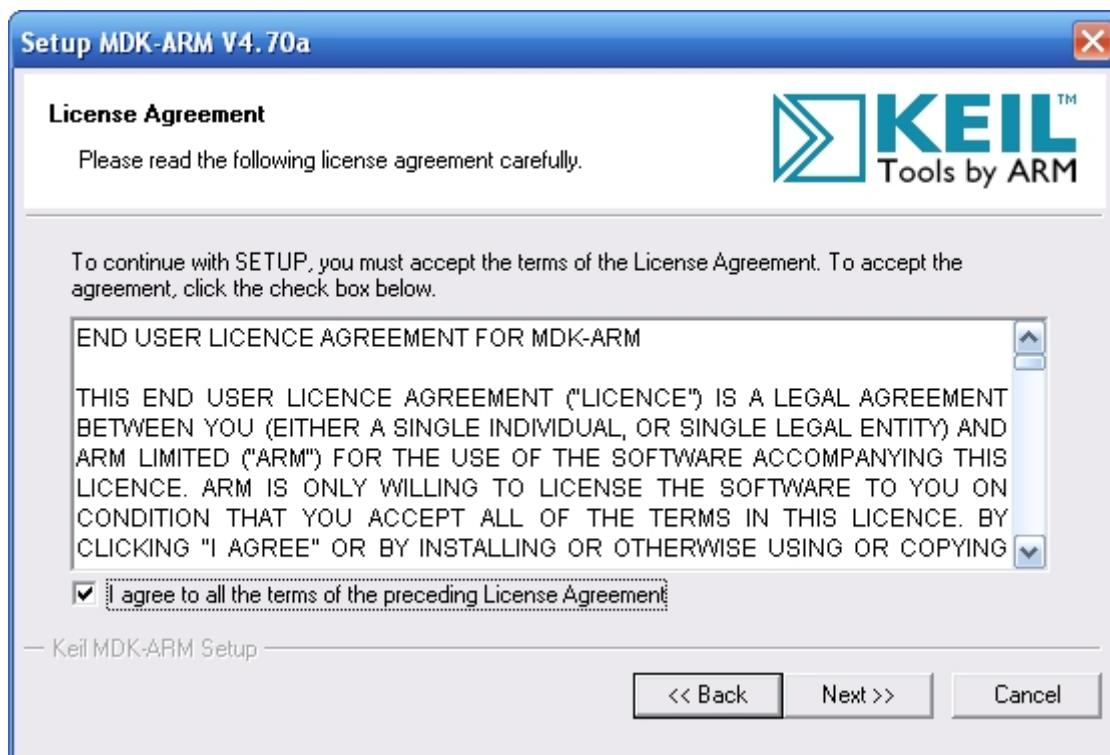


Для начала освоения программирования для ARM рекомендуем использовать бесплатный пакет от Keil: <http://www.keil.com/arm/mdk.asp> — ограничения бесплатной версии в 32К кода вполне достаточно для начального освоения программирования под процессоры семейства Cortex-Mx, а затем уже можно переползать на открытое ПО: GNU toolchain 10.1, Eclipse 11.1 и Linux XIII.

Процесс установки и первоначальной настройки описан в 9. В этом разделе будет рассмотрен только процесс установки и начальной настройки, подробно о пакете Keil MDK-ARM см.

Качаем пакет с официального сайта, заполнив анкету: <https://www.keil.com/demo/eval/arm.htm>.

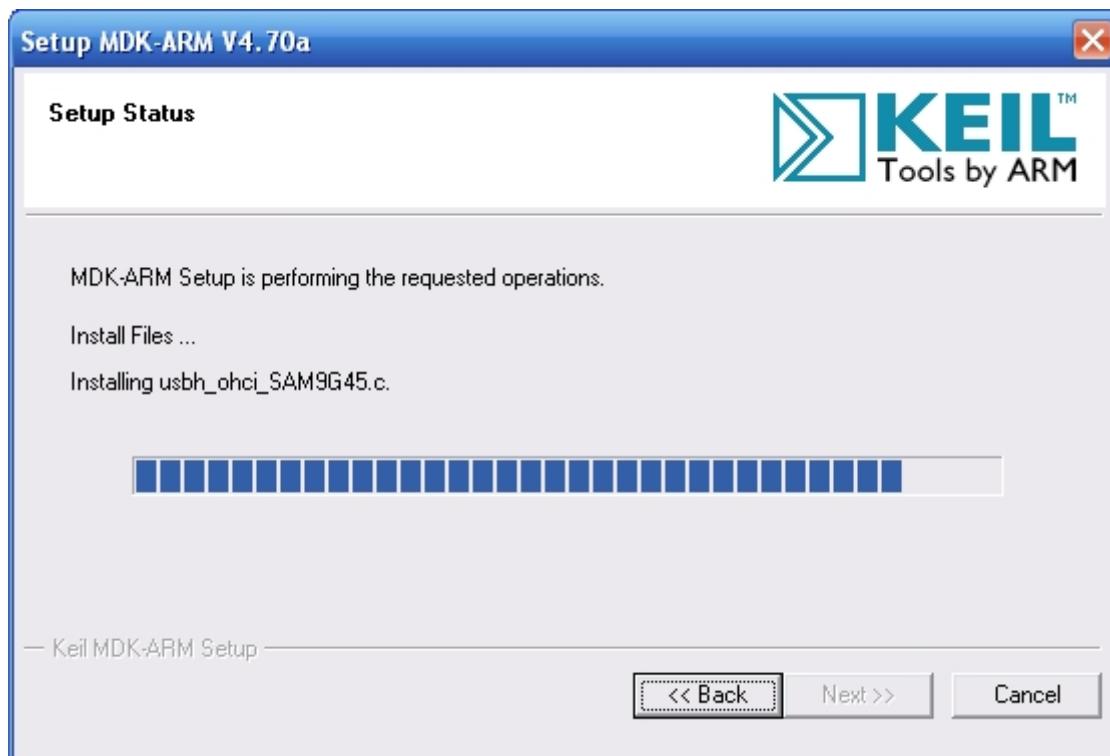


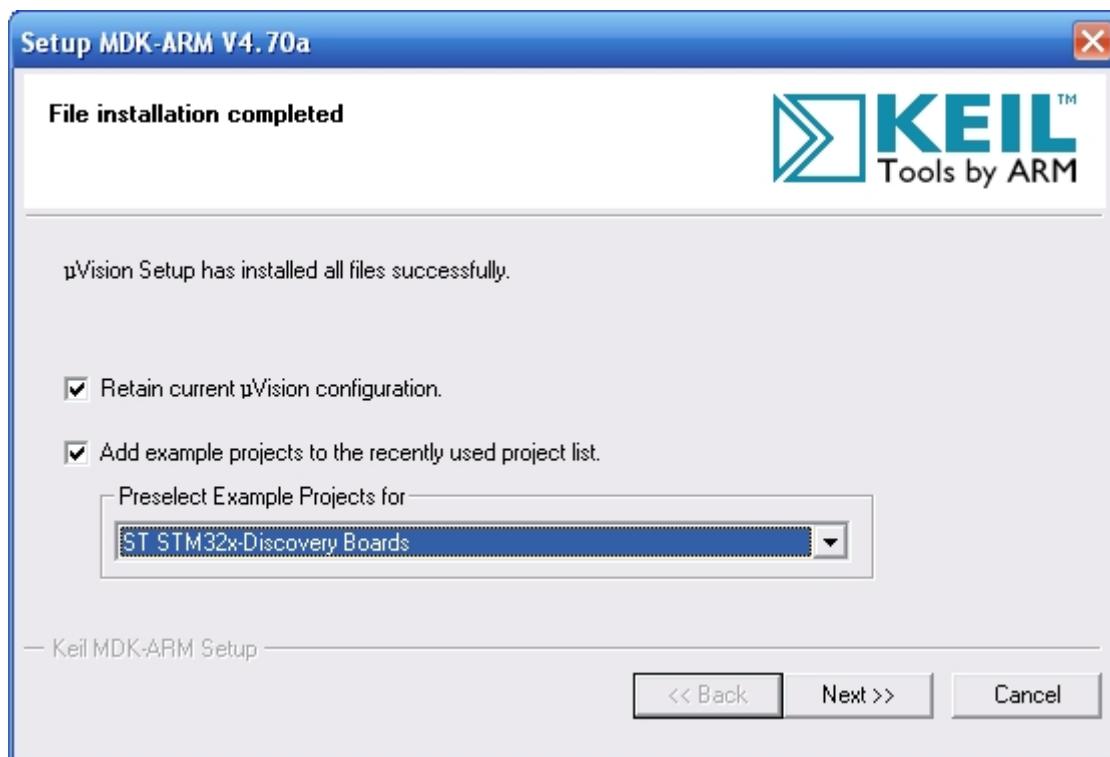


Путь установки пакета

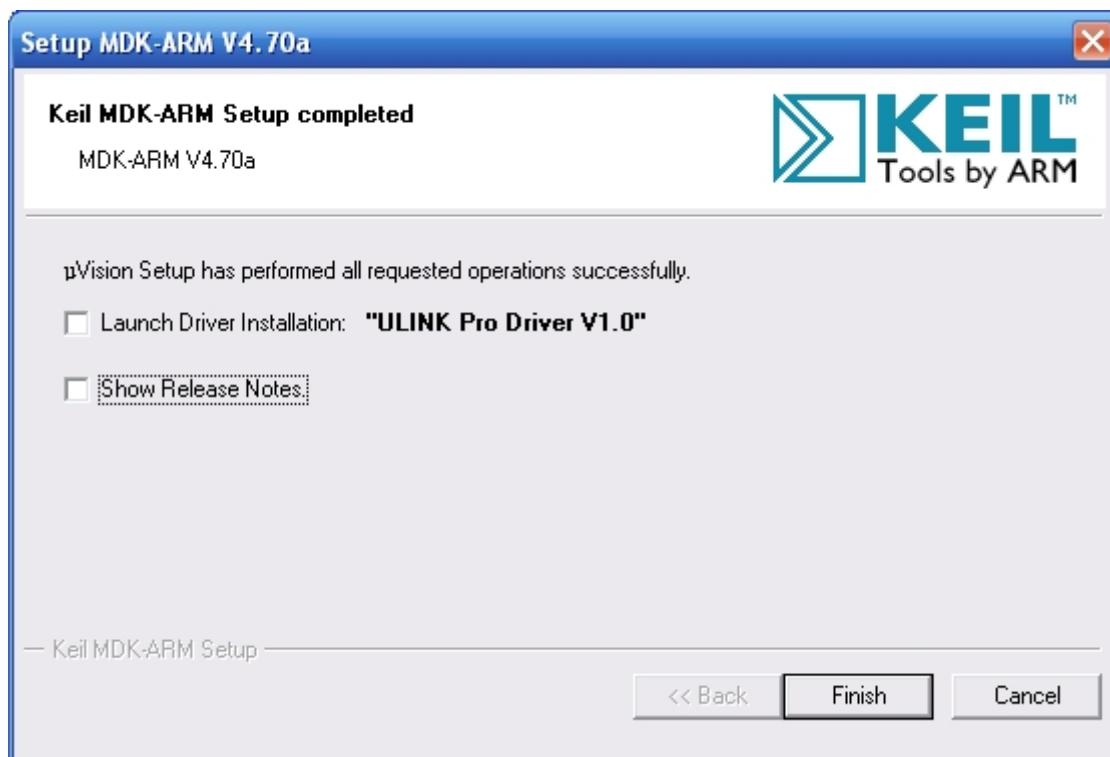


Личные данные: имя, название компании или hobbit, адрес электронной почты.





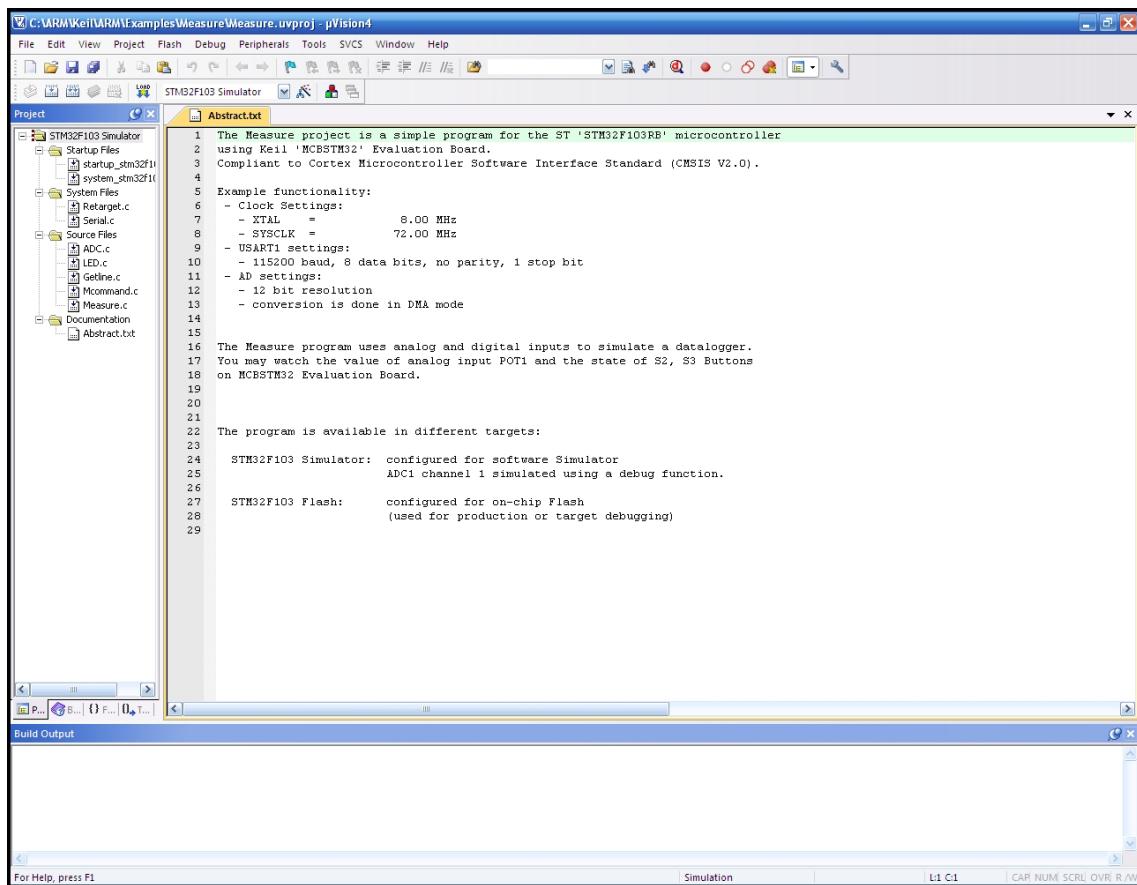
Укажите какие примеры кода добавить в список recently used project list: для работы с другими типами микропроцессоров выберете соответствующий раздел, или оставьте Simulation Hardware по умолчанию.



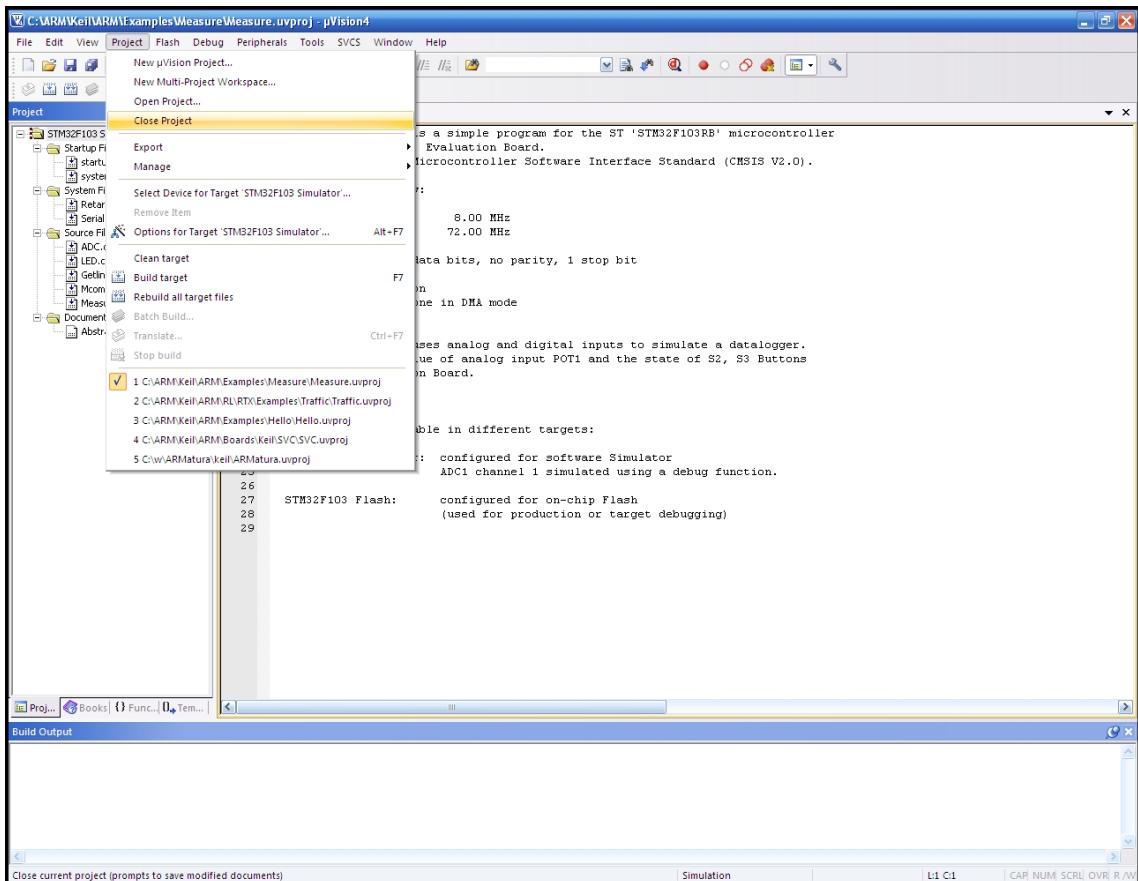
Снять установку драйвера программатора ULINK (если у вас его нет) и вывод текстового файла с последними изменениями Keil.



После запуска открывается проект по умолчанию, настроенный для программного симулятора STM32F103.



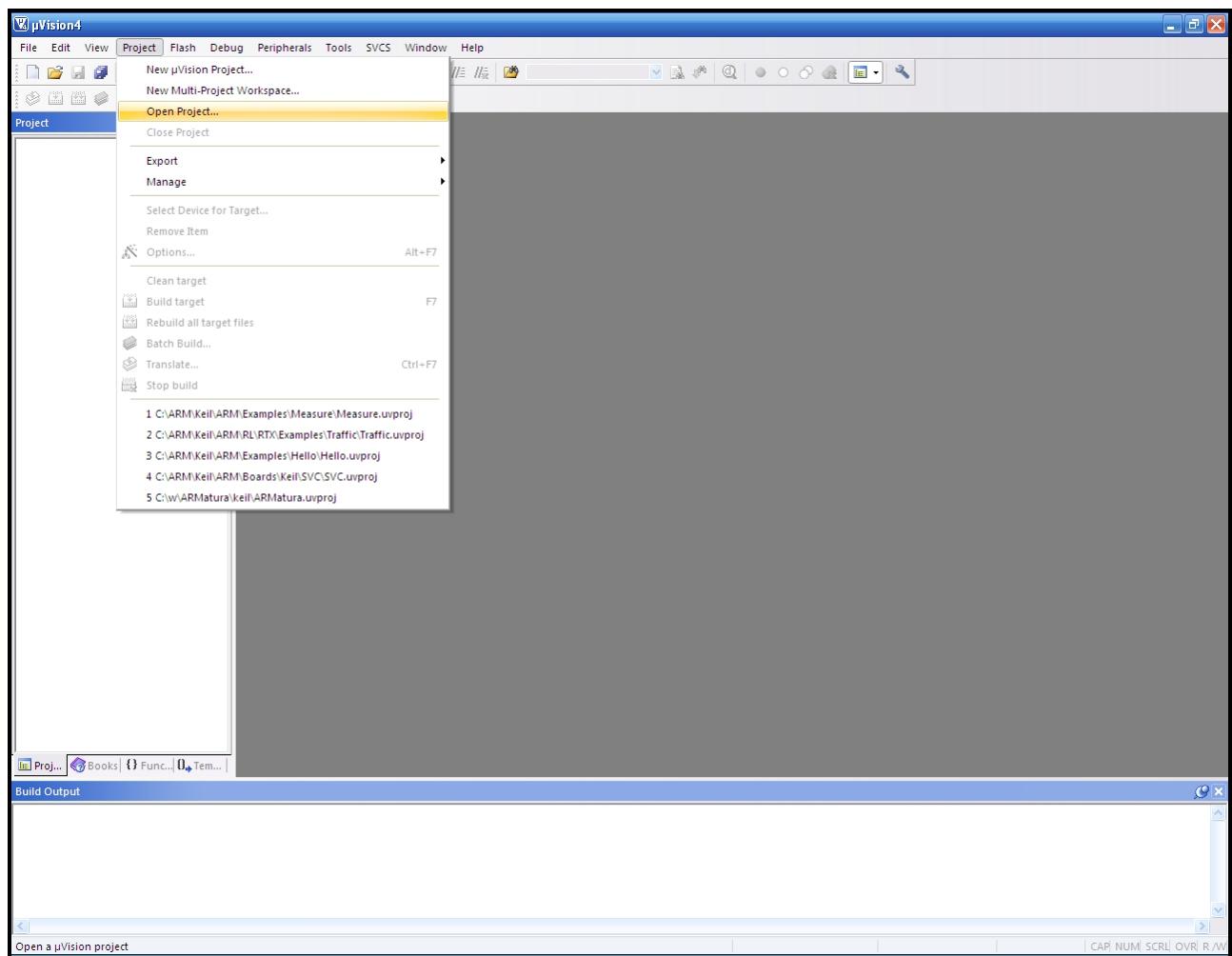
Нужно его закрыть,

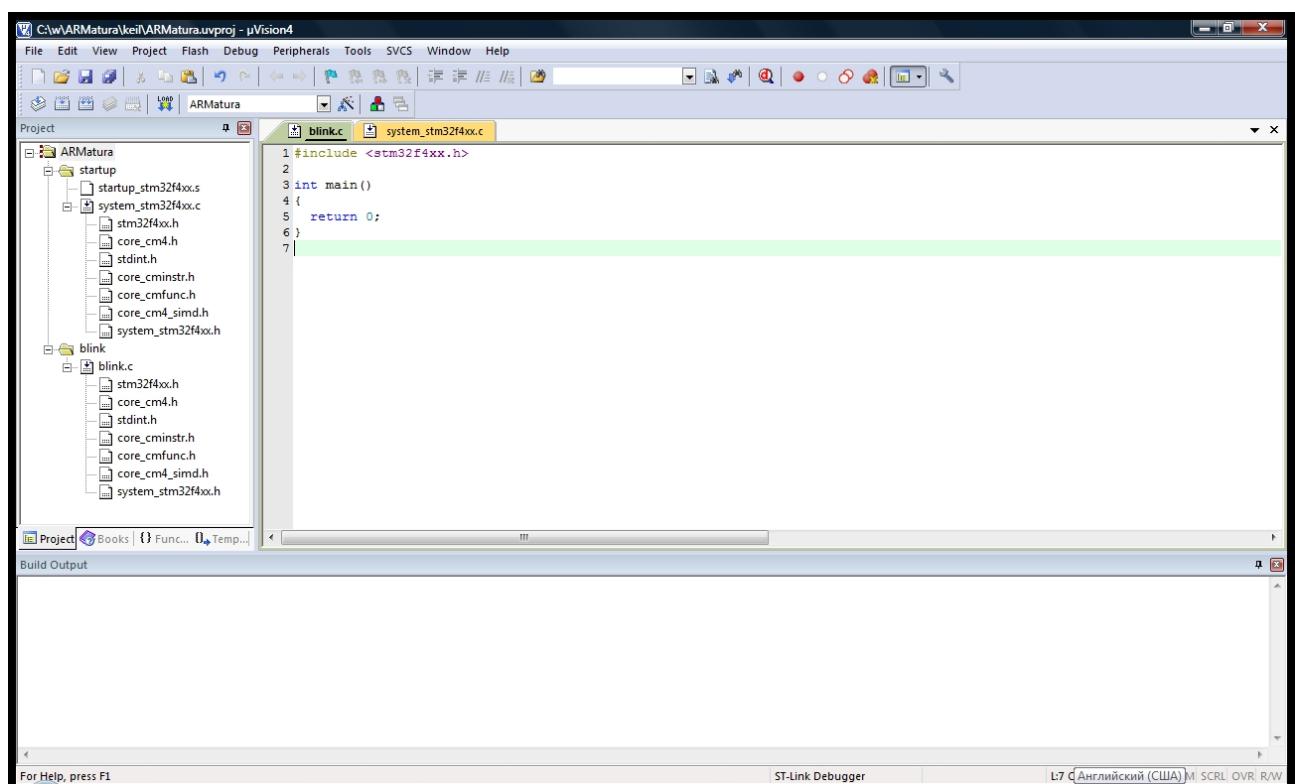
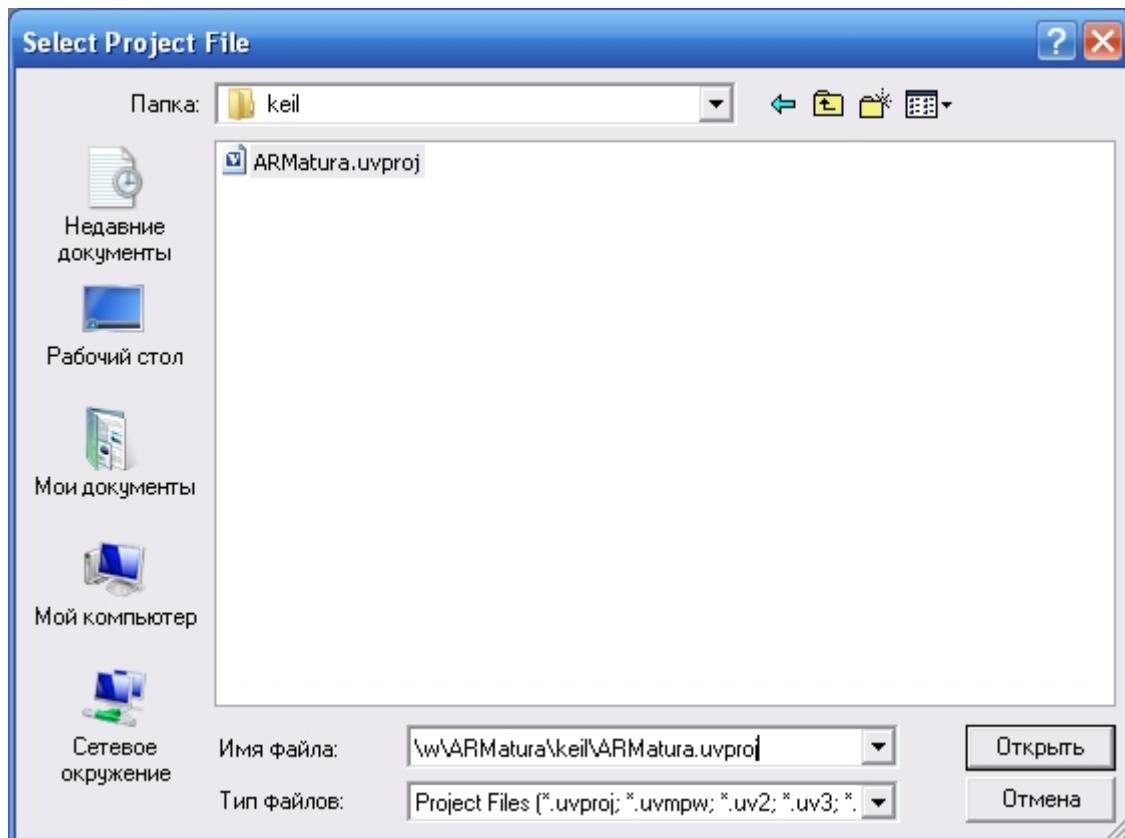


и открыть первый самый простой проект:

# Глава 7

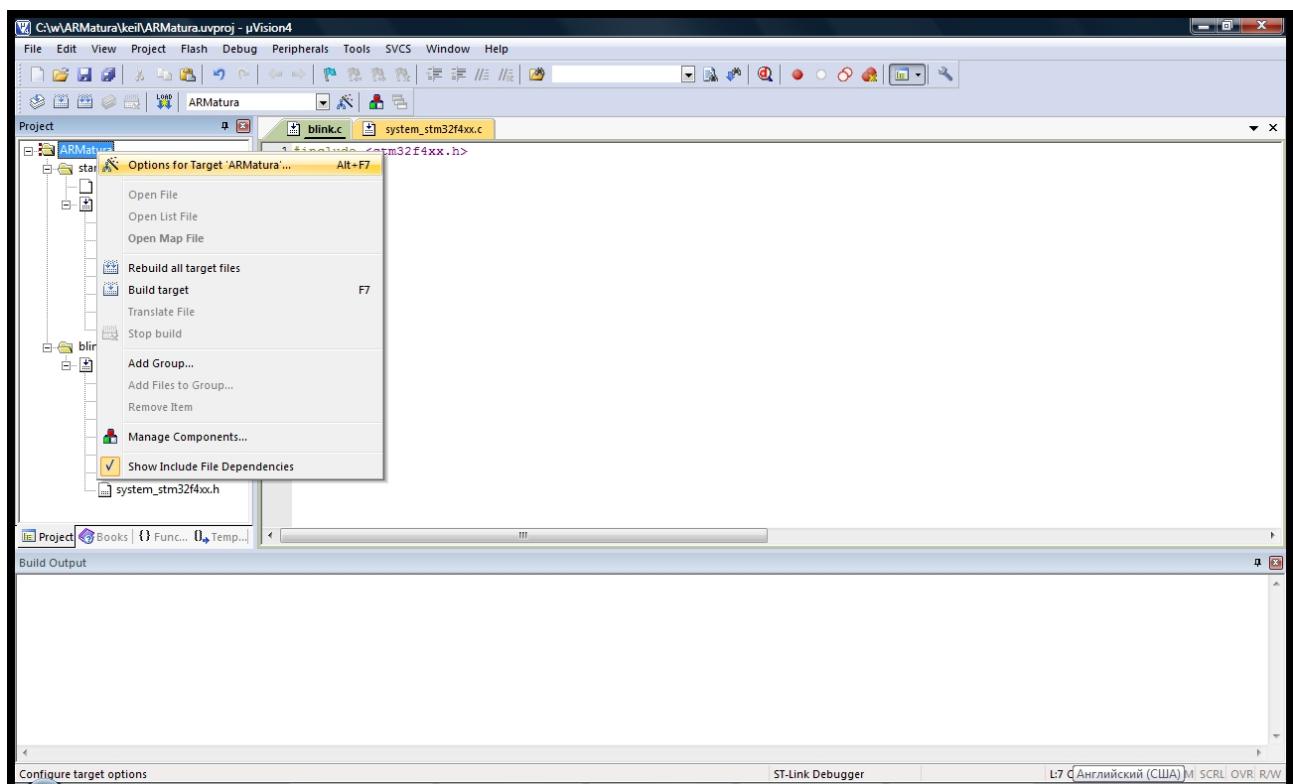
## Первый проект: blink



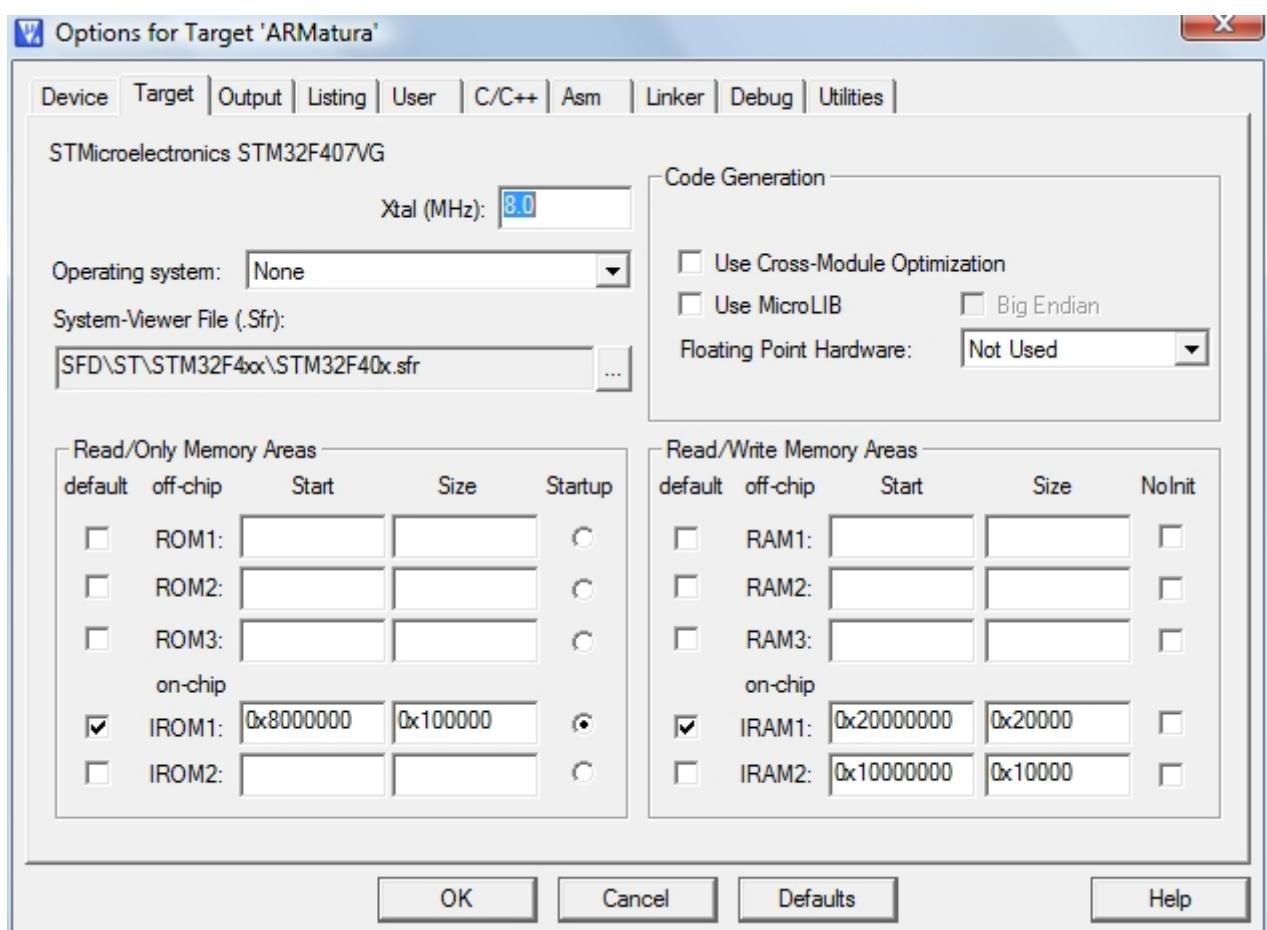


## 7.1 Настройки проекта в Keil

Настройки проекта вызываются из меню **Project** **> Options for Target 'ARMatura' ...**, комбинацией клавиш **Alt + F7**, или выбором аналогичной опции из контекстного меню, вызываемого щелчком правой кнопкой мыши на корне дерева проекта **ARMatura** в левом окне **Project**.

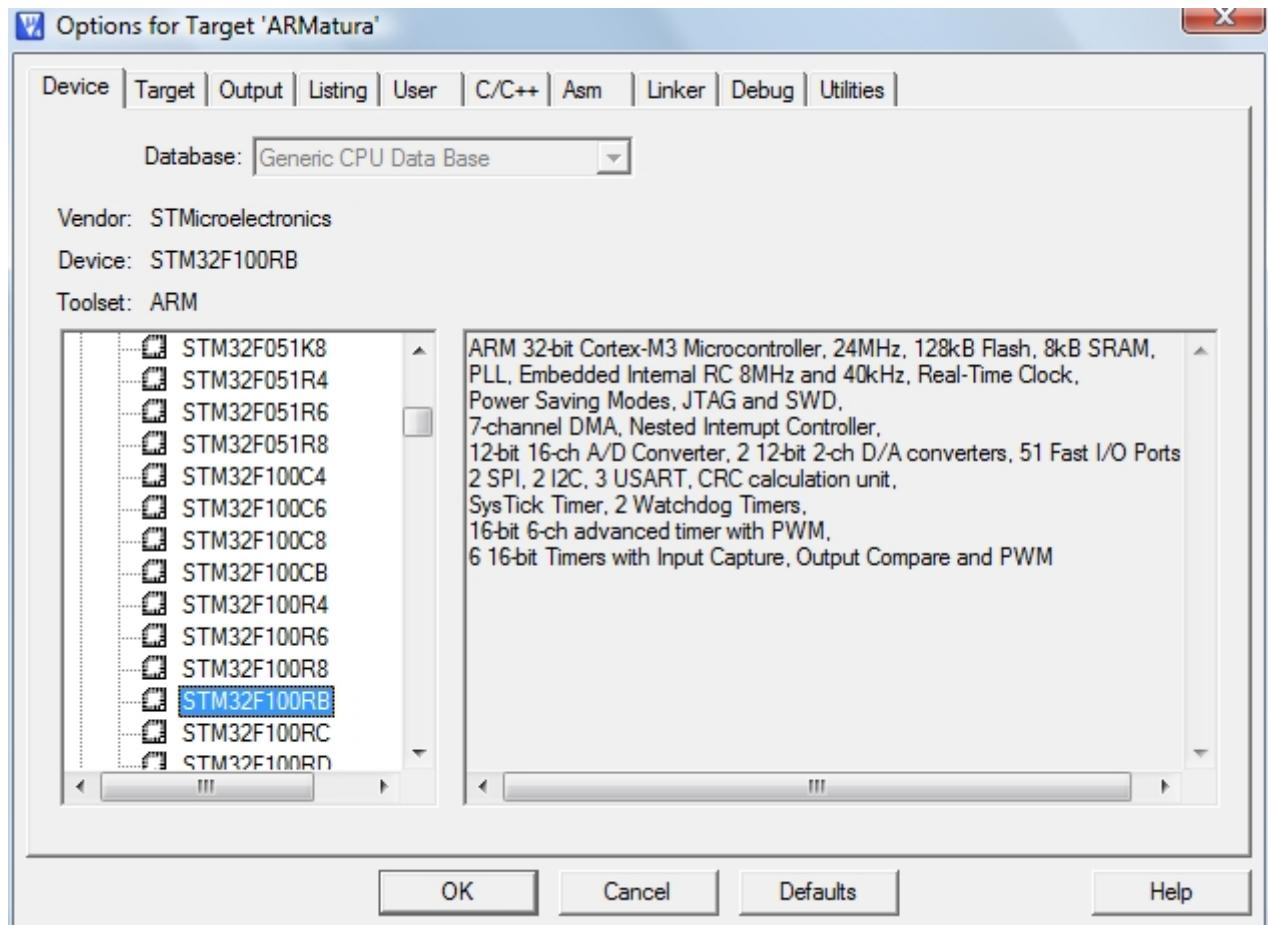


По умолчанию открывается вкладка **Target**:

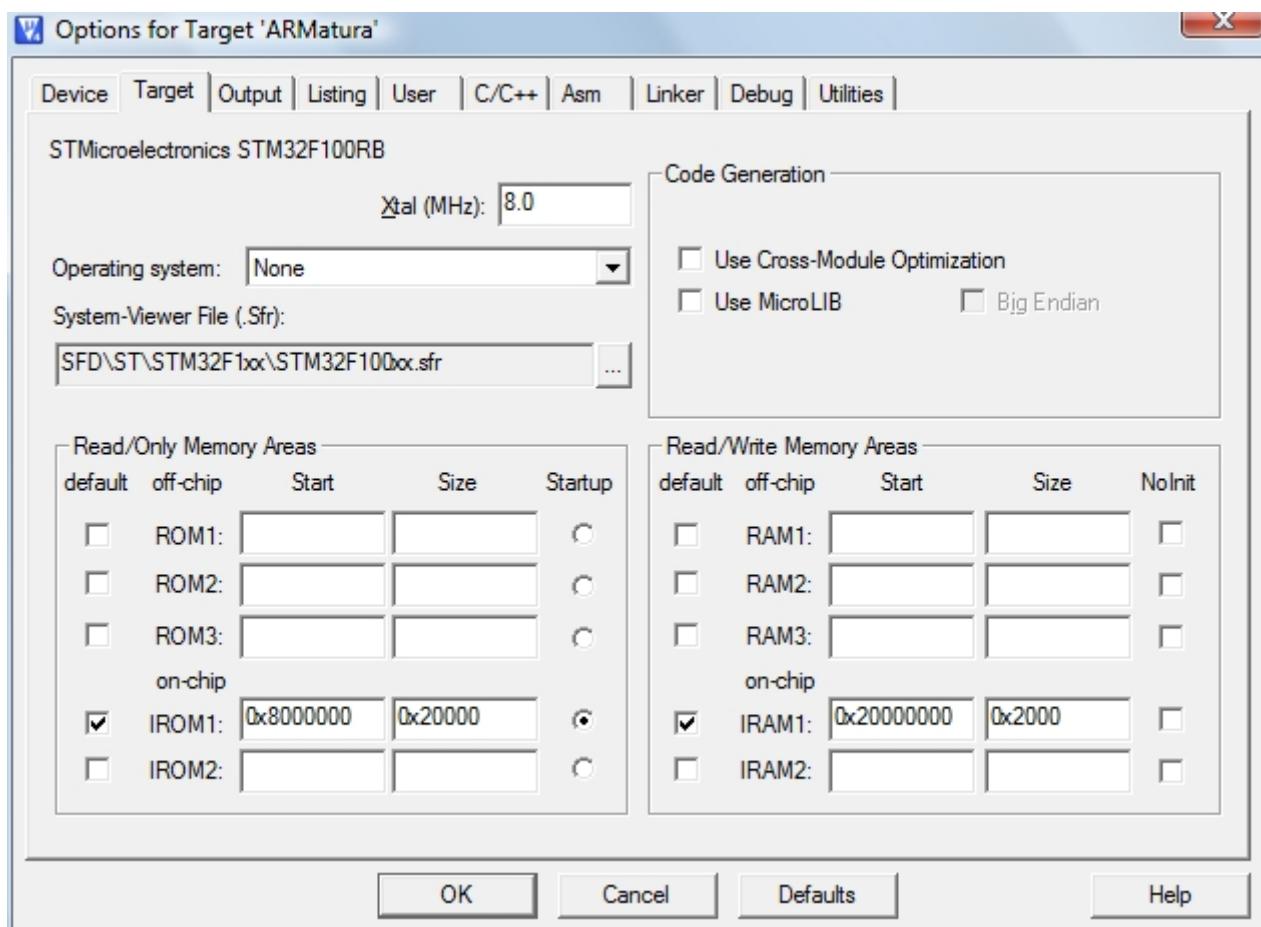


Xtal (MHz)	8.0	текущий процессор частота внешнего кварца на платах Discovery обычно стоит 8
Operating system:	None	или OCPB Keil RTX
Floating Point Hardware:	Not Used	использовать libc от Keil для Cortex-M4 доступен аппаратный FPU
		На этой вкладке также прописывается карта встроенной и внешней памяти Flash/SRAM.

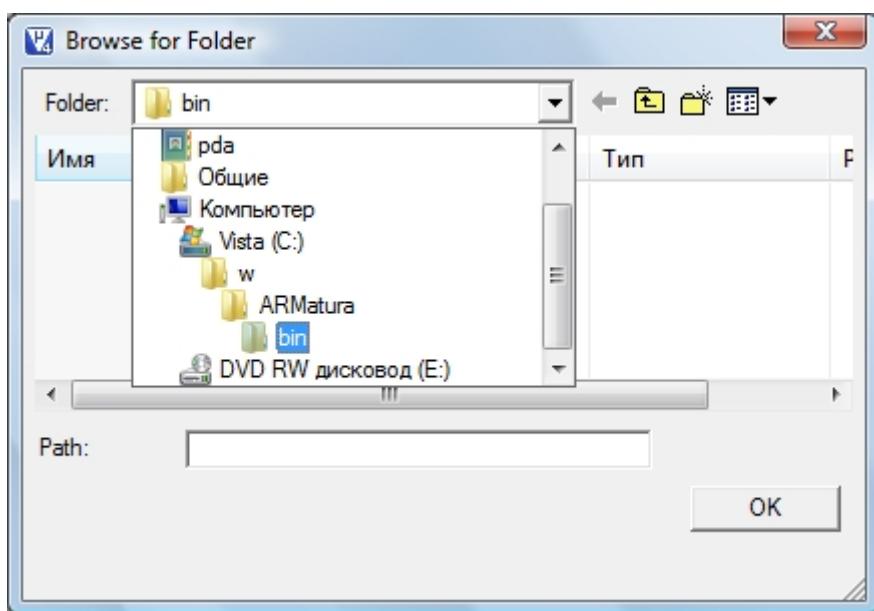
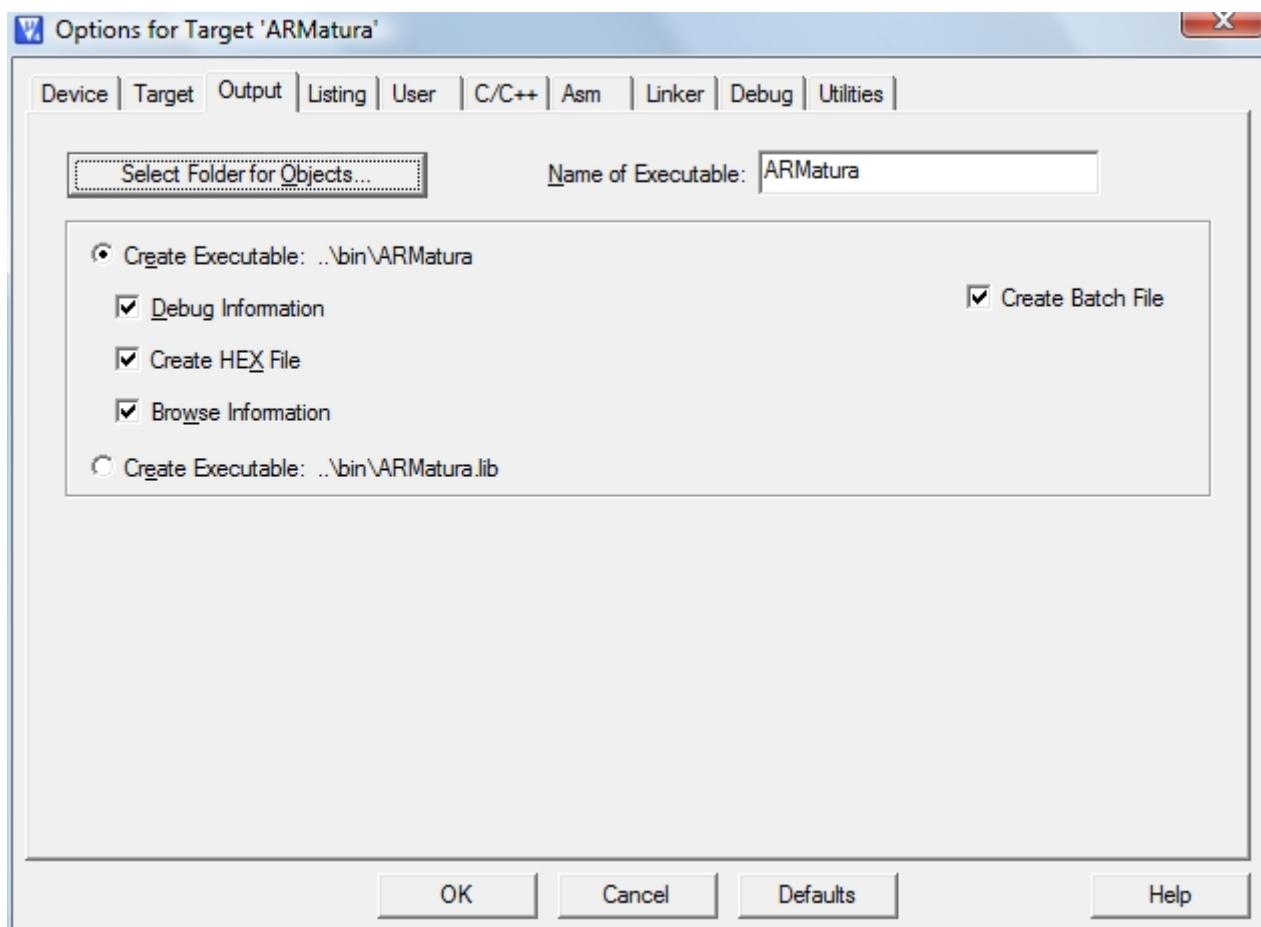
При необходимости собрать проект под другой процессор открываем вкладку **Device**, переконфигурируем проект под другую плату – STM32VLDISCOVERY 1:



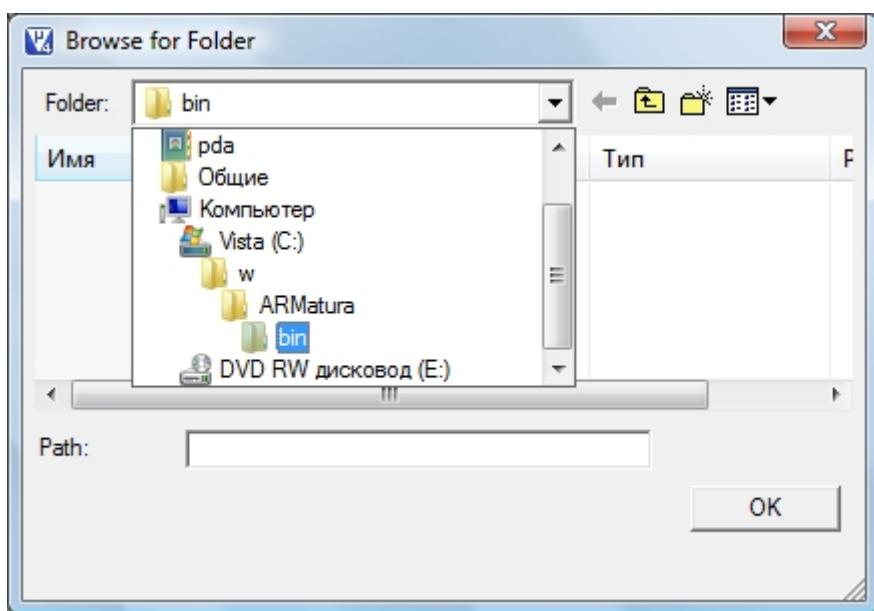
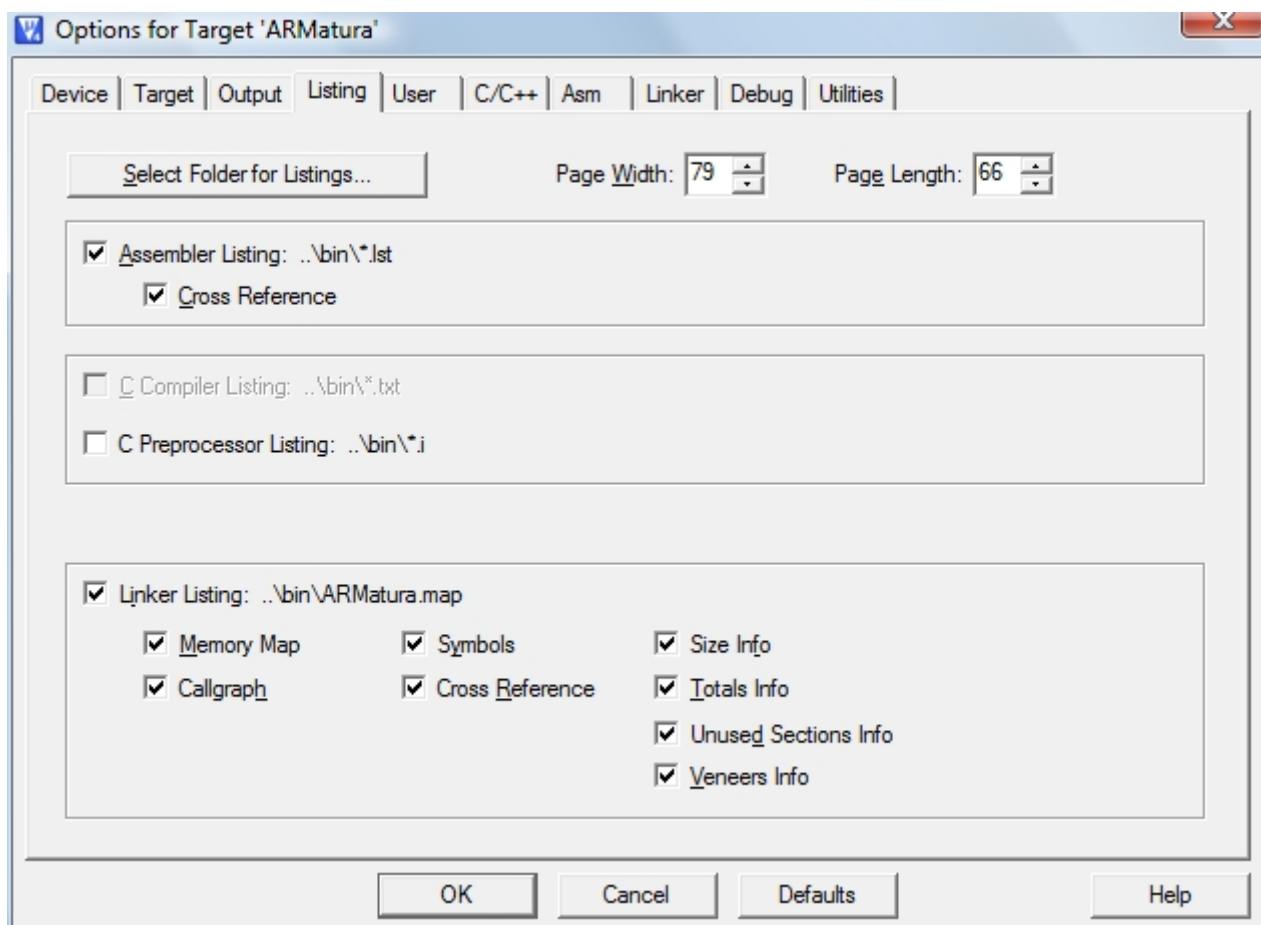
Обратите внимание что на вкладке **Target** изменился чип и настройки памяти:



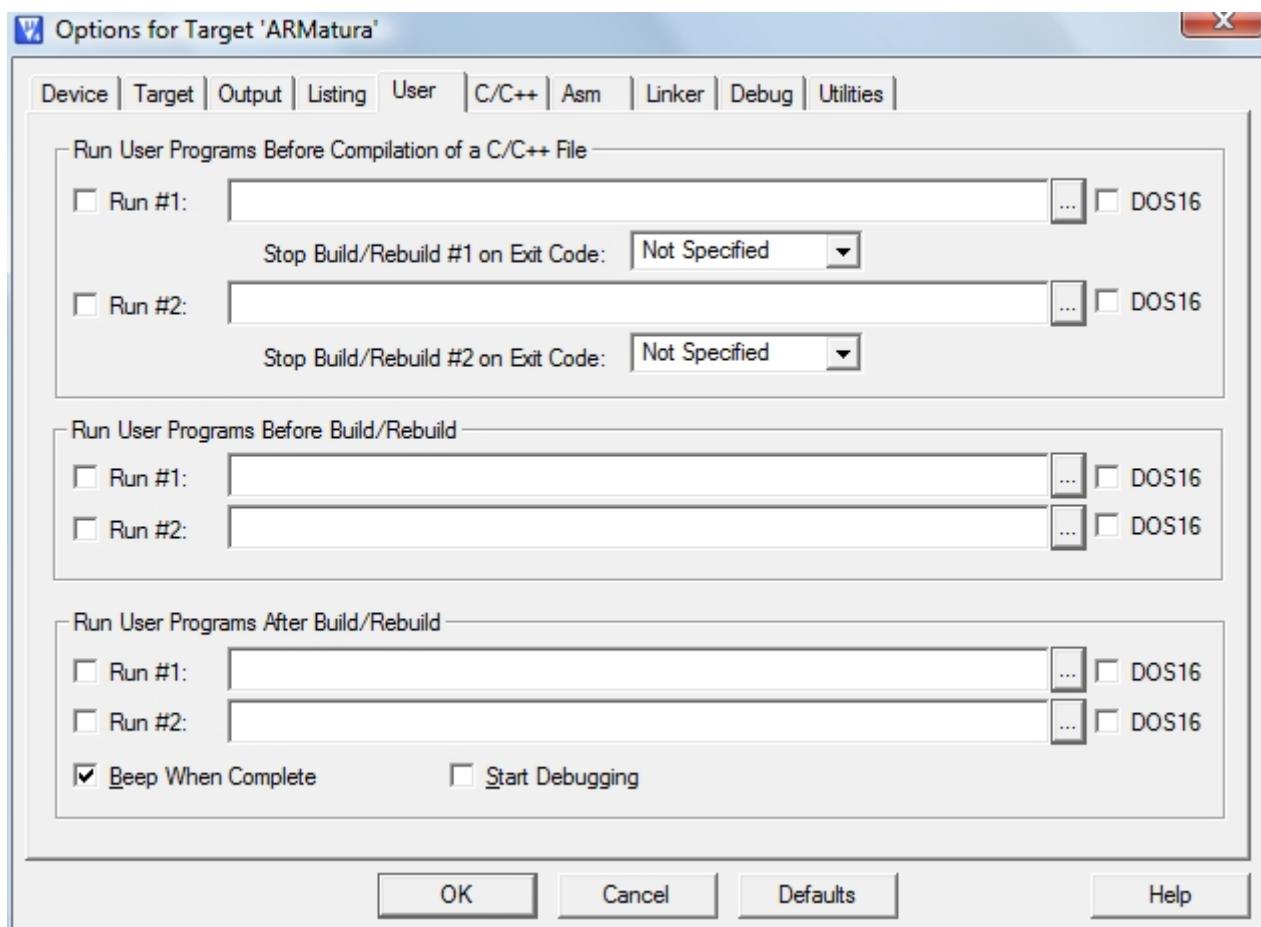
**Output** Настройки выходных файлов: каталог для бинарных файлов прошивки = firmware, при необходимости можно использовать .hex файл прошивки, имя файла прошивки



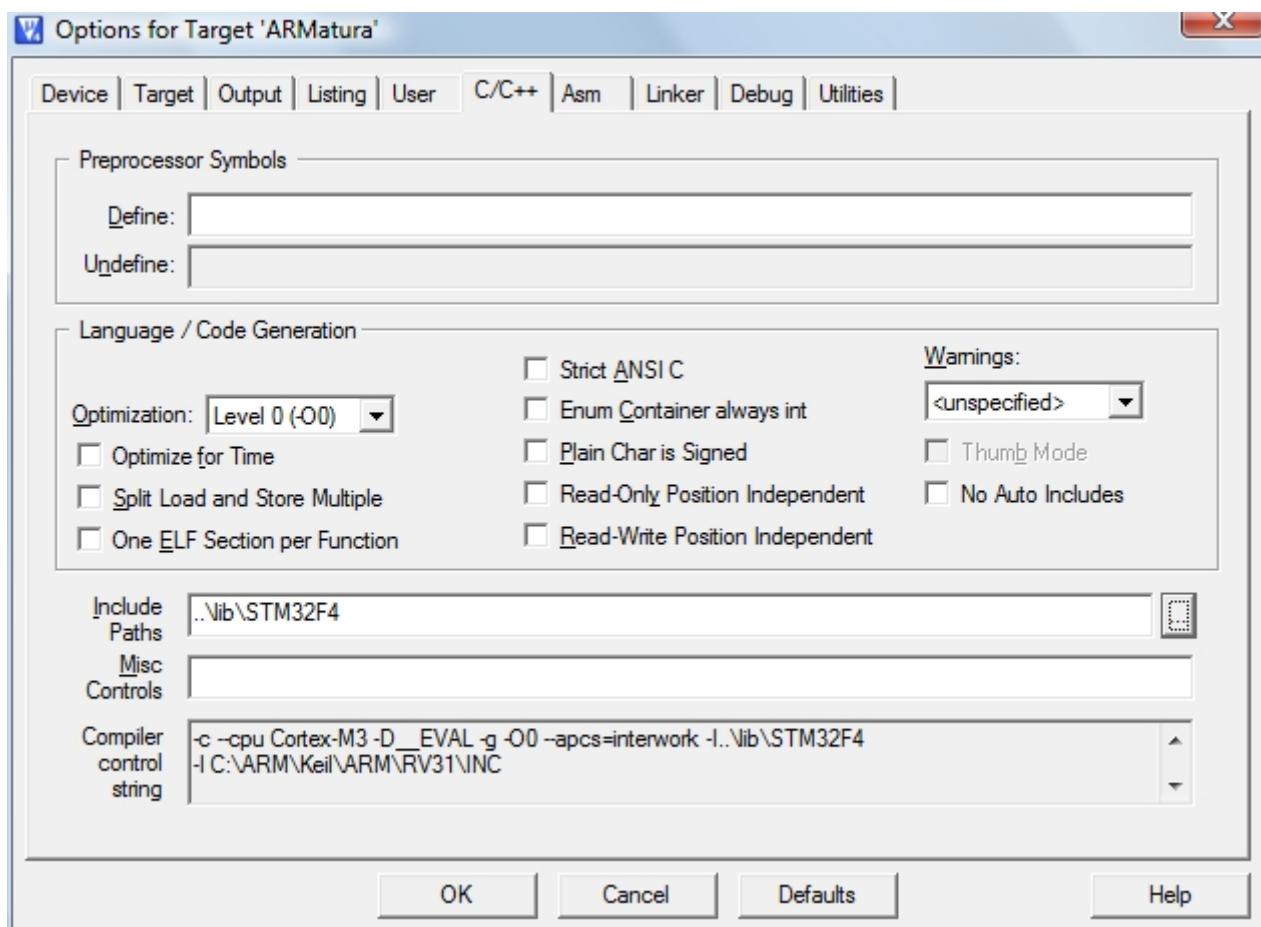
[Listing] Настройки генерации ассемблерных листингов в тот же каталог с прошивкой



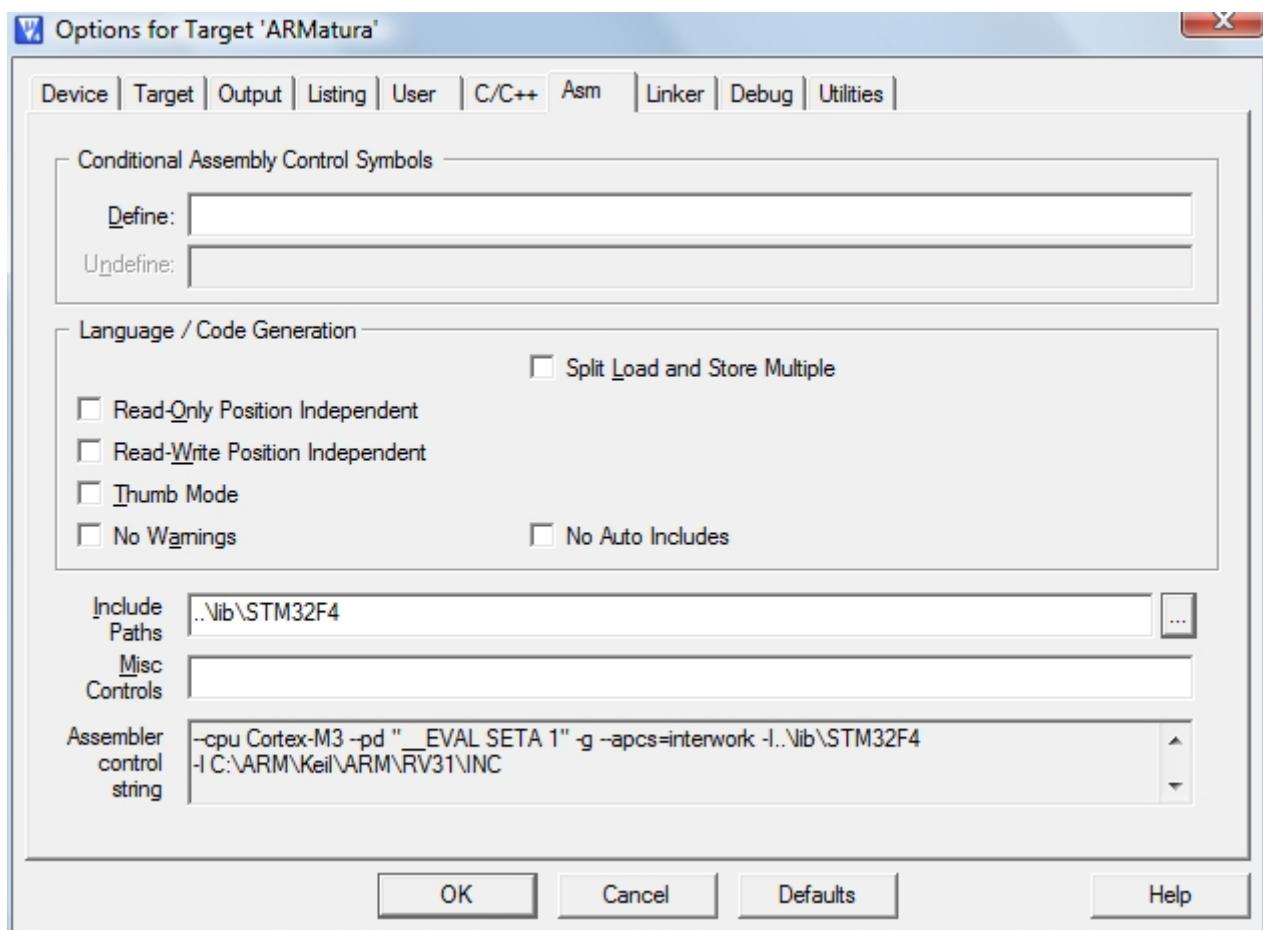
**User** Запуск пользовательских программ во время сборки проекта: пока не используем



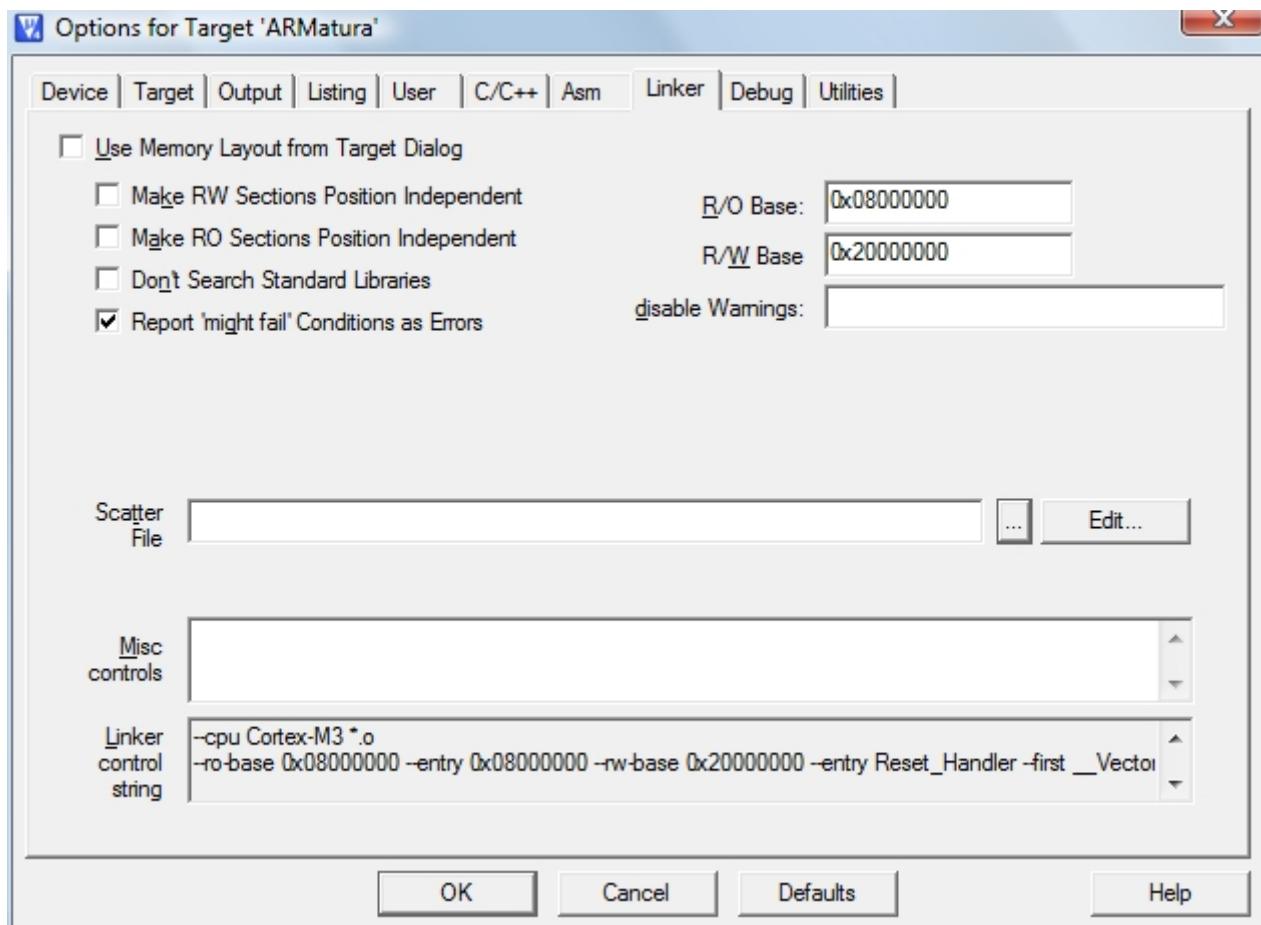
**C/C++** Настройки компилятора  $C^{++}$ : опции оптимизации, каталоги библиотек и включаемых .h файлов, в нижнем поле прописывается полная командная строка вызова компилятора, которая может вам в дальнейшем понадобится, если потребуется компилировать без использования Keil IDE.



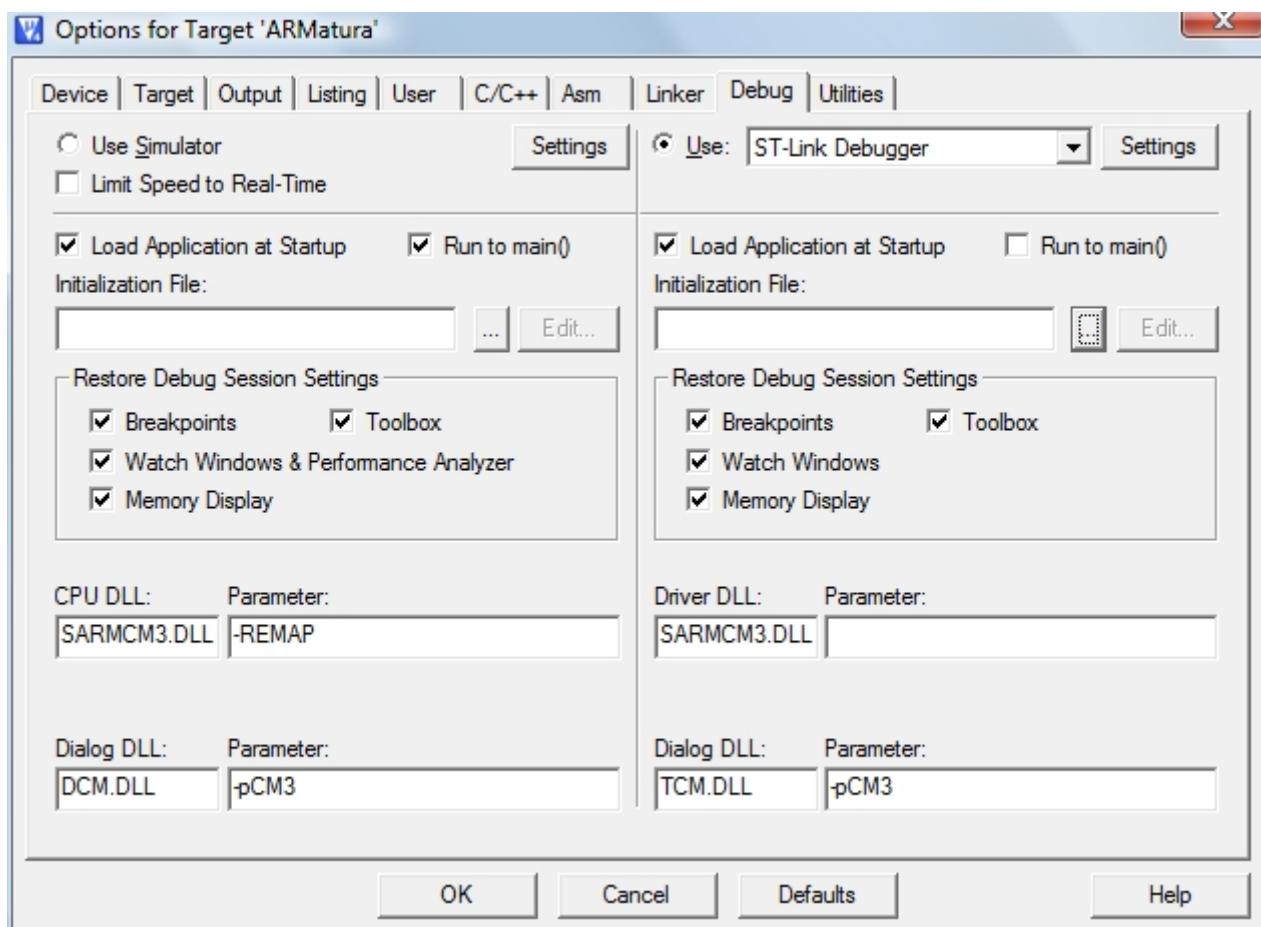
**Asm** Настройки ассемблера, приблизительно те же что и для  $C^{++}$



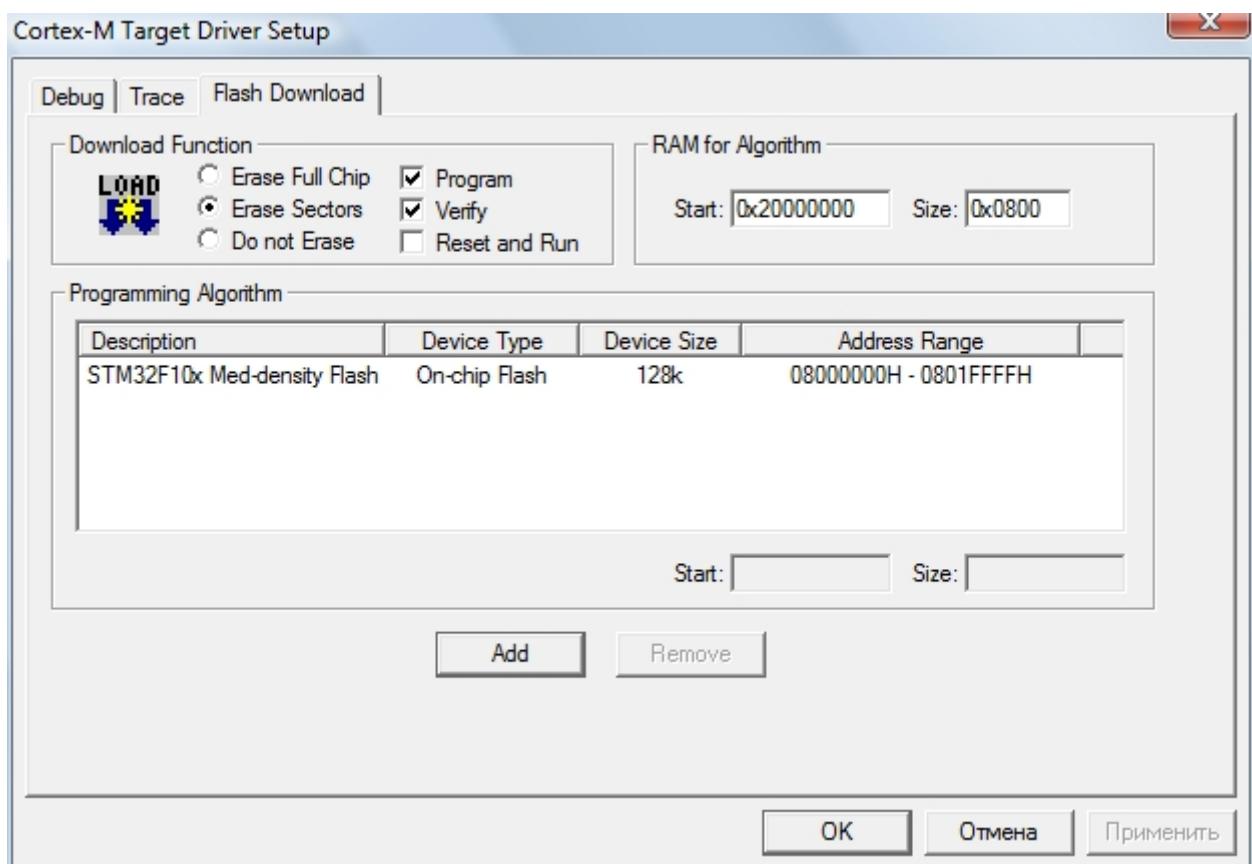
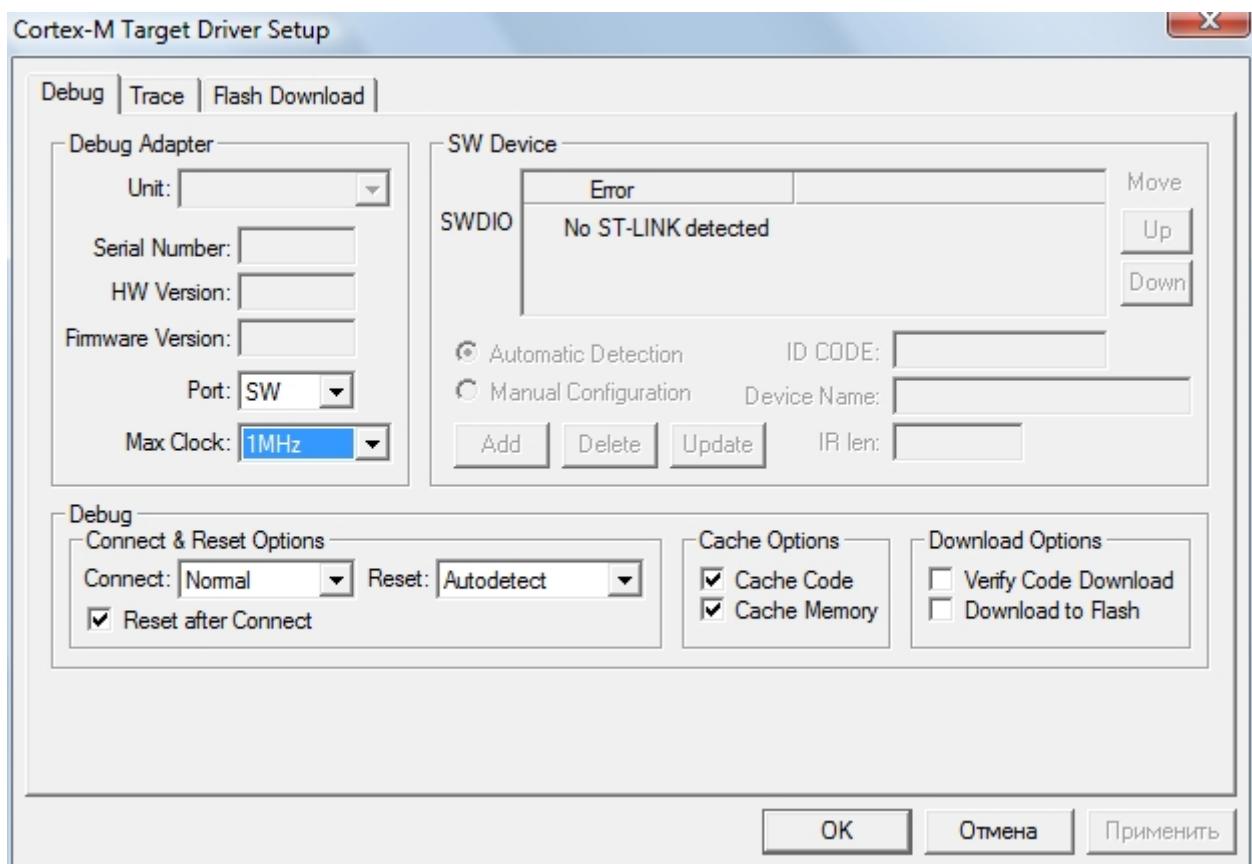
**Linker** Настройки линкера, который собирает объектные файлы .o, в которые компилируются каждый программный файл (модуль) по отдельности, в один готовый файл прошивки (настройки карты памяти контроллера, адреса точки входа программы и т.п.)



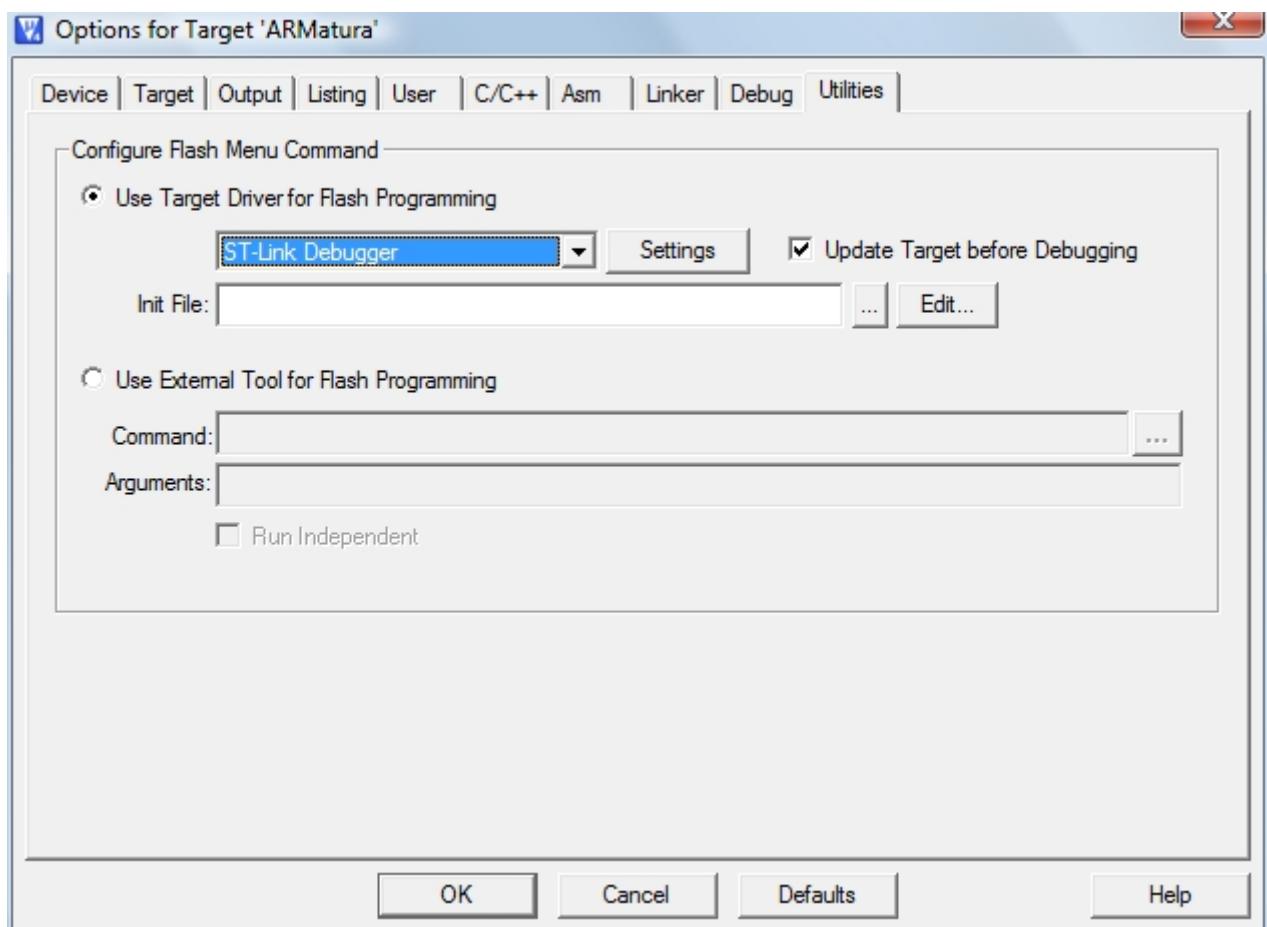
**Debug** Настройки отладки с помощью адаптера – STlink (внешний или встроенный в оценочную плату), JTAG.



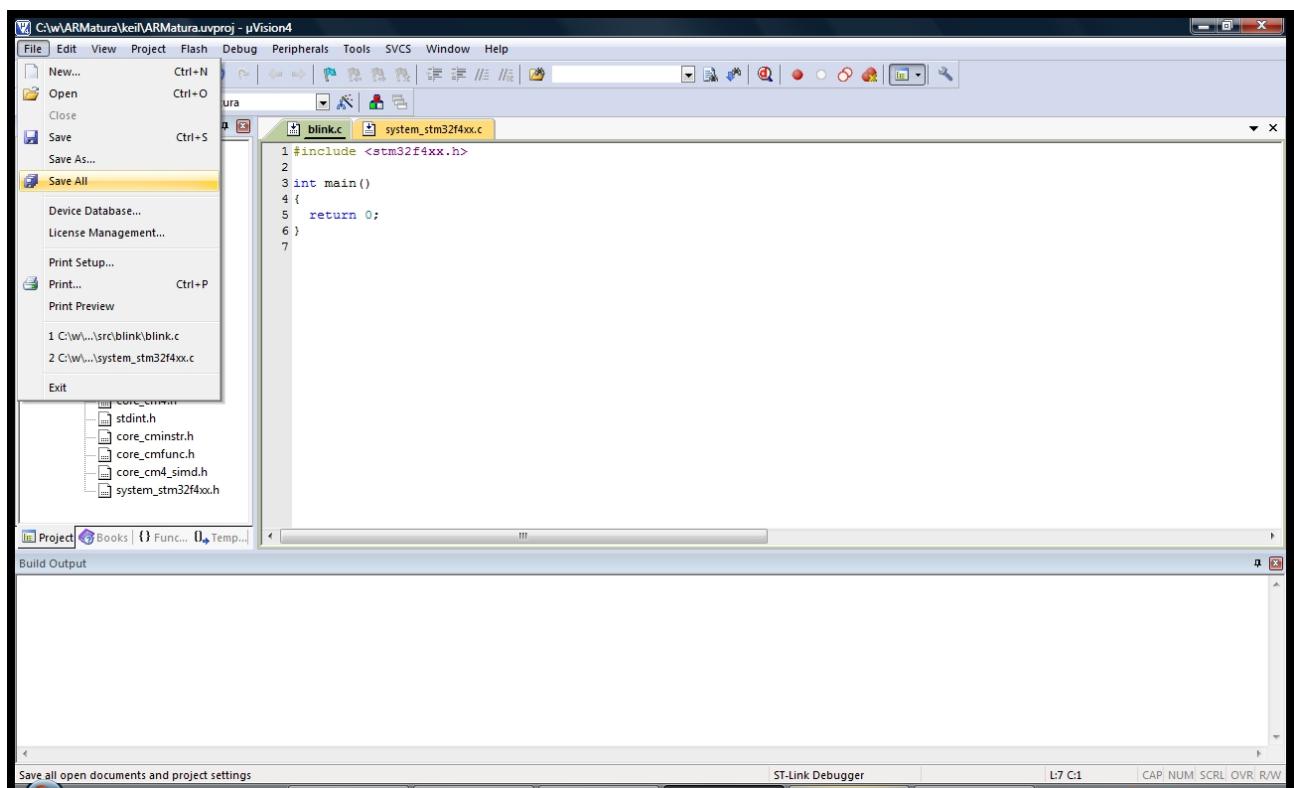
Настройки адаптера STlink – включаем режим SWD для варианта встроенного на оценочную плату, и обнаруживаем что не установили пакет поддержки STlink и драйвера.



**Utilities** Настраиваем в качестве программатора для прошивки тот же STlink



Сохраняем настроенный проект



## 7.2 Структура файлов

```

ARMatura
└── startup ..... код инициализации ядра процессора
    └── keil_startup_stm32f4xx.s ..... ассемблерный код инициализации
        └── system_stm32f4xx.c ..... синый код инициализации STM32F4
            └── system_stm32f4xx.h ..... синый код инициализации STM32F4
                └── stm32f4xx.h ..... STM32F4 CMSIS Cortex-M4 Device Peripheral
                    └── core_cm4.h ..... CMSIS библиотека поддержки ядра Cortex-M4
                        └── stdint.h ..... стандартные целые типы C+
                        └── core_cmInstr.h ..... CMSIS Core Instruction Interface
                        └── core_cmFunc.h ..... CMSIS Core Register Access Functions
                        └── core_cm4_simd.h ..... CMSIS Cortex-M4 расширение SIMD/DSP
    └── blink
        └── blink.c ..... простая программа мигания светодиодом или дрыгания ногой
            └── stm32f4xx.h

```

В отличие от кристаллов типа Atmel AVR, при программировании для архитектуры Cortex-Mx принято использовать готовую библиотеку CMSIS, для которой проводится активная стандартизация среди производителей процессоров.

Cortex > Microcontroller > Software > Interface > Standard

<http://www.arm.com/products/processors/cortex-m/cortex-microcontroller-software-interface-standard.php>

Кроме основной части CMSIS, общей для всех производителей МК Cortex-Mx, каждый подставщик предоставляет аппаратно-зависимую часть библиотеки:

- стартовый код инициализации ядра процессора
- библиотеку периферии например STM32F Standard Peripheral Library, включающую код обеспечивающий доступ к устройствам, встроенным в кристалл в т.ч.
- USB OTG Host & Device
- DSP /SIMD Extension (для Cortex-M4F) системо-зависимую часть расширения CMSIS DSP

Библиотеки доступны для загрузки тут:

**STSW-STM32065** STM32F4 DSP and standard peripherals library, including 82 examples for 26 different peripherals and template project for 5 different IDEs

[http://www.st.com/st-web-ui/static/active/en/st\\_prod\\_software\\_internet/resource/technical/software/firmware/stm32f4\\_dsp\\_stdperiph\\_lib.zip](http://www.st.com/st-web-ui/static/active/en/st_prod_software_internet/resource/technical/software/firmware/stm32f4_dsp_stdperiph_lib.zip)

## 7.3 blink.c

Сначала посмотрим код основной программы:

Listing 7.1: /src/blink/blink.c

```
/*
 * ARMatura book sample
 * https://github.com/ponyatov/ARMatura
 */

#include <stm32f4xx.h> // загрузить CMSIS-библиотеку STM32Fx

int main()
{
    return 0;
    char X [] = "HelloWorld!";
}
```

Как видим — в ней нет абсолютно ничего сложного: только подключение библиотеки нашего МК и единственная функция `int main(void)`, содержащая только возврат в стартовый код CMSIS значения по умолчанию — 0.

Файл `stm32f4xx.h` подробно рассмотрен в 21.1 — на текущем слое знаний не стоит в него углубляться: он слишком большой и содержит практически весь список функционала, обеспечиваемый CMSIS.

То же самое можно сказать и о других файлах — все они относятся либо к библиотеке CMSIS, либо к файлам поддержки производителя чипов.

Подробности будут рассмотрены далее в разделе VII, а пока важнее получить практический результат.

Глава 8

Hell Of World

## **Часть IV**

# **Средства разработки**

# Глава 9

## Keil MDK-ARM



Для начала освоения программирования для ARM рекомендуем использовать бесплатный пакет от Keil: <http://www.keil.com/arm/mdk.asp> — ограничения бесплатной версии в 32К кода вполне достаточно для начального освоения программирования под процессоры семейства Cortex-Mx, а затем уже можно переползать на открытое ПО: GNU toolchain 10.1, Eclipse 11.1 и Linux XIII.

Процесс установки и первоначальной настройки описан в 9.

# Глава 10

## Компиляторы

10.1 GCC

10.2 KeilCC

10.3 IAR

# **Глава 11**

## **IDE**

**11.1 Eclipse**

**11.2 Code::Blocks**

**11.3 gVim**

**11.4 Keil uVision**

**11.5 IAR**

# Глава 12

## Программаторы

12.1 STlink

12.2 Serial Boot

# Глава 13

## Отладчики

13.1 JTAG

13.2 STM32 SWD

13.3 GDB

STlink gdbserver

OpenOCD

# Часть V

## Основы языка $C^{+^+}$

# Глава 14

## Синтаксис

# Глава 15

## Типы данных

# Глава 16

## Стандартная библиотека libc

## Часть VI

### Отладка

# Глава 17

## JTAG

# Глава 18

## GDB

# Глава 19

## OpenOCD

## **Часть VII**

### **CMSIS**

# Глава 20

## Startup: файлы стартового кода

keil\_startup\_stm32f4xx.s

Listing 20.1: /lib/STM32F4/keil\_startup\_stm32f4xx.s

```
;***** (C) COPYRIGHT 2012 STMicroelectronics *****
;* File Name          : startup_stm32f4xx.s
;* Author             : MCD Application Team
;* Version            : V1.0.2
;* Date               : 05-March-2012
;* Description        : STM32F4xx devices vector table for MDK-ARM toolchain.
;*
;* This module performs :
;* - Set the initial SP
;* - Set the initial PC == Reset_Handler
;* - Set the vector table entries with the exceptions ISR address
;* - Configure the system clock and the external SRAM mounted on
;* STM324xG-EVAL board to be used as data memory (optional,
;* to be enabled by user)
;* - Branches to --main in the C library (which eventually
;* calls main()).
;* After Reset the CortexM4 processor is in Thread mode,
;* priority is Privileged, and the Stack is set to Main.
;* <<< Use Configuration Wizard in Context Menu >>>
```

```
*****  
;  
; Licensed under MCD-ST Liberty SW License Agreement V2, ( the "License");  
; You may not use this file except in compliance with the License.  
;  
; You may obtain a copy of the License at:  
;  
; http://www.st.com/software-license-agreement-liberty-v2  
;  
; Unless required by applicable law or agreed to in writing, software  
; distributed under the License is distributed on an "AS IS" BASIS,  
; WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
; See the License for the specific language governing permissions and  
; limitations under the License.  
;  
*****  
;  
; Amount of memory (in bytes) allocated for Stack  
; Tailor this value to your application needs  
<h> Stack Configuration  
;  
; <o> Stack Size (in Bytes) <0x0-0xFFFFFFF:8>  
;  
; </h>  
;  
Stack_Size EQU 0x000000400  
;  
;  
Stack_Mem AREA STACK, NOINIT, READWRITE, ALIGN=3  
Stack_Space SPACE Stack_Size  
--initial_sp  
;  
;  
; <h> Heap Configuration  
; <o> Heap Size (in Bytes) <0x0-0xFFFFFFF:8>  
;  
; </h>  
;  
Heap_Size EQU 0x000000200
```

```
--heap_base          AREA    HEAP, NOINIT, READWRITE, ALIGN=3
Heap_Mem           SPACE   Heap_Size
--heap_limit

PRESERVE8
THUMB

; Vector Table Mapped to Address 0 at Reset
AREA   RESET, DATA, READONLY
EXPORT --Vectors
EXPORT --Vectors_End
EXPORT --Vectors_Size

--Vectors
DCD    --initial_sp
DCD    Reset_Handler
DCD    NMI_Handler
DCD    HardFault_Handler
DCD    MemManage_Handler
DCD    BusFault_Handler
DCD    UsageFault_Handler
DCD    0
DCD    0
DCD    0
DCD    0
DCD    0
DCD    SVC_Handler
DCD    DebugMon_Handler
DCD    0
DCD    PendSV_Handler
DCD    SysTick_Handler

; External Interrupts
DCD    WWDG_IRQHandler
DCD    PVD_IRQHandler
; Window WatchDog
; PVD through EXTI Line detection
```

```
 ; TAMP and TimeStamps through the EXTI line
 DCD          ; Tamper and TimeStamps through the EXTI line
 DCD          ; RTC Wakeup through the EXTI line
 DCD          ; FLASH
 DCD          ; RCC
 DCD          ; EXTI Line0
 DCD          ; EXTI Line1
 DCD          ; EXTI Line2
 DCD          ; EXTI Line3
 DCD          ; EXTI Line4
 DCD          ; DMA1 Stream 0
 DCD          ; DMA1 Stream 1
 DCD          ; DMA1 Stream 2
 DCD          ; DMA1 Stream 3
 DCD          ; DMA1 Stream 4
 DCD          ; DMA1 Stream 5
 DCD          ; DMA1 Stream 6
 DCD          ; ADC1 , ADC2 and ADC3s
 DCD          ; CAN1 TX
 DCD          ; CAN1 RX0
 DCD          ; CAN1 RX1
 DCD          ; CAN1 SCE
 DCD          ; External Line [9:5] s
 DCD          ; TIM1 Break and TIM9
 DCD          ; TIM1 Update and TIM10
 DCD          ; TIM1 Trigger and Commutation and TIM11
 DCD          ; TIM1 Capture Compare
 DCD          ; TIM2
 DCD          ; TIM3
 DCD          ; TIM4
 DCD          ; I2C1 Event
 DCD          ; I2C1 Error
 DCD          ; I2C2 Event
 DCD          ; I2C2 Error
 DCD          ; SPI1
 DCD          ; SPI2
 DCD          ; ADC
 DCD          ; CAN1_RX
 DCD          ; CAN1_RXO
 DCD          ; CAN1_RX1
 DCD          ; CAN1_SCE
 DCD          ; External Line [9:5] s
 DCD          ; TIM1_BRK_TIM9_IRQHandler
 DCD          ; TIM1_UP_TIM10_IRQHandler
 DCD          ; TIM1_TRG_COM_TIM11_IRQHandler
 DCD          ; TIM1_CC_IRQHandler
 DCD          ; TIM2_IRQHandler
 DCD          ; TIM3_IRQHandler
 DCD          ; TIM4_IRQHandler
 DCD          ; I2C1_EV_IRQHandler
 DCD          ; I2C1_ER_IRQHandler
 DCD          ; I2C2_EV_IRQHandler
 DCD          ; I2C2_ER_IRQHandler
 DCD          ; SPI1_IRQHandler
 DCD          ; SPI2_IRQHandler
```

```

    DCD          USART1_IRQHandler
    DCD          USART2_IRQHandler
    DCD          USART3_IRQHandler
    DCD          EXTI15_10_IRQHandler
    DCD          RTC_Alarm_IRQHandler
    DCD          OTG_FS_WKUP_IRQHandler
    DCD          TIM8_BRK_TIM12_IRQHandler
    DCD          TIM8_UP_TIM13_IRQHandler
    DCD          TIM8_TRG_COM_TIM14_IRQHandler
    DCD          TIM8_CC_IRQHandler
    DCD          DMA1_Stream7_IRQHandler
    DCD          FSMC_IRQHandler
    DCD          SDIO_IRQHandler
    DCD          TIM5_IRQHandler
    DCD          SPI3_IRQHandler
    DCD          UART4_IRQHandler
    DCD          UART5_IRQHandler
    DCD          TIM6_DAC_IRQHandler
    DCD          TIM7_IRQHandler
    DCD          DMA2_Stream0_IRQHandler
    DCD          DMA2_Stream1_IRQHandler
    DCD          DMA2_Stream2_IRQHandler
    DCD          DMA2_Stream3_IRQHandler
    DCD          DMA2_Stream4_IRQHandler
    DCD          ETH_IRQHandler
    DCD          ETH_WKUP_IRQHandler
    DCD          CAN2_Tx_IRQHandler
    DCD          CAN2_RXO_IRQHandler
    DCD          CAN2_RX1_IRQHandler
    DCD          CAN2_SCE_IRQHandler
    DCD          OTG_FS_IRQHandler
    DCD          DMA2_Stream5_IRQHandler
    DCD          DMA2_Stream6_IRQHandler
    DCD          DMA2_Stream7_IRQHandler
    DCD          USART6_IRQHandler

    ; USART1
    ; USART2
    ; USART3
    ; External Line [15:10] s
    ; RTC Alarm (A and B) through EXTI Line
    ; USB OTG FS Wakeup through EXTI line
    ; TIM8 Break and TIM12
    ; TIM8 Update and TIM13
    ; TIM8 Trigger and Commutation and TIM14
    ; TIM8 Capture Compare
    ; DMA1 Stream7
    ; FSMC
    ; SDIO
    ; TIM5
    ; SPI3
    ; UART4
    ; UART5
    ; TIM6 and DAC1&2 underrun errors
    ; TIM7
    ; DMA2 Stream 0
    ; DMA2 Stream 1
    ; DMA2 Stream 2
    ; DMA2 Stream 3
    ; DMA2 Stream 4
    ; Ethernet
    ; CAN2 TX
    ; CAN2 RXO
    ; CAN2 RX1
    ; CAN2 SCE
    ; USB OTG FS
    ; DMA2 Stream 5
    ; DMA2 Stream 6
    ; DMA2 Stream 7
    ; USART6

```

```

    ; I2C3 event
    ; I2C3 error
    ; USB OTG HS End Point 1 Out
    ; USB OTG HS End Point 1 In
    ; USB OTG HS Wakeup through EXTI
    ; USB OTG HS
    ; USB OTG HS
    ; DCMI
    ; CRYP crypto
    ; Hash and Rng
    ; FPU

--Vectors_End
    DCD    I2C3_EV_IRQHandler
    DCD    I2C3_ER_IRQHandler
    DCD    OTG_HS_EP1_OUT_IRQHandler
    DCD    OTG_HS_EP1_IN_IRQHandler
    DCD    OTG_HS_WKUP_IRQHandler
    DCD    OTG_HS_IRQHandler
    DCD    DCMI_IRQHandler
    DCD    CRYP_IRQHandler
    DCD    HASH_RNG_IRQHandler
    DCD    FPU_IRQHandler

--Vectors_Size    EQU    --Vectors_End - --Vectors

    AREA   | .text | , CODE, READONLY

; Reset handler
Reset_Handler    PROC
    EXPORT  Reset_Handler    [WEAK]
    IMPORT  SystemInit
    IMPORT  __main
    ENDP

    LDR    R0 , =SystemInit
    BLX    R0
    LDR    R0 , =__main
    BX    R0
    ENDP

; Dummy Exception Handlers (infinite loops which can be modified)
NMI_Handler    PROC
    EXPORT  NMI_Handler    [WEAK]
    B
    ENDP

```

```
HardFault_Handler \
    PROC
    EXPORT HardFault_Handler [WEAK]
    B
ENDP

MemManage_Handler \
    PROC
    EXPORT MemManage_Handler [WEAK]
    B
ENDP

BusFault_Handler \
    PROC
    EXPORT BusFault_Handler [WEAK]
    B
ENDP

UsageFault_Handler \
    PROC
    EXPORT UsageFault_Handler [WEAK]
    B
ENDP

SVC_Handler \
    PROC
    EXPORT SVC_Handler [WEAK]
    B
ENDP

DebugMon_Handler \
    PROC
    EXPORT DebugMon_Handler [WEAK]
    B
ENDP

PendSV_Handler \
    PROC
    EXPORT PendSV_Handler [WEAK]
    B
ENDP

SysTick_Handler \
    PROC
    EXPORT SysTick_Handler [WEAK]
```

---

```

B          .
ENDP

Default_Handler PROC
    EXPORT WWDG_IRQHandler [WEAK]
    EXPORT PVD_IRQHandler [WEAK]
    EXPORT TAMP_STAMP_IRQHandler [WEAK]
    EXPORT RTC_WKUP_IRQHandler [WEAK]
    EXPORT FLASH_IRQHandler [WEAK]
    EXPORT RCC_IRQHandler [WEAK]
    EXPORT EXTI0_IRQHandler [WEAK]
    EXPORT EXTI1_IRQHandler [WEAK]
    EXPORT EXTI2_IRQHandler [WEAK]
    EXPORT EXTI3_IRQHandler [WEAK]
    EXPORT EXTI4_IRQHandler [WEAK]
    EXPORT DMA1_Stream0_IRQHandler [WEAK]
    EXPORT DMA1_Stream1_IRQHandler [WEAK]
    EXPORT DMA1_Stream2_IRQHandler [WEAK]
    EXPORT DMA1_Stream3_IRQHandler [WEAK]
    EXPORT DMA1_Stream4_IRQHandler [WEAK]
    EXPORT DMA1_Stream5_IRQHandler [WEAK]
    EXPORT DMA1_Stream6_IRQHandler [WEAK]
    EXPORT ADC_IRQHandler [WEAK]
    EXPORT CAN1_Tx_IRQHandler [WEAK]
    EXPORT CAN1_RXO_IRQHandler [WEAK]
    EXPORT CAN1_RX1_IRQHandler [WEAK]
    EXPORT CAN1_SCE_IRQHandler [WEAK]
    EXPORT EXTI9_5_IRQHandler [WEAK]
    EXPORT TIM1_BRK_TIM9_IRQHandler [WEAK]
    EXPORT TIM1_UP_TIM10_IRQHandler [WEAK]
    EXPORT TIM1_TRG_COM_TIM11_IRQHandler [WEAK]
    EXPORT TIM1_CC_IRQHandler [WEAK]
    EXPORT TIM2_IRQHandler [WEAK]
    EXPORT TIM3_IRQHandler [WEAK]

```

---

```
EXPORT TIM4_IRQHandler [WEAK]
EXPORT I2C1_EV_IRQHandler [WEAK]
EXPORT I2C1_ER_IRQHandler [WEAK]
EXPORT I2C2_EV_IRQHandler [WEAK]
EXPORT I2C2_ER_IRQHandler [WEAK]
EXPORT SPI1_IRQHandler [WEAK]
EXPORT SPI2_IRQHandler [WEAK]
EXPORT USART1_IRQHandler [WEAK]
EXPORT USART2_IRQHandler [WEAK]
EXPORT USART3_IRQHandler [WEAK]
EXPORT EXTI15_10_IRQHandler [WEAK]
EXPORT RTC_Alarm_IRQHandler [WEAK]
EXPORT OTG_FS_WKUP_IRQHandler [WEAK]
EXPORT TIM8_BRK_IRQHandler [WEAK]
EXPORT TIM8_UP_IRQHandler [WEAK]
EXPORT TIM8_TIM13_IRQHandler [WEAK]
EXPORT TIM8_TRG_COM_TIM14_IRQHandler [WEAK]
EXPORT TIM8_CC_IRQHandler [WEAK]
EXPORT DMA1_Stream7_IRQHandler [WEAK]
EXPORT FSMC_IRQHandler [WEAK]
EXPORT SDIO_IRQHandler [WEAK]
EXPORT TIM5_IRQHandler [WEAK]
EXPORT SPI3_IRQHandler [WEAK]
EXPORT UART4_IRQHandler [WEAK]
EXPORT UART5_IRQHandler [WEAK]
EXPORT TIM6_DAC_IRQHandler [WEAK]
EXPORT TIM7_IRQHandler [WEAK]
EXPORT DMA2_Stream0_IRQHandler [WEAK]
EXPORT DMA2_Stream1_IRQHandler [WEAK]
EXPORT DMA2_Stream2_IRQHandler [WEAK]
EXPORT DMA2_Stream3_IRQHandler [WEAK]
EXPORT DMA2_Stream4_IRQHandler [WEAK]
EXPORT ETH_IRQHandler [WEAK]
EXPORT ETH_WKUP_IRQHandler [WEAK]
EXPORT CAN2_Tx_IRQHandler [WEAK]
EXPORT CAN2_Rx0_IRQHandler [WEAK]
```

```
EXPORT CAN2_RX1_IRQHandler [WEAK]
EXPORT CAN2_SCE_IRQHandler [WEAK]
EXPORT OTG_FS_IRQHandler [WEAK]
EXPORT DMA2_Stream5_IRQHandler [WEAK]
EXPORT DMA2_Stream6_IRQHandler [WEAK]
EXPORT DMA2_Stream7_IRQHandler [WEAK]
EXPORT USART6_IRQHandler [WEAK]
EXPORT I2C3_EV_IRQHandler [WEAK]
EXPORT I2C3_ER_IRQHandler [WEAK]
EXPORT OTG_HS_EP1_OUT_IRQHandler [WEAK]
EXPORT OTG_HS_EP1_IN_IRQHandler [WEAK]
EXPORT OTG_HS_WKUP_IRQHandler [WEAK]
EXPORT OTG_HS_IRQHandler [WEAK]
EXPORT DCMI_IRQHandler [WEAK]
EXPORT CRYP_IRQHandler [WEAK]
EXPORT HASH_RNG_IRQHandler [WEAK]
EXPORT FPU_IRQHandler [WEAK]

WWDG_IRQHandler
PVD_IRQHandler
TAMP_STAMP_IRQHandler
RTC_WKUP_IRQHandler
FLASH_IRQHandler
RCC_IRQHandler
EXTIO_IRQHandler
EXTI1_IRQHandler
EXTI2_IRQHandler
EXTI3_IRQHandler
EXTI4_IRQHandler
DMA1_Stream0_IRQHandler
DMA1_Stream1_IRQHandler
DMA1_Stream2_IRQHandler
DMA1_Stream3_IRQHandler
DMA1_Stream4_IRQHandler
DMA1_Stream5_IRQHandler
```

---

```
DMA1_Stream6_IRQHandler
ADC_IRQHandler
CAN1_TX_IRQHandler
CAN1_RXO_IRQHandler
CAN1_RXI_IRQHandler
CAN1_SCE_IRQHandler
EXTI9_5_IRQHandler
TIM1_BRK_TIM9_IRQHandler
TIM1_UP_TIM10_IRQHandler
TIM1_TRG_COM_TIM11_IRQHandler
TIM1_CC_IRQHandler
TIM2_IRQHandler
TIM3_IRQHandler
TIM4_IRQHandler
I2C1_EV_IRQHandler
I2C1_ER_IRQHandler
I2C2_EV_IRQHandler
I2C2_ER_IRQHandler
SPI1_IRQHandler
SPI2_IRQHandler
USART1_IRQHandler
USART2_IRQHandler
USART3_IRQHandler
EXTI15_10_IRQHandler
RTC_Alarm_IRQHandler
OTG_FS_WKUP_IRQHandler
TIM8_BRK_TIM12_IRQHandler
TIM8_UP_TIM13_IRQHandler
TIM8_TRG_COM_TIM14_IRQHandler
TIM8_CC_IRQHandler
DMA1_Stream7_IRQHandler
FSMC_IRQHandler
SDIO_IRQHandler
TIM5_IRQHandler
SPI3_IRQHandler
```

---

---

```
UART4_IRQHandler
UART5_IRQHandler
TIM6_DAC_IRQHandler
TIM7_IRQHandler
DMA2_Stream0_IRQHandler
DMA2_Stream1_IRQHandler
DMA2_Stream2_IRQHandler
DMA2_Stream3_IRQHandler
DMA2_Stream4_IRQHandler
ETH_IRQHandler
ETH_WKUP_IRQHandler
CAN2_TX_IRQHandler
CAN2_RX0_IRQHandler
CAN2_RX1_IRQHandler
CAN2_SCE_IRQHandler
OTG_FS_IRQHandler
DMA2_Stream5_IRQHandler
DMA2_Stream6_IRQHandler
DMA2_Stream7_IRQHandler
USART6_IRQHandler
I2C3_EV_IRQHandler
I2C3_ER_IRQHandler
OTG_HS_EP1_OUT_IRQHandler
OTG_HS_EP1_IN_IRQHandler
OTG_HS_WKUP_IRQHandler
OTG_HS_IRQHandler
DCMI_IRQHandler
CRYPT_IRQHandler
HASH_RNG_IRQHandler
FPU_IRQHandler
```

B

ENDP

```

ALIGN
; *****
; User Stack and Heap initialization
; *****

IF      :DEF : --MICROLIB

    EXPORT --initial_sp
    EXPORT --heap_base
    EXPORT --heap_limit

ELSE

    IMPORT --use_two_region_memory
    EXPORT --user_initial_stackheap

--user_initial_stackheap

    LDR   R0, = Heap_Mem
    LDR   R1, = (Stack_Mem + Stack_Size)
    LDR   R2, = (Heap_Mem + Heap_Size)
    LDR   R3, = Stack_Mem
    BX    LR

ALIGN
ENDIF
END

;***** (C) COPYRIGHT STMicroelectronics *****END OF FILE*****

```

## Глава 21

### Стандартная библиотека STM32

## 21.1 stm32f4xx.h

Listing 21.1: /lib/STM32F4/stm32f4xx.h

```
/*
 * @file      stm32f4xx.h
 * @author    MCD Application Team
 * @version   V1.0.2
 * @date      05-March-2012
 * @brief     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.
 *
 * 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 peripheral's drivers in application code(i.e.
 *     code will be based on direct access to peripheral's 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 peripheral's registers hardware
 *
 * <h2><center>&copy; COPYRIGHT 2012 STMicroelectronics</center></h2>
 *
 * Licensed under MCD-ST Liberty SW License Agreement V2, (the "License");
 * You may not use this file except in compliance with the License.
 * You may obtain a copy of the License at:
 *
 * @attention
 *
```

```
* http://www.st.com/software-license-agreement-liberty-v2
*
* Unless required by applicable law or agreed to in writing, software
* distributed under the License is distributed on an "AS IS" BASIS,
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
* See the License for the specific language governing permissions and
* limitations under the License.
*/
*****
/* @addtogroup CMSIS
 * @{
 */
/* @addtogroup stm32f4xx
 * @{
 */
#ifndef __STM32F4XX_H
#define __STM32F4XX_H

#ifdef __cplusplus
extern "C" {
#endif /* __cplusplus */

/* @library_configuration_section
 * @{
 */
/*
 * Uncomment the line below according to the target STM32 device used in your
 * application
 */

```

```

#ifndef !defined (STM32F4XX)
#define STM32F4XX
#endif

/* Tip: To avoid modifying this file each time you need to switch between these
   devices, you can define the device in your toolchain compiler preprocessor.
*/



#if !defined (STM32F4XX)
#error "Please select first the target STM32F4XX device used in your application"
#endif

#if !defined (USE_STDPERIPH_DRIVER)
/*
 * Obrief 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
*/
#ifndef USE_STDPERIPH_DRIVER
#endif /* USE_STDPERIPH_DRIVER */
#endif /* USE_STDPERIPH_DRIVER */

/*
 * Obrief 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.
*/



#if !defined (HSE_VALUE)
#define HSE_VALUE ((uint32_t)25000000) /*!< Value of the External oscillator in Hz */
#endif /* HSE_VALUE */

/*
 * Obrief In the following line adjust the External High Speed oscillator (HSE) Startup
*/

```

```

Timeout value
*/
#ifndef !defined (HSE_STARTUP_TIMEOUT)
#define HSE_STARTUP_TIMEOUT ((uint16_t)0x0500) /*!< Time out for HSE start up */
#endif /* HSE_STARTUP_TIMEOUT */

#ifndef !defined (HSI_VALUE)
#define HSI_VALUE ((uint32_t)16000000) /*!< Value of the Internal oscillator in Hz */
#endif /* HSI_VALUE */

/**
 * @brief STM32F4XX Standard Peripherals Library version number V1.0.2
 */
#define --STM32F4XX_STDPERIPH_VERSION_MAIN (0x01) /*!< [31:24] main version */
#define --STM32F4XX_STDPERIPH_VERSION_SUB1 (0x00) /*!< [23:16] sub1 version */
#define --STM32F4XX_STDPERIPH_VERSION_SUB2 (0x02) /*!< [15:8] sub2 version */
#define --STM32F4XX_STDPERIPH_VERSION_RC (0x00) /*!< [7:0] release candidate */
((--STM32F4XX_STDPERIPH_VERSION_MAIN << 24) \
|(--STM32F4XX_STDPERIPH_VERSION_SUB1 << 16) \
|(--STM32F4XX_STDPERIPH_VERSION_SUB2 << 8) \
|(--STM32F4XX_STDPERIPH_VERSION_RC))

/**
 * @addtogroup Configuration_section_for\CMSIS
 */
/* @{

 /**
 * @brief Configuration of the Cortex-M4 Processor and Core Peripherals
 */
#ifndef !defined __CM4_REV
#define __CM4_REV 0x0001 /*!< Core revision r0p1 */
#endif /* __CM4_REV */
#ifndef !defined __MPU_PRESENT
#define __MPU_PRESENT 1 /*!< STM32F4XX provides an MPU */
#endif /* __MPU_PRESENT */
*/

```

---

```

#define __NVIC_PRIO_BITS      4      /* !< STM32F4XX uses 4 Bits for the Priority Levels */
#define __Vendor_SysTickConfig 0      /* !< Set to 1 if different SysTick Config is used */
#define __FPU_PRESENT         1      /* !< FPU present */

/* @brief STM32F4XX Interrupt Number Definition, according to the selected device
   in @ref Library_configuration_section
*/
typedef enum IRQn
{
    /* **** Cortex-M4 Processor Exceptions Numbers ****/
    NonMaskableInt_IRQn      = -14, /* !< 2 Non Maskable Interrupt
    */
    MemoryManagement_IRQn     = -12, /* !< 4 Cortex-M4 Memory Management Interrupt
    */
    BusFault_IRQn              = -11, /* !< 5 Cortex-M4 Bus Fault Interrupt
    */
    UsageFault_IRQn            = -10, /* !< 6 Cortex-M4 Usage Fault Interrupt
    */
    SVCall_IRQn                = -5,  /* !< 11 Cortex-M4 SV Call Interrupt
    */
    DebugMonitor_IRQn          = -4,  /* !< 12 Cortex-M4 Debug Monitor Interrupt
    */
    PendSV_IRQn                = -2,  /* !< 14 Cortex-M4 Pend SV Interrupt
    */
    SysTick_IRQn                = -1,  /* !< 15 Cortex-M4 System Tick Interrupt
    */
    /* **** STM32 specific Interrupt Numbers ****/
    WWDG_IRQn                  = 0,   /* !< Window WatchDog Interrupt
    */
    PVD_IRQn                   = 1,   /* !< PVD through EXTI Line detection Interrupt
    */
    TAMP_STAMP_IRQn             = 2,   /* !< Tamper andTimeStamp interrupts through the EXTI line
    */
}

```

---

---

```

RTC_WKUP_IRQn           = 3,          /* !< RTC Wakeup interrupt through the EXTI line
/*/FLASH_IRQn             = 4,          /* !< FLASH global Interrupt
/*/RCC_IRQn                = 5,          /* !< RCC global Interrupt
/*/EXTIO_IRQn              = 6,          /* !< EXTI Line0 Interrupt
/*/EXTI1_IRQn               = 7,          /* !< EXTI Line1 Interrupt
/*/EXTI2_IRQn               = 8,          /* !< EXTI Line2 Interrupt
/*/EXTI3_IRQn               = 9,          /* !< EXTI Line3 Interrupt
/*/EXTI4_IRQn               = 10,         /* !< EXTI Line4 Interrupt
/*/DMA1_Stream0_IRQn        = 11,         /* !< DMA1 Stream 0 global Interrupt
/*/DMA1_Stream1_IRQn        = 12,         /* !< DMA1 Stream 1 global Interrupt
/*/DMA1_Stream2_IRQn        = 13,         /* !< DMA1 Stream 2 global Interrupt
/*/DMA1_Stream3_IRQn        = 14,         /* !< DMA1 Stream 3 global Interrupt
/*/DMA1_Stream4_IRQn        = 15,         /* !< DMA1 Stream 4 global Interrupt
/*/DMA1_Stream5_IRQn        = 16,         /* !< DMA1 Stream 5 global Interrupt
/*/DMA1_Stream6_IRQn        = 17,         /* !< DMA1 Stream 6 global Interrupt
/*/ADC_IRQn                 = 18,         /* !< ADC1, ADC2 and ADC3 global Interrupts
/*/CAN1_TX_IRQn             = 19,         /* !< CAN1 TX Interrupt
/*/

```

---

---

```

    = 20,      /* !< CAN1_RX0 Interrupt
 */
    = 21,      /* !< CAN1_RX1 Interrupt
 */
    = 22,      /* !< CAN1_SCE Interrupt
 */
    = 23,      /* !< EXTI9_5 Interrupt
 */
    = 24,      /* !< TIM1_BRK(TIM9) Interrupt
 */
    = 25,      /* !< TIM1_UP(TIM10) Interrupt and TIM10 global interrupt
 */
    = 26,      /* !< TIM1_TRIGGER and Commutation Interrupt and TIM11 global interrupt
 */
    = 27,      /* !< TIM1_CC Capture Compare Interrupt
 */
    = 28,      /* !< TIM2 global Interrupt
 */
    = 29,      /* !< TIM3 global Interrupt
 */
    = 30,      /* !< TIM4 global Interrupt
 */
    = 31,      /* !< I2C1 Event Interrupt
 */
    = 32,      /* !< I2C1_Error Interrupt
 */
    = 33,      /* !< I2C2 Event Interrupt
 */
    = 34,      /* !< I2C2_Error Interrupt
 */
    = 35,      /* !< SPI1 global Interrupt
 */
    = 36,      /* !< SPI2 global Interrupt
 */
    = 37,      /* !< USART1 Interrupt
 */

```

---

---

```

USART2_IRQHandler      = 38,      /* !< USART2 global Interrupt
*/
USART3_IRQHandler      = 39,      /* !< USART3 global Interrupt
*/
EXTI15_10_IRQHandler   = 40,      /* !< External Line[15:10] Interrupts
*/
RTC_Alarm_IRQHandler   = 41,      /* !< RTC Alarm (A and B) through EXTI Line Interrupt
*/
OTG_FS_WKUP_IRQHandler = 42,      /* !< USB OTG FS Wakeup through EXTI line interrupt
*/
TIM8_BRK_TIM12_IRQHandler = 43,    /* !< TIM8 Break Interrupt and TIM12 global interrupt
*/
TIM8_UP_TIM13_IRQHandler = 44,    /* !< TIM8 Update Interrupt and TIM13 global interrupt
*/
TIM8_TRG_COM_TIM14_IRQHandler = 45,    /* !< TIM8 Trigger and Commutation Interrupt and TIM14 global interrupt
TIM8_CC_IRQHandler     = 46,      /* !< TIM8 Capture Compare Interrupt
*/
DMA1_Stream7_IRQHandler = 47,      /* !< DMA1 Stream7 Interrupt
*/
FSMC_IRQHandler        = 48,      /* !< FSMC global Interrupt
*/
SDIO_IRQHandler         = 49,      /* !< SDIO global Interrupt
*/
TIM5_IRQHandler         = 50,      /* !< TIM5 global Interrupt
*/
SPI3_IRQHandler         = 51,      /* !< SPI3 global Interrupt
*/
UART4_IRQHandler        = 52,      /* !< UART4 global Interrupt
*/
UART5_IRQHandler        = 53,      /* !< UART5 global Interrupt
*/
TIM6_DAC_IRQHandler     = 54,      /* !< TIM6 global and DAC1&2 underrun error interrupts
*/
TIM7_IRQHandler         = 55,      /* !< TIM7 global interrupt
*/

```

---

---

```

DMA2_Stream0_IRQHandler      = 56,      /* !< DMA2 Stream 0 global Interrupt
*/
DMA2_Stream1_IRQHandler      = 57,      /* !< DMA2 Stream 1 global Interrupt
*/
DMA2_Stream2_IRQHandler      = 58,      /* !< DMA2 Stream 2 global Interrupt
*/
DMA2_Stream3_IRQHandler      = 59,      /* !< DMA2 Stream 3 global Interrupt
*/
DMA2_Stream4_IRQHandler      = 60,      /* !< DMA2 Stream 4 global Interrupt
*/
ETH_IRQHandler                = 61,      /* !< Ethernet global Interrupt
*/
ETH_WKUP_IRQHandler          = 62,      /* !< Ethernet Wakeup through EXTI line Interrupt
*/
CAN2_TX_IRQHandler           = 63,      /* !< CAN2 TX Interrupt
*/
CAN2_RXO_IRQHandler          = 64,      /* !< CAN2 RXO Interrupt
*/
CAN2_RX1_IRQHandler          = 65,      /* !< CAN2 RX1 Interrupt
*/
CAN2_SCE_IRQHandler          = 66,      /* !< CAN2 SCE Interrupt
*/
OTG_FS_IRQHandler             = 67,      /* !< USB OTG FS global Interrupt
*/
DMA2_Stream5_IRQHandler      = 68,      /* !< DMA2 Stream 5 global interrupt
*/
DMA2_Stream6_IRQHandler      = 69,      /* !< DMA2 Stream 6 global interrupt
*/
DMA2_Stream7_IRQHandler      = 70,      /* !< DMA2 Stream 7 global interrupt
*/
USART6_IRQHandler              = 71,      /* !< USART6 global interrupt
*/
I2C3_EV_IRQHandler           = 72,      /* !< I2C3 event interrupt
*/

```

---

---

```

    I2C3_ER_IRQn           = 73,          /* !< I2C3 error interrupt
*/
    OTG_HS_EP1_OUT_IRQn    = 74,          /* !< USB OTG HS End Point 1 Out global interrupt
*/
    OTG_HS_EP1_IN_IRQn     = 75,          /* !< USB OTG HS End Point 1 In global interrupt
*/
    OTG_HS_WKUP_IRQn       = 76,          /* !< USB OTG HS Wakeup through EXTI interrupt
*/
    OTG_HS_IRQn             = 77,          /* !< USB OTG HS global interrupt
*/
    DCMI_IRQn               = 78,          /* !< DCMI global interrupt
*/
    CRYPT_IRQn              = 79,          /* !< CRYPT crypto global interrupt
*/
    HASH_RNG_IRQn           = 80,          /* !< Hash and Rng global interrupt
*/
    FPU_IRQn                = 81,          /* !< FPU global interrupt
*/
} IRQn_Type;

/* * * * */

#include "core_cm4.h"          /* Cortex-M4 processor and core peripherals */
#include "system_stm32f4xx.h"
#include <stdint.h>

/* * * @addtogroup Exported_types
*/
/* !< STM32F10x Standard Peripheral Library old types (maintained for legacy purpose) */
typedef int32_t s32;
typedef int16_t s16;
typedef int8_t s8;

```

---

```

typedef const int32_t sc32; /*!< Read Only */
typedef const int16_t sc16; /*!< Read Only */
typedef const int8_t sc8; /*!< Read Only */

typedef __IO int32_t vs32;
typedef __IO int16_t vs16;
typedef __IO int8_t vs8;

typedef __I int32_t vsc32; /*!< Read Only */
typedef __I int16_t vsc16; /*!< Read Only */
typedef __I int8_t vsc8; /*!< Read Only */

typedef uint32_t u32;
typedef uint16_t u16;
typedef uint8_t u8;

typedef const uint32_t uc32; /*!< Read Only */
typedef const uint16_t uc16; /*!< Read Only */
typedef const uint8_t uc8; /*!< Read Only */

typedef __IO uint32_t vu32;
typedef __IO uint16_t vu16;
typedef __IO uint8_t vu8;

typedef __I uint32_t vuc32; /*!< Read Only */
typedef __I uint16_t vuc16; /*!< Read Only */
typedef __I uint8_t vuc8; /*!< Read Only */

typedef enum {RESET = 0, SET = !RESET} FlagStatus, ITStatus;

typedef enum {DISABLE = 0, ENABLE = !DISABLE} FunctionalState;
#define IS_FUNCTIONAL_STATE(STATE) ((STATE) == DISABLE) || ((STATE) == ENABLE)

typedef enum {ERROR = 0, SUCCESS = !ERROR} ErrorStatus;

```

---

```

/** * @defgroup Peripheral_registers_structures
 * @{
 */

/** * @brief Analog to Digital Converter
 */

typedef struct
{
    __IO uint32_t SR;          /* !< ADC status register, Address offset: 0x00 */
    __IO uint32_t CR1;         /* !< ADC control register 1, Address offset: 0x04 */
    __IO uint32_t CR2;         /* !< ADC control register 2, Address offset: 0x08 */
    __IO uint32_t SMPR1;       /* !< ADC sample time register 1, Address offset: 0x0C */
    __IO uint32_t SMPR2;       /* !< ADC sample time register 2, Address offset: 0x10 */
    __IO uint32_t JOFR1;       /* !< ADC injected channel data offset register 1, Address offset: 0x14 */
    __IO uint32_t JOFR2;       /* !< ADC injected channel data offset register 2, Address offset: 0x18 */
    __IO uint32_t JOFR3;       /* !< ADC injected channel data offset register 3, Address offset: 0x1C */
    __IO uint32_t JOFR4;       /* !< ADC injected channel data offset register 4, Address offset: 0x20 */
    __IO uint32_t HTR;         /* !< ADC watchdog higher threshold register, Address offset: 0x24 */
    __IO uint32_t LTR;         /* !< ADC watchdog lower threshold register, Address offset: 0x28 */
    __IO uint32_t SQR1;        /* !< ADC regular sequence register 1, Address offset: 0x2C */
    __IO uint32_t SQR2;        /* !< ADC regular sequence register 2, Address offset: 0x30 */
    __IO uint32_t SQR3;        /* !< ADC regular sequence register 3, Address offset: 0x34 */
    __IO uint32_t JSQR;        /* !< ADC injected sequence register, Address offset: 0x38 */
    __IO uint32_t JDR1;        /* !< ADC injected data register 1, Address offset: 0x3C */
    __IO uint32_t JDR2;        /* !< ADC injected data register 2, Address offset: 0x40 */
    __IO uint32_t JDR3;        /* !< ADC injected data register 3, Address offset: 0x44 */
    __IO uint32_t JDR4;        /* !< ADC injected data register 4, Address offset: 0x48 */
    __IO uint32_t DR;          /* !< ADC regular data register, Address offset: 0x4C */
}

```

---

```

} ADC_TypeDef;

typedef struct
{
    --IO uint32_t CSR;      /* !< ADC Common status register,
    --IO uint32_t CCR;      /* !< ADC common control register,
    --IO uint32_t CDR;      /* !< ADC common regular data register for dual
    --IO uint32_t CDR;      AND triple modes,
} ADC_Common_TypeDef;

/*
 * @brief Controller Area Network TxMailBox
 */

typedef struct
{
    --IO uint32_t TIR;      /* !< CAN TX mailbox identifier register */
    --IO uint32_t TDTR;     /* !< CAN mailbox data length control and time stamp register */
    --IO uint32_t TDLR;     /* !< CAN mailbox data low register */
    --IO uint32_t TDHR;     /* !< CAN mailbox data high register
} CAN_TxMailBox_TypeDef;

/*
 * @brief Controller Area Network FIFOMailBox
 */

typedef struct
{
    --IO uint32_t RIR;      /* !< CAN receive FIFO mailbox identifier register */
    --IO uint32_t RDTR;     /* !< CAN receive FIFO mailbox data length control and time stamp register */
    --IO uint32_t RDLR;     /* !< CAN receive FIFO mailbox data low register */
    --IO uint32_t RDHR;     /* !< CAN receive FIFO mailbox data high register
} CAN_FIFOMailBox_TypeDef;

```

```

    /**
     * @brief Controller Area Network FilterRegister
     */
    typedef struct
    {
        --IO uint32_t FR1; /*!< CAN Filter bank register 1 */
        --IO uint32_t FR2; /*!< CAN Filter bank register 1 */
    } CAN_FilterRegister_TypeDef;

    /**
     * @brief Controller Area Network
     */
    typedef struct
    {
        --IO uint32_t MCR; /*!< CAN master control register , Address offset: 0x0022 */
        --IO uint32_t MSR; /*!< CAN master status register , Address offset: 0x04 */
        --IO uint32_t TSR; /*!< CAN transmit status register , Address offset: 0x08 */
        --IO uint32_t RFOR; /*!< CAN receive FIFO 0 register , Address offset: 0x0C */
        --IO uint32_t RF1R; /*!< CAN receive FIFO 1 register , Address offset: 0x10 */
        --IO uint32_t IER; /*!< CAN interrupt enable register , Address offset: 0x14 */
        --IO uint32_t ESR; /*!< CAN error status register , Address offset: 0x18 */
        --IO uint32_t BTR; /*!< CAN bit timing register , Address offset: 0x1C */
        uint32_t RESERVED0 [88]; /*!< Reserved , 0x020 - 0x17F */
        uint32_t sTxMailBox [3]; /*!< CAN Tx MailBox , Address offset: 0x180 - 0x188
    }

```

```

CAN_FIFOMailBox_TypeDef      sFIFOMailBox[2];      /* !< CAN FIFO MailBox , Address offset: 0x1E0
uint32_t          RESERVED1[12];      /* !< Reserved , 0x1D0 - 0x1FF
                                         FMR;      /* !< CAN filter master register , Address offset: 0x200
                                         --IO uint32_t      /* !< CAN filter master register register , Address offset: 0x204
                                         --IO uint32_t      /* !< CAN filter mode register , Address offset: 0x20C
                                         uint32_t          RESERVED2;      /* !< Reserved , 0x208
                                         --IO uint32_t      /* !< CAN filter scale register , Address offset: 0x214
                                         uint32_t          FS1R;      /* !< Reserved , 0x210
                                         RESERVED3;      /* !< Reserved , 0x218
                                         --IO uint32_t      /* !< CAN filter FIFO assignment register , Address offset: 0x21C
                                         uint32_t          RESERVED4;      /* !< Reserved , 0x21E
                                         --IO uint32_t      /* !< CAN filter activation register , Address offset: 0x220
                                         uint32_t          FA1R;      /* !< Reserved , 0x228
                                         RESERVED5[8];      /* !< Reserved , 0x220-0x23F
                                         --IO uint32_t      /* !< CAN Filter Register Register , Address offset: 0x240-0x24C
                                         CAN_FilterRegister_TypeDef sFilterRegister[28]; /* !< CAN Filter Register ,
                                         * CAN_TypeDef;
                                         */
                                         /* @brief CRC calculation unit
                                         */
typedef struct
{
    --IO uint32_t      DR;      /* !< CRC Data register , Address offset: 0x00 */
    --IO uint8_t       IDR;      /* !< CRC Independent data register , Address offset: 0x04 */
    uint8_t           RESERVED0; /* !< Reserved , 0x05 */
    uint16_t          RESERVED1; /* !< Reserved , 0x06 */
}

```

```

/*!< CRC Control register, Address offset: 0x08 */
} CRC_TypeDef;

/** @brief Digital to Analog Converter */
typedef struct
{
    __IO uint32_t CR;          /*!< DAC control register, Address offset: 0x00 */
    __IO uint32_t SWTRIGR;     /*!< DAC software trigger register, Address offset: 0x04 */
    __IO uint32_t DHR12R1;     /*!< DAC channel1 12-bit right-aligned data holding register, Address offset: 0x08 */
    __IO uint32_t DHR12L1;     /*!< DAC channel1 12-bit left aligned data holding register, Address offset: 0x0C */
    __IO uint32_t DHR8R1;      /*!< DAC channel1 8-bit right aligned data holding register, Address offset: 0x10 */
    __IO uint32_t DHR12R2;     /*!< DAC channel12 12-bit right aligned data holding register, Address offset: 0x14 */
    __IO uint32_t DHR12L2;     /*!< DAC channel12 12-bit left aligned data holding register, Address offset: 0x18 */
    __IO uint32_t DHR8R2;      /*!< DAC channel12 8-bit right-aligned data holding register, Address offset: 0x1C */
    __IO uint32_t DHR12RD;     /*!< Dual DAC 12-bit right-aligned data holding register, Address offset: 0x20 */
    __IO uint32_t DHR12LD;     /*!< Dual DAC 12-bit left aligned data holding register, Address offset: 0x24 */
    __IO uint32_t DHR8RD;      /*!< Dual DAC 8-bit right aligned data holding register, Address offset: 0x28 */
    __IO uint32_t DOR1;        /*!< DAC channel11 data output register, Address offset: 0x2C */
    __IO uint32_t DOR2;        /*!< DAC channel12 data output register, Address offset: 0x30 */
    __IO uint32_t SR;          /*!< DAC status register, Address offset: 0x34 */
} DAC_TypeDef;
}

/** @brief Debug MCU */
typedef struct
{
    __IO uint32_t IDCODE;      /*!< MCU device ID code, Address offset: 0x00 */
    __IO uint32_t CR;          /*!< Debug MCU configuration register, Address offset: 0x04 */
    __IO uint32_t APB1FZ;      /*!< Debug MCU APB1 freeze register, Address offset: 0x08 */
    __IO uint32_t APB2FZ;      /*!< Debug MCU APB2 freeze register, Address offset: 0x0C */
}

```

```

} DBGMCU_TypeDef;

/* * @brief DCMI */
 */

typedef struct
{
    __IO uint32_t CR;          /* !< DCMI control register 1 , Address offset: 0x00 */
    __IO uint32_t SR;          /* !< DCMI status register, Address offset: 0x04 */
    __IO uint32_t RISR;        /* !< DCMI raw interrupt status register, Address offset: 0x08 */
    __IO uint32_t IER;         /* !< DCMI interrupt enable register, Address offset: 0x0C */
    __IO uint32_t MISR;        /* !< DCMI masked interrupt status register, Address offset: 0x10 */
    __IO uint32_t ICR;         /* !< DCMI interrupt clear register, Address offset: 0x14 */
    __IO uint32_t ESCR;        /* !< DCMI embedded synchronization code register, Address offset: 0x18 */
    __IO uint32_t ESUR;        /* !< DCMI embedded synchronization unmask register, Address offset: 0x1C */
    __IO uint32_t CWSTRTR;     /* !< DCMI crop window start, Address offset: 0x20 */
    __IO uint32_t CWSIZER;     /* !< DCMI crop window size, Address offset: 0x24 */
    __IO uint32_t DR;          /* !< DCMI data register, Address offset: 0x28 */
} DCMI_TypeDef;

/* * @brief DMA Controller */
 */

typedef struct
{
    __IO uint32_t CR;          /* !< DMA stream x configuration register */
    __IO uint32_t NDTR;        /* !< DMA stream x number of data register */
    __IO uint32_t PAR;         /* !< DMA stream x peripheral address register */
    __IO uint32_t MOAR;        /* !< DMA stream x memory 0 address register */
    __IO uint32_t M1AR;        /* !< DMA stream x memory 1 address register */
    __IO uint32_t FCR;         /* !< DMA stream x FIFO control register */
} DMA_Stream_TypeDef;

```

---

```

typedef struct
{
    --I0 uint32_t LISR;      /* !< DMA low interrupt status register,      Address offset: 0x00 */
    --I0 uint32_t HISR;      /* !< DMA high interrupt status register,     Address offset: 0x04 */
    --I0 uint32_t LIFCR;     /* !< DMA low interrupt flag clear register,   Address offset: 0x08 */
    --I0 uint32_t HIFCR;     /* !< DMA high interrupt flag clear register,   Address offset: 0x0C */
} DMA_TypeDef;

/*
 * @brief Ethernet MAC
 */

typedef struct
{
    --I0 uint32_t MACCR;      MACCR;          /* */             Address offset: 0x00
    --I0 uint32_t MACFFR;     MACFFR;         /* */             Address offset: 0x04
    --I0 uint32_t MACHTHR;    MACHTHR;        /* */             Address offset: 0x08
    --I0 uint32_t MACTLRL;    MACTLRL;        /* */             Address offset: 0x0C
    --I0 uint32_t MACMIIAR;   MACMIIAR;       /* */             Address offset: 0x10
    --I0 uint32_t MACMIIDR;   MACMIIDR;      /* */             Address offset: 0x14
    --I0 uint32_t MACFCR;     MACFCR;         /* */             Address offset: 0x18
    --I0 uint32_t MACVLANTR;  MACVLANTR;     /* */             Address offset: 0x1C
    uint32_t RESERVED0 [2];  /* */             Address offset: 0x20
    --I0 uint32_t MACRWUFR;   MACRWUFR;       /* */             Address offset: 0x24
    --I0 uint32_t MACPMTCSR;  MACPMTCSR;     /* */             Address offset: 0x28
    uint32_t RESERVED1 [2];  /* */             Address offset: 0x2C
    --I0 uint32_t MACCSR;    MACCSR;         /* */             Address offset: 0x30
    --I0 uint32_t MACIMR;    MACIMR;         /* */             Address offset: 0x34
    --I0 uint32_t MACAOHR;   MACAOHR;        /* */             Address offset: 0x38
    --I0 uint32_t MACAOLR;   MACAOLR;        /* */             Address offset: 0x3C
    --I0 uint32_t MACA1HR;   MACA1HR;        /* */             Address offset: 0x40
    --I0 uint32_t MACA1LR;   MACA1LR;        /* */             Address offset: 0x44
    --I0 uint32_t MACA2HR;   MACA2HR;        /* */             Address offset: 0x48
    --I0 uint32_t MACA2LR;   MACA2LR;        /* */             Address offset: 0x4C
    --I0 uint32_t MACA3HR;   MACA3HR;        /* */             Address offset: 0x50
}

```

---

---

```

-- IO uint32_t MACA3LR;          /* 24 */
-- IO uint32_t RESERVED2 [40];    /* 65 */
-- IO uint32_t MMCCR;           /* 69 */
-- IO uint32_t MMCRIR;          /* 84 */
-- IO uint32_t MMCTIR;          /* 84 */
-- IO uint32_t MMCRIMR;         /* 84 */
-- IO uint32_t MMCTIMR;         /* 84 */
-- IO uint32_t MMCTGFSCCR;      /* 84 */
-- IO uint32_t MMCTGFMSCCR;     /* 84 */
-- IO uint32_t MMCTGFOR;        /* 84 */
-- IO uint32_t MMCRFCECR;       /* 84 */
-- IO uint32_t MMCRFAECCR;      /* 84 */
-- IO uint32_t MMCRGUFCR;        /* 84 */
-- IO uint32_t MMCRGUFCCR;      /* 84 */
-- IO uint32_t PTPTSCR;         /* 84 */
-- IO uint32_t PTPSSIR;         /* 84 */
-- IO uint32_t PTPTSHR;         /* 84 */
-- IO uint32_t PTPTSLR;         /* 84 */
-- IO uint32_t PTPTSHUR;        /* 84 */
-- IO uint32_t PTPTSLUR;        /* 84 */
-- IO uint32_t PTPTSAR;         /* 84 */
-- IO uint32_t PTPTTHR;         /* 84 */
-- IO uint32_t PTPTTLR;         /* 84 */
-- IO uint32_t RESERVED8;        /* 84 */
-- IO uint32_t PTPTSSR;         /* 84 */
-- IO uint32_t RESERVED9 [565];   /* 84 */
-- IO uint32_t DMABMR;          /* 84 */
-- IO uint32_t DMAATPDR;        /* 84 */
-- IO uint32_t DMARPDR;         /* 84 */
-- IO uint32_t DMARDLAR;        /* 84 */
-- IO uint32_t DMATDLAR;        /* 84 */

```

---

---

```

-- IO  uint32_t DMASR;
-- IO  uint32_t DMAOMR;
-- IO  uint32_t DMAIER;
-- IO  uint32_t DMAMFBOCR;
-- IO  uint32_t DMARSWTR;
-- IO  uint32_t RESERVED10 [8];
-- IO  uint32_t DMACHTDR;
-- IO  uint32_t DMACHRDR;
-- IO  uint32_t DMACHTBAR;
-- IO  uint32_t DMACHRBAR;
} ETH_TypeDef;

/*
 * @brief External Interrupt/Event Controller
 */

typedef struct
{
    -- IO  uint32_t IMR;      /* !< EXTI Interrupt mask register, Address offset: 0x00 */
    -- IO  uint32_t EMR;      /* !< EXTI Event mask register, Address offset: 0x04 */
    -- IO  uint32_t RTSR;     /* !< EXTI Rising trigger selection register, Address offset: 0x08 */
    -- IO  uint32_t FTSR;     /* !< EXTI Falling trigger selection register, Address offset: 0x0C */
    -- IO  uint32_t SWIER;    /* !< EXTI Software interrupt event register, Address offset: 0x10 */
    -- IO  uint32_t PR;       /* !< EXTI Pending register, Address offset: 0x14 */
} EXTI_TypeDef;

/*
 * @brief FLASH Registers
 */

typedef struct
{
    -- IO  uint32_t ACR;      /* !< FLASH access control register, Address offset: 0x00 */
    -- IO  uint32_t KEYR;     /* !< FLASH key register, Address offset: 0x04 */
    -- IO  uint32_t OPTKEYR;  /* !< FLASH option key register, Address offset: 0x08 */
}

```

---

---

```

-- IO uint32_t SR;          /* !< FLASH status register,           Address offset: 0x0C */
-- IO uint32_t CR;          /* !< FLASH control register,        Address offset: 0x10 */
-- IO uint32_t OPTCR;       /* !< FLASH option control register, Address offset: 0x14 */
} FLASH_TypeDef;

/**
 * @brief Flexible Static Memory Controller
 */

typedef struct
{
    -- IO uint32_t BTCR[8];      /* !< NOR/PSRAM chip-select control register(BCR) and chip-select timing register(BTR), A
} FSMC_Bank1_TypeDef;

/**
 * @brief Flexible Static Memory Controller Bank1E
 */

typedef struct
{
    -- IO uint32_t BWTR[7];      /* !< NOR/PSRAM write timing registers, Address offset: 0x104-0x11C */
} FSMC_Bank1E_TypeDef;

/**
 * @brief Flexible Static Memory Controller Bank2
 */

typedef struct
{
    -- IO uint32_t PCR2;         /* !< NAND Flash control register 2,           Address offset: 0x60 */
    -- IO uint32_t SR2;          /* !< NAND Flash FIFO status and interrupt register 2, Address offset: 0x64 */
    -- IO uint32_t PMEM2;        /* !< NAND Flash Common memory space timing register 2, Address offset: 0x68 */
    -- IO uint32_t PATT2;        /* !< NAND Flash Attribute memory space timing register 2, Address offset: 0x6C */
    uint32_t RESERVED;          /* !< Reserved, 0x70 */
    -- IO uint32_t ECCR2;        /* !< NAND Flash ECC result registers 2,       Address offset: 0x74 */
} F

```

---

```

} FSMC_Bank2_TypeDef;

/**
 * @brief Flexible Static Memory Controller Bank3
 */

typedef struct
{
    --IO uint32_t PCR3;          /*!< NAND Flash control register 3,
    --IO uint32_t SR3;           /*!< NAND Flash FIFO status and interrupt register 3,
    --IO uint32_t PMEM3;         /*!< NAND Flash Common memory space timing register 3,
    --IO uint32_t PATT3;         /*!< NAND Flash Attribute memory space timing register 3,
    uint32_t RESERVED0;         /*!< Reserved , 0x90
    --IO uint32_t ECCR3;         /*!< NAND Flash ECC result registers 3,
} FSMC_Bank3_TypeDef;

/**
 * @brief Flexible Static Memory Controller Bank4
 */

typedef struct
{
    --IO uint32_t PCR4;          /*!< PC Card control register 4,
    --IO uint32_t SR4;           /*!< PC Card FIFO status and interrupt register 4,
    --IO uint32_t PMEM4;         /*!< PC Card Common memory space timing register 4,
    --IO uint32_t PATT4;         /*!< PC Card Attribute memory space timing register 4,
    --IO uint32_t PIO4;          /*!< PC Card I/O space timing register 4,
} FSMC_Bank4_TypeDef;

/**
 * @brief General Purpose I/O
 */

typedef struct
{

```

---

```

--> IO uint32_t MODER;          /*!< GPIO port mode register , Address offset: 0x00
--> IO uint32_t OTYPER;         /*!< GPIO port output type register , Address offset: 0x04
--> IO uint32_t OSPEEDR;        /*!< GPIO port output speed register , Address offset: 0x08
--> IO uint32_t PUPDR;          /*!< GPIO port pull-up/pull-down register , Address offset: 0x0C
--> IO uint32_t IDR;           /*!< GPIO port input data register , Address offset: 0x10
--> IO uint32_t ODR;           /*!< GPIO port output data register , Address offset: 0x14
--> IO uint16_t BSRRL;          /*!< GPIO port bit set/reset low register , Address offset: 0x18
--> IO uint16_t BSRRH;          /*!< GPIO port bit set/reset high register , Address offset: 0x1A
--> IO uint32_t LCKR;           /*!< GPIO port configuration lock register , Address offset: 0x1C
--> IO uint32_t AFR[2];         /*!< GPIO alternate function registers , Address offset: 0x20-0x24 */
} GPIO_TypeDef;

/*
 * @brief System configuration controller
 */

typedef struct
{
    --> IO uint32_t MEMRMP;      /*!< SYSCFG memory remap register , Address offset: 0x00
    --> IO uint32_t PMC;          /*!< SYSCFG peripheral mode configuration register , Address offset: 0x04
    --> IO uint32_t EXTICR[4];     /*!< SYSCFG external interrupt configuration registers , Address offset: 0x08-0x14 */
    uint32_t RESERVED[2];          /*!< Reserved , 0x18-0x1C
    --> IO uint32_t CMPCR;        /*!< SYSCFG Compensation cell control register , Address offset: 0x20
} SYSCFG_TypeDef;

/*
 * @brief Inter-integrated Circuit Interface
 */

typedef struct
{

```

```

-- I0 uint16_t CR1;          /*!< I2C Control register 1,           Address offset: 0x00 */
-- uint16_t RESERVED0;       /*!< Reserved, 0x02                 */
-- I0 uint16_t CR2;          /*!< I2C Control register 2,           Address offset: 0x04 */
-- uint16_t RESERVED1;       /*!< Reserved, 0x06                 */
-- I0 uint16_t OAR1;         /*!< I2C Own address register 1,     Address offset: 0x08 */
-- uint16_t RESERVED2;       /*!< Reserved, 0xA0                 */
-- I0 uint16_t OAR2;         /*!< I2C Own address register 2,     Address offset: 0x0C */
-- uint16_t RESERVED3;       /*!< Reserved, 0xE0                 */
-- I0 uint16_t DR;           /*!< I2C Data register,             Address offset: 0x10 */
-- uint16_t RESERVED4;       /*!< Reserved, 0x12                 */
-- I0 uint16_t SR1;          /*!< I2C Status register 1,          Address offset: 0x14 */
-- uint16_t RESERVED5;       /*!< Reserved, 0x16                 */
-- I0 uint16_t SR2;          /*!< I2C Status register 2,          Address offset: 0x18 */
-- uint16_t RESERVED6;       /*!< Reserved, 0x1A                 */
-- I0 uint16_t CCR;          /*!< I2C Clock control register,   Address offset: 0x1C */
-- uint16_t RESERVED7;       /*!< Reserved, 0x1E                 */
-- I0 uint16_t TRISE;        /*!< I2C TRISE register,            Address offset: 0x20 */
-- uint16_t RESERVED8;       /*!< Reserved, 0x22                 */

} I2C_TypeDef;

/**
 * @brief Independent WATCHDOG
 */
typedef struct
{
    -- I0 uint32_t KR;          /*!< IWDG Key register,             Address offset: 0x00 */
    -- I0 uint32_t PR;          /*!< IWDG Prescaler register,      Address offset: 0x04 */
    -- I0 uint32_t RLR;         /*!< IWDG Reload register,         Address offset: 0x08 */
    -- I0 uint32_t SR;          /*!< IWDG Status register,         Address offset: 0x0C */
} IWDG_TypeDef;

/**
 * @brief Power Control
*/

```

```

typedef struct {
    --IO uint32_t CR;      /* !< PWR power control register, Address offset: 0x00 */
    --IO uint32_t CSR;     /* !< PWR power control/status register, Address offset: 0x04 */
} PWR_TypeDef;

/*
 * @brief Reset and Clock Control
 */

typedef struct {
    --IO uint32_t CR;      /* !< RCC clock control register,
    --IO uint32_t PLLCFGR; /* !< RCC PLL configuration register,
    --IO uint32_t CFGR;   /* !< RCC clock configuration register,
    --IO uint32_t CIR;    /* !< RCC clock interrupt register,
    --IO uint32_t AHB1RSTR; /* !< RCC AHB1 peripheral reset register,
    --IO uint32_t AHB2RSTR; /* !< RCC AHB2 peripheral reset register,
    --IO uint32_t AHB3RSTR; /* !< RCC AHB3 peripheral reset register,
    uint32_t RESERVED0;  /* !< Reserved, 0x1C
} RCC_TypeDef;

/*
 * @brief APB1 and APB2 peripherals
 */

typedef struct {
    --IO uint32_t APB1RSTR; /* !< RCC APB1 peripheral reset register,
    --IO uint32_t APB2RSTR; /* !< RCC APB2 peripheral reset register,
    uint32_t RESERVED1[2]; /* !< Reserved, 0x28-0x2C
} RCC_APB_Peripheral_TypeDef;

/*
 * @brief AHB peripherals
 */

typedef struct {
    --IO uint32_t AHB1ENR;  /* !< RCC AHB1 peripheral clock register,
    --IO uint32_t AHB2ENR;  /* !< RCC AHB2 peripheral clock register,
    --IO uint32_t AHB3ENR;  /* !< RCC AHB3 peripheral clock register,
    uint32_t RESERVED2;    /* !< Reserved, 0x3C
} RCC_AHB_Peripheral_TypeDef;

/*
 * @brief APB peripherals
 */

typedef struct {
    --IO uint32_t APB1ENR;  /* !< RCC APB1 peripheral clock enable register,
    --IO uint32_t APB2ENR;  /* !< RCC APB2 peripheral clock enable register,
    uint32_t RESERVED3[2]; /* !< Reserved, 0x48-0x4C
} RCC_APB_Peripheral_TypeDef;

```

## Глава 21. Стандартная библиотека STM32

```
/*!
 * @file RCC.h
 * @brief Real-Time Clock
 */

#ifndef __RCC_H
#define __RCC_H

#include "stm32f10x.h"

/* RCC registers definitions */

/* RCC clock enable register */
#define RCC_CENR ((RCC寄存器地址) + 0x00000000)
/* RCC clock disable register */
#define RCC_CDDR ((RCC寄存器地址) + 0x00000004)
/* RCC clock control register */
#define RCC_CCER ((RCC寄存器地址) + 0x00000008)
/* RCC clock enable in low power mode register */
#define RCC_LPENR ((RCC寄存器地址) + 0x0000001C)
/* RCC peripheral clock enable in low power mode register */
#define RCC_APB1LPENR ((RCC寄存器地址) + 0x00000020)
/* RCC peripheral clock enable in low power mode register */
#define RCC_APB2LPENR ((RCC寄存器地址) + 0x00000024)
/* RCC peripheral clock enable in low power mode register */
#define RCC_AHB1LPENR ((RCC寄存器地址) + 0x00000028)
/* RCC peripheral clock enable in low power mode register */
#define RCC_AHB2LPENR ((RCC寄存器地址) + 0x0000002C)
/* RCC peripheral clock enable in low power mode register */
#define RCC_AHB3LPENR ((RCC寄存器地址) + 0x00000030)
/* Reserved */
#define RCC_RESERVED4 ((RCC寄存器地址) + 0x00000034)
/* Reserved */
#define RCC_RESERVED5 ((RCC寄存器地址) + 0x00000038)
/* RCC Backup domain control register */
#define RCC_BDCR ((RCC寄存器地址) + 0x00000040)
/* RCC clock control & status register */
#define RCC_CSR ((RCC寄存器地址) + 0x00000044)
/* RCC spread spectrum clock generation register */
#define RCC_SSCGR ((RCC寄存器地址) + 0x00000048)
/* RCC PLLI2S configuration register */
#define RCC_PLLI2SCFGR ((RCC寄存器地址) + 0x0000004C)
/* RCC prescaler register */
#define RCC_PRER ((RCC寄存器地址) + 0x00000050)
/* RCC wakeup timer register */
#define RCC_WUTR ((RCC寄存器地址) + 0x00000054)
/* RCC calibration register */
#define RCC_CALIBR ((RCC寄存器地址) + 0x00000058)
/* RCC alarm A register */
#define RCC_ALRMAR ((RCC寄存器地址) + 0x0000005C)
/* RCC alarm B register */
#define RCC_ALRMBR ((RCC寄存器地址) + 0x00000060)
/* RCC write protection register */
#define RCC_WPR ((RCC寄存器地址) + 0x00000064)
/* RCC sub second register */
#define RCC_SSRR ((RCC寄存器地址) + 0x00000068)
/* RCC shift control register */
#define RCC_SHIFTFR ((RCC寄存器地址) + 0x0000006C)

/* RCC clock enable in low power mode register offset */
#define RCC_LPENR_OFFSET 0x00000000
/* RCC peripheral clock enable in low power mode register offset */
#define RCC_APB1LPENR_OFFSET 0x00000020
/* RCC peripheral clock enable in low power mode register offset */
#define RCC_APB2LPENR_OFFSET 0x00000024
/* RCC peripheral clock enable in low power mode register offset */
#define RCC_AHB1LPENR_OFFSET 0x00000028
/* RCC peripheral clock enable in low power mode register offset */
#define RCC_AHB2LPENR_OFFSET 0x0000002C
/* RCC peripheral clock enable in low power mode register offset */
#define RCC_AHB3LPENR_OFFSET 0x00000030
/* Reserved */
#define RCC_RESERVED4_OFFSET 0x00000034
/* Reserved */
#define RCC_RESERVED5_OFFSET 0x00000038
/* RCC Backup domain control register offset */
#define RCC_BDCR_OFFSET 0x00000040
/* RCC clock control & status register offset */
#define RCC_CSR_OFFSET 0x00000044
/* RCC spread spectrum clock generation register offset */
#define RCC_SSCGR_OFFSET 0x00000048
/* RCC PLLI2S configuration register offset */
#define RCC_PLLI2SCFGR_OFFSET 0x0000004C
/* RCC prescaler register offset */
#define RCC_PRER_OFFSET 0x00000050
/* RCC wakeup timer register offset */
#define RCC_WUTR_OFFSET 0x00000054
/* RCC calibration register offset */
#define RCC_CALIBR_OFFSET 0x00000058
/* RCC alarm A register offset */
#define RCC_ALRMAR_OFFSET 0x0000005C
/* RCC alarm B register offset */
#define RCC_ALRMBR_OFFSET 0x00000060
/* RCC write protection register offset */
#define RCC_WPR_OFFSET 0x00000064
/* RCC sub second register offset */
#define RCC_SSRR_OFFSET 0x00000068
/* RCC shift control register offset */
#define RCC_SHIFTFR_OFFSET 0x0000006C

/* RCC time register */
#define RCC_TR ((RCC寄存器地址) + 0x00000070)
/* RCC date register */
#define RCC_DR ((RCC寄存器地址) + 0x00000074)
/* RTC control register */
#define RCC_CR ((RCC寄存器地址) + 0x00000078)
/* RTC initialization and status register */
#define RCC_ISR ((RCC寄存器地址) + 0x0000007C)
/* RTC prescaler register */
#define RCC_PRER ((RCC寄存器地址) + 0x00000080)
/* RTC wakeup timer register */
#define RCC_WUTR ((RCC寄存器地址) + 0x00000084)
/* RTC calibration register */
#define RCC_CALIBR ((RCC寄存器地址) + 0x00000088)
/* RTC alarm A register */
#define RCC_ALRMAR ((RCC寄存器地址) + 0x0000008C)
/* RTC alarm B register */
#define RCC_ALRMBR ((RCC寄存器地址) + 0x00000090)
/* RTC write protection register */
#define RCC_WPR ((RCC寄存器地址) + 0x00000094)
/* RTC sub second register */
#define RCC_SSRR ((RCC寄存器地址) + 0x00000098)
/* RTC shift control register */
#define RCC_SHIFTFR ((RCC寄存器地址) + 0x000000A0)

/* RTC time register offset */
#define RCC_TR_OFFSET 0x00000070
/* RTC date register offset */
#define RCC_DR_OFFSET 0x00000074
/* RTC control register offset */
#define RCC_CR_OFFSET 0x00000078
/* RTC initialization and status register offset */
#define RCC_ISR_OFFSET 0x0000007C
/* RTC prescaler register offset */
#define RCC_PRER_OFFSET 0x00000080
/* RTC wakeup timer register offset */
#define RCC_WUTR_OFFSET 0x00000084
/* RTC calibration register offset */
#define RCC_CALIBR_OFFSET 0x00000088
/* RTC alarm A register offset */
#define RCC_ALRMAR_OFFSET 0x0000008C
/* RTC alarm B register offset */
#define RCC_ALRMBR_OFFSET 0x00000090
/* RTC write protection register offset */
#define RCC_WPR_OFFSET 0x00000094
/* RTC sub second register offset */
#define RCC_SSRR_OFFSET 0x00000098
/* RTC shift control register offset */
#define RCC_SHIFTFR_OFFSET 0x000000A0

/* RTC time register */
#define RTC_TDR ((RTC寄存器地址) + 0x00000000)
/* RTC date register */
#define RTC_DDR ((RTC寄存器地址) + 0x00000004)
/* RTC control register */
#define RTC_CCR ((RTC寄存器地址) + 0x00000008)
/* RTC initialization and status register */
#define RTC_ISR ((RTC寄存器地址) + 0x0000000C)
/* RTC prescaler register */
#define RTC_PRR ((RTC寄存器地址) + 0x00000010)
/* RTC wakeup timer register */
#define RTC_WUTR ((RTC寄存器地址) + 0x00000014)
/* RTC calibration register */
#define RTC_CALIBR ((RTC寄存器地址) + 0x00000018)
/* RTC alarm A register */
#define RTC_ALRMAR ((RTC寄存器地址) + 0x0000001C)
/* RTC alarm B register */
#define RTC_ALRMBR ((RTC寄存器地址) + 0x00000020)
/* RTC write protection register */
#define RTC_WPR ((RTC寄存器地址) + 0x00000024)
/* RTC sub second register */
#define RTC_SSRR ((RTC寄存器地址) + 0x00000028)
/* RTC shift control register */
#define RTC_SHIFTFR ((RTC寄存器地址) + 0x0000002C)

/* RTC time register offset */
#define RTC_TDR_OFFSET 0x00000000
/* RTC date register offset */
#define RTC_DDR_OFFSET 0x00000004
/* RTC control register offset */
#define RTC_CCR_OFFSET 0x00000008
/* RTC initialization and status register offset */
#define RTC_ISR_OFFSET 0x0000000C
/* RTC prescaler register offset */
#define RTC_PRR_OFFSET 0x00000010
/* RTC wakeup timer register offset */
#define RTC_WUTR_OFFSET 0x00000014
/* RTC calibration register offset */
#define RTC_CALIBR_OFFSET 0x00000018
/* RTC alarm A register offset */
#define RTC_ALRMAR_OFFSET 0x0000001C
/* RTC alarm B register offset */
#define RTC_ALRMBR_OFFSET 0x00000020
/* RTC write protection register offset */
#define RTC_WPR_OFFSET 0x00000024
/* RTC sub second register offset */
#define RTC_SSRR_OFFSET 0x00000028
/* RTC shift control register offset */
#define RTC_SHIFTFR_OFFSET 0x0000002C
```

## Глава 21. Стандартная библиотека STM32

```

--IO uint32_t TSTR;          /*!< RTC time stamp time register,
--IO uint32_t TSDR;          /*!< RTC time stamp date register,
--IO uint32_t TSSSR;         /*!< RTC time-stamp sub second register,
--IO uint32_t CALR;          /*!< RTC calibration register,
--IO uint32_t TAFCR;         /*!< RTC tamper and alternate function configuration register , Address offset: 0x40
--IO uint32_t ALRMASSR;      /*!< RTC alarm A sub second register, Address offset: 0x44
--IO uint32_t ALRMBSSR;      /*!< RTC alarm B sub second register, Address offset: 0x48
--IO uint32_t RESERVED7;     /*!< Reserved , 0x4C
*/
--IO uint32_t BKPOR;          /*!< RTC backup register 1,
--IO uint32_t BKP1R;          /*!< RTC backup register 1,
--IO uint32_t BKP2R;          /*!< RTC backup register 2,
--IO uint32_t BKP3R;          /*!< RTC backup register 3,
--IO uint32_t BKP4R;          /*!< RTC backup register 4,
--IO uint32_t BKP5R;          /*!< RTC backup register 5,
--IO uint32_t BKP6R;          /*!< RTC backup register 6,
--IO uint32_t BKP7R;          /*!< RTC backup register 7,
--IO uint32_t BKP8R;          /*!< RTC backup register 8,
--IO uint32_t BKP9R;          /*!< RTC backup register 9,
--IO uint32_t BKP10R;         /*!< RTC backup register 10,
--IO uint32_t BKP11R;         /*!< RTC backup register 11,
--IO uint32_t BKP12R;         /*!< RTC backup register 12,
--IO uint32_t BKP13R;         /*!< RTC backup register 13,
--IO uint32_t BKP14R;         /*!< RTC backup register 14,
--IO uint32_t BKP15R;         /*!< RTC backup register 15,
--IO uint32_t BKP16R;         /*!< RTC backup register 16,
--IO uint32_t BKP17R;         /*!< RTC backup register 17,
--IO uint32_t BKP18R;         /*!< RTC backup register 18,
--IO uint32_t BKP19R;         /*!< RTC backup register 19,
}
RTC_TypeDef;
/*
 * @brief SD host Interface
 */

```

```

typedef struct
{
    --I0 uint32_t POWER;           /*!< SDIO power control register , Address offset: 0x00 */
    --I0 uint32_t CLKCR;          /*!< SDI clock control register , Address offset: 0x04 */
    --I0 uint32_t ARG;            /*!< SDIO argument register , Address offset: 0x08 */
    --I0 uint32_t CMD;            /*!< SDIO command register , Address offset: 0x0C */
    --I0 uint32_t RESPCMD;        /*!< SDIO command response register , Address offset: 0x10 */
    --I0 uint32_t RESP1;          /*!< SDIO response 1 register , Address offset: 0x14 */
    --I0 uint32_t RESP2;          /*!< SDIO response 2 register , Address offset: 0x18 */
    --I0 uint32_t RESP3;          /*!< SDIO response 3 register , Address offset: 0x1C */
    --I0 uint32_t RESP4;          /*!< SDIO response 4 register , Address offset: 0x20 */
    --I0 uint32_t DTIMER;         /*!< SDIO data timer register , Address offset: 0x24 */
    --I0 uint32_t DLEN;           /*!< SDIO data length register , Address offset: 0x28 */
    --I0 uint32_t DCTRL;          /*!< SDIO data control register , Address offset: 0x2C */
    --I0 uint32_t DCOUNT;         /*!< SDIO data counter register , Address offset: 0x30 */
    --I0 uint32_t STA;            /*!< SDIO status register , Address offset: 0x34 */
    --I0 uint32_t ICR;            /*!< SDIO interrupt clear register , Address offset: 0x38 */
    --I0 uint32_t MASK;           /*!< SDIO mask register , Address offset: 0x3C */
    uint32_t RESERVED[2];         /*!< Reserved , 0x40-0x44 */
    --I0 uint32_t FIFOCNT;        /*!< SDIO FIFO counter register , Address offset: 0x48 */
    uint32_t RESERVED1[13];       /*!< Reserved , 0x4C-0x7C */
    --I0 uint32_t FIFO;            /*!< SDIO data FIFO register , Address offset: 0x80 */
} SDIO_TypeDef;
/* * @brief Serial Peripheral Interface */

typedef struct
{
    --I0 uint16_t CR1;             /*!< SPI control register 1 (not used in I2S mode) , Address offset: 0x00 */
    uint16_t RESERVED;            /*!< Reserved , 0x02 */
    --I0 uint16_t CR2;             /*!< SPI control register 2 , Address offset: 0x04 */
    uint16_t RESERVED1;           /*!< Reserved , 0x06 */
    --I0 uint16_t SR;              /*!< SPI status register , Address offset: 0x08 */
}

```

```

/*!
 * !< Reserved , 0x0A
 * !< SPI data register , Address offset: 0x0C */
--IO uint16_t DR; /* !< Reserved , 0xE0
/* !< SPI CRC polynomial register (not used in I2S mode) , Address offset: 0x10 */
--IO uint16_t CRCPR; /* !< SPI CRC register , Address offset: 0x14 */
/* !< SPI RX CRC register (not used in I2S mode) , Address offset: 0x18 */
--IO uint16_t RESERVED4; /* !< Reserved , 0x12
/* !< SPI TX CRC register (not used in I2S mode) , Address offset: 0x1C */
--IO uint16_t RXCRCR; /* !< Reserved , 0x16
/* !< SPI_I2S configuration register , Address offset: 0x20 */
--IO uint16_t RESERVED5; /* !< Reserved , 0x1A
/* !< SPI_I2S prescaler register , Address offset: 0x22 */
--IO uint16_t TXCRCR; /* !< Reserved , 0x1E
/* !< SPI_I2S prescaler register , Address offset: 0x26 */
--IO uint16_t RESERVED6; /* !< Reserved , 0x2A
/* !< Reserved , 0x2E
--IO uint16_t I2SCFGR; /* !< Reserved , 0x32
/* !< Reserved , 0x36
--IO uint16_t RESERVED7; /* !< Reserved , 0x3A
/* !< Reserved , 0x3E
--IO uint16_t I2SPR; /* !< Reserved , 0x42
/* !< Reserved , 0x46
--IO uint16_t RESERVED8; /* !< Reserved , 0x4A
/* !< Reserved , 0x4E

} SPI_TypeDef;

/**
 * @brief TIM
 */
typedef struct
{
    /* !
     * !< TIM control register 1 , Address offset: 0x00 */
--IO uint16_t CR1; /* !< Reserved , 0x02
/* !< TIM control register 2 , Address offset: 0x04 */
--IO uint16_t RESERVED0; /* !< Reserved , 0x06
/* !< TIM slave mode control register , Address offset: 0x08 */
--IO uint16_t CR2; /* !< Reserved , 0x0A
/* !< TIM DMA/interrupt enable register , Address offset: 0x0C */
--IO uint16_t RESERVED1; /* !< Reserved , 0x0E
/* !< Reserved , 0x12
/* !< TIM status register , Address offset: 0x10 */
--IO uint16_t SMCR; /* !< Reserved , 0x14
/* !< TIM event generation register , Address offset: 0x18 */
--IO uint16_t DIER; /* !< Reserved , 0x16
/* !< Reserved , 0x1A
/* !< Reserved , 0x1E
--IO uint16_t RESERVED3; /* !< Reserved , 0x22
/* !< Reserved , 0x26
/* !< Reserved , 0x2A
--IO uint16_t SR; /* !< Reserved , 0x2E
/* !< Reserved , 0x32
/* !< Reserved , 0x36
--IO uint16_t EGR; /* !< Reserved , 0x3A
/* !< Reserved , 0x3E
--IO uint16_t RESERVED4; /* !< Reserved , 0x42
/* !< Reserved , 0x46
/* !< Reserved , 0x4A
--IO uint16_t CCMR1; /* !< Reserved , 0x52
/* !< Reserved , 0x56
--IO uint16_t RESERVED5; /* !< Reserved , 0x5A
/* !< Reserved , 0x5E
--IO uint16_t CCMR2; /* !< Reserved , 0x62
/* !< Reserved , 0x66
--IO uint16_t RESERVED6; /* !< Reserved , 0x6A
/* !< Reserved , 0x6E
}

```

```

--> IO uint16_t CCMR2;          /*!< TIM capture/compare mode register 2, Address offset: 0x1C */
--> uint16_t RESERVED7;         /*!< Reserved, 0x1E
--> --> IO uint16_t CCER;        /*!< TIM capture/compare enable register, Address offset: 0x20 */
--> --> uint16_t RESERVED8;      /*!< Reserved, 0x22
--> --> IO uint32_t CNT;        /*!< TIM counter register,
--> --> IO uint16_t PSC;        /*!< TIM prescaler,
--> --> uint16_t RESERVED9;      /*!< Reserved, 0x2A
--> --> IO uint32_t ARR;        /*!< TIM auto-reload register,
--> --> IO uint16_t RCR;        /*!< TIM repetition counter register,
--> --> uint16_t RESERVED10;     /*!< Reserved, 0x32
--> --> IO uint32_t CCR1;       /*!< TIM capture/compare register 1,
--> --> IO uint32_t CCR2;       /*!< TIM capture/compare register 2,
--> --> IO uint32_t CCR3;       /*!< TIM capture/compare register 3,
--> --> IO uint32_t CCR4;       /*!< TIM capture/compare register 4,
--> --> IO uint16_t BDTR;       /*!< TIM break and dead-time register,
--> --> uint16_t RESERVED11;     /*!< Reserved, 0x46
--> --> IO uint16_t DCR;        /*!< TIM DMA control register,
--> --> uint16_t RESERVED12;     /*!< Reserved, 0x4A
--> --> IO uint16_t DMAR;       /*!< TIM DMA address for full transfer,
--> --> uint16_t RESERVED13;     /*!< Reserved, 0x4E
--> --> IO uint16_t OR;         /*!< TIM option register,
--> --> uint16_t RESERVED14;     /*!< Reserved, 0x52
} TIM_TypeDef;

/*
 * @brief Universal Synchronous Asynchronous Receiver Transmitter
 */

typedef struct
{
    --> IO uint16_t SR;           /*!< USART Status register,
    uint16_t RESERVED0;           /*!< Reserved, 0x02
    --> IO uint16_t DR;           /*!< USART Data register,
    uint16_t RESERVED1;           /*!< Reserved, 0x06
    --> IO uint16_t BRR;          /*!< USART Baud rate register,
} USART_TypeDef;

```

```

/*!
 * !< Reserved , 0x0A
 * !< USART Control register 1,
 * !< Reserved , 0xE
 * !< USART Control register 2,
 * !< USART Control register 3,
 * !< USART Control register 4,
 * !< Reserved , 0x12
 * !< USART Control register 5,
 * !< Reserved , 0x16
 * !< USART Guard time and prescaler register , Address offset: 0x18 */
} USART_TypeDef;

/* @brief Window WATCHDOG
 */
typedef struct
{
    __IO uint32_t CR;      /* !< WWDG Control register , Address offset: 0x00 */
    __IO uint32_t CFR;     /* !< WWDG Configuration register , Address offset: 0x04 */
    __IO uint32_t SR;      /* !< WWDG Status register , Address offset: 0x08 */
} WWDG_TypeDef;

/* @brief Crypto Processor
 */
typedef struct
{
    __IO uint32_t CR;      /* !< CRYPT control register , Address offset: 0x00 */
    __IO uint32_t SR;      /* !< CRYPT status register , Address offset: 0x04 */
    __IO uint32_t DR;      /* !< CRYPT data input register , Address offset: 0x08 */
    __IO uint32_t DOUT;    /* !< CRYPT data output register , Address offset: 0x0C */
    __IO uint32_t DMACR;   /* !< CRYPT DMA control register , Address offset: 0x10 */
    __IO uint32_t IMSCR;   /* !< CRYPT interrupt mask set/clear register , Address offset: 0x14 */
    __IO uint32_t RISR;    /* !< CRYPT raw interrupt status register , Address offset: 0x18 */
}

```

```

-- IO uint32_t MISR;
-- IO uint32_t KOLR;
-- IO uint32_t KORR;
-- IO uint32_t K1LR;
-- IO uint32_t K1RR;
-- IO uint32_t K2LR;
-- IO uint32_t K2RR;
-- IO uint32_t IVOLR;
-- IO uint32_t K3LR;
-- IO uint32_t K3RR;
-- IO uint32_t IV0RR;
-- IO uint32_t IV1LR;
-- IO uint32_t IV1RR;
} CRYP_TypeDef;

/*
 * @brief HASH
 */
typedef struct
{
    -- 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];
} HASH_TypeDef;

```

---

```

Address offset: 0x1C /*
Address offset: 0x20 /*
Address offset: 0x24 /*
Address offset: 0x28 /*
Address offset: 0x2C /*
Address offset: 0x30 /*
Address offset: 0x34 /*
Address offset: 0x38 /*
Address offset: 0x3C /*
Address offset: 0x40 /*
Address offset: 0x44 /*
Address offset: 0x48 /*
Address offset: 0x4C */

/*
 * @brief HASH
 */

```

```

typedef struct
{
    -- IO  uint32_t CR;      /* !< RNG control register, Address offset: 0x00 */
    -- IO  uint32_t SR;      /* !< RNG status register, Address offset: 0x04 */
    -- IO  uint32_t DR;      /* !< RNG data register,   Address offset: 0x08 */
} RNG_TypeDef;

/** @addtogroup Peripheral_memory_map
 * @{
 */

#define FLASH_BASE ((uint32_t)0x08000000) /* !< FLASH(up to 1 MB) base address in the alias region
#define CCMDATARAM_BASE ((uint32_t)0x10000000) /* !< CCM(core coupled memory) data RAM(64 KB) base address in the alias region
#define SRAM1_BASE ((uint32_t)0x20000000) /* !< SRAM1(112 KB) base address in the alias region
#define SRAM2_BASE ((uint32_t)0x2001C000) /* !< SRAM2(16 KB) base address in the alias region
#define PERIPH_BASE ((uint32_t)0x40000000) /* !< Peripheral base address in the alias region
#define BKPSRAM_BASE ((uint32_t)0x40024000) /* !< Backup SRAM(4 KB) base address in the alias region
#define FSMC_R_BASE ((uint32_t)0xA0000000) /* !< FSMC registers base address
#define CCMDATARAM_BB_BASE ((uint32_t)0x12000000) /* !< CCM(core coupled memory) data RAM(64 KB) base address in the alias region
#define SRAM1_BB_BASE ((uint32_t)0x22000000) /* !< SRAM1(112 KB) base address in the bit-band region
#define SRAM2_BB_BASE ((uint32_t)0x2201C000) /* !< SRAM2(16 KB) base address in the bit-band region
*/

```

---

```

#define PERIPH_BB_BASE ((uint32_t)0x42000000) /*!< Peripheral base address in the bit-band region
*/
#define BKPSRAM_BB_BASE ((uint32_t)0x42024000) /*!< Backup SRAM(4 KB) base address in the bit-band region
*/

/* Legacy defines */
#define SRAM_BB_BASE SRAM1_BB_BASE

/*!< Peripheral memory map */
#define PERIPH_BASE (PERIPH_BASE + 0x0000)
#define PERIPH_BASE (PERIPH_BASE + 0x0400)
#define PERIPH_BASE (PERIPH_BASE + 0x0800)
#define PERIPH_BASE (PERIPH_BASE + 0x0C00)
#define PERIPH_BASE (PERIPH_BASE + 0x1000)
#define PERIPH_BASE (PERIPH_BASE + 0x1400)
#define PERIPH_BASE (PERIPH_BASE + 0x1800)
#define PERIPH_BASE (PERIPH_BASE + 0x1C00)
#define PERIPH_BASE (PERIPH_BASE + 0x2000)
#define PERIPH_BASE (PERIPH_BASE + 0x2800)
#define PERIPH_BASE (PERIPH_BASE + 0x2C00)
#define PERIPH_BASE (PERIPH_BASE + 0x3000)
#define PERIPH_BASE (PERIPH_BASE + 0x3400)
#define PERIPH_BASE (PERIPH_BASE + 0x3800)
#define PERIPH_BASE (PERIPH_BASE + 0x3C00)
#define PERIPH_BASE (PERIPH_BASE + 0x4000)
#define PERIPH_BASE (PERIPH_BASE + 0x4400)
#define PERIPH_BASE (PERIPH_BASE + 0x4800)

/*!< APB1 peripherals */
#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)

/* !< APB2 peripherals */
#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)

/* !< AHB1 peripherals */
#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)

/* !< AHB2 peripherals */
#define DCMI_BASE            (AHB2PERIPH_BASE + 0x50000)
#define CRYP_BASE             (AHB2PERIPH_BASE + 0x60000)
#define HASH_BASE             (AHB2PERIPH_BASE + 0x60400)
#define RNG_BASE              (AHB2PERIPH_BASE + 0x60800)

```

---

```

/* !< FSMC_Bankx registers base address */
#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)

/* Debug MCU registers base address */
#define DBGMCU_BASE ((uint32_t)0xE0042000)

/**
 * @}
 */

/** @addtogroup Peripheral_declaration
 */
#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 ((I2S2ext_TypeDef *) I2S2ext_BASE)
#define SPI2 ((SPI_TypeDef *) SPI2_BASE)
#define SPI3 ((SPI_TypeDef *) SPI3_BASE)
#define I2S3ext ((I2S3ext_TypeDef *) I2S3ext_BASE)
#define USART2 ((USART_TypeDef *) USART2_BASE)
#define USART3 ((USART_TypeDef *) USART3_BASE)

```

---

---

```

#define UART4      ((USART_TypeDef *) USART4_BASE)
#define UART5      ((USART_TypeDef *) USART5_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_StreamTypeDef *) DMA1_Stream0_BASE)
#define DMA1_Stream1 ((DMA_StreamTypeDef *) DMA1_Stream1_BASE)
#define DMA1_Stream2 ((DMA_StreamTypeDef *) DMA1_Stream2_BASE)
#define DMA1_Stream3 ((DMA_StreamTypeDef *) DMA1_Stream3_BASE)
#define DMA1_Stream4 ((DMA_StreamTypeDef *) DMA1_Stream4_BASE)
#define DMA1_Stream5 ((DMA_StreamTypeDef *) DMA1_Stream5_BASE)
#define DMA1_Stream6 ((DMA_StreamTypeDef *) DMA1_Stream6_BASE)
#define DMA1_Stream7 ((DMA_StreamTypeDef *) DMA1_Stream7_BASE)

#define DMA2 ((DMA_TypeDef *) DMA2_BASE)
#define DMA2_Stream0 ((DMA_StreamTypeDef *) DMA2_Stream0_BASE)
#define DMA2_Stream1 ((DMA_StreamTypeDef *) DMA2_Stream1_BASE)
#define DMA2_Stream2 ((DMA_StreamTypeDef *) DMA2_Stream2_BASE)
#define DMA2_Stream3 ((DMA_StreamTypeDef *) DMA2_Stream3_BASE)
#define DMA2_Stream4 ((DMA_StreamTypeDef *) DMA2_Stream4_BASE)
#define DMA2_Stream5 ((DMA_StreamTypeDef *) DMA2_Stream5_BASE)
#define DMA2_Stream6 ((DMA_StreamTypeDef *) DMA2_Stream6_BASE)
#define DMA2_Stream7 ((DMA_StreamTypeDef *) DMA2_Stream7_BASE)

#define ERH ((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)

/* */
*/
```

```

/** @addtogroup Exported_constants
 */
/* @defgroup Peripheral_Registers_Bits_Definition
 */
/* **** Peripheral Registers_Bits_Definition ****/
/* **** Analog to Digital Converter ****/
/* **** Bit definition for ADC_SR register ****/
/* !<ADC_SR_AWD ((uint8_t)0x01) /* !<Analog watchdog flag */
/* !<ADC_SR_EOC ((uint8_t)0x02) /* !<End of conversion */
/* !<ADC_SR_JEOC ((uint8_t)0x04) /* !<Injected channel end of conversion */
/* !<ADC_SR_JSTRT ((uint8_t)0x08) /* !<Injected channel Start flag */
/* !<ADC_SR_STRI ((uint8_t)0x10) /* !<Regular channel Start flag */
/* !<ADC_SR_OVR ((uint8_t)0x20) /* !<Overrun flag */

/* **** Bit definition for ADC_CR1 register ****/
/* !<ADC_CR1_AWDCH ((uint32_t)0x0000001F) /* !<AWDCH[4:0] bits (Analog watchdog chan
/* !<ADC_CR1_AWDCH_0 ((uint32_t)0x00000001) /* !<Bit 0 */
/* !<ADC_CR1_AWDCH_1 ((uint32_t)0x00000002) /* !<Bit 1 */
/* !<ADC_CR1_AWDCH_2 ((uint32_t)0x00000004) /* !<Bit 2 */
/* !<ADC_CR1_AWDCH_3 ((uint32_t)0x00000008) /* !<Bit 3 */
/* !<ADC_CR1_AWDCH_4 ((uint32_t)0x00000010) /* !<Bit 4 */
/* !<ADC_CR1_EOCIE ((uint32_t)0x00000020) /* !<Interrupt enable for EOC */
/* !<ADC_CR1_AWDIE ((uint32_t)0x00000040) /* !<Analog Watchdog interrupt enable */
/* !<ADC_CR1_JEOCIE ((uint32_t)0x00000080) /* !<Interrupt enable for injected channel
*/

```

```

/* !<Scan mode */
/* !<Enable the watchdog on a single channel */
/* !<Automatic injected group conversion */
/* !<Discontinuous mode on regular channel */
/* !<Discontinuous mode on injected channel */
/* !<DISCNUM [2:0] bits (Discontinuous mode)
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */
/* !<Analog watchdog enable on injected channel */
/* !<Analog watchdog enable on regular channel */
/* !<RES [2:0] bits (Resolution) */
/* !<Bit 0 */
/* !<Bit 1 */
/* !<overrun interrupt enable */

/* **** Bit definition for ADC_CR2 register **** */
/* !<A/D Converter ON / OFF */
/* !<Continuous Conversion */
/* !<Direct Memory access mode */
/* !<DMA disable selection (Single ADC) */
/* !<End of conversion selection */
/* !<Data Alignment */
/* !<JEXTSEL[3:0] bits (External event select)
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */
/* !<Bit 3 */
/* !<JEXTEN[1:0] bits (External Trigger Control)
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Start Conversion of injected channels */
/* !<EXTSEL[3:0] bits (External Event Selection)
/* !<Bit 0 */
/* !<Bit 1 */

/* **** ADC_CR2_DEFINITION **** */
#define ADC_CR1_SCAN ((uint32_t)0x000000100)
#define ADC_CR1_AWDGSL ((uint32_t)0x000000200)
#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)0x00040000)
#define ADC_CR1_AWDEN ((uint32_t)0x00080000)
#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)0x00100000)
#define ADC_CR2_JEXTSEL_1 ((uint32_t)0x00200000)
#define ADC_CR2_JEXTSEL_2 ((uint32_t)0x00400000)
#define ADC_CR2_JEXTSEL_3 ((uint32_t)0x00800000)
#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_JEXTEN_2 ((uint32_t)0x00400000)
#define ADC_CR2_JEXTEN_3 ((uint32_t)0x00800000)
#define ADC_CR2_JSWSTART ((uint32_t)0x00F00000)
#define ADC_CR2_EXTSEL ((uint32_t)0x01000000)
#define ADC_CR2_EXTSEL_0 ((uint32_t)0x02000000)
#define ADC_CR2_EXTSEL_1 ((uint32_t)0x04000000)

```

```

Глава 21. Старт стандартной библиотеки STM32

/*
 * Bit definition for ADC_SMPR1 register *****
 */
#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)

/*
***** Bit definition for ADC_SMPR1 register *****
*/
#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_SMP13_3 ((uint32_t)0x00007000)
#define ADC_SMPR1_SMP14 ((uint32_t)0x00001000)
#define ADC_SMPR1_SMP14_0 ((uint32_t)0x00002000)
#define ADC_SMPR1_SMP14_1 ((uint32_t)0x00004000)
#define ADC_SMPR1_SMP14_2 ((uint32_t)0x00003800)
#define ADC_SMPR1_SMP14_3 ((uint32_t)0x00008000)
#define ADC_SMPR1_SMP14_4 ((uint32_t)0x00010000)
#define ADC_SMPR1_SMP14_5 ((uint32_t)0x00020000)
#define ADC_SMPR1_SMP14_6 ((uint32_t)0x00040000)
#define ADC_SMPR1_SMP14_7 ((uint32_t)0x00080000)
#define ADC_SMPR1_SMP14_8 ((uint32_t)0x00100000)
#define ADC_SMPR1_SMP14_9 ((uint32_t)0x00200000)
#define ADC_SMPR1_SMP14_10 ((uint32_t)0x00400000)
#define ADC_SMPR1_SMP14_11 ((uint32_t)0x00800000)
#define ADC_SMPR1_SMP14_12 ((uint32_t)0x01000000)
#define ADC_SMPR1_SMP14_13 ((uint32_t)0x02000000)
#define ADC_SMPR1_SMP14_14 ((uint32_t)0x04000000)
#define ADC_SMPR1_SMP14_15 ((uint32_t)0x08000000)
#define ADC_SMPR1_SMP14_16 ((uint32_t)0x10000000)
#define ADC_SMPR1_SMP14_17 ((uint32_t)0x20000000)
#define ADC_SMPR1_SMP14_18 ((uint32_t)0x40000000)
#define ADC_SMPR1_SMP14_19 ((uint32_t)0x80000000)
#define ADC_SMPR1_SMP14_20 ((uint32_t)0x100000000)
#define ADC_SMPR1_SMP14_21 ((uint32_t)0x200000000)
#define ADC_SMPR1_SMP14_22 ((uint32_t)0x400000000)
#define ADC_SMPR1_SMP14_23 ((uint32_t)0x800000000)
#define ADC_SMPR1_SMP14_24 ((uint32_t)0x1000000000)
#define ADC_SMPR1_SMP14_25 ((uint32_t)0x2000000000)
#define ADC_SMPR1_SMP14_26 ((uint32_t)0x4000000000)
#define ADC_SMPR1_SMP14_27 ((uint32_t)0x8000000000)
#define ADC_SMPR1_SMP14_28 ((uint32_t)0x10000000000)
#define ADC_SMPR1_SMP14_29 ((uint32_t)0x20000000000)
#define ADC_SMPR1_SMP14_30 ((uint32_t)0x40000000000)
#define ADC_SMPR1_SMP14_31 ((uint32_t)0x80000000000)

/*
 * Bit definition for External Trigger Control
 */
#define EXTE_N ((uint32_t)0x00000001)
#define EXTE_T ((uint32_t)0x00000002)
#define EXTE_E ((uint32_t)0x00000004)
#define EXTE_R ((uint32_t)0x00000008)
#define EXTE_W ((uint32_t)0x00000010)
#define EXTE_M ((uint32_t)0x00000020)
#define EXTE_C ((uint32_t)0x00000040)
#define EXTE_S ((uint32_t)0x00000080)
#define EXTE_T1 ((uint32_t)0x00000100)
#define EXTE_T2 ((uint32_t)0x00000200)
#define EXTE_T3 ((uint32_t)0x00000400)
#define EXTE_T4 ((uint32_t)0x00000800)
#define EXTE_T5 ((uint32_t)0x00001000)
#define EXTE_T6 ((uint32_t)0x00002000)
#define EXTE_T7 ((uint32_t)0x00004000)
#define EXTE_T8 ((uint32_t)0x00008000)
#define EXTE_T9 ((uint32_t)0x00010000)
#define EXTE_T10 ((uint32_t)0x00020000)
#define EXTE_T11 ((uint32_t)0x00040000)
#define EXTE_T12 ((uint32_t)0x00080000)
#define EXTE_T13 ((uint32_t)0x00100000)
#define EXTE_T14 ((uint32_t)0x00200000)
#define EXTE_T15 ((uint32_t)0x00400000)
#define EXTE_T16 ((uint32_t)0x00800000)
#define EXTE_T17 ((uint32_t)0x01000000)
#define EXTE_T18 ((uint32_t)0x02000000)
#define EXTE_T19 ((uint32_t)0x04000000)
#define EXTE_T20 ((uint32_t)0x08000000)
#define EXTE_T21 ((uint32_t)0x10000000)
#define EXTE_T22 ((uint32_t)0x20000000)
#define EXTE_T23 ((uint32_t)0x40000000)
#define EXTE_T24 ((uint32_t)0x80000000)
#define EXTE_T25 ((uint32_t)0x100000000)
#define EXTE_T26 ((uint32_t)0x200000000)
#define EXTE_T27 ((uint32_t)0x400000000)
#define EXTE_T28 ((uint32_t)0x800000000)
#define EXTE_T29 ((uint32_t)0x1000000000)
#define EXTE_T30 ((uint32_t)0x2000000000)
#define EXTE_T31 ((uint32_t)0x4000000000)

```

```

/* !<Bit 2 */          /* !<SMP17[2:0] bits (Channel 17 Sample time)
((uint32_t)0x00100000) ((uint32_t)0x00E00000) ((uint32_t)0x00200000)
((uint32_t)0x00400000) ((uint32_t)0x00800000) ((uint32_t)0x00080000)
((uint32_t)0x00008000) ((uint32_t)0x00000800) ((uint32_t)0x00000080)
((uint32_t)0x00000008) ((uint32_t)0x00000000) ((uint32_t)0x00000000)

/* !<Bit 0 */          /* !<SMP0[2:0] bits (Channel 0 Sample time)
((uint32_t)0x00000007) ((uint32_t)0x00000001) ((uint32_t)0x00000002)
((uint32_t)0x00000004) ((uint32_t)0x00000038) ((uint32_t)0x00000008)
((uint32_t)0x00000010) ((uint32_t)0x00000020) ((uint32_t)0x00000040)
((uint32_t)0x00000080) ((uint32_t)0x00000100) ((uint32_t)0x00000000)
((uint32_t)0x00000000) ((uint32_t)0x00000000) ((uint32_t)0x00000000)

/* !<Bit 1 */          /* !<SMP1[2:0] bits (Channel 1 Sample time)
((uint32_t)0x00000002) ((uint32_t)0x00000004) ((uint32_t)0x00000008)
((uint32_t)0x00000010) ((uint32_t)0x00000020) ((uint32_t)0x00000040)
((uint32_t)0x00000080) ((uint32_t)0x00000100) ((uint32_t)0x00000000)
((uint32_t)0x00000000) ((uint32_t)0x00000000) ((uint32_t)0x00000000)

/* !<Bit 2 */          /* !<SMP2[2:0] bits (Channel 2 Sample time)
((uint32_t)0x00000001) ((uint32_t)0x00000003) ((uint32_t)0x00000007)
((uint32_t)0x00000011) ((uint32_t)0x00000023) ((uint32_t)0x00000047)
((uint32_t)0x00000087) ((uint32_t)0x00000107) ((uint32_t)0x00000000)
((uint32_t)0x00000000) ((uint32_t)0x00000000) ((uint32_t)0x00000000)

/* !<Bit 0 */          /* !<SMP3[2:0] bits (Channel 3 Sample time)
((uint32_t)0x00000000) ((uint32_t)0x00000002) ((uint32_t)0x00000006)
((uint32_t)0x00000004) ((uint32_t)0x00000008) ((uint32_t)0x00000012)
((uint32_t)0x00000020) ((uint32_t)0x00000040) ((uint32_t)0x00000000)
((uint32_t)0x00000000) ((uint32_t)0x00000000) ((uint32_t)0x00000000)

/* !<Bit 1 */          /* !<SMP4[2:0] bits (Channel 4 Sample time)
((uint32_t)0x00000001) ((uint32_t)0x00000003) ((uint32_t)0x00000007)
((uint32_t)0x00000005) ((uint32_t)0x00000009) ((uint32_t)0x00000013)
((uint32_t)0x00000021) ((uint32_t)0x00000041) ((uint32_t)0x00000000)
((uint32_t)0x00000000) ((uint32_t)0x00000000) ((uint32_t)0x00000000)

/* !<Bit 2 */          /* !<SMP5[2:0] bits (Channel 5 Sample time)
((uint32_t)0x00000000) ((uint32_t)0x00000002) ((uint32_t)0x00000006)
((uint32_t)0x00000004) ((uint32_t)0x00000008) ((uint32_t)0x00000012)
((uint32_t)0x00000020) ((uint32_t)0x00000040) ((uint32_t)0x00000000)
((uint32_t)0x00000000) ((uint32_t)0x00000000) ((uint32_t)0x00000000)

/* !<Bit 0 */          /* !<Bit 0 */
/* !<Bit 1 */          /* !<Bit 1 */
/* !<Bit 2 */          /* !<Bit 2 */

/* ***** Bit definition for ADC_SMPR2 register *****/
#define ADC_SMPR2_SMPO ((uint32_t)0x00000000)
#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)0x00000008)
#define ADC_SMPR2_SMP1_0 ((uint32_t)0x00000010)
#define ADC_SMPR2_SMP1_1 ((uint32_t)0x00000011)
#define ADC_SMPR2_SMP1_2 ((uint32_t)0x00000012)
#define ADC_SMPR2_SMP2 ((uint32_t)0x00000014)
#define ADC_SMPR2_SMP2_0 ((uint32_t)0x00000015)
#define ADC_SMPR2_SMP2_1 ((uint32_t)0x00000016)
#define ADC_SMPR2_SMP2_2 ((uint32_t)0x00000017)
#define ADC_SMPR2_SMP3 ((uint32_t)0x0000001B)
#define ADC_SMPR2_SMP3_0 ((uint32_t)0x0000001C)
#define ADC_SMPR2_SMP3_1 ((uint32_t)0x0000001D)
#define ADC_SMPR2_SMP3_2 ((uint32_t)0x0000001E)
#define ADC_SMPR2_SMP4 ((uint32_t)0x0000001F)
#define ADC_SMPR2_SMP4_0 ((uint32_t)0x00000020)
#define ADC_SMPR2_SMP4_1 ((uint32_t)0x00000021)
#define ADC_SMPR2_SMP4_2 ((uint32_t)0x00000022)
#define ADC_SMPR2_SMP5 ((uint32_t)0x00000023)
#define ADC_SMPR2_SMP5_0 ((uint32_t)0x00000024)
#define ADC_SMPR2_SMP5_1 ((uint32_t)0x00000025)
#define ADC_SMPR2_SMP5_2 ((uint32_t)0x00000026)

```

```

#define ADC_SMPR2_SMP6 ((uint32_t)0x0001C00000)
#define ADC_SMPR2_SMP6_0 ((uint32_t)0x0000400000)
#define ADC_SMPR2_SMP6_1 ((uint32_t)0x0000800000)
#define ADC_SMPR2_SMP6_2 ((uint32_t)0x0001000000)
#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)

/**************** Bit definition for ADC_JOFR1 register */
#define ADC_JOFR1_JOFFSET1 ((uint16_t)0x0FFF) /*!<Data offset for injected channel 1 */

/**************** Bit definition for ADC_JOFR2 register */
#define ADC_JOFR2_JOFFSET2 ((uint16_t)0x0FFF) /*!<Data offset for injected channel 2 */

/**************** Bit definition for ADC_JOFR3 register */
#define ADC_JOFR3_JOFFSET3 ((uint16_t)0x0FFF) /*!<Data offset for injected channel 3 */

/**************** Bit definition for ADC_JOFR4 register */
#define ADC_JOFR4_JOFFSET4 ((uint16_t)0x0FFF) /*!<Data offset for injected channel 4 */

/**************** Bit definition for ADC_HTR register */
#define ADC_HTR_HT ((uint16_t)0x0FFF) /*!<Analog watchdog high threshold */

/**************** Bit definition for ADC_LTR register */
#define ADC_LTR_LT ((uint16_t)0x0FFF) /*!<Analog watchdog low threshold */

```

```

/*
***** Bit definition for ADC_SQR1 register *****/
#define ADC_SQR1_SQ13 ((uint32_t)0x00000001F)
#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)0x0000003E0)
#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)0x000007C00)
#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)

/*
***** Bit definition for ADC_SQR2 register *****/
#define ADC_SQR2_SQ7 ((uint32_t)0x0000001F)
#define ADC_SQR2_SQ7_0 ((uint32_t)0x00000001)
#define ADC_SQR2_SQ7_1 ((uint32_t)0x00000002)

/*
***** Bit definition for ADC_SQR3 register *****/
#define ADC_SQR3_SQ15 ((uint32_t)0x00000001F)
#define ADC_SQR3_SQ15_0 ((uint32_t)0x00000001)
#define ADC_SQR3_SQ15_1 ((uint32_t)0x00000002)
#define ADC_SQR3_SQ15_2 ((uint32_t)0x00000004)
#define ADC_SQR3_SQ15_3 ((uint32_t)0x00000008)
#define ADC_SQR3_SQ15_4 ((uint32_t)0x00000010)
#define ADC_SQR3_SQ15_5 ((uint32_t)0x00000020)
#define ADC_SQR3_SQ15_6 ((uint32_t)0x00000040)
#define ADC_SQR3_SQ15_7 ((uint32_t)0x00000080)
#define ADC_SQR3_SQ15_8 ((uint32_t)0x00000100)
#define ADC_SQR3_SQ15_9 ((uint32_t)0x00000200)
#define ADC_SQR3_SQ15_10 ((uint32_t)0x00000400)
#define ADC_SQR3_SQ15_11 ((uint32_t)0x00000800)
#define ADC_SQR3_SQ15_12 ((uint32_t)0x00001000)
#define ADC_SQR3_SQ15_13 ((uint32_t)0x00002000)
#define ADC_SQR3_SQ15_14 ((uint32_t)0x00004000)
#define ADC_SQR3_SQ15_15 ((uint32_t)0x00008000)
#define ADC_SQR3_SQ15_16 ((uint32_t)0x000F0000)
#define ADC_SQR3_SQ15_17 ((uint32_t)0x00100000)
#define ADC_SQR3_SQ15_18 ((uint32_t)0x00200000)
#define ADC_SQR3_SQ15_19 ((uint32_t)0x00400000)
#define ADC_SQR3_SQ15_20 ((uint32_t)0x00800000)
#define ADC_SQR3_SQ15_21 ((uint32_t)0x0F000000)
#define ADC_SQR3_SQ15_22 ((uint32_t)0x10000000)
#define ADC_SQR3_SQ15_23 ((uint32_t)0x20000000)
#define ADC_SQR3_SQ15_24 ((uint32_t)0x40000000)
#define ADC_SQR3_SQ15_25 ((uint32_t)0x80000000)
#define ADC_SQR3_SQ15_26 ((uint32_t)0xF0000000)
#define ADC_SQR3_SQ15_27 ((uint32_t)0x100000000)
#define ADC_SQR3_SQ15_28 ((uint32_t)0x200000000)
#define ADC_SQR3_SQ15_29 ((uint32_t)0x400000000)
#define ADC_SQR3_SQ15_30 ((uint32_t)0x800000000)
#define ADC_SQR3_SQ15_31 ((uint32_t)0xF00000000)

```

```

#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)0x00004000)
#define ADC_SQR2_SQ9_1 ((uint32_t)0x00008000)
#define ADC_SQR2_SQ9_2 ((uint32_t)0x00010000)
#define ADC_SQR2_SQ9_3 ((uint32_t)0x00020000)
#define ADC_SQR2_SQ9_4 ((uint32_t)0x00040000)
#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)

/* !<Bit 2 */ ((uint32_t)0x00000008)
/* !<Bit 3 */ ((uint32_t)0x00000010)
/* !<Bit 4 */ ((uint32_t)0x000003E0)
/* !<SQ8 [4:0] bits (8th conversion in regu
/* !<Bit 0 */ ((uint32_t)0x00000020)
/* !<Bit 1 */ ((uint32_t)0x00000040)
/* !<Bit 2 */ ((uint32_t)0x00000080)
/* !<Bit 3 */ ((uint32_t)0x00000100)
/* !<Bit 4 */ ((uint32_t)0x00000200)
/* !<SQ9 [4:0] bits (9th conversion in regu
/* !<Bit 0 */ ((uint32_t)0x00000400)
/* !<Bit 1 */ ((uint32_t)0x00000800)
/* !<Bit 2 */ ((uint32_t)0x00001000)
/* !<Bit 3 */ ((uint32_t)0x00002000)
/* !<Bit 4 */ ((uint32_t)0x00004000)
/* !<SQ10 [4:0] bits (10th conversion in regu
/* !<Bit 0 */ ((uint32_t)0x00008000)
/* !<Bit 1 */ ((uint32_t)0x00010000)
/* !<Bit 2 */ ((uint32_t)0x00020000)
/* !<Bit 3 */ ((uint32_t)0x00040000)
/* !<Bit 4 */ ((uint32_t)0x00080000)
/* !<SQ11 [4:0] bits (11th conversion in regu
/* !<Bit 0 */ ((uint32_t)0x00100000)
/* !<Bit 1 */ ((uint32_t)0x00200000)
/* !<Bit 2 */ ((uint32_t)0x00400000)
/* !<Bit 3 */ ((uint32_t)0x00800000)
/* !<Bit 4 */ ((uint32_t)0x01000000)
/* !<SQ12 [4:0] bits (12th conversion in regu
/* !<Bit 0 */ ((uint32_t)0x02000000)
/* !<Bit 1 */ ((uint32_t)0x04000000)
/* !<Bit 2 */ ((uint32_t)0x08000000)
/* !<Bit 3 */ ((uint32_t)0x10000000)
/* !<Bit 4 */ ((uint32_t)0x20000000)

***** Bit definition for ADC_SQR3 register *****

```

```

#define ADC_SQR3_SQ1 ((uint32_t)0x00000001F)
#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)0x0000003E0)
#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)0x00007C0)
#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)0x0000F8000)
#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_SQ5_5 ((uint32_t)0x02000000)
#define ADC_SQR3_SQ5_6 ((uint32_t)0x04000000)
#define ADC_SQR3_SQ5_7 ((uint32_t)0x08000000)
#define ADC_SQR3_SQ5_8 ((uint32_t)0x10000000)

/* !<SQ1 [4:0] bits (1st conversion in regu
   ADC_SQR3_SQ1_0
   ADC_SQR3_SQ1_1
   ADC_SQR3_SQ1_2
   ADC_SQR3_SQ1_3
   ADC_SQR3_SQ1_4
   ADC_SQR3_SQ2_0
   ADC_SQR3_SQ2_1
   ADC_SQR3_SQ2_2
   ADC_SQR3_SQ2_3
   ADC_SQR3_SQ2_4
   ADC_SQR3_SQ3_0
   ADC_SQR3_SQ3_1
   ADC_SQR3_SQ3_2
   ADC_SQR3_SQ3_3
   ADC_SQR3_SQ3_4
   ADC_SQR3_SQ4_0
   ADC_SQR3_SQ4_1
   ADC_SQR3_SQ4_2
   ADC_SQR3_SQ4_3
   ADC_SQR3_SQ4_4
   ADC_SQR3_SQ5_0
   ADC_SQR3_SQ5_1
   ADC_SQR3_SQ5_2
   ADC_SQR3_SQ5_3
   ADC_SQR3_SQ5_4
   ADC_SQR3_SQ5_5
   ADC_SQR3_SQ5_6
   ADC_SQR3_SQ5_7
   ADC_SQR3_SQ5_8
   ADC_SQR3_SQ6_0
   ADC_SQR3_SQ6_1
   ADC_SQR3_SQ6_2
   ADC_SQR3_SQ6_3

/* !<Bit 0 */
   ADC_SQR3_SQ1_0
   ADC_SQR3_SQ1_1
   ADC_SQR3_SQ1_2
   ADC_SQR3_SQ1_3
   ADC_SQR3_SQ1_4
   ADC_SQR3_SQ2_0
   ADC_SQR3_SQ2_1
   ADC_SQR3_SQ2_2
   ADC_SQR3_SQ2_3
   ADC_SQR3_SQ2_4
   ADC_SQR3_SQ3_0
   ADC_SQR3_SQ3_1
   ADC_SQR3_SQ3_2
   ADC_SQR3_SQ3_3
   ADC_SQR3_SQ3_4
   ADC_SQR3_SQ4_0
   ADC_SQR3_SQ4_1
   ADC_SQR3_SQ4_2
   ADC_SQR3_SQ4_3
   ADC_SQR3_SQ4_4
   ADC_SQR3_SQ5_0
   ADC_SQR3_SQ5_1
   ADC_SQR3_SQ5_2
   ADC_SQR3_SQ5_3
   ADC_SQR3_SQ5_4
   ADC_SQR3_SQ5_5
   ADC_SQR3_SQ5_6
   ADC_SQR3_SQ5_7
   ADC_SQR3_SQ5_8
   ADC_SQR3_SQ6_0
   ADC_SQR3_SQ6_1
   ADC_SQR3_SQ6_2
   ADC_SQR3_SQ6_3

/* !<Bit 1 */
   ADC_SQR3_SQ1_1
   ADC_SQR3_SQ1_2
   ADC_SQR3_SQ1_3
   ADC_SQR3_SQ1_4
   ADC_SQR3_SQ2_1
   ADC_SQR3_SQ2_2
   ADC_SQR3_SQ2_3
   ADC_SQR3_SQ2_4
   ADC_SQR3_SQ3_1
   ADC_SQR3_SQ3_2
   ADC_SQR3_SQ4_1
   ADC_SQR3_SQ4_2
   ADC_SQR3_SQ4_3
   ADC_SQR3_SQ4_4
   ADC_SQR3_SQ5_1
   ADC_SQR3_SQ5_2
   ADC_SQR3_SQ5_3
   ADC_SQR3_SQ5_4
   ADC_SQR3_SQ5_5
   ADC_SQR3_SQ5_6
   ADC_SQR3_SQ5_7
   ADC_SQR3_SQ5_8
   ADC_SQR3_SQ6_1
   ADC_SQR3_SQ6_2
   ADC_SQR3_SQ6_3

/* !<Bit 2 */
   ADC_SQR3_SQ1_2
   ADC_SQR3_SQ1_3
   ADC_SQR3_SQ1_4
   ADC_SQR3_SQ2_2
   ADC_SQR3_SQ2_3
   ADC_SQR3_SQ3_2
   ADC_SQR3_SQ4_2
   ADC_SQR3_SQ4_3
   ADC_SQR3_SQ5_2
   ADC_SQR3_SQ5_3
   ADC_SQR3_SQ5_4
   ADC_SQR3_SQ5_5
   ADC_SQR3_SQ5_6
   ADC_SQR3_SQ5_7
   ADC_SQR3_SQ5_8
   ADC_SQR3_SQ6_2
   ADC_SQR3_SQ6_3

/* !<Bit 3 */
   ADC_SQR3_SQ1_3
   ADC_SQR3_SQ1_4
   ADC_SQR3_SQ2_3
   ADC_SQR3_SQ3_3
   ADC_SQR3_SQ4_3
   ADC_SQR3_SQ5_3
   ADC_SQR3_SQ5_4
   ADC_SQR3_SQ5_5
   ADC_SQR3_SQ5_6
   ADC_SQR3_SQ5_7
   ADC_SQR3_SQ5_8
   ADC_SQR3_SQ6_3

/* !<Bit 4 */
   ADC_SQR3_SQ1_4
   ADC_SQR3_SQ2_4
   ADC_SQR3_SQ3_4
   ADC_SQR3_SQ4_4
   ADC_SQR3_SQ5_4
   ADC_SQR3_SQ5_5
   ADC_SQR3_SQ5_6
   ADC_SQR3_SQ5_7
   ADC_SQR3_SQ5_8

/* !<Bit 5 */
   ADC_SQR3_SQ2_4
   ADC_SQR3_SQ3_4
   ADC_SQR3_SQ4_4
   ADC_SQR3_SQ5_4
   ADC_SQR3_SQ5_5
   ADC_SQR3_SQ5_6
   ADC_SQR3_SQ5_7
   ADC_SQR3_SQ5_8

/* !<Bit 6 */
   ADC_SQR3_SQ3_4
   ADC_SQR3_SQ4_4
   ADC_SQR3_SQ5_4
   ADC_SQR3_SQ5_5
   ADC_SQR3_SQ5_6
   ADC_SQR3_SQ5_7
   ADC_SQR3_SQ5_8

```

```

#define ADC_SQR3_SQ6_4 ((uint32_t)0x20000000) /* !<Bit 4 */

/* ***** Bit definition for ADC_JSQR register *****/
#define ADC_JSQR_JSQ1 ((uint32_t)0x0000001F) /* !<JSQ1 [4:0] bits (1st conversion in inj)
#define ADC_JSQR_JSQ1_0 ((uint32_t)0x00000001) /* !<Bit 0 */
#define ADC_JSQR_JSQ1_1 ((uint32_t)0x00000002) /* !<Bit 1 */
#define ADC_JSQR_JSQ1_2 ((uint32_t)0x00000004) /* !<Bit 2 */
#define ADC_JSQR_JSQ1_3 ((uint32_t)0x00000008) /* !<Bit 3 */
#define ADC_JSQR_JSQ1_4 ((uint32_t)0x00000010) /* !<Bit 4 */
#define ADC_JSQR_JSQ2 ((uint32_t)0x000003E0) /* !<JSQ2 [4:0] bits (2nd conversion in inj)
#define ADC_JSQR_JSQ2_0 ((uint32_t)0x00000020) /* !<Bit 0 */
#define ADC_JSQR_JSQ2_1 ((uint32_t)0x00000040) /* !<Bit 1 */
#define ADC_JSQR_JSQ2_2 ((uint32_t)0x00000080) /* !<Bit 2 */
#define ADC_JSQR_JSQ2_3 ((uint32_t)0x00000100) /* !<Bit 3 */
#define ADC_JSQR_JSQ2_4 ((uint32_t)0x00000200) /* !<Bit 4 */
#define ADC_JSQR_JSQ3 ((uint32_t)0x000007C0) /* !<JSQ3 [4:0] bits (3rd conversion in inj)
#define ADC_JSQR_JSQ3_0 ((uint32_t)0x00000400) /* !<Bit 0 */
#define ADC_JSQR_JSQ3_1 ((uint32_t)0x00000800) /* !<Bit 1 */
#define ADC_JSQR_JSQ3_2 ((uint32_t)0x00001000) /* !<Bit 2 */
#define ADC_JSQR_JSQ3_3 ((uint32_t)0x00002000) /* !<Bit 3 */
#define ADC_JSQR_JSQ3_4 ((uint32_t)0x00004000) /* !<Bit 4 */
#define ADC_JSQR_JSQ4 ((uint32_t)0x000F8000) /* !<JSQ4 [4:0] bits (4th conversion in inj)
#define ADC_JSQR_JSQ4_0 ((uint32_t)0x00008000) /* !<Bit 0 */
#define ADC_JSQR_JSQ4_1 ((uint32_t)0x00010000) /* !<Bit 1 */
#define ADC_JSQR_JSQ4_2 ((uint32_t)0x00020000) /* !<Bit 2 */
#define ADC_JSQR_JSQ4_3 ((uint32_t)0x00040000) /* !<Bit 3 */
#define ADC_JSQR_JSQ4_4 ((uint32_t)0x00080000) /* !<Bit 4 */
#define ADC_JSQR_JL ((uint32_t)0x00300000) /* !<JL [1:0] bits (Injected Sequence length)
#define ADC_JSQR_JL_0 ((uint32_t)0x00100000) /* !<Bit 0 */
#define ADC_JSQR_JL_1 ((uint32_t)0x00200000) /* !<Bit 1 */

/* ***** Bit definition for ADC_JDR1 register *****/
#define ADC_JDR1_JDATA ((uint16_t)0xFFFF) /* !<Injected data */

/* ***** Bit definition for ADC_JDR2 register *****/
#define ADC_JDR1_JDATA

```

```

/* !< Injected data */
((uint16_t)0xFFFF)

/* ***** Bit definition for ADC_JDR3 register *****/
#define ADC_JDR3_JDATA ((uint16_t)0xFFFF) /* !< Injected data */

/* ***** Bit definition for ADC_JDR4 register *****/
#define ADC_JDR4_JDATA ((uint16_t)0xFFFF) /* !< Injected data */

/* ***** Bit definition for ADC_DR register *****/
#define ADC_DR_DATA ((uint32_t)0x0000FFFF) /* !< Regular data */
#define ADC_DR_ADC2DATA ((uint32_t)0xFFFFF0000) /* !< ADC2 data */

/* ***** Bit definition for ADC_CSR register *****/
#define ADC_CSR_AWD1 ((uint32_t)0x00000001) /* !< ADC1 Analog watchdog flag */
#define ADC_CSR_EOC1 ((uint32_t)0x00000002) /* !< ADC1 End of conversion */
#define ADC_CSR_JEOC1 ((uint32_t)0x00000004) /* !< ADC1 Injected channel end of conversion */
#define ADC_CSR_JSTRT1 ((uint32_t)0x00000008) /* !< ADC1 Injected channel Start flag */
#define ADC_CSR_STRT1 ((uint32_t)0x00000010) /* !< ADC1 Regular channel Start flag */
#define ADC_CSR_DOVR1 ((uint32_t)0x00000020) /* !< ADC1 DMA overrun flag */
#define ADC_CSR_AWD2 ((uint32_t)0x00000100) /* !< ADC2 Analog watchdog flag */
#define ADC_CSR_EOC2 ((uint32_t)0x00000200) /* !< ADC2 End of conversion */
#define ADC_CSR_JEOC2 ((uint32_t)0x00000400) /* !< ADC2 Injected channel end of conversion */
#define ADC_CSR_JSTRT2 ((uint32_t)0x00000800) /* !< ADC2 Injected channel Start flag */
#define ADC_CSR_STRT2 ((uint32_t)0x00001000) /* !< ADC2 Regular channel Start flag */
#define ADC_CSR_DOVR2 ((uint32_t)0x00002000) /* !< ADC2 DMA overrun flag */
#define ADC_CSR_AWD3 ((uint32_t)0x00001000) /* !< ADC3 Analog watchdog flag */
#define ADC_CSR_EOC3 ((uint32_t)0x00002000) /* !< ADC3 End of conversion */
#define ADC_CSR_JEOC3 ((uint32_t)0x00004000) /* !< ADC3 Injected channel end of conversion */
#define ADC_CSR_JSTRT3 ((uint32_t)0x00008000) /* !< ADC3 Injected channel Start flag */
#define ADC_CSR_STRT3 ((uint32_t)0x00100000) /* !< ADC3 Regular channel Start flag */
#define ADC_CSR_DOVR3 ((uint32_t)0x00200000) /* !< ADC3 DMA overrun flag */

/* ***** Bit definition for ADC_CCR register *****/
#define ADC_CCR_MULTI ((uint32_t)0x0000001F) /* !< MULTI[4:0] bits (Multi-ADC mode selection)
#define ADC_CCR_MULTI_0 ((uint32_t)0x00000001) /* !< Bit 0 */

```

```
#define ADC_JDR2_JDATA ((uint16_t)0xFFFF) /* !< Injected data */
```

```
/* ***** Bit definition for ADC_JDR3 register *****/
#define ADC_JDR3_JDATA ((uint16_t)0xFFFF) /* !< Injected data */
```

```
/* ***** Bit definition for ADC_JDR4 register *****/
#define ADC_JDR4_JDATA ((uint16_t)0xFFFF) /* !< Injected data */
```

```
/* ***** Bit definition for ADC_DR register *****/
#define ADC_DR_DATA ((uint32_t)0x0000FFFF) /* !< Regular data */
#define ADC_DR_ADC2DATA ((uint32_t)0xFFFFF0000) /* !< ADC2 data */
```

```
/* ***** Bit definition for ADC_CSR register *****/
#define ADC_CSR_AWD1 ((uint32_t)0x00000001) /* !< ADC1 Analog watchdog flag */
#define ADC_CSR_EOC1 ((uint32_t)0x00000002) /* !< ADC1 End of conversion */
#define ADC_CSR_JEOC1 ((uint32_t)0x00000004) /* !< ADC1 Injected channel end of conversion */
#define ADC_CSR_JSTRT1 ((uint32_t)0x00000008) /* !< ADC1 Injected channel Start flag */
#define ADC_CSR_STRT1 ((uint32_t)0x00000010) /* !< ADC1 Regular channel Start flag */
#define ADC_CSR_DOVR1 ((uint32_t)0x00000020) /* !< ADC1 DMA overrun flag */
#define ADC_CSR_AWD2 ((uint32_t)0x00000100) /* !< ADC2 Analog watchdog flag */
#define ADC_CSR_EOC2 ((uint32_t)0x00000200) /* !< ADC2 End of conversion */
#define ADC_CSR_JEOC2 ((uint32_t)0x00000400) /* !< ADC2 Injected channel end of conversion */
#define ADC_CSR_JSTRT2 ((uint32_t)0x00000800) /* !< ADC2 Injected channel Start flag */
#define ADC_CSR_STRT2 ((uint32_t)0x00001000) /* !< ADC2 Regular channel Start flag */
#define ADC_CSR_DOVR2 ((uint32_t)0x00002000) /* !< ADC2 DMA overrun flag */
#define ADC_CSR_AWD3 ((uint32_t)0x00001000) /* !< ADC3 Analog watchdog flag */
#define ADC_CSR_EOC3 ((uint32_t)0x00002000) /* !< ADC3 End of conversion */
#define ADC_CSR_JEOC3 ((uint32_t)0x00004000) /* !< ADC3 Injected channel end of conversion */
#define ADC_CSR_JSTRT3 ((uint32_t)0x00008000) /* !< ADC3 Injected channel Start flag */
#define ADC_CSR_STRT3 ((uint32_t)0x00100000) /* !< ADC3 Regular channel Start flag */
#define ADC_CSR_DOVR3 ((uint32_t)0x00200000) /* !< ADC3 DMA overrun flag */
```

```
/* ***** Bit definition for ADC_CCR register *****/
#define ADC_CCR_MULTI ((uint32_t)0x0000001F) /* !< MULTI[4:0] bits (Multi-ADC mode selection)
#define ADC_CCR_MULTI_0 ((uint32_t)0x00000001) /* !< Bit 0 */
```

```

#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)0x000000F0)
#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_TSURREFE ((uint32_t)0x00800000)

/* !<Bit 1 */ ((uint32_t)0x00000002)
/* !<Bit 2 */ ((uint32_t)0x00000004)
/* !<Bit 3 */ ((uint32_t)0x00000008)
/* !<Bit 4 */ ((uint32_t)0x00000010)
/* !<DELAY [3:0] bits (Delay between 2 Samp*/ ((uint32_t)0x000000F0)
/* !<Bit 0 */ ((uint32_t)0x00000100)
/* !<Bit 1 */ ((uint32_t)0x00000200)
/* !<Bit 2 */ ((uint32_t)0x00000400)
/* !<Bit 3 */ ((uint32_t)0x00000800)
/* !<DMA disable selection (Multi-ADC mode*/ ((uint32_t)0x00002000)
/* !<DMA [1:0] bits (Direct Memory Access m*/ ((uint32_t)0x0000C000)
/* !<Bit 0 */ ((uint32_t)0x00004000)
/* !<Bit 1 */ ((uint32_t)0x00008000)
/* !<ADCprescaler)*/ ((uint32_t)0x00030000)
/* !<Bit 0 */ ((uint32_t)0x00010000)
/* !<Bit 1 */ ((uint32_t)0x00020000)
/* !<VBAT Enable */ ((uint32_t)0x00400000)
/* !<Temperature Sensor and VREFINT Enable */ ((uint32_t)0x00800000)

/* Bit definition for ADC_CDR register *****/
#define ADC_CDR_DATA1 ((uint32_t)0x00000FFF)
#define ADC_CDR_DATA2 ((uint32_t)0xFFFFF0000)

/* !<CAN control and status registers */
/* !<CAN Area Network */
/* !<Initialization Request */
#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)

/* !<Sleep Mode Request */
/* !<Transmit FIFO Priority */
/* !<Receive FIFO Locked Mode */
/* !<No Automatic Retransmission */

```

```

/* !<Automatic Wakeup Mode */          /* !<Initialization Acknowledge */
((uint16_t)0x00020)                  /* !<Sleep Acknowledge */
((uint16_t)0x00040)                  /* !<Error Interrupt */
((uint16_t)0x00080)                  /* !<Wakeup Interrupt */
((uint16_t)0x80000)                  /* !<Sleep Acknowledge Interrupt */
/* !<Transmit Mode */
/* !<Receive Mode */
/* !<Last Sample Point */
/* !<CAN Rx Signal */

/* !<Request Completed Mailbox0 */
((uint32_t)0x00000001)              /* !<Transmission OK of Mailbox0 */
((uint32_t)0x00000002)              /* !<Arbitration Lost for Mailbox0 */
((uint32_t)0x00000004)              /* !<Transmission Error of Mailbox0 */
((uint32_t)0x00000008)              /* !<Abort Request for Mailbox0 */
((uint32_t)0x00000000)              /* !<Request Completed Mailbox1 */
((uint32_t)0x00000010)              /* !<Transmission OK of Mailbox1 */
((uint32_t)0x00000020)              /* !<Arbitration Lost for Mailbox1 */
((uint32_t)0x00000040)              /* !<Transmission Error of Mailbox1 */
((uint32_t)0x00000080)              /* !<Abort Request for Mailbox1 */
((uint32_t)0x00000001)              /* !<Request Completed Mailbox2 */
((uint32_t)0x00000020)              /* !<Transmission OK of Mailbox2 */
((uint32_t)0x00000040)              /* !<Arbitration Lost for mailbox2 */
((uint32_t)0x00000080)              /* !<Transmission Error of Mailbox2 */
((uint32_t)0x00000000)              /* !<Abort Request for Mailbox2 */
/* !<Mailbox Code */

/* !<TME[2:0] bits */
((uint32_t)0x10000000)

CAN_TSR_TME

CAN_MCR_AWUM
CAN_MCR_ABOM
CAN_MCR_TTCM
CAN_MCR_RESET

CAN_MSR_INAK
CAN_MSR_SLAK
CAN_MSR_ERRI
CAN_MSR_WKUI
CAN_MSR_SLAKI
CAN_MSR_TXM
CAN_MSR_RXM
CAN_MSR_SAMP
CAN_MSR_RX

CAN_TSR_RQCP0
CAN_TSR_TXOK0
CAN_TSR_ALST0
CAN_TSR_TERRO
CAN_TSR_ABRQ0
CAN_TSR_RQCP1
CAN_TSR_TXOK1
CAN_TSR_ALST1
CAN_TSR_TERR1
CAN_TSR_ABRQ1
CAN_TSR_RQCP2
CAN_TSR_TXOK2
CAN_TSR_ALST2
CAN_TSR_TERR2
CAN_TSR_ABRQ2
CAN_TSR_CODE

```

```

#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_LOWO ((uint32_t)0x20000000)
#define CAN_TSR_LOW1 ((uint32_t)0x40000000)
#define CAN_TSR_LOW2 ((uint32_t)0x80000000)

/**************** Bit definition for CAN_RFOR register *****/
#define CAN_RFOR_FMPO ((uint8_t)0x03)
#define CAN_RFOR_FULL0 ((uint8_t)0x08)
#define CAN_RFOR_FOVR0 ((uint8_t)0x10)
#define CAN_RFOR_RFOM0 ((uint8_t)0x20)

/**************** Bit definition for CAN_RF1R register *****/
#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)

/**************** Bit definition for CAN_IER register *****/
#define CAN_IER_TMEIE ((uint32_t)0x00000001)
#define CAN_IER_FMPIEO ((uint32_t)0x00000002)
#define CAN_IER_FFIEO ((uint32_t)0x00000004)
#define CAN_IER_FOVIEO ((uint32_t)0x00000008)
#define CAN_IER_FMPIE1 ((uint32_t)0x00000010)
#define CAN_IER_FFIIE1 ((uint32_t)0x00000020)
#define CAN_IER_FOVIE1 ((uint32_t)0x00000040)
#define CAN_IER_EWGLIE ((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)0x00000800)
#define CAN_IER_WKUIE ((uint32_t)0x00010000)

/* !<Transmit Mailbox 0 Empty */
/* !<Transmit Mailbox 1 Empty */
/* !<Transmit Mailbox 2 Empty */

/* !<LOW[2:0] bits */
/* !<Lowest Priority Flag for Mailbox 0 */
/* !<Lowest Priority Flag for Mailbox 1 */
/* !<Lowest Priority Flag for Mailbox 2 */

/* !<FIFO 0 Message Pending */
/* !<FIFO 0 Full */
/* !<FIFO 0 Overrun */
/* !<Release FIFO 0 Output Mailbox */

/* !<FIFO 1 Message Pending */
/* !<FIFO 1 Full */
/* !<FIFO 1 Overrun */
/* !<Release FIFO 1 Output Mailbox */

/* !<Transmit Mailbox Empty Interrupt Enable */
/* !<FIFO Message Pending Interrupt Enable */
/* !<FIFO Full Interrupt Enable */
/* !<FIFO Overrun Interrupt Enable */
/* !<FIFO Message Pending Interrupt Enable */
/* !<FIFO Full Interrupt Enable */
/* !<FIFO Overrun Interrupt Enable */
/* !<Error Warning Interrupt Enable */
/* !<Error Passive Interrupt Enable */
/* !<Bus-Off Interrupt Enable */
/* !<Last Error Code Interrupt Enable */
/* !<Error Interrupt Enable */
/* !<WakeUp Interrupt Enable */

```

```

#define CAN_IER_SLKIE ((uint32_t)0x0000200000) /* !<Sleep Interrupt Enable */

/* ***** Bit definition for CAN_ESR register *****/
#define CAN_ESR_EWGF ((uint32_t)0x00000001) /* !<Error Warning Flag */
#define CAN_ESR_EPVF ((uint32_t)0x00000002) /* !<Error Passive Flag */
#define CAN_ESR_BOFF ((uint32_t)0x00000004) /* !<Bus-Off Flag */

#define CAN_ESR_LEC ((uint32_t)0x00000070) /* !<LEC[2:0] bits (Last Error Code)
#define CAN_ESR_LEC_0 ((uint32_t)0x00000010) /* !<Bit 0 */
#define CAN_ESR_LEC_1 ((uint32_t)0x00000020) /* !<Bit 1 */
#define CAN_ESR_LEC_2 ((uint32_t)0x00000040) /* !<Bit 2 */

#define CAN_ESR_TEC ((uint32_t)0x000FF0000) /* !<Least significant byte of the 9-bit T
#define CAN_ESR_REC ((uint32_t)0xFFFF000000) /* !<Receive Error Counter */

/* ***** Bit definition for CAN_BTR register *****/
#define CAN_BTR_BRP ((uint32_t)0x0000003FF) /* !<Baud Rate Prescaler */
#define CAN_BTR_TS1 ((uint32_t)0x000F0000) /* !<Time Segment 1 */
#define CAN_BTR_TS2 ((uint32_t)0x00700000) /* !<Time Segment 2 */
#define CAN_BTR_SJW ((uint32_t)0x03000000) /* !<Resynchronization Jump Width */
#define CAN_BTR_LBKM ((uint32_t)0x40000000) /* !<Loop Back Mode (Debug) */
#define CAN_BTR_STIM ((uint32_t)0x80000000) /* !<Silent Mode */

/* !<Mailbox registers */
/* ***** Bit definition for CAN_TIOR register *****/
#define CAN_TIOR_TXRQ ((uint32_t)0x00000001) /* !<Transmit Mailbox Request */
#define CAN_TIOR_RTR ((uint32_t)0x00000002) /* !<Remote Transmission Request */
#define CAN_TIOR_IDE ((uint32_t)0x00000004) /* !<Identifier Extension */
#define CAN_TIOR_EXID ((uint32_t)0x001FFFF8) /* !<Extended Identifier */
#define CAN_TIOR_STID ((uint32_t)0xFFE00000) /* !<Standard Identifier or Extended Identifier */

/* ***** Bit definition for CAN_TDTR register *****/
#define CAN_TDTR_DLCL ((uint32_t)0x0000000F) /* !<Data Length Code */
#define CAN_TDTR_TGT ((uint32_t)0x00000100) /* !<Transmit Global Time */
#define CAN_TDTR_TIME ((uint32_t)0xFFFFF000) /* !<Message Time Stamp */

```

```

/*
***** Bit definition for CAN_TDLOR register *****/
#define CAN_TDLOR_DATA0 ((uint32_t)0x000000FF) /*!<Data byte 0 */
#define CAN_TDLOR_DATA1 ((uint32_t)0x0000FF00) /*!<Data byte 1 */
#define CAN_TDLOR_DATA2 ((uint32_t)0x00FF0000) /*!<Data byte 2 */
#define CAN_TDLOR_DATA3 ((uint32_t)0xFF000000) /*!<Data byte 3 */

/*
***** Bit definition for CAN_TDHor register *****/
#define CAN_TDHor_DATA4 ((uint32_t)0x000000FF) /*!<Data byte 4 */
#define CAN_TDHor_DATA5 ((uint32_t)0x0000FF00) /*!<Data byte 5 */
#define CAN_TDHor_DATA6 ((uint32_t)0x00FF0000) /*!<Data byte 6 */
#define CAN_TDHor_DATA7 ((uint32_t)0xFF000000) /*!<Data byte 7 */

/*
***** Bit definition for CAN_TI1R register *****/
#define CAN_TI1R_TXRQ ((uint32_t)0x00000001) /*!<Transmit Mailbox Request */
#define CAN_TI1R_RTR ((uint32_t)0x00000002) /*!<Remote Transmission Request */
#define CAN_TI1R_IDE ((uint32_t)0x00000004) /*!<Identifier Extension */
#define CAN_TI1R_EXID ((uint32_t)0x001FFFF8) /*!<Extended Identifier */
#define CAN_TI1R_STID ((uint32_t)0xFFE00000) /*!<Standard Identifier or Extended Identifier */

/*
***** Bit definition for CAN_TDT1R register *****/
#define CAN_TDT1R_DLCode ((uint32_t)0x0000000F) /*!<Data Length Code */
#define CAN_TDT1R_GTime ((uint32_t)0x00000100) /*!<Transmit Global Time */
#define CAN_TDT1R_STamp ((uint32_t)0xFFFFF000) /*!<Message Time Stamp */

/*
***** Bit definition for CAN_TDL1R register *****/
#define CAN_TDL1R_DLCode ((uint32_t)0x000000FF) /*!<Data byte 0 */
#define CAN_TDL1R_GTime ((uint32_t)0x0000FF00) /*!<Data byte 1 */
#define CAN_TDL1R_STamp ((uint32_t)0x00FF0000) /*!<Data byte 2 */
#define CAN_TDL1R_STamp3 ((uint32_t)0xFF000000) /*!<Data byte 3 */

/*
***** Bit definition for CAN_TD1H1R register *****/
#define CAN_TD1H1R_DLCode ((uint32_t)0x000000FF) /*!<Data byte 4 */
#define CAN_TD1H1R_GTime ((uint32_t)0x0000FF00) /*!<Data byte 5 */
#define CAN_TD1H1R_STamp ((uint32_t)0x00FF0000) /*!<Data byte 6 */

```

```

#define CAN_TDHR_DATA7 ((uint32_t)0xFFFF000000) /* !<Data byte 7 */

/* ***** Bit definition for CAN_TI2R register *****/
#define CAN_TI2R_TXRQ ((uint32_t)0x00000001) /* !<Transmit Mailbox Request */
#define CAN_TI2R_RTR ((uint32_t)0x00000002) /* !<Remote Transmission Request */
#define CAN_TI2R_IDE ((uint32_t)0x00000004) /* !<Identifier Extension */
#define CAN_TI2R_EXID ((uint32_t)0x001FFFF8) /* !<Extended Identifier */
#define CAN_TI2R_STID ((uint32_t)0xFFFE0000) /* !<Standard Identifier or Extended Identifier */

/* ***** Bit definition for CAN_TDT2R register *****/
#define CAN_TDT2R_DLCL ((uint32_t)0x0000000F) /* !<Data Length Code */
#define CAN_TDT2R_TGT ((uint32_t)0x00000100) /* !<Transmit Global Time */
#define CAN_TDT2R_TIME ((uint32_t)0xFFFFF000) /* !<Message Time Stamp */

/* ***** Bit definition for CAN_TDL2R register *****/
#define CAN_TDL2R_DATA0 ((uint32_t)0x000000FF) /* !<Data byte 0 */
#define CAN_TDL2R_DATA1 ((uint32_t)0x0000FF00) /* !<Data byte 1 */
#define CAN_TDL2R_DATA2 ((uint32_t)0x00FF0000) /* !<Data byte 2 */
#define CAN_TDL2R_DATA3 ((uint32_t)0xFF000000) /* !<Data byte 3 */

/* ***** Bit definition for CAN_TDHL2R register *****/
#define CAN_TDHL2R_DATA4 ((uint32_t)0x000000FF) /* !<Data byte 4 */
#define CAN_TDHL2R_DATA5 ((uint32_t)0x0000FF00) /* !<Data byte 5 */
#define CAN_TDHL2R_DATA6 ((uint32_t)0x00FF0000) /* !<Data byte 6 */
#define CAN_TDHL2R_DATA7 ((uint32_t)0xFF000000) /* !<Data byte 7 */

/* ***** Bit definition for CAN_TDH2R register *****/
#define CAN_TDH2R_DATA4 ((uint32_t)0x00000002) /* !<Remote Transmission Request */
#define CAN_TDH2R_DATA5 ((uint32_t)0x00000004) /* !<Identifier Extension */
#define CAN_TDH2R_DATA6 ((uint32_t)0x001FFFF8) /* !<Extended Identifier */
#define CAN_TDH2R_DATA7 ((uint32_t)0xFFFE0000) /* !<Standard Identifier or Extended Identifier */

/* ***** Bit definition for CAN_RDTOR register *****/
#define CAN_RDTOR_DLCL ((uint32_t)0x0000000F) /* !<Data Length Code */
#define CAN_RDTOR_FMI ((uint32_t)0x0000FF00) /* !<Filter Match Index */

```

```

#define CAN_RDTOR_TIME ((uint32_t)0xFFFFF0000) /* !<Message Time Stamp */

/* ***** Bit definition for CAN_RDLOR register *****/
#define CAN_RDLOR_DATA0 ((uint32_t)0x000000FF) /* !<Data byte 0 */
#define CAN_RDLOR_DATA1 ((uint32_t)0x0000FF00) /* !<Data byte 1 */
#define CAN_RDLOR_DATA2 ((uint32_t)0x00FF0000) /* !<Data byte 2 */
#define CAN_RDLOR_DATA3 ((uint32_t)0xFF000000) /* !<Data byte 3 */

/* ***** Bit definition for CAN_RDHOR register *****/
#define CAN_RDHOR_DATA4 ((uint32_t)0x0000000F) /* !<Data byte 4 */
#define CAN_RDHOR_DATA5 ((uint32_t)0x0000FF00) /* !<Data byte 5 */
#define CAN_RDHOR_DATA6 ((uint32_t)0x00FF0000) /* !<Data byte 6 */
#define CAN_RDHOR_DATA7 ((uint32_t)0xFF000000) /* !<Data byte 7 */

/* ***** Bit definition for CAN_RIIR register *****/
#define CAN_RIIR_RTR ((uint32_t)0x00000002) /* !<Remote Transmission Request */
#define CAN_RIIR_IDE ((uint32_t)0x00000004) /* !<Identifier Extension */
#define CAN_RIIR_EXID ((uint32_t)0x001FFFF8) /* !<Extended identifier */
#define CAN_RIIR_STID ((uint32_t)0xFFE00000) /* !<Standard Identifier or Extended Identifier */

/* ***** Bit definition for CAN_RDT1R register *****/
#define CAN_RDT1R_DL0 ((uint32_t)0x0000000F) /* !<Data Length Code */
#define CAN_RDT1R_DL1 ((uint32_t)0x0000FF00) /* !<Filter Match Index */
#define CAN_RDT1R_DL2 ((uint32_t)0x00FF0000) /* !<Message Time Stamp */

/* ***** Bit definition for CAN_RDL1R register *****/
#define CAN_RDL1R_DL0 ((uint32_t)0x000000FF) /* !<Data byte 0 */
#define CAN_RDL1R_DL1 ((uint32_t)0x0000FF00) /* !<Data byte 1 */
#define CAN_RDL1R_DL2 ((uint32_t)0x00FF0000) /* !<Data byte 2 */
#define CAN_RDL1R_DL3 ((uint32_t)0xFF000000) /* !<Data byte 3 */

/* ***** Bit definition for CAN_RDH1R register *****/
#define CAN_RDH1R_DL4 ((uint32_t)0x000000FF) /* !<Data byte 4 */
#define CAN_RDH1R_DL5 ((uint32_t)0x0000FF00) /* !<Data byte 5 */
#define CAN_RDH1R_DL6 ((uint32_t)0x00FF0000) /* !<Data byte 6 */

```

```

#define CAN_RDH1R_DATA7 ((uint32_t)0xFFFF000000)

/* !<CAN filter registers */
/* ***** Bit definition for CAN_FMR register *****/
#define CAN_FMR_INIT ((uint8_t)0x01) /* !<Filter Init Mode */

/* ***** Bit definition for CAN_FM1R register *****/
#define CAN_FM1R_FBM ((uint16_t)0x3FFF)
#define CAN_FM1R_FBMO ((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)

/* ***** Bit definition for CAN_FS1R register *****/
#define CAN_FS1R_FSC ((uint16_t)0x3FFF)
#define CAN_FS1R_FSCO ((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)

/* !<Filter Scale Configuration */
bit 0 */ /* !<Filter Scale Configuration bit 0 */
bit 1 */ /* !<Filter Scale Configuration bit 1 */
bit 2 */ /* !<Filter Scale Configuration bit 2 */
bit 3 */ /* !<Filter Scale Configuration bit 3 */
bit 4 */ /* !<Filter Scale Configuration bit 4 */
bit 5 */ /* !<Filter Scale Configuration bit 5 */
bit 6 */ /* !<Filter Scale Configuration bit 6 */
bit 7 */ /* !<Filter Scale Configuration bit 7 */
bit 8 */ /* !<Filter Scale Configuration bit 8 */
bit 9 */ /* !<Filter Scale Configuration bit 9 */
bit 10 */ /* !<Filter Scale Configuration bit 10 */
bit 11 */ /* !<Filter Scale Configuration bit 11 */
bit 12 */ /* !<Filter Scale Configuration bit 12 */
bit 13 */ /* !<Filter Scale Configuration bit 13 */

```

```

/*!<Filter Scale Configuration bit 16 */
#define CAN_FS1R_FSC11 ((uint16_t)0x0400)
#define CAN_FS1R_FSC12 ((uint16_t)0x0800)
#define CAN_FS1R_FSC13 ((uint16_t)0x1000)
#define CAN_FS1R_FSC14 ((uint16_t)0x2000)

***** Bit definition for CAN_FFA1R register *****/

```

```

#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)

***** Bit definition for CAN_FA1R register *****/

```

```

#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)

***** Bit definition for CAN_FA1R register *****/

```

```

/*!<Filter Scale Configuration bit 17 */
#define CAN_FA1R_FACT1 ((uint16_t)0x0001)
#define CAN_FA1R_FACT2 ((uint16_t)0x0002)
#define CAN_FA1R_FACT3 ((uint16_t)0x0004)
#define CAN_FA1R_FACT4 ((uint16_t)0x0008)
#define CAN_FA1R_FACT5 ((uint16_t)0x0010)
#define CAN_FA1R_FACT6 ((uint16_t)0x0020)
#define CAN_FA1R_FACT7 ((uint16_t)0x0040)
#define CAN_FA1R_FACT8 ((uint16_t)0x0080)
#define CAN_FA1R_FACT9 ((uint16_t)0x0100)
#define CAN_FA1R_FACT10 ((uint16_t)0x0200)

/*!<Filter Scale Configuration bit 18 */
#define CAN_FA1R_FACT11 ((uint16_t)0x0001)
#define CAN_FA1R_FACT12 ((uint16_t)0x0002)
#define CAN_FA1R_FACT13 ((uint16_t)0x0004)

```

```

#define CAN_FA1R_FACT11 ((uint16_t)0x0800)
#define CAN_FA1R_FACT12 ((uint16_t)0x1000)
#define CAN_FA1R_FACT13 ((uint16_t)0x2000)

/**************** Bit definition for CAN_FOR1 register *****/
#define CAN_FOR1_FBO ((uint32_t)0x00000001)
#define CAN_FOR1_FB1 ((uint32_t)0x00000002)
#define CAN_FOR1_FB2 ((uint32_t)0x00000004)
#define CAN_FOR1_FB3 ((uint32_t)0x00000008)
#define CAN_FOR1_FB4 ((uint32_t)0x00000010)
#define CAN_FOR1_FB5 ((uint32_t)0x00000020)
#define CAN_FOR1_FB6 ((uint32_t)0x00000040)
#define CAN_FOR1_FB7 ((uint32_t)0x00000080)
#define CAN_FOR1_FB8 ((uint32_t)0x00000100)
#define CAN_FOR1_FB9 ((uint32_t)0x00000200)
#define CAN_FOR1_FB10 ((uint32_t)0x00000400)
#define CAN_FOR1_FB11 ((uint32_t)0x00000800)
#define CAN_FOR1_FB12 ((uint32_t)0x00001000)
#define CAN_FOR1_FB13 ((uint32_t)0x00002000)
#define CAN_FOR1_FB14 ((uint32_t)0x00004000)
#define CAN_FOR1_FB15 ((uint32_t)0x00008000)
#define CAN_FOR1_FB16 ((uint32_t)0x00010000)
#define CAN_FOR1_FB17 ((uint32_t)0x00020000)
#define CAN_FOR1_FB18 ((uint32_t)0x00040000)
#define CAN_FOR1_FB19 ((uint32_t)0x00080000)
#define CAN_FOR1_FB20 ((uint32_t)0x00100000)
#define CAN_FOR1_FB21 ((uint32_t)0x00200000)
#define CAN_FOR1_FB22 ((uint32_t)0x00400000)
#define CAN_FOR1_FB23 ((uint32_t)0x00800000)
#define CAN_FOR1_FB24 ((uint32_t)0x01000000)
#define CAN_FOR1_FB25 ((uint32_t)0x02000000)
#define CAN_FOR1_FB26 ((uint32_t)0x04000000)
#define CAN_FOR1_FB27 ((uint32_t)0x08000000)
#define CAN_FOR1_FB28 ((uint32_t)0x10000000)
#define CAN_FOR1_FB29 ((uint32_t)0x20000000)

/* !<Filter 11 Active */
/* !<Filter 12 Active */
/* !<Filter 13 Active */
/* !<Filter 14 Active */
/* !<Filter 15 Active */
/* !<Filter 16 Active */
/* !<Filter 17 Active */
/* !<Filter 18 Active */
/* !<Filter 19 Active */
/* !<Filter 20 Active */
/* !<Filter 21 Active */
/* !<Filter 22 Active */
/* !<Filter 23 Active */
/* !<Filter 24 Active */
/* !<Filter 25 Active */
/* !<Filter 26 Active */
/* !<Filter 27 Active */
/* !<Filter 28 Active */
/* !<Filter 29 Active */

```

---

```

#define CAN_FOR1_FB30      ((uint32_t)0x40000000)
#define CAN_FOR1_FB31      ((uint32_t)0x80000000)

/* ***** Bit definition for CAN_F1R1 register *****/
/* !<Filter bit 30 */
/* !<Filter bit 31 */

/* ***** Bit definition for CAN_F1R1 register *****/
/* !<Filter bit 0 */
/* !<Filter bit 1 */
/* !<Filter bit 2 */
/* !<Filter bit 3 */
/* !<Filter bit 4 */
/* !<Filter bit 5 */
/* !<Filter bit 6 */
/* !<Filter bit 7 */
/* !<Filter bit 8 */
/* !<Filter bit 9 */
/* !<Filter bit 10 */
/* !<Filter bit 11 */
/* !<Filter bit 12 */
/* !<Filter bit 13 */
/* !<Filter bit 14 */
/* !<Filter bit 15 */
/* !<Filter bit 16 */
/* !<Filter bit 17 */
/* !<Filter bit 18 */
/* !<Filter bit 19 */
/* !<Filter bit 20 */
/* !<Filter bit 21 */
/* !<Filter bit 22 */
/* !<Filter bit 23 */
/* !<Filter bit 24 */
/* !<Filter bit 25 */
/* !<Filter bit 26 */
/* !<Filter bit 27 */
/* !<Filter bit 28 */
/* !<Filter bit 29 */
/* !<Filter bit 30 */

/* ***** Bit definition for CAN_F1R1 register *****/
#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)

/* ***** Bit definition for CAN_F2R1 register *****/
CAN_F2R1_FBO ((uint32_t)0x00000001) /* !<Filter bit 0 */
CAN_F2R1_FB1 ((uint32_t)0x00000002) /* !<Filter bit 1 */
CAN_F2R1_FB2 ((uint32_t)0x00000004) /* !<Filter bit 2 */
CAN_F2R1_FB3 ((uint32_t)0x00000008) /* !<Filter bit 3 */
CAN_F2R1_FB4 ((uint32_t)0x00000010) /* !<Filter bit 4 */
CAN_F2R1_FB5 ((uint32_t)0x00000020) /* !<Filter bit 5 */
CAN_F2R1_FB6 ((uint32_t)0x00000040) /* !<Filter bit 6 */
CAN_F2R1_FB7 ((uint32_t)0x00000080) /* !<Filter bit 7 */
CAN_F2R1_FB8 ((uint32_t)0x00000100) /* !<Filter bit 8 */
CAN_F2R1_FB9 ((uint32_t)0x00000200) /* !<Filter bit 9 */
CAN_F2R1_FB10 ((uint32_t)0x00000400) /* !<Filter bit 10 */
CAN_F2R1_FB11 ((uint32_t)0x00000800) /* !<Filter bit 11 */
CAN_F2R1_FB12 ((uint32_t)0x00001000) /* !<Filter bit 12 */
CAN_F2R1_FB13 ((uint32_t)0x00002000) /* !<Filter bit 13 */
CAN_F2R1_FB14 ((uint32_t)0x00004000) /* !<Filter bit 14 */
CAN_F2R1_FB15 ((uint32_t)0x00008000) /* !<Filter bit 15 */
CAN_F2R1_FB16 ((uint32_t)0x00010000) /* !<Filter bit 16 */
CAN_F2R1_FB17 ((uint32_t)0x00020000) /* !<Filter bit 17 */
CAN_F2R1_FB18 ((uint32_t)0x00040000) /* !<Filter bit 18 */
CAN_F2R1_FB19 ((uint32_t)0x00080000) /* !<Filter bit 19 */
CAN_F2R1_FB20 ((uint32_t)0x00100000) /* !<Filter bit 20 */
CAN_F2R1_FB21 ((uint32_t)0x00200000) /* !<Filter bit 21 */
CAN_F2R1_FB22 ((uint32_t)0x00400000) /* !<Filter bit 22 */
CAN_F2R1_FB23 ((uint32_t)0x00800000) /* !<Filter bit 23 */
CAN_F2R1_FB24 ((uint32_t)0x01000000) /* !<Filter bit 24 */
CAN_F2R1_FB25 ((uint32_t)0x02000000) /* !<Filter bit 25 */
CAN_F2R1_FB26 ((uint32_t)0x04000000) /* !<Filter bit 26 */
CAN_F2R1_FB27 ((uint32_t)0x08000000) /* !<Filter bit 27 */
CAN_F2R1_FB28 ((uint32_t)0x10000000) /* !<Filter bit 28 */
CAN_F2R1_FB29 ((uint32_t)0x20000000) /* !<Filter bit 29 */
CAN_F2R1_FB30 ((uint32_t)0x40000000) /* !<Filter bit 30 */
CAN_F2R1_FB31 ((uint32_t)0x80000000) /* !<Filter bit 31 */

```

---

```

/*
***** Bit definition for CAN_F3R1 register *****/
#define CAN_F3R1_FB0 ((uint32_t)0x00000001) /* !<Filter bit 0 */
#define CAN_F3R1_FB1 ((uint32_t)0x00000002) /* !<Filter bit 1 */
#define CAN_F3R1_FB2 ((uint32_t)0x00000004) /* !<Filter bit 2 */
#define CAN_F3R1_FB3 ((uint32_t)0x00000008) /* !<Filter bit 3 */
#define CAN_F3R1_FB4 ((uint32_t)0x00000010) /* !<Filter bit 4 */
#define CAN_F3R1_FB5 ((uint32_t)0x00000020) /* !<Filter bit 5 */
#define CAN_F3R1_FB6 ((uint32_t)0x00000040) /* !<Filter bit 6 */
#define CAN_F3R1_FB7 ((uint32_t)0x00000080) /* !<Filter bit 7 */
#define CAN_F3R1_FB8 ((uint32_t)0x00000100) /* !<Filter bit 8 */
#define CAN_F3R1_FB9 ((uint32_t)0x00000200) /* !<Filter bit 9 */
#define CAN_F3R1_FB10 ((uint32_t)0x00000400) /* !<Filter bit 10 */
#define CAN_F3R1_FB11 ((uint32_t)0x00000800) /* !<Filter bit 11 */
#define CAN_F3R1_FB12 ((uint32_t)0x00001000) /* !<Filter bit 12 */
#define CAN_F3R1_FB13 ((uint32_t)0x00002000) /* !<Filter bit 13 */
#define CAN_F3R1_FB14 ((uint32_t)0x00004000) /* !<Filter bit 14 */
#define CAN_F3R1_FB15 ((uint32_t)0x00008000) /* !<Filter bit 15 */
#define CAN_F3R1_FB16 ((uint32_t)0x00010000) /* !<Filter bit 16 */
#define CAN_F3R1_FB17 ((uint32_t)0x00020000) /* !<Filter bit 17 */
#define CAN_F3R1_FB18 ((uint32_t)0x00040000) /* !<Filter bit 18 */
#define CAN_F3R1_FB19 ((uint32_t)0x00080000) /* !<Filter bit 19 */
#define CAN_F3R1_FB20 ((uint32_t)0x00100000) /* !<Filter bit 20 */
#define CAN_F3R1_FB21 ((uint32_t)0x00200000) /* !<Filter bit 21 */
#define CAN_F3R1_FB22 ((uint32_t)0x00400000) /* !<Filter bit 22 */
#define CAN_F3R1_FB23 ((uint32_t)0x00800000) /* !<Filter bit 23 */
#define CAN_F3R1_FB24 ((uint32_t)0x01000000) /* !<Filter bit 24 */
#define CAN_F3R1_FB25 ((uint32_t)0x02000000) /* !<Filter bit 25 */
#define CAN_F3R1_FB26 ((uint32_t)0x04000000) /* !<Filter bit 26 */
#define CAN_F3R1_FB27 ((uint32_t)0x08000000) /* !<Filter bit 27 */
#define CAN_F3R1_FB28 ((uint32_t)0x10000000) /* !<Filter bit 28 */
#define CAN_F3R1_FB29 ((uint32_t)0x20000000) /* !<Filter bit 29 */
#define CAN_F3R1_FB30 ((uint32_t)0x40000000) /* !<Filter bit 30 */
#define CAN_F3R1_FB31 ((uint32_t)0x80000000) /* !<Filter bit 31 */

```

---

```

/*
***** Bit definition for CAN_F4R1 register *****/
#define CAN_F4R1_FBO ((uint32_t)0x00000001) /*!<Filter bit 0 */
#define CAN_F4R1_FB1 ((uint32_t)0x00000002) /*!<Filter bit 1 */
#define CAN_F4R1_FB2 ((uint32_t)0x00000004) /*!<Filter bit 2 */
#define CAN_F4R1_FB3 ((uint32_t)0x00000008) /*!<Filter bit 3 */
#define CAN_F4R1_FB4 ((uint32_t)0x00000010) /*!<Filter bit 4 */
#define CAN_F4R1_FB5 ((uint32_t)0x00000020) /*!<Filter bit 5 */
#define CAN_F4R1_FB6 ((uint32_t)0x00000040) /*!<Filter bit 6 */
#define CAN_F4R1_FB7 ((uint32_t)0x00000080) /*!<Filter bit 7 */
#define CAN_F4R1_FB8 ((uint32_t)0x00000100) /*!<Filter bit 8 */
#define CAN_F4R1_FB9 ((uint32_t)0x00000200) /*!<Filter bit 9 */
#define CAN_F4R1_FB10 ((uint32_t)0x00000400) /*!<Filter bit 10 */
#define CAN_F4R1_FB11 ((uint32_t)0x00000800) /*!<Filter bit 11 */
#define CAN_F4R1_FB12 ((uint32_t)0x00001000) /*!<Filter bit 12 */
#define CAN_F4R1_FB13 ((uint32_t)0x00002000) /*!<Filter bit 13 */
#define CAN_F4R1_FB14 ((uint32_t)0x00004000) /*!<Filter bit 14 */
#define CAN_F4R1_FB15 ((uint32_t)0x00008000) /*!<Filter bit 15 */
#define CAN_F4R1_FB16 ((uint32_t)0x00010000) /*!<Filter bit 16 */
#define CAN_F4R1_FB17 ((uint32_t)0x00020000) /*!<Filter bit 17 */
#define CAN_F4R1_FB18 ((uint32_t)0x00040000) /*!<Filter bit 18 */
#define CAN_F4R1_FB19 ((uint32_t)0x00080000) /*!<Filter bit 19 */
#define CAN_F4R1_FB20 ((uint32_t)0x00100000) /*!<Filter bit 20 */
#define CAN_F4R1_FB21 ((uint32_t)0x00200000) /*!<Filter bit 21 */
#define CAN_F4R1_FB22 ((uint32_t)0x00400000) /*!<Filter bit 22 */
#define CAN_F4R1_FB23 ((uint32_t)0x00800000) /*!<Filter bit 23 */
#define CAN_F4R1_FB24 ((uint32_t)0x01000000) /*!<Filter bit 24 */
#define CAN_F4R1_FB25 ((uint32_t)0x02000000) /*!<Filter bit 25 */
#define CAN_F4R1_FB26 ((uint32_t)0x04000000) /*!<Filter bit 26 */
#define CAN_F4R1_FB27 ((uint32_t)0x08000000) /*!<Filter bit 27 */
#define CAN_F4R1_FB28 ((uint32_t)0x10000000) /*!<Filter bit 28 */
#define CAN_F4R1_FB29 ((uint32_t)0x20000000) /*!<Filter bit 29 */
#define CAN_F4R1_FB30 ((uint32_t)0x40000000) /*!<Filter bit 30 */
#define CAN_F4R1_FB31 ((uint32_t)0x80000000) /*!<Filter bit 31 */

/*
***** Bit definition for CAN_F5R1 register *****/

```

---

```

#define CAN_F5R1_FBO ((uint32_t)0x00000000)
#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)

/* !<Filter bit 0 */
/* !<Filter bit 1 */
/* !<Filter bit 2 */
/* !<Filter bit 3 */
/* !<Filter bit 4 */
/* !<Filter bit 5 */
/* !<Filter bit 6 */
/* !<Filter bit 7 */
/* !<Filter bit 8 */
/* !<Filter bit 9 */
/* !<Filter bit 10 */
/* !<Filter bit 11 */
/* !<Filter bit 12 */
/* !<Filter bit 13 */
/* !<Filter bit 14 */
/* !<Filter bit 15 */
/* !<Filter bit 16 */
/* !<Filter bit 17 */
/* !<Filter bit 18 */
/* !<Filter bit 19 */
/* !<Filter bit 20 */
/* !<Filter bit 21 */
/* !<Filter bit 22 */
/* !<Filter bit 23 */
/* !<Filter bit 24 */
/* !<Filter bit 25 */
/* !<Filter bit 26 */
/* !<Filter bit 27 */
/* !<Filter bit 28 */
/* !<Filter bit 29 */
/* !<Filter bit 30 */
/* !<Filter bit 31 */

/* Bit definition for CAN_F6R1 register **** */
#define CAN_F6R1_FBO ((uint32_t)0x00000001) /* !<Filter bit 0 */

```

```

#define CAN_F6R1_FB1 ((uint32_t)0x00000002) /* !<Filter bit 1 */
#define CAN_F6R1_FB2 ((uint32_t)0x00000004) /* !<Filter bit 2 */
#define CAN_F6R1_FB3 ((uint32_t)0x00000008) /* !<Filter bit 3 */
#define CAN_F6R1_FB4 ((uint32_t)0x00000010) /* !<Filter bit 4 */
#define CAN_F6R1_FB5 ((uint32_t)0x00000020) /* !<Filter bit 5 */
#define CAN_F6R1_FB6 ((uint32_t)0x00000040) /* !<Filter bit 6 */
#define CAN_F6R1_FB7 ((uint32_t)0x00000080) /* !<Filter bit 7 */
#define CAN_F6R1_FB8 ((uint32_t)0x00000100) /* !<Filter bit 8 */
#define CAN_F6R1_FB9 ((uint32_t)0x00000200) /* !<Filter bit 9 */
#define CAN_F6R1_FB10 ((uint32_t)0x00000400) /* !<Filter bit 10 */
#define CAN_F6R1_FB11 ((uint32_t)0x00000800) /* !<Filter bit 11 */
#define CAN_F6R1_FB12 ((uint32_t)0x00001000) /* !<Filter bit 12 */
#define CAN_F6R1_FB13 ((uint32_t)0x00002000) /* !<Filter bit 13 */
#define CAN_F6R1_FB14 ((uint32_t)0x00004000) /* !<Filter bit 14 */
#define CAN_F6R1_FB15 ((uint32_t)0x00008000) /* !<Filter bit 15 */
#define CAN_F6R1_FB16 ((uint32_t)0x00010000) /* !<Filter bit 16 */
#define CAN_F6R1_FB17 ((uint32_t)0x00020000) /* !<Filter bit 17 */
#define CAN_F6R1_FB18 ((uint32_t)0x00040000) /* !<Filter bit 18 */
#define CAN_F6R1_FB19 ((uint32_t)0x00080000) /* !<Filter bit 19 */
#define CAN_F6R1_FB20 ((uint32_t)0x00100000) /* !<Filter bit 20 */
#define CAN_F6R1_FB21 ((uint32_t)0x00200000) /* !<Filter bit 21 */
#define CAN_F6R1_FB22 ((uint32_t)0x00400000) /* !<Filter bit 22 */
#define CAN_F6R1_FB23 ((uint32_t)0x00800000) /* !<Filter bit 23 */
#define CAN_F6R1_FB24 ((uint32_t)0x01000000) /* !<Filter bit 24 */
#define CAN_F6R1_FB25 ((uint32_t)0x02000000) /* !<Filter bit 25 */
#define CAN_F6R1_FB26 ((uint32_t)0x04000000) /* !<Filter bit 26 */
#define CAN_F6R1_FB27 ((uint32_t)0x08000000) /* !<Filter bit 27 */
#define CAN_F6R1_FB28 ((uint32_t)0x10000000) /* !<Filter bit 28 */
#define CAN_F6R1_FB29 ((uint32_t)0x20000000) /* !<Filter bit 29 */
#define CAN_F6R1_FB30 ((uint32_t)0x40000000) /* !<Filter bit 30 */
#define CAN_F6R1_FB31 ((uint32_t)0x80000000) /* !<Filter bit 31 */

***** Bit definition for CAN_F7R1 register *****/
#define CAN_F7R1_FBO ((uint32_t)0x00000001) /* !<Filter bit 0 */
#define CAN_F7R1_FBI ((uint32_t)0x00000002) /* !<Filter bit 1 */

```

```

#define CAN_F7R1_FB2 ((uint32_t)0x00000004) /* !<Filter bit 2 */
#define CAN_F7R1_FB3 ((uint32_t)0x00000008) /* !<Filter bit 3 */
#define CAN_F7R1_FB4 ((uint32_t)0x00000010) /* !<Filter bit 4 */
#define CAN_F7R1_FB5 ((uint32_t)0x00000020) /* !<Filter bit 5 */
#define CAN_F7R1_FB6 ((uint32_t)0x00000040) /* !<Filter bit 6 */
#define CAN_F7R1_FB7 ((uint32_t)0x00000080) /* !<Filter bit 7 */
#define CAN_F7R1_FB8 ((uint32_t)0x00000100) /* !<Filter bit 8 */
#define CAN_F7R1_FB9 ((uint32_t)0x00000200) /* !<Filter bit 9 */
#define CAN_F7R1_FB10 ((uint32_t)0x00000400) /* !<Filter bit 10 */
#define CAN_F7R1_FB11 ((uint32_t)0x00000800) /* !<Filter bit 11 */
#define CAN_F7R1_FB12 ((uint32_t)0x00001000) /* !<Filter bit 12 */
#define CAN_F7R1_FB13 ((uint32_t)0x00002000) /* !<Filter bit 13 */
#define CAN_F7R1_FB14 ((uint32_t)0x00004000) /* !<Filter bit 14 */
#define CAN_F7R1_FB15 ((uint32_t)0x00008000) /* !<Filter bit 15 */
#define CAN_F7R1_FB16 ((uint32_t)0x00010000) /* !<Filter bit 16 */
#define CAN_F7R1_FB17 ((uint32_t)0x00020000) /* !<Filter bit 17 */
#define CAN_F7R1_FB18 ((uint32_t)0x00040000) /* !<Filter bit 18 */
#define CAN_F7R1_FB19 ((uint32_t)0x00080000) /* !<Filter bit 19 */
#define CAN_F7R1_FB20 ((uint32_t)0x00100000) /* !<Filter bit 20 */
#define CAN_F7R1_FB21 ((uint32_t)0x00200000) /* !<Filter bit 21 */
#define CAN_F7R1_FB22 ((uint32_t)0x00400000) /* !<Filter bit 22 */
#define CAN_F7R1_FB23 ((uint32_t)0x00800000) /* !<Filter bit 23 */
#define CAN_F7R1_FB24 ((uint32_t)0x01000000) /* !<Filter bit 24 */
#define CAN_F7R1_FB25 ((uint32_t)0x02000000) /* !<Filter bit 25 */
#define CAN_F7R1_FB26 ((uint32_t)0x04000000) /* !<Filter bit 26 */
#define CAN_F7R1_FB27 ((uint32_t)0x08000000) /* !<Filter bit 27 */
#define CAN_F7R1_FB28 ((uint32_t)0x10000000) /* !<Filter bit 28 */
#define CAN_F7R1_FB29 ((uint32_t)0x20000000) /* !<Filter bit 29 */
#define CAN_F7R1_FB30 ((uint32_t)0x40000000) /* !<Filter bit 30 */
#define CAN_F7R1_FB31 ((uint32_t)0x80000000) /* !<Filter bit 31 */

/* ***** Bit definition for CAN_F8R1 register *****/
#define CAN_F8R1_FBO ((uint32_t)0x00000001) /* !<Filter bit 0 */
#define CAN_F8R1_FB1 ((uint32_t)0x00000002) /* !<Filter bit 1 */
#define CAN_F8R1_FB2 ((uint32_t)0x00000004) /* !<Filter bit 2 */

```

```

#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)

/* !<Filter bit 3 */
/* !<Filter bit 4 */
/* !<Filter bit 5 */
/* !<Filter bit 6 */
/* !<Filter bit 7 */
/* !<Filter bit 8 */
/* !<Filter bit 9 */
/* !<Filter bit 10 */
/* !<Filter bit 11 */
/* !<Filter bit 12 */
/* !<Filter bit 13 */
/* !<Filter bit 14 */
/* !<Filter bit 15 */
/* !<Filter bit 16 */
/* !<Filter bit 17 */
/* !<Filter bit 18 */
/* !<Filter bit 19 */
/* !<Filter bit 20 */
/* !<Filter bit 21 */
/* !<Filter bit 22 */
/* !<Filter bit 23 */
/* !<Filter bit 24 */
/* !<Filter bit 25 */
/* !<Filter bit 26 */
/* !<Filter bit 27 */
/* !<Filter bit 28 */
/* !<Filter bit 29 */
/* !<Filter bit 30 */
/* !<Filter bit 31 */

/**************** Bit definition for CAN_F9R1 register *****/
#define CAN_F9R1_FBO ((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)0x000000010)
#define CAN_F9R1_FB5 ((uint32_t)0x000000020)
#define CAN_F9R1_FB6 ((uint32_t)0x000000040)
#define CAN_F9R1_FB7 ((uint32_t)0x000000080)
#define CAN_F9R1_FB8 ((uint32_t)0x000000100)
#define CAN_F9R1_FB9 ((uint32_t)0x000000200)
#define CAN_F9R1_FB10 ((uint32_t)0x000000400)
#define CAN_F9R1_FB11 ((uint32_t)0x000000800)
#define CAN_F9R1_FB12 ((uint32_t)0x000001000)
#define CAN_F9R1_FB13 ((uint32_t)0x000002000)
#define CAN_F9R1_FB14 ((uint32_t)0x000004000)
#define CAN_F9R1_FB15 ((uint32_t)0x000008000)
#define CAN_F9R1_FB16 ((uint32_t)0x000010000)
#define CAN_F9R1_FB17 ((uint32_t)0x000020000)
#define CAN_F9R1_FB18 ((uint32_t)0x000040000)
#define CAN_F9R1_FB19 ((uint32_t)0x000080000)
#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)

/* !<Filter bit 4 */
/* !<Filter bit 5 */
/* !<Filter bit 6 */
/* !<Filter bit 7 */
/* !<Filter bit 8 */
/* !<Filter bit 9 */
/* !<Filter bit 10 */
/* !<Filter bit 11 */
/* !<Filter bit 12 */
/* !<Filter bit 13 */
/* !<Filter bit 14 */
/* !<Filter bit 15 */
/* !<Filter bit 16 */
/* !<Filter bit 17 */
/* !<Filter bit 18 */
/* !<Filter bit 19 */
/* !<Filter bit 20 */
/* !<Filter bit 21 */
/* !<Filter bit 22 */
/* !<Filter bit 23 */
/* !<Filter bit 24 */
/* !<Filter bit 25 */
/* !<Filter bit 26 */
/* !<Filter bit 27 */
/* !<Filter bit 28 */
/* !<Filter bit 29 */
/* !<Filter bit 30 */
/* !<Filter bit 31 */

***** Bit definition for CAN_F10R1 register *****

#define CAN_F10R1_FBO ((uint32_t)0x000000001)
#define CAN_F10R1_FBI ((uint32_t)0x000000002)
#define CAN_F10R1_FB2 ((uint32_t)0x000000004)
#define CAN_F10R1_FB3 ((uint32_t)0x000000008)
#define CAN_F10R1_FB4 ((uint32_t)0x000000010)

/* !<Filter bit 0 */
/* !<Filter bit 1 */
/* !<Filter bit 2 */
/* !<Filter bit 3 */
/* !<Filter bit 4 */

```

```

#define CAN_F1OR1_FB5      (((uint32_t)0x000000020)/* !<Filter bit 5 */)
#define CAN_F1OR1_FB6      (((uint32_t)0x000000040)/* !<Filter bit 6 */)
#define CAN_F1OR1_FB7      (((uint32_t)0x000000080)/* !<Filter bit 7 */)
#define CAN_F1OR1_FB8      (((uint32_t)0x000000100)/* !<Filter bit 8 */)
#define CAN_F1OR1_FB9      (((uint32_t)0x000000200)/* !<Filter bit 9 */)
#define CAN_F1OR1_FB10     (((uint32_t)0x000000400)/* !<Filter bit 10 */)
#define CAN_F1OR1_FB11     (((uint32_t)0x000000800)/* !<Filter bit 11 */)
#define CAN_F1OR1_FB12     (((uint32_t)0x000001000)/* !<Filter bit 12 */)
#define CAN_F1OR1_FB13     (((uint32_t)0x000002000)/* !<Filter bit 13 */)
#define CAN_F1OR1_FB14     (((uint32_t)0x000004000)/* !<Filter bit 14 */)
#define CAN_F1OR1_FB15     (((uint32_t)0x000008000)/* !<Filter bit 15 */)
#define CAN_F1OR1_FB16     (((uint32_t)0x000010000)/* !<Filter bit 16 */)
#define CAN_F1OR1_FB17     (((uint32_t)0x000020000)/* !<Filter bit 17 */)
#define CAN_F1OR1_FB18     (((uint32_t)0x000040000)/* !<Filter bit 18 */)
#define CAN_F1OR1_FB19     (((uint32_t)0x000080000)/* !<Filter bit 19 */)
#define CAN_F1OR1_FB20     (((uint32_t)0x001000000)/* !<Filter bit 20 */)
#define CAN_F1OR1_FB21     (((uint32_t)0x002000000)/* !<Filter bit 21 */)
#define CAN_F1OR1_FB22     (((uint32_t)0x004000000)/* !<Filter bit 22 */)
#define CAN_F1OR1_FB23     (((uint32_t)0x008000000)/* !<Filter bit 23 */)
#define CAN_F1OR1_FB24     (((uint32_t)0x010000000)/* !<Filter bit 24 */)
#define CAN_F1OR1_FB25     (((uint32_t)0x020000000)/* !<Filter bit 25 */)
#define CAN_F1OR1_FB26     (((uint32_t)0x040000000)/* !<Filter bit 26 */)
#define CAN_F1OR1_FB27     (((uint32_t)0x080000000)/* !<Filter bit 27 */)
#define CAN_F1OR1_FB28     (((uint32_t)0x100000000)/* !<Filter bit 28 */)
#define CAN_F1OR1_FB29     (((uint32_t)0x200000000)/* !<Filter bit 29 */)
#define CAN_F1OR1_FB30     (((uint32_t)0x400000000)/* !<Filter bit 30 */)
#define CAN_F1OR1_FB31     (((uint32_t)0x800000000)/* !<Filter bit 31 */)

/********************* Bit definition for CAN_F11R1 register *****/
#define CAN_F11R1_FBO      ((uint32_t)0x000000001)/* !<Filter bit 0 */
#define CAN_F11R1_FB1      ((uint32_t)0x000000002)/* !<Filter bit 1 */
#define CAN_F11R1_FB2      ((uint32_t)0x000000004)/* !<Filter bit 2 */
#define CAN_F11R1_FB3      ((uint32_t)0x000000008)/* !<Filter bit 3 */
#define CAN_F11R1_FB4      ((uint32_t)0x000000010)/* !<Filter bit 4 */
#define CAN_F11R1_FB5      ((uint32_t)0x000000020)/* !<Filter bit 5 */

```

```

#define CAN_F11R1_FB6 ((uint32_t)0x000000040)
#define CAN_F11R1_FB7 ((uint32_t)0x000000080)
#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)

/* !<Filter bit 6 */
/* !<Filter bit 7 */
/* !<Filter bit 8 */
/* !<Filter bit 9 */
/* !<Filter bit 10 */
/* !<Filter bit 11 */
/* !<Filter bit 12 */
/* !<Filter bit 13 */
/* !<Filter bit 14 */
/* !<Filter bit 15 */
/* !<Filter bit 16 */
/* !<Filter bit 17 */
/* !<Filter bit 18 */
/* !<Filter bit 19 */
/* !<Filter bit 20 */
/* !<Filter bit 21 */
/* !<Filter bit 22 */
/* !<Filter bit 23 */
/* !<Filter bit 24 */
/* !<Filter bit 25 */
/* !<Filter bit 26 */
/* !<Filter bit 27 */
/* !<Filter bit 28 */
/* !<Filter bit 29 */
/* !<Filter bit 30 */
/* !<Filter bit 31 */

// Bit definition for CAN_F12R1 register ****
#define CAN_F12R1_FBO ((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)0x000000080)
#define CAN_F12R1_FB8 ((uint32_t)0x000000100)
#define CAN_F12R1_FB9 ((uint32_t)0x000000200)
#define CAN_F12R1_FB10 ((uint32_t)0x000000400)
#define CAN_F12R1_FB11 ((uint32_t)0x000000800)
#define CAN_F12R1_FB12 ((uint32_t)0x000001000)
#define CAN_F12R1_FB13 ((uint32_t)0x000002000)
#define CAN_F12R1_FB14 ((uint32_t)0x000004000)
#define CAN_F12R1_FB15 ((uint32_t)0x000008000)
#define CAN_F12R1_FB16 ((uint32_t)0x000010000)
#define CAN_F12R1_FB17 ((uint32_t)0x000020000)
#define CAN_F12R1_FB18 ((uint32_t)0x000040000)
#define CAN_F12R1_FB19 ((uint32_t)0x000080000)
#define CAN_F12R1_FB20 ((uint32_t)0x000100000)
#define CAN_F12R1_FB21 ((uint32_t)0x000200000)
#define CAN_F12R1_FB22 ((uint32_t)0x000400000)
#define CAN_F12R1_FB23 ((uint32_t)0x000800000)
#define CAN_F12R1_FB24 ((uint32_t)0x001000000)
#define CAN_F12R1_FB25 ((uint32_t)0x002000000)
#define CAN_F12R1_FB26 ((uint32_t)0x004000000)
#define CAN_F12R1_FB27 ((uint32_t)0x008000000)
#define CAN_F12R1_FB28 ((uint32_t)0x010000000)
#define CAN_F12R1_FB29 ((uint32_t)0x020000000)
#define CAN_F12R1_FB30 ((uint32_t)0x040000000)
#define CAN_F12R1_FB31 ((uint32_t)0x080000000)

/* !<Filter bit 7 */
/* !<Filter bit 8 */
/* !<Filter bit 9 */
/* !<Filter bit 10 */
/* !<Filter bit 11 */
/* !<Filter bit 12 */
/* !<Filter bit 13 */
/* !<Filter bit 14 */
/* !<Filter bit 15 */
/* !<Filter bit 16 */
/* !<Filter bit 17 */
/* !<Filter bit 18 */
/* !<Filter bit 19 */
/* !<Filter bit 20 */
/* !<Filter bit 21 */
/* !<Filter bit 22 */
/* !<Filter bit 23 */
/* !<Filter bit 24 */
/* !<Filter bit 25 */
/* !<Filter bit 26 */
/* !<Filter bit 27 */
/* !<Filter bit 28 */
/* !<Filter bit 29 */
/* !<Filter bit 30 */
/* !<Filter bit 31 */

// Bit definition for CAN_F13R1 register ****
#define CAN_F13R1_FBO ((uint32_t)0x000000001)
#define CAN_F13R1_FBI ((uint32_t)0x000000002)
#define CAN_F13R1_FB2 ((uint32_t)0x000000004)
#define CAN_F13R1_FB3 ((uint32_t)0x000000008)
#define CAN_F13R1_FB4 ((uint32_t)0x000000010)
#define CAN_F13R1_FB5 ((uint32_t)0x000000020)
#define CAN_F13R1_FB6 ((uint32_t)0x000000040)
#define CAN_F13R1_FB7 ((uint32_t)0x000000080)

```

```

#define CAN_F13R1_FB8 ((uint32_t)0x000000100)
#define CAN_F13R1_FB9 ((uint32_t)0x000000200)
#define CAN_F13R1_FB10 ((uint32_t)0x000000400)
#define CAN_F13R1_FB11 ((uint32_t)0x000000800)
#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)

/* Bit definition for CAN_FOR2 register *****/
#define CAN_FOR2_FBO ((uint32_t)0x00000001)
#define CAN_FOR2_FB1 ((uint32_t)0x00000002)
#define CAN_FOR2_FB2 ((uint32_t)0x00000004)
#define CAN_FOR2_FB3 ((uint32_t)0x00000008)
#define CAN_FOR2_FB4 ((uint32_t)0x00000010)
#define CAN_FOR2_FB5 ((uint32_t)0x00000020)
#define CAN_FOR2_FB6 ((uint32_t)0x00000040)
#define CAN_FOR2_FB7 ((uint32_t)0x00000080)
#define CAN_FOR2_FB8 ((uint32_t)0x00000100)

/* Bit definition for CAN_FOR register *****/
#define CAN_FOR_FBO ((uint32_t)0x00000001)
#define CAN_FOR_FB1 ((uint32_t)0x00000002)
#define CAN_FOR_FB2 ((uint32_t)0x00000004)
#define CAN_FOR_FB3 ((uint32_t)0x00000008)
#define CAN_FOR_FB4 ((uint32_t)0x00000010)
#define CAN_FOR_FB5 ((uint32_t)0x00000020)
#define CAN_FOR_FB6 ((uint32_t)0x00000040)
#define CAN_FOR_FB7 ((uint32_t)0x00000080)
#define CAN_FOR_FB8 ((uint32_t)0x00000100)

```

```

#define CAN_FOR2_FB9          (((uint32_t)0x000000200)/* !<Filter bit 9 */
#define CAN_FOR2_FB10         (((uint32_t)0x000000400)/* !<Filter bit 10 */
#define CAN_FOR2_FB11         (((uint32_t)0x000000800)/* !<Filter bit 11 */
#define CAN_FOR2_FB12         (((uint32_t)0x000001000)/* !<Filter bit 12 */
#define CAN_FOR2_FB13         (((uint32_t)0x000002000)/* !<Filter bit 13 */
#define CAN_FOR2_FB14         (((uint32_t)0x000004000)/* !<Filter bit 14 */
#define CAN_FOR2_FB15         (((uint32_t)0x000008000)/* !<Filter bit 15 */
#define CAN_FOR2_FB16         (((uint32_t)0x000010000)/* !<Filter bit 16 */
#define CAN_FOR2_FB17         (((uint32_t)0x000020000)/* !<Filter bit 17 */
#define CAN_FOR2_FB18         (((uint32_t)0x000040000)/* !<Filter bit 18 */
#define CAN_FOR2_FB19         (((uint32_t)0x000080000)/* !<Filter bit 19 */
#define CAN_FOR2_FB20         (((uint32_t)0x000100000)/* !<Filter bit 20 */
#define CAN_FOR2_FB21         (((uint32_t)0x000200000)/* !<Filter bit 21 */
#define CAN_FOR2_FB22         (((uint32_t)0x000400000)/* !<Filter bit 22 */
#define CAN_FOR2_FB23         (((uint32_t)0x000800000)/* !<Filter bit 23 */
#define CAN_FOR2_FB24         (((uint32_t)0x010000000)/* !<Filter bit 24 */
#define CAN_FOR2_FB25         (((uint32_t)0x020000000)/* !<Filter bit 25 */
#define CAN_FOR2_FB26         (((uint32_t)0x040000000)/* !<Filter bit 26 */
#define CAN_FOR2_FB27         (((uint32_t)0x080000000)/* !<Filter bit 27 */
#define CAN_FOR2_FB28         (((uint32_t)0x100000000)/* !<Filter bit 28 */
#define CAN_FOR2_FB29         (((uint32_t)0x200000000)/* !<Filter bit 29 */
#define CAN_FOR2_FB30         (((uint32_t)0x400000000)/* !<Filter bit 30 */
#define CAN_FOR2_FB31         (((uint32_t)0x800000000)/* !<Filter bit 31 */

/* ***** Bit definition for CAN_F1R2 register *****

#define CAN_F1R2_FBO          ((uint32_t)0x000000001)/* !<Filter bit 0 */
#define CAN_F1R2_FB1          ((uint32_t)0x000000002)/* !<Filter bit 1 */
#define CAN_F1R2_FB2          ((uint32_t)0x000000004)/* !<Filter bit 2 */
#define CAN_F1R2_FB3          ((uint32_t)0x000000008)/* !<Filter bit 3 */
#define CAN_F1R2_FB4          ((uint32_t)0x000000010)/* !<Filter bit 4 */
#define CAN_F1R2_FB5          ((uint32_t)0x000000020)/* !<Filter bit 5 */
#define CAN_F1R2_FB6          ((uint32_t)0x000000040)/* !<Filter bit 6 */
#define CAN_F1R2_FB7          ((uint32_t)0x000000080)/* !<Filter bit 7 */
#define CAN_F1R2_FB8          ((uint32_t)0x000000100)/* !<Filter bit 8 */
#define CAN_F1R2_FB9          ((uint32_t)0x000000200)/* !<Filter bit 9 */

```

```

/* !<Filter bit 10 */
((uint32_t)0x000000400)
CAN_F1R2_FB10
CAN_F1R2_FB11
CAN_F1R2_FB12
CAN_F1R2_FB13
CAN_F1R2_FB14
CAN_F1R2_FB15
CAN_F1R2_FB16
CAN_F1R2_FB17
CAN_F1R2_FB18
CAN_F1R2_FB19
CAN_F1R2_FB20
CAN_F1R2_FB21
CAN_F1R2_FB22
CAN_F1R2_FB23
CAN_F1R2_FB24
CAN_F1R2_FB25
CAN_F1R2_FB26
CAN_F1R2_FB27
CAN_F1R2_FB28
CAN_F1R2_FB29
CAN_F1R2_FB30
CAN_F1R2_FB31

/* !<Filter bit 11 */
((uint32_t)0x000000800)
((uint32_t)0x00001000)
((uint32_t)0x00002000)
((uint32_t)0x00004000)
((uint32_t)0x00008000)
((uint32_t)0x00010000)
((uint32_t)0x00020000)
((uint32_t)0x00040000)
((uint32_t)0x00080000)
((uint32_t)0x00100000)
((uint32_t)0x00200000)
((uint32_t)0x00400000)
((uint32_t)0x00800000)
((uint32_t)0x01000000)
((uint32_t)0x02000000)
((uint32_t)0x04000000)
((uint32_t)0x08000000)
((uint32_t)0x10000000)
((uint32_t)0x20000000)
((uint32_t)0x40000000)
((uint32_t)0x80000000)

/* !<Filter bit 12 */
((uint32_t)0x00000001)
((uint32_t)0x00000002)
((uint32_t)0x00000004)
((uint32_t)0x00000008)
((uint32_t)0x00000010)
((uint32_t)0x00000020)
((uint32_t)0x00000040)
((uint32_t)0x00000080)
((uint32_t)0x00001000)
((uint32_t)0x00002000)
((uint32_t)0x00004000)

/* !<Filter bit 13 */
((uint32_t)0x00000000)
(* !<Filter bit 14 */
((uint32_t)0x00000000)
(* !<Filter bit 15 */
((uint32_t)0x00000000)
(* !<Filter bit 16 */
((uint32_t)0x00000000)
(* !<Filter bit 17 */
((uint32_t)0x00000000)
(* !<Filter bit 18 */
((uint32_t)0x00000000)
(* !<Filter bit 19 */
((uint32_t)0x00000000)
(* !<Filter bit 20 */
((uint32_t)0x00000000)
(* !<Filter bit 21 */
((uint32_t)0x00000000)
(* !<Filter bit 22 */
((uint32_t)0x00000000)
(* !<Filter bit 23 */
((uint32_t)0x00000000)
(* !<Filter bit 24 */
((uint32_t)0x00000000)
(* !<Filter bit 25 */
((uint32_t)0x00000000)
(* !<Filter bit 26 */
((uint32_t)0x00000000)
(* !<Filter bit 27 */
((uint32_t)0x00000000)
(* !<Filter bit 28 */
((uint32_t)0x00000000)
(* !<Filter bit 29 */
((uint32_t)0x00000000)
(* !<Filter bit 30 */
((uint32_t)0x00000000)
(* !<Filter bit 31 */

***** Bit definition for CAN_F2R2 register *****/
CAN_F2R2_FBO
CAN_F2R2_FB1
CAN_F2R2_FB2
CAN_F2R2_FB3
CAN_F2R2_FB4
CAN_F2R2_FB5
CAN_F2R2_FB6
CAN_F2R2_FB7
CAN_F2R2_FB8
CAN_F2R2_FB9
CAN_F2R2_FB10
CAN_F2R2_FB11
CAN_F2R2_FB12
CAN_F2R2_FB13
CAN_F2R2_FB14
CAN_F2R2_FB15
CAN_F2R2_FB16
CAN_F2R2_FB17
CAN_F2R2_FB18
CAN_F2R2_FB19
CAN_F2R2_FB20
CAN_F2R2_FB21
CAN_F2R2_FB22
CAN_F2R2_FB23
CAN_F2R2_FB24
CAN_F2R2_FB25
CAN_F2R2_FB26
CAN_F2R2_FB27
CAN_F2R2_FB28
CAN_F2R2_FB29
CAN_F2R2_FB30
CAN_F2R2_FB31

```

```

#define CAN_F2R2_FB11 ((uint32_t)0x000000800)
#define CAN_F2R2_FB12 ((uint32_t)0x000001000)
#define CAN_F2R2_FB13 ((uint32_t)0x000002000)
#define CAN_F2R2_FB14 ((uint32_t)0x000004000)
#define CAN_F2R2_FB15 ((uint32_t)0x000008000)
#define CAN_F2R2_FB16 ((uint32_t)0x000100000)
#define CAN_F2R2_FB17 ((uint32_t)0x000200000)
#define CAN_F2R2_FB18 ((uint32_t)0x000400000)
#define CAN_F2R2_FB19 ((uint32_t)0x000800000)
#define CAN_F2R2_FB20 ((uint32_t)0x001000000)
#define CAN_F2R2_FB21 ((uint32_t)0x002000000)
#define CAN_F2R2_FB22 ((uint32_t)0x004000000)
#define CAN_F2R2_FB23 ((uint32_t)0x008000000)
#define CAN_F2R2_FB24 ((uint32_t)0x010000000)
#define CAN_F2R2_FB25 ((uint32_t)0x020000000)
#define CAN_F2R2_FB26 ((uint32_t)0x040000000)
#define CAN_F2R2_FB27 ((uint32_t)0x080000000)
#define CAN_F2R2_FB28 ((uint32_t)0x100000000)
#define CAN_F2R2_FB29 ((uint32_t)0x200000000)
#define CAN_F2R2_FB30 ((uint32_t)0x400000000)
#define CAN_F2R2_FB31 ((uint32_t)0x800000000)

/* ***** Bit definition for CAN_F3R2 register *****/
#define CAN_F3R2_FBO ((uint32_t)0x000000001)
#define CAN_F3R2_FB1 ((uint32_t)0x000000002)
#define CAN_F3R2_FB2 ((uint32_t)0x000000004)
#define CAN_F3R2_FB3 ((uint32_t)0x000000008)
#define CAN_F3R2_FB4 ((uint32_t)0x000000010)
#define CAN_F3R2_FB5 ((uint32_t)0x000000020)
#define CAN_F3R2_FB6 ((uint32_t)0x000000040)
#define CAN_F3R2_FB7 ((uint32_t)0x000000080)
#define CAN_F3R2_FB8 ((uint32_t)0x000000100)
#define CAN_F3R2_FB9 ((uint32_t)0x000000200)
#define CAN_F3R2_FB10 ((uint32_t)0x000000400)
#define CAN_F3R2_FB11 ((uint32_t)0x000000800)

/* !<Filter bit 11 */
/* !<Filter bit 12 */
/* !<Filter bit 13 */
/* !<Filter bit 14 */
/* !<Filter bit 15 */
/* !<Filter bit 16 */
/* !<Filter bit 17 */
/* !<Filter bit 18 */
/* !<Filter bit 19 */
/* !<Filter bit 20 */
/* !<Filter bit 21 */
/* !<Filter bit 22 */
/* !<Filter bit 23 */
/* !<Filter bit 24 */
/* !<Filter bit 25 */
/* !<Filter bit 26 */
/* !<Filter bit 27 */
/* !<Filter bit 28 */
/* !<Filter bit 29 */
/* !<Filter bit 30 */
/* !<Filter bit 31 */

```

```

#define CAN_F3R2_FB12 ((uint32_t)0x000001000)
#define CAN_F3R2_FB13 ((uint32_t)0x000002000)
#define CAN_F3R2_FB14 ((uint32_t)0x000004000)
#define CAN_F3R2_FB15 ((uint32_t)0x000008000)
#define CAN_F3R2_FB16 ((uint32_t)0x000010000)
#define CAN_F3R2_FB17 ((uint32_t)0x000020000)
#define CAN_F3R2_FB18 ((uint32_t)0x000040000)
#define CAN_F3R2_FB19 ((uint32_t)0x000080000)
#define CAN_F3R2_FB20 ((uint32_t)0x001000000)
#define CAN_F3R2_FB21 ((uint32_t)0x002000000)
#define CAN_F3R2_FB22 ((uint32_t)0x004000000)
#define CAN_F3R2_FB23 ((uint32_t)0x008000000)
#define CAN_F3R2_FB24 ((uint32_t)0x010000000)
#define CAN_F3R2_FB25 ((uint32_t)0x020000000)
#define CAN_F3R2_FB26 ((uint32_t)0x040000000)
#define CAN_F3R2_FB27 ((uint32_t)0x080000000)
#define CAN_F3R2_FB28 ((uint32_t)0x100000000)
#define CAN_F3R2_FB29 ((uint32_t)0x200000000)
#define CAN_F3R2_FB30 ((uint32_t)0x400000000)
#define CAN_F3R2_FB31 ((uint32_t)0x800000000)

/* !<Filter bit 12 */
/* !<Filter bit 13 */
/* !<Filter bit 14 */
/* !<Filter bit 15 */
/* !<Filter bit 16 */
/* !<Filter bit 17 */
/* !<Filter bit 18 */
/* !<Filter bit 19 */
/* !<Filter bit 20 */
/* !<Filter bit 21 */
/* !<Filter bit 22 */
/* !<Filter bit 23 */
/* !<Filter bit 24 */
/* !<Filter bit 25 */
/* !<Filter bit 26 */
/* !<Filter bit 27 */
/* !<Filter bit 28 */
/* !<Filter bit 29 */
/* !<Filter bit 30 */
/* !<Filter bit 31 */

/* ***** Bit definition for CAN_F4R2 register *****/
#define CAN_F4R2_FBO ((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)0x0000020000)
#define CAN_F4R2_FB14 ((uint32_t)0x0000040000)
#define CAN_F4R2_FB15 ((uint32_t)0x0000080000)
#define CAN_F4R2_FB16 ((uint32_t)0x0001000000)
#define CAN_F4R2_FB17 ((uint32_t)0x0002000000)
#define CAN_F4R2_FB18 ((uint32_t)0x0004000000)
#define CAN_F4R2_FB19 ((uint32_t)0x0008000000)
#define CAN_F4R2_FB20 ((uint32_t)0x0010000000)
#define CAN_F4R2_FB21 ((uint32_t)0x0020000000)
#define CAN_F4R2_FB22 ((uint32_t)0x0040000000)
#define CAN_F4R2_FB23 ((uint32_t)0x0080000000)
#define CAN_F4R2_FB24 ((uint32_t)0x0100000000)
#define CAN_F4R2_FB25 ((uint32_t)0x0200000000)
#define CAN_F4R2_FB26 ((uint32_t)0x0400000000)
#define CAN_F4R2_FB27 ((uint32_t)0x0800000000)
#define CAN_F4R2_FB28 ((uint32_t)0x1000000000)
#define CAN_F4R2_FB29 ((uint32_t)0x2000000000)
#define CAN_F4R2_FB30 ((uint32_t)0x4000000000)
#define CAN_F4R2_FB31 ((uint32_t)0x8000000000)

/* !<Filter bit 13 */
/* !<Filter bit 14 */
/* !<Filter bit 15 */
/* !<Filter bit 16 */
/* !<Filter bit 17 */
/* !<Filter bit 18 */
/* !<Filter bit 19 */
/* !<Filter bit 20 */
/* !<Filter bit 21 */
/* !<Filter bit 22 */
/* !<Filter bit 23 */
/* !<Filter bit 24 */
/* !<Filter bit 25 */
/* !<Filter bit 26 */
/* !<Filter bit 27 */
/* !<Filter bit 28 */
/* !<Filter bit 29 */
/* !<Filter bit 30 */
/* !<Filter bit 31 */

/**************** Bit definition for CAN_F5R2 register *****/
#define CAN_F5R2_FBO ((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)0x000004000)
#define CAN_F5R2_FB15 ((uint32_t)0x000008000)
#define CAN_F5R2_FB16 ((uint32_t)0x000100000)
#define CAN_F5R2_FB17 ((uint32_t)0x000200000)
#define CAN_F5R2_FB18 ((uint32_t)0x000400000)
#define CAN_F5R2_FB19 ((uint32_t)0x000800000)
#define CAN_F5R2_FB20 ((uint32_t)0x001000000)
#define CAN_F5R2_FB21 ((uint32_t)0x002000000)
#define CAN_F5R2_FB22 ((uint32_t)0x004000000)
#define CAN_F5R2_FB23 ((uint32_t)0x008000000)
#define CAN_F5R2_FB24 ((uint32_t)0x010000000)
#define CAN_F5R2_FB25 ((uint32_t)0x020000000)
#define CAN_F5R2_FB26 ((uint32_t)0x040000000)
#define CAN_F5R2_FB27 ((uint32_t)0x080000000)
#define CAN_F5R2_FB28 ((uint32_t)0x100000000)
#define CAN_F5R2_FB29 ((uint32_t)0x200000000)
#define CAN_F5R2_FB30 ((uint32_t)0x400000000)
#define CAN_F5R2_FB31 ((uint32_t)0x800000000)

/* !<Filter bit 14 */
/* !<Filter bit 15 */
/* !<Filter bit 16 */
/* !<Filter bit 17 */
/* !<Filter bit 18 */
/* !<Filter bit 19 */
/* !<Filter bit 20 */
/* !<Filter bit 21 */
/* !<Filter bit 22 */
/* !<Filter bit 23 */
/* !<Filter bit 24 */
/* !<Filter bit 25 */
/* !<Filter bit 26 */
/* !<Filter bit 27 */
/* !<Filter bit 28 */
/* !<Filter bit 29 */
/* !<Filter bit 30 */
/* !<Filter bit 31 */

/* Bit definition for CAN_F6R2 register *****/
#define CAN_F6R2_FBO ((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)

/* !<Filter bit 0 */
/* !<Filter bit 1 */
/* !<Filter bit 2 */
/* !<Filter bit 3 */
/* !<Filter bit 4 */
/* !<Filter bit 5 */
/* !<Filter bit 6 */
/* !<Filter bit 7 */
/* !<Filter bit 8 */
/* !<Filter bit 9 */
/* !<Filter bit 10 */
/* !<Filter bit 11 */
/* !<Filter bit 12 */
/* !<Filter bit 13 */
/* !<Filter bit 14 */

```

```

#define CAN_F6R2_FB15 ((uint32_t)0x000008000)
#define CAN_F6R2_FB16 ((uint32_t)0x000010000)
#define CAN_F6R2_FB17 ((uint32_t)0x000020000)
#define CAN_F6R2_FB18 ((uint32_t)0x000040000)
#define CAN_F6R2_FB19 ((uint32_t)0x000080000)
#define CAN_F6R2_FB20 ((uint32_t)0x001000000)
#define CAN_F6R2_FB21 ((uint32_t)0x002000000)
#define CAN_F6R2_FB22 ((uint32_t)0x004000000)
#define CAN_F6R2_FB23 ((uint32_t)0x008000000)
#define CAN_F6R2_FB24 ((uint32_t)0x010000000)
#define CAN_F6R2_FB25 ((uint32_t)0x020000000)
#define CAN_F6R2_FB26 ((uint32_t)0x040000000)
#define CAN_F6R2_FB27 ((uint32_t)0x080000000)
#define CAN_F6R2_FB28 ((uint32_t)0x100000000)
#define CAN_F6R2_FB29 ((uint32_t)0x200000000)
#define CAN_F6R2_FB30 ((uint32_t)0x400000000)
#define CAN_F6R2_FB31 ((uint32_t)0x800000000)

/**************** Bit definition for CAN_F7R2 register *****/
#define CAN_F7R2_FBO ((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)

/* !<Filter bit 15 */
/* !<Filter bit 16 */
/* !<Filter bit 17 */
/* !<Filter bit 18 */
/* !<Filter bit 19 */
/* !<Filter bit 20 */
/* !<Filter bit 21 */
/* !<Filter bit 22 */
/* !<Filter bit 23 */
/* !<Filter bit 24 */
/* !<Filter bit 25 */
/* !<Filter bit 26 */
/* !<Filter bit 27 */
/* !<Filter bit 28 */
/* !<Filter bit 29 */
/* !<Filter bit 30 */
/* !<Filter bit 31 */

```

```

#define CAN_F7R2_FB16 ((uint32_t)0x000010000)
#define CAN_F7R2_FB17 ((uint32_t)0x000020000)
#define CAN_F7R2_FB18 ((uint32_t)0x000040000)
#define CAN_F7R2_FB19 ((uint32_t)0x000080000)
#define CAN_F7R2_FB20 ((uint32_t)0x001000000)
#define CAN_F7R2_FB21 ((uint32_t)0x002000000)
#define CAN_F7R2_FB22 ((uint32_t)0x004000000)
#define CAN_F7R2_FB23 ((uint32_t)0x008000000)
#define CAN_F7R2_FB24 ((uint32_t)0x010000000)
#define CAN_F7R2_FB25 ((uint32_t)0x020000000)
#define CAN_F7R2_FB26 ((uint32_t)0x040000000)
#define CAN_F7R2_FB27 ((uint32_t)0x080000000)
#define CAN_F7R2_FB28 ((uint32_t)0x100000000)
#define CAN_F7R2_FB29 ((uint32_t)0x200000000)
#define CAN_F7R2_FB30 ((uint32_t)0x400000000)
#define CAN_F7R2_FB31 ((uint32_t)0x800000000)

***** Bit definition for CAN_F8R2 register *****/
#define CAN_F8R2_FBO ((uint32_t)0x000000001)
#define CAN_F8R2_FB1 ((uint32_t)0x000000002)
#define CAN_F8R2_FB2 ((uint32_t)0x000000004)
#define CAN_F8R2_FB3 ((uint32_t)0x000000008)
#define CAN_F8R2_FB4 ((uint32_t)0x000000010)
#define CAN_F8R2_FB5 ((uint32_t)0x000000020)
#define CAN_F8R2_FB6 ((uint32_t)0x000000040)
#define CAN_F8R2_FB7 ((uint32_t)0x000000080)
#define CAN_F8R2_FB8 ((uint32_t)0x000000100)
#define CAN_F8R2_FB9 ((uint32_t)0x000000200)
#define CAN_F8R2_FB10 ((uint32_t)0x000000400)
#define CAN_F8R2_FB11 ((uint32_t)0x000000800)
#define CAN_F8R2_FB12 ((uint32_t)0x000001000)
#define CAN_F8R2_FB13 ((uint32_t)0x000002000)
#define CAN_F8R2_FB14 ((uint32_t)0x000004000)
#define CAN_F8R2_FB15 ((uint32_t)0x000008000)
#define CAN_F8R2_FB16 ((uint32_t)0x000010000)

/* !<Filter bit 16 */
/* !<Filter bit 17 */
/* !<Filter bit 18 */
/* !<Filter bit 19 */
/* !<Filter bit 20 */
/* !<Filter bit 21 */
/* !<Filter bit 22 */
/* !<Filter bit 23 */
/* !<Filter bit 24 */
/* !<Filter bit 25 */
/* !<Filter bit 26 */
/* !<Filter bit 27 */
/* !<Filter bit 28 */
/* !<Filter bit 29 */
/* !<Filter bit 30 */
/* !<Filter bit 31 */

***** Bit definition for CAN_F8R2 register *****/
/* !<Filter bit 0 */
/* !<Filter bit 1 */
/* !<Filter bit 2 */
/* !<Filter bit 3 */
/* !<Filter bit 4 */
/* !<Filter bit 5 */
/* !<Filter bit 6 */
/* !<Filter bit 7 */
/* !<Filter bit 8 */
/* !<Filter bit 9 */
/* !<Filter bit 10 */
/* !<Filter bit 11 */
/* !<Filter bit 12 */
/* !<Filter bit 13 */
/* !<Filter bit 14 */
/* !<Filter bit 15 */
/* !<Filter bit 16 */

```

```

#define CAN_F8R2_FB17 ((uint32_t)0x0000200000)
#define CAN_F8R2_FB18 ((uint32_t)0x0000400000)
#define CAN_F8R2_FB19 ((uint32_t)0x0000800000)
#define CAN_F8R2_FB20 ((uint32_t)0x0010000000)
#define CAN_F8R2_FB21 ((uint32_t)0x0020000000)
#define CAN_F8R2_FB22 ((uint32_t)0x0040000000)
#define CAN_F8R2_FB23 ((uint32_t)0x0080000000)
#define CAN_F8R2_FB24 ((uint32_t)0x0100000000)
#define CAN_F8R2_FB25 ((uint32_t)0x0200000000)
#define CAN_F8R2_FB26 ((uint32_t)0x0400000000)
#define CAN_F8R2_FB27 ((uint32_t)0x0800000000)
#define CAN_F8R2_FB28 ((uint32_t)0x1000000000)
#define CAN_F8R2_FB29 ((uint32_t)0x2000000000)
#define CAN_F8R2_FB30 ((uint32_t)0x4000000000)
#define CAN_F8R2_FB31 ((uint32_t)0x8000000000)

/* !<Filter bit 17 */
/* !<Filter bit 18 */
/* !<Filter bit 19 */
/* !<Filter bit 20 */
/* !<Filter bit 21 */
/* !<Filter bit 22 */
/* !<Filter bit 23 */
/* !<Filter bit 24 */
/* !<Filter bit 25 */
/* !<Filter bit 26 */
/* !<Filter bit 27 */
/* !<Filter bit 28 */
/* !<Filter bit 29 */
/* !<Filter bit 30 */
/* !<Filter bit 31 */

/* **** Bit definition for CAN_F9R2 register ****/
/* **** CAN_F9R2_FBO */
#define CAN_F9R2_FB0 ((uint32_t)0x0000000001)
#define CAN_F9R2_FB1 ((uint32_t)0x0000000002)
#define CAN_F9R2_FB2 ((uint32_t)0x0000000004)
#define CAN_F9R2_FB3 ((uint32_t)0x0000000008)
#define CAN_F9R2_FB4 ((uint32_t)0x0000000010)
#define CAN_F9R2_FB5 ((uint32_t)0x0000000020)
#define CAN_F9R2_FB6 ((uint32_t)0x0000000040)
#define CAN_F9R2_FB7 ((uint32_t)0x0000000080)
#define CAN_F9R2_FB8 ((uint32_t)0x0000001000)
#define CAN_F9R2_FB9 ((uint32_t)0x0000002000)
#define CAN_F9R2_FB10 ((uint32_t)0x0000004000)
#define CAN_F9R2_FB11 ((uint32_t)0x0000008000)
#define CAN_F9R2_FB12 ((uint32_t)0x0000100000)
#define CAN_F9R2_FB13 ((uint32_t)0x0000200000)
#define CAN_F9R2_FB14 ((uint32_t)0x0000400000)
#define CAN_F9R2_FB15 ((uint32_t)0x0000800000)
#define CAN_F9R2_FB16 ((uint32_t)0x0001000000)
#define CAN_F9R2_FB17 ((uint32_t)0x0002000000)

```

```

#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)

/* Bit definition for CAN_F10R2 register */
#define CAN_F10R2_FBO ((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)

/* Bit definition for CAN_F11R2 register */
#define CAN_F11R2_FBO ((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_F1OR2_FB19 ((uint32_t)0x00080000)
#define CAN_F1OR2_FB20 ((uint32_t)0x00100000)
#define CAN_F1OR2_FB21 ((uint32_t)0x00200000)
#define CAN_F1OR2_FB22 ((uint32_t)0x00400000)
#define CAN_F1OR2_FB23 ((uint32_t)0x00800000)
#define CAN_F1OR2_FB24 ((uint32_t)0x01000000)
#define CAN_F1OR2_FB25 ((uint32_t)0x02000000)
#define CAN_F1OR2_FB26 ((uint32_t)0x04000000)
#define CAN_F1OR2_FB27 ((uint32_t)0x08000000)
#define CAN_F1OR2_FB28 ((uint32_t)0x10000000)
#define CAN_F1OR2_FB29 ((uint32_t)0x20000000)
#define CAN_F1OR2_FB30 ((uint32_t)0x40000000)
#define CAN_F1OR2_FB31 ((uint32_t)0x80000000)

/* ***** Bit definition for CAN_F11R2 register *****/
#define CAN_F11R2_FBO ((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)

/* !<Filter bit 19 */
/* !<Filter bit 20 */
/* !<Filter bit 21 */
/* !<Filter bit 22 */
/* !<Filter bit 23 */
/* !<Filter bit 24 */
/* !<Filter bit 25 */
/* !<Filter bit 26 */
/* !<Filter bit 27 */
/* !<Filter bit 28 */
/* !<Filter bit 29 */
/* !<Filter bit 30 */
/* !<Filter bit 31 */

/* !<Filter bit 0 */
/* !<Filter bit 1 */
/* !<Filter bit 2 */
/* !<Filter bit 3 */
/* !<Filter bit 4 */
/* !<Filter bit 5 */
/* !<Filter bit 6 */
/* !<Filter bit 7 */
/* !<Filter bit 8 */
/* !<Filter bit 9 */
/* !<Filter bit 10 */
/* !<Filter bit 11 */
/* !<Filter bit 12 */
/* !<Filter bit 13 */
/* !<Filter bit 14 */
/* !<Filter bit 15 */
/* !<Filter bit 16 */
/* !<Filter bit 17 */
/* !<Filter bit 18 */
/* !<Filter bit 19 */

```

```

#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)

/**************** Bit definition for CAN_F12R2 register *****/
#define CAN_F12R2_FBO ((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)

/* Bit definition for CAN_F13R2 register *****/
#define CAN_F13R2_FBO ((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)0x00100000)
#define CAN_F13R2_FB17 ((uint32_t)0x00200000)
#define CAN_F13R2_FB18 ((uint32_t)0x00400000)
#define CAN_F13R2_FB19 ((uint32_t)0x00800000)
#define CAN_F13R2_FB20 ((uint32_t)0x01000000)
#define CAN_F13R2_FB21 ((uint32_t)0x02000000)

/* !<Filter bit 21 */
/* !<Filter bit 22 */
/* !<Filter bit 23 */
/* !<Filter bit 24 */
/* !<Filter bit 25 */
/* !<Filter bit 26 */
/* !<Filter bit 27 */
/* !<Filter bit 28 */
/* !<Filter bit 29 */
/* !<Filter bit 30 */
/* !<Filter bit 31 */

/* !<Filter bit 0 */
/* !<Filter bit 1 */
/* !<Filter bit 2 */
/* !<Filter bit 3 */
/* !<Filter bit 4 */
/* !<Filter bit 5 */
/* !<Filter bit 6 */
/* !<Filter bit 7 */
/* !<Filter bit 8 */
/* !<Filter bit 9 */
/* !<Filter bit 10 */
/* !<Filter bit 11 */
/* !<Filter bit 12 */
/* !<Filter bit 13 */
/* !<Filter bit 14 */
/* !<Filter bit 15 */
/* !<Filter bit 16 */
/* !<Filter bit 17 */
/* !<Filter bit 18 */
/* !<Filter bit 19 */
/* !<Filter bit 20 */
/* !<Filter bit 21 */

```

```

#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)

/* !<Filter bit 22 */
/* !<Filter bit 23 */
/* !<Filter bit 24 */
/* !<Filter bit 25 */
/* !<Filter bit 26 */
/* !<Filter bit 27 */
/* !<Filter bit 28 */
/* !<Filter bit 29 */

/* !< Data register bits */

/* CRC calculation unit */

/* Bit definition for CRC_DR register */
#define CRC_DR (((uint32_t)0xFFFFFFF) /*!< Data register bits */

/* Bit definition for CRC_IDR register */
#define CRC_IDR (((uint8_t)0xFF) /*!< General-purpose 8-bit data register bits */

/* Bit definition for CRC_CR register */
#define CRC_CR (((uint8_t)0x01) /*!< RESET bit */

/* Bit definition for CRYP_CR register */
#define CRYP_CR (((uint8_t)0x00) /*!< RESET bit */

/* Bit definition for CRYP_CR register */
#define CRYP_CR_ALGODIR (((uint32_t)0x00000004))

```

---

```

#define CRYP_CR_ALGOMODE (((uint32_t)0x000000038)
#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)0x0000000C0)
#define CRYP_CR_DATATYPE_0 ((uint32_t)0x00000040)
#define CRYP_CR_DATATYPE_1 ((uint32_t)0x00000080)
#define CRYP_CR_KEYSIZE (((uint32_t)0x000000300)
#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_CRYPTEN ((uint32_t)0x00008000)

/****** Bits definition for CRYP_SR register *****/
#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)

/****** Bits definition for CRYP_IMSCR register *****/
#define CRYP_IMSCR_INIM (((uint32_t)0x00000001)
#define CRYP_IMSCR_OUTIM (((uint32_t)0x00000002)

/****** Bits definition for CRYP_RISR register *****/
#define CRYP_RISR_OUTRIS (((uint32_t)0x00000001))

```

---

```

#define CRYP_RISR_INRIS ((uint32_t)0x000000002)
/****** Bits definition for CRYP_MISR register *****/
#define CRYP_MISR_INMIS ((uint32_t)0x00000001)
#define CRYP_MISR_OUTMIS ((uint32_t)0x00000002)

/****** Bit definition for DAC_CR register *****/
#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
#define DAC_CR_WAVE1_0
#define DAC_CR_WAVE1_1
#define DAC_CR_WAVE1_2
#define DAC_CR_WAVE1_3

#define DAC_CR_MAMP1
#define DAC_CR_MAMP1_0
#define DAC_CR_MAMP1_1
#define DAC_CR_MAMP1_2
#define DAC_CR_MAMP1_3

#define DAC_CR_DMAEN1
#define DAC_CR_EN2
#define DAC_CR_BOFF2
#define DAC_CR_TEN2

/* Digital to Analog Converter */
/* **** Bit definition for DAC channel1 ****/
/* !<DAC channel1 enable */
/* !<DAC channel1 output buffer disable */
/* !<DAC channel1 Trigger enable */

/* !<TSEL1[2:0] (DAC channel1 Trigger select)
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */

/* !<WAVE1[1:0] (DAC channel1 noise/triang
/* !<Bit 0 */
/* !<Bit 1 */

/* !<MAMP1[3:0] (DAC channel1 Mask/amplitude)
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */
/* !<Bit 3 */

/* !<DAC channel1 DMA enable */
/* !<DAC channel2 enable */
/* !<DAC channel2 output buffer disable */
/* !<DAC channel2 Trigger enable */

```

```

Глава 21. Стандартная библиотека STM32

Trigger
Gsel
((uint32_t)0x00380000) /* !<TSEL2 [2:0] (DAC channel12 Trigger
#define DAC_CR_TSEL2 ((uint32_t)0x00080000) /* !<Bit 0 */
#define DAC_CR_TSEL2_0 ((uint32_t)0x00100000) /* !<Bit 1 */
#define DAC_CR_TSEL2_1 ((uint32_t)0x00200000) /* !<Bit 2 */

#define DAC_CR_WAVE2 ((uint32_t)0x00C00000) /* !<WAVE2 [1:0] (DAC channel12 noise/Amplitude
#define DAC_CR_WAVE2_0 ((uint32_t)0x00400000) /* !<Bit 0 */
#define DAC_CR_WAVE2_1 ((uint32_t)0x00800000) /* !<Bit 1 */

#define DAC_CR_MAMP2 ((uint32_t)0x00F00000) /* !<MAMP2 [3:0] (DAC channel12 Mask/Ambit)
#define DAC_CR_MAMP2_0 ((uint32_t)0x01000000) /* !<Bit 0 */
#define DAC_CR_MAMP2_1 ((uint32_t)0x02000000) /* !<Bit 1 */
#define DAC_CR_MAMP2_2 ((uint32_t)0x04000000) /* !<Bit 2 */
#define DAC_CR_MAMP2_3 ((uint32_t)0x08000000) /* !<Bit 3 */

#define DAC_CR_DMAEN2 ((uint32_t)0x10000000) /* !<DAC channel12 DMA enabled */

***** Bit definition for DAC_SWTRIGR register *****/
#define DAC_SWTRIGR_SWTRIG1 ((uint8_t)0x01) /* !<DAC channel1 software trigger */
#define DAC_SWTRIGR_SWTRIG2 ((uint8_t)0x02) /* !<DAC channel2 software trigger */

***** Bit definition for DAC_DHR12R1 register *****/
#define DAC_DHR12R1_DACC1DHR ((uint16_t)0x0FFF) /* !<DAC channel1 12-bit Right aligned data

***** Bit definition for DAC_DHR12L1 register *****/
#define DAC_DHR12L1_DACC1DHR ((uint16_t)0x0FF0) /* !<DAC channel1 12-bit Left aligned data

***** Bit definition for DAC_DHR8R1 register *****/
#define DAC_DHR8R1_DACC1DHR ((uint8_t)0xFF) /* !<DAC channel1 8-bit Right aligned data

***** Bit definition for DAC_DHR12R2 register *****/
#define DAC_DHR12R2_DACC2DHR ((uint16_t)0xFFFF) /* !<DAC channel2 12-bit Right aligned data

***** Bit definition for DAC_DHR12L2 register *****/
#define DAC_DHR12L2_DACC2DHR ((uint16_t)0xFFFF0) /* !<DAC channel2 12-bit Left aligned data

```

```

/*
***** Bit definition for DAC_DHR8R2 register *****/
#define DAC_DHR8R2_DACC2DHR ((uint8_t)0xFF) /* !<DAC channel2 8-bit Right aligned data

/*
***** Bit definition for DAC_DHR12RD register *****/
#define DAC_DHR12RD_DACC1DHR ((uint32_t)0x00000FFF) /* !<DAC channel1 12-bit Right aligned data
#define DAC_DHR12RD_DACC2DHR ((uint32_t)0x0FFF0000) /* !<DAC channel2 12-bit Right aligned data

/*
***** Bit definition for DAC_DHR12LD register *****/
#define DAC_DHR12LD_DACC1DHR ((uint32_t)0x00000FFF) /* !<DAC channel1 12-bit Left aligned data
#define DAC_DHR12LD_DACC2DHR ((uint32_t)0xFFFF0000) /* !<DAC channel2 12-bit Left aligned data

/*
***** Bit definition for DAC_DHR8RD register *****/
#define DAC_DHR8RD_DACC1DHR ((uint16_t)0x0FFF) /* !<DAC channel1 8-bit Right aligned data
#define DAC_DHR8RD_DACC2DHR ((uint16_t)0xFFFF00) /* !<DAC channel2 8-bit Right aligned data

/*
***** Bit definition for DAC_DHR12LD register *****/
#define DAC_DHR12LD_DACC1DHR ((uint16_t)0x0FFF) /* !<DAC channel1 8-bit Left aligned data
#define DAC_DHR12LD_DACC2DHR ((uint16_t)0xFFFF00) /* !<DAC channel2 12-bit Left aligned data

/*
***** Bit definition for DAC_DOR1 register *****/
#define DAC_DOR1_DACC1DOR ((uint16_t)0x0FFF) /* !<DAC channel1 data output */

/*
***** Bit definition for DAC_DOR2 register *****/
#define DAC_DOR2_DACC2DOR ((uint16_t)0x0FFF) /* !<DAC channel2 data output */

/*
***** Bit definition for DAC_SR register *****/
#define DAC_SR_DMAUDR1 ((uint32_t)0x000002000) /* !<DAC channel1 DMA underrun flag */
#define DAC_SR_DMAUDR2 ((uint32_t)0x20000000) /* !<DAC channel2 DMA underrun flag */

/*
***** Bit definition for DAC_SR register *****/
/* Debug MCU */
/* DCM1 */

```

```

/*
 **** Bits definition for DCMI_CR register ****
#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)

/*
 **** Bits definition for DCMI_SR register ****
#define DCMI_SR_HSYNC ((uint32_t)0x00000001)
#define DCMI_SR_VSYNC ((uint32_t)0x00000002)
#define DCMI_SR_FNE ((uint32_t)0x00000004)

/*
 **** Bits definition for DCMI_RISR register ****
#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)

/*
 **** Bits definition for DCMI_IER register ****
#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)0x000000010)

/* **** Bits definition for DCMI_MISR register **** */
#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)

/* **** Bits definition for DCMI_ICR register **** */
#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)

/* **** Bits definition for DMA_SxCR register **** */
/* DMA Controller */
/*
DMA_SxCR_CHSEL_0 ((uint32_t)0x0E000000)
DMA_SxCR_CHSEL_1 ((uint32_t)0x02000000)
DMA_SxCR_CHSEL_2 ((uint32_t)0x04000000)
DMA_SxCR_MBURST ((uint32_t)0x08000000)
DMA_SxCR_MBURST_0 ((uint32_t)0x00800000)
DMA_SxCR_MBURST_1 ((uint32_t)0x01000000)
DMA_SxCR_PBURST ((uint32_t)0x00600000)
DMA_SxCR_PBURST_0 ((uint32_t)0x00200000)
DMA_SxCR_PBURST_1 ((uint32_t)0x00400000)
DMA_SxCR_ACK ((uint32_t)0x00100000)
DMA_SxCR_CT ((uint32_t)0x00080000)
DMA_SxCR_DBM ((uint32_t)0x00040000)
*/

```

```

#define DMA_SxCR_PL ((uint32_t)0x0000300000)
#define DMA_SxCR_PL_0 ((uint32_t)0x000010000)
#define DMA_SxCR_PL_1 ((uint32_t)0x000020000)
#define DMA_SxCR_PINCOS ((uint32_t)0x000008000)
#define DMA_SxCR_MSIZE ((uint32_t)0x000006000)
#define DMA_SxCR_MSIZE_0 ((uint32_t)0x000002000)
#define DMA_SxCR_MSIZE_1 ((uint32_t)0x000004000)
#define DMA_SxCR_PSIZE ((uint32_t)0x000018000)
#define DMA_SxCR_PSIZE_0 ((uint32_t)0x000008000)
#define DMA_SxCR_PSIZE_1 ((uint32_t)0x000001000)
#define DMA_SxCR_MINC ((uint32_t)0x000004000)
#define DMA_SxCR_PINC ((uint32_t)0x000002000)
#define DMA_SxCR_CIRC ((uint32_t)0x000001000)
#define DMA_SxCR_DIR ((uint32_t)0x000000C00)
#define DMA_SxCR_DIR_0 ((uint32_t)0x000000400)
#define DMA_SxCR_DIR_1 ((uint32_t)0x000000800)
#define DMA_SxCR_PFCTRL ((uint32_t)0x000000020)
#define DMA_SxCR_TCIE ((uint32_t)0x000000010)
#define DMA_SxCR_HTIE ((uint32_t)0x000000008)
#define DMA_SxCR_TEIE ((uint32_t)0x000000004)
#define DMA_SxCR_DMEIE ((uint32_t)0x000000002)
#define DMA_SxCR_EN ((uint32_t)0x000000001)

/**************** Bits definition for DMA_SxNDTR register *****/
#define DMA_SxNDT ((uint32_t)0x00000FFF)
#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)0x000000400))
#define DMA_SxNDT_11 (((uint32_t)0x000000800))
#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))

/* ***** Bits definition for DMA_SxFCR register *****/
#define DMA_SxFCR_FEIE ((uint32_t)0x000000080)
#define DMA_SxFCR_FS ((uint32_t)0x000000038)
#define DMA_SxFCR_FS_0 (((uint32_t)0x00000008))
#define DMA_SxFCR_FS_1 (((uint32_t)0x000000010))
#define DMA_SxFCR_FS_2 (((uint32_t)0x000000020))
#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))

/* ***** Bits definition for DMA_LISR register *****/
#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_HTIIF2 (((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)0x00008000))
#define DMA_LISR_HTIIF1 (((uint32_t)0x00004000))
#define DMA_LISR_TEIF1 (((uint32_t)0x00002000))
#define DMA_LISR_DMEIF1 (((uint32_t)0x00001000))
#define DMA_LISR_FEIF1 (((uint32_t)0x00000400))
#define DMA_LISR_TCIFO (((uint32_t)0x0000020))

```

---

```

#define DMA_LISR_HTIFO ((uint32_t)0x000000010)
#define DMA_LISR_TEIFO ((uint32_t)0x00000008)
#define DMA_LISR_DMEIFO ((uint32_t)0x00000004)
#define DMA_LISR_FEIFO ((uint32_t)0x00000001)

/**************** Bits definition for DMA_HISR register *****/
#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)0x00008000)
#define DMA_HISR_HTIF5 ((uint32_t)0x00004000)
#define DMA_HISR_TEIF5 ((uint32_t)0x00002000)
#define DMA_HISR_DMEIF5 ((uint32_t)0x00001000)
#define DMA_HISR_FEIF5 ((uint32_t)0x00000400)
#define DMA_HISR_TCIF4 ((uint32_t)0x00000200)
#define DMA_HISR_HTIF4 ((uint32_t)0x00000100)
#define DMA_HISR_TEIF4 ((uint32_t)0x00000080)
#define DMA_HISR_DMEIF4 ((uint32_t)0x00000004)
#define DMA_HISR_FEIF4 ((uint32_t)0x00000001)

/**************** Bits definition for DMA_LIFCR register *****/
#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)0x0000800000)
#define DMA_LIFCR_CDMEIF2 ((uint32_t)0x0000400000)
#define DMA_LIFCR_CFEIF2 ((uint32_t)0x0001000000)
#define DMA_LIFCR_CTCIIF1 ((uint32_t)0x0000080000)
#define DMA_LIFCR_CHTIIF1 ((uint32_t)0x0000040000)
#define DMA_LIFCR_CTEIF1 ((uint32_t)0x0000020000)
#define DMA_LIFCR_CDMEIF1 ((uint32_t)0x0000010000)
#define DMA_LIFCR_CFEIF1 ((uint32_t)0x0000004000)
#define DMA_LIFCR_CTCIIFO ((uint32_t)0x0000002000)
#define DMA_LIFCR_CHTIIFO ((uint32_t)0x0000001000)
#define DMA_LIFCR_CTEIF0 ((uint32_t)0x0000000800)
#define DMA_LIFCR_CDMEIF0 ((uint32_t)0x0000000400)
#define DMA_LIFCR_CFEIF0 ((uint32_t)0x0000000100)

/* ***** Bits definition for DMA_HIFCR register *****/
#define DMA_HIFCR_CTCIIF7 ((uint32_t)0x0800000000)
#define DMA_HIFCR_CHTIIF7 ((uint32_t)0x0400000000)
#define DMA_HIFCR_CTEIF7 ((uint32_t)0x0200000000)
#define DMA_HIFCR_CDMEIF7 ((uint32_t)0x0100000000)
#define DMA_HIFCR_CFEIF7 ((uint32_t)0x0040000000)
#define DMA_HIFCR_CTCIIF6 ((uint32_t)0x0020000000)
#define DMA_HIFCR_CHTIIF6 ((uint32_t)0x0010000000)
#define DMA_HIFCR_CTEIF6 ((uint32_t)0x0008000000)
#define DMA_HIFCR_CDMEIF6 ((uint32_t)0x0004000000)
#define DMA_HIFCR_CFEIF6 ((uint32_t)0x0001000000)
#define DMA_HIFCR_CTCIIF5 ((uint32_t)0x0000800000)
#define DMA_HIFCR_CHTIIF5 ((uint32_t)0x0000400000)
#define DMA_HIFCR_CTEIF5 ((uint32_t)0x0000200000)
#define DMA_HIFCR_CDMEIF5 ((uint32_t)0x0000100000)
#define DMA_HIFCR_CFEIF5 ((uint32_t)0x0000040000)
#define DMA_HIFCR_CTCIIF4 ((uint32_t)0x0000020000)
#define DMA_HIFCR_CHTIIF4 ((uint32_t)0x0000010000)
#define DMA_HIFCR_CTEIF4 ((uint32_t)0x0000004000)
#define DMA_HIFCR_CDMEIF4 ((uint32_t)0x0000000800)
#define DMA_HIFCR_CFEIF4 ((uint32_t)0x0000000400)

```

---

```

/*
 */
/* External Interrupt/Event Controller */
/*
***** Bit definition for EXTI_IMR register *****/
#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)

/*
***** Bit definition for EXTI_EMR register *****/
#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)

/*
***** Bit definition for EXTI_EMR register *****/
/* !< Event Mask on line 0 */
/* !< Event Mask on line 1 */
/* !< Event Mask on line 2 */
/* !< Event Mask on line 3 */
/* !< Event Mask on line 4 */
/* !< Event Mask on line 5 */

```

```

#define EXTI_EMR_MR6 ((uint32_t)0x000000040)
#define EXTI_EMR_MR7 ((uint32_t)0x000000080)
#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)

/* Bit definition for EXTI_RTSR register *****/
#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)

/* Bit definition for EXTI_RTSR register *****/
#define EXTI_RTSR_MR6 ((uint32_t)0x000000040)
#define EXTI_RTSR_MR7 ((uint32_t)0x000000080)
#define EXTI_RTSR_MR8 ((uint32_t)0x00000100)
#define EXTI_RTSR_MR9 ((uint32_t)0x00000200)
#define EXTI_RTSR_MR10 ((uint32_t)0x00000400)
#define EXTI_RTSR_MR11 ((uint32_t)0x00000800)
#define EXTI_RTSR_MR12 ((uint32_t)0x00001000)
#define EXTI_RTSR_MR13 ((uint32_t)0x00002000)
#define EXTI_RTSR_MR14 ((uint32_t)0x00004000)
#define EXTI_RTSR_MR15 ((uint32_t)0x00008000)
#define EXTI_RTSR_MR16 ((uint32_t)0x00010000)
#define EXTI_RTSR_MR17 ((uint32_t)0x00020000)
#define EXTI_RTSR_MR18 ((uint32_t)0x00040000)

```

#define EXTI\_RTSR\_TR19

((uint32\_t)0x00080000)

/\* !< Rising trigger event configuration b

```
***** Bit definition for EXTI_FTSR register *****/
#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)

/* !< Falling trigger event configuration
```

```
***** Bit definition for EXTI_SWIER register *****/
#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)

/* !< Software Interrupt on line 0 */
/* !< Software Interrupt on line 1 */
/* !< Software Interrupt on line 2 */
/* !< Software Interrupt on line 3 */
/* !< Software Interrupt on line 4 */
/* !< Software Interrupt on line 5 */
/* !< Software Interrupt on line 6 */
/* !< Software Interrupt on line 7 */
/* !< Software Interrupt on line 8 */
/* !< Software Interrupt on line 9 */
```

Глava 21

173

```

/* !< Software Interrupt on line 10
((uint32_t)0x000000400)
((uint32_t)0x000000800)
((uint32_t)0x00001000)
((uint32_t)0x00002000)
((uint32_t)0x00004000)
((uint32_t)0x00008000)
((uint32_t)0x00010000)
((uint32_t)0x00020000)
((uint32_t)0x00040000)
((uint32_t)0x00080000)
(* !< Software Interrupt on line 11
(* !< Software Interrupt on line 12
(* !< Software Interrupt on line 13
(* !< Software Interrupt on line 14
(* !< Software Interrupt on line 15
(* !< Software Interrupt on line 16
(* !< Software Interrupt on line 17
(* !< Software Interrupt on line 18
(* !< Software Interrupt on line 19

/* !< Pending bit for line 0 */
(* !< Pending bit for line 1 */
(* !< Pending bit for line 2 */
(* !< Pending bit for line 3 */
(* !< Pending bit for line 4 */
(* !< Pending bit for line 5 */
(* !< Pending bit for line 6 */
(* !< Pending bit for line 7 */
(* !< Pending bit for line 8 */
(* !< Pending bit for line 9 */
(* !< Pending bit for line 10 */
(* !< Pending bit for line 11 */
(* !< Pending bit for line 12 */
(* !< Pending bit for line 13 */
(* !< Pending bit for line 14 */
(* !< Pending bit for line 15 */
(* !< Pending bit for line 16 */
(* !< Pending bit for line 17 */
(* !< Pending bit for line 18 */
(* !< Pending bit for line 19 */

/* **** Bit definition for EXTI_PR register *****/
Bit definition for EXTI_PR register *****
(* !< Software Interrupt on line 0 */
(* !< Software Interrupt on line 1 */
(* !< Software Interrupt on line 2 */
(* !< Software Interrupt on line 3 */
(* !< Software Interrupt on line 4 */
(* !< Software Interrupt on line 5 */
(* !< Software Interrupt on line 6 */
(* !< Software Interrupt on line 7 */
(* !< Software Interrupt on line 8 */
(* !< Software Interrupt on line 9 */
(* !< Software Interrupt on line 10 */
(* !< Software Interrupt on line 11 */
(* !< Software Interrupt on line 12 */
(* !< Software Interrupt on line 13 */
(* !< Software Interrupt on line 14 */
(* !< Software Interrupt on line 15 */
(* !< Software Interrupt on line 16 */
(* !< Software Interrupt on line 17 */
(* !< Software Interrupt on line 18 */
(* !< Software Interrupt on line 19 */

/* **** */
/*

```

```

/*
 * **** Bits definition for FLASH_ACR register ****
 * ****
 * #define FLASH_ACR_LATENCY ((uint32_t)0x00000007)
 * #define FLASH_ACR_OWS ((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_PRFSEN ((uint32_t)0x000000100)
#define FLASH_ACR_ICEN ((uint32_t)0x000000200)
#define FLASH_ACR_DCEN ((uint32_t)0x000000400)
#define FLASH_ACR_ICRST ((uint32_t)0x000000800)
#define FLASH_ACR_DCRST ((uint32_t)0x000001000)
#define FLASH_ACR_BYTE0_ADDRESS ((uint32_t)0x40023C00)
#define FLASH_ACR_BYTE2_ADDRESS ((uint32_t)0x40023C03)

**** Bits definition for FLASH_SR register ****
#define FLASH_SR_EOP ((uint32_t)0x000000001)
#define FLASH_SR_SOP ((uint32_t)0x000000002)
#define FLASH_SR_WRPERR ((uint32_t)0x000000010)
#define FLASH_SR_PGAERR ((uint32_t)0x000000020)
#define FLASH_SR_PGPERR ((uint32_t)0x000000040)
#define FLASH_SR_PGSERR ((uint32_t)0x000000080)
#define FLASH_SR_BSY ((uint32_t)0x000010000)

**** Bits definition for FLASH_CR register ****
#define FLASH_CR_PG ((uint32_t)0x000000001)
#define FLASH_CR_SER ((uint32_t)0x000000002)
#define FLASH_CR_MER ((uint32_t)0x000000004)

```

---

```

#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)

/**************** Bits definition for FLASH_OPTCR register *****/
#define FLASH_OPTCR_OPTLOCK ((uint32_t)0x00000001)
#define FLASH_OPTCR_OPTSTRT ((uint32_t)0x00000002)
#define FLASH_OPTCR_BORLEV_0 ((uint32_t)0x00000004)
#define FLASH_OPTCR_BORLEV_1 ((uint32_t)0x00000008)
#define FLASH_OPTCR_BORLEV ((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)0x00010000)
#define FLASH_OPTCR_nWRP_0 ((uint32_t)0x00020000)
#define FLASH_OPTCR_nWRP_1 ((uint32_t)0x00040000)
#define FLASH_OPTCR_nWRP_2 ((uint32_t)0x00080000)
#define FLASH_OPTCR_nWRP_3 ((uint32_t)0x00100000)
#define FLASH_OPTCR_nWRP_4 ((uint32_t)0x00200000)
#define FLASH_OPTCR_nWRP_5 ((uint32_t)0x00400000)
#define FLASH_OPTCR_nWRP_6 ((uint32_t)0x00800000)
#define FLASH_OPTCR_nWRP_7 ((uint32_t)0x00100000)

```

---

```

#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)

/********************* Bit definition for FSMC_BCR1 register ********************/
/* Address/data multiplexing enable bit */

#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_FACCE           ((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)0x00008000)

/********************* Bit definition for FSMC_BCR2 register ********************/
#define FSMC_BCR2_MBKEN           ((uint32_t)0x00000001)

```

```

/* !<Address/data multiplexing enable bit */

#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_FACCE ((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)0x0000800000)

/* ***** Bit definition for FSMC_BCR3 register *****/
#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_FACCE ((uint32_t)0x00000040)
#define FSMC_BCR3_BURSTEN ((uint32_t)0x00000100)

/* !<MTYP [1:0] bits (Memory type) */
/* !<Bit 0 */
/* !<Bit 1 */

/* !<MWID [1:0] bits (Memory type) */
/* !<Bit 0 */
/* !<Bit 1 */

/* !<Flash access enable */
/* !<Burst enable bit */
/* !<Wait signal polarity bit */
/* !<Wrapped burst mode support */
/* !<Wait timing configuration */
/* !<Write enable bit */
/* !<Wait enable bit */
/* !<Extended mode enable */
/* !<Asynchronous wait */
/* !<Write burst enable */

/* !<Memory bank enable bit */
/* !<Address/data multiplexing enable bit */

/* !<MTYP [1:0] bits (Memory type) */
/* !<Bit 0 */
/* !<Bit 1 */

/* !<MWID [1:0] bits (Memory type) */
/* !<Bit 0 */
/* !<Bit 1 */

/* !<Flash access enable */
/* !<Burst enable bit */

```

```

#define FSMC_BCR3_WAITPOL ((uint32_t)0x000000200)
#define FSMC_BCR3_WRAPMOD ((uint32_t)0x000000400)
#define FSMC_BCR3_WAITCFG ((uint32_t)0x000000800)
#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)0x000080000)

/* !<Wait signal polarity bit. */
/* !<Wrapped burst mode support */
/* !<Wait timing configuration */
/* !<Write enable bit */
/* !<Write enable bit */
/* !<Wait enable bit */
/* !<Extended mode enable */
/* !<Asynchronous wait */
/* !<Write burst enable */

/* !<Memory bank enable bit */
/* !<Address/data multiplexing enablebit */

/* !<MTYP[1:0] bits (Memory type) */
/* !<Bit 0 */
/* !<Bit 1 */

/* !<MWID[1:0] bits (Memory data bus width)
/* !<Bit 0 */
/* !<Bit 1 */

/* !<Flash access enable */
/* !<Burst enable bit */
/* !<Wait signal polarity bit */
/* !<Wrapped burst mode support */
/* !<Wait timing configuration */
/* !<Write enable bit */
/* !<Wait enable bit */
/* !<Extended mode enable */
/* !<Asynchronous wait */
/* !<Write burst enable */

/* !<BTR1 register */
((uint32_t)0x0000000F)
((uint32_t)0x00000001)

/* !<ADDSET[3:0] bits (Address setup phase */
/* !<Bit 0 */

```

```

#define FSMC_BTR1_ADDSET_1          /* !<Bit 1 */
#define FSMC_BTR1_ADDSET_2          /* !<Bit 2 */
#define FSMC_BTR1_ADDSET_3          /* !<Bit 3 */

#define FSMC_BTR1_ADDHLD           /* !<ADDHLD [3:0] bits (Address-hold phase)
#define FSMC_BTR1_ADDHLD_0          /* !<Bit 0 */
#define FSMC_BTR1_ADDHLD_1          /* !<Bit 1 */
#define FSMC_BTR1_ADDHLD_2          /* !<Bit 2 */
#define FSMC_BTR1_ADDHLD_3          /* !<Bit 3 */

#define FSMC_BTR1_DATAST           /* !<DATAST [3:0] bits (Data-phase duration)
#define FSMC_BTR1_DATAST_0          /* !<Bit 0 */
#define FSMC_BTR1_DATAST_1          /* !<Bit 1 */
#define FSMC_BTR1_DATAST_2          /* !<Bit 2 */
#define FSMC_BTR1_DATAST_3          /* !<Bit 3 */

#define FSMC_BTR1_BUSTURN          /* !<BUSTURN [3:0] bits (Bus turnaround time)
#define FSMC_BTR1_BUSTURN_0          /* !<Bit 0 */
#define FSMC_BTR1_BUSTURN_1          /* !<Bit 1 */
#define FSMC_BTR1_BUSTURN_2          /* !<Bit 2 */
#define FSMC_BTR1_BUSTURN_3          /* !<Bit 3 */

#define FSMC_BTR1_CLKDIV            /* !<CLKDIV [3:0] bits (Clock divide ratio)
#define FSMC_BTR1_CLKDIV_0          /* !<Bit 0 */
#define FSMC_BTR1_CLKDIV_1          /* !<Bit 1 */
#define FSMC_BTR1_CLKDIV_2          /* !<Bit 2 */
#define FSMC_BTR1_CLKDIV_3          /* !<Bit 3 */

#define FSMC_BTR1_DATLAT            /* !<DATLAT [3:0] bits (Data latency)
#define FSMC_BTR1_DATLAT_0          /* !<Bit 0 */
#define FSMC_BTR1_DATLAT_1          /* !<Bit 1 */
#define FSMC_BTR1_DATLAT_2          /* !<Bit 2 */
#define FSMC_BTR1_DATLAT_3          /* !<Bit 3 */

#define FSMC_BTR1_ACCMOD           /* !<ACCMOD [1:0] bits (Access mode)

```

```

#define FSMC_BTR1_ACCMOD_0 ((uint32_t)0x10000000) /* !<Bit 0 */
#define FSMC_BTR1_ACCMOD_1 ((uint32_t)0x20000000) /* !<Bit 1 */

/* ***** Bit definition for FSMC_BTR2 register *****/
#define FSMC_BTR2_ADDSET ((uint32_t)0x0000000F) /* !<ADDSET[3:0] bits (Address setup phase)
#define FSMC_BTR2_ADDSET_0 ((uint32_t)0x00000001) /* !<Bit 0 */
#define FSMC_BTR2_ADDSET_1 ((uint32_t)0x00000002) /* !<Bit 1 */
#define FSMC_BTR2_ADDSET_2 ((uint32_t)0x00000004) /* !<Bit 2 */
#define FSMC_BTR2_ADDSET_3 ((uint32_t)0x00000008) /* !<Bit 3 */

#define FSMC_BTR2_ADDHLD ((uint32_t)0x000000F0) /* !<ADDHLD[3:0] bits (Address-hold phase)
#define FSMC_BTR2_ADDHLD_0 ((uint32_t)0x00000010) /* !<Bit 0 */
#define FSMC_BTR2_ADDHLD_1 ((uint32_t)0x00000020) /* !<Bit 1 */
#define FSMC_BTR2_ADDHLD_2 ((uint32_t)0x00000040) /* !<Bit 2 */
#define FSMC_BTR2_ADDHLD_3 ((uint32_t)0x00000080) /* !<Bit 3 */

#define FSMC_BTR2_DATAST ((uint32_t)0x00000FF00) /* !<DATAST[3:0] bits (Data - phase duration)
#define FSMC_BTR2_DATAST_0 ((uint32_t)0x00000100) /* !<Bit 0 */
#define FSMC_BTR2_DATAST_1 ((uint32_t)0x00000200) /* !<Bit 1 */
#define FSMC_BTR2_DATAST_2 ((uint32_t)0x00000400) /* !<Bit 2 */
#define FSMC_BTR2_DATAST_3 ((uint32_t)0x00000800) /* !<Bit 3 */

#define FSMC_BTR2_BUSTURN ((uint32_t)0x000F0000) /* !<BUSTURN[3:0] bits (Bus turnaround phase)
#define FSMC_BTR2_BUSTURN_0 ((uint32_t)0x00010000) /* !<Bit 0 */
#define FSMC_BTR2_BUSTURN_1 ((uint32_t)0x00020000) /* !<Bit 1 */
#define FSMC_BTR2_BUSTURN_2 ((uint32_t)0x00040000) /* !<Bit 2 */
#define FSMC_BTR2_BUSTURN_3 ((uint32_t)0x00080000) /* !<Bit 3 */

#define FSMC_BTR2_CLKDIV ((uint32_t)0x00F00000) /* !<CLKDIV[3:0] bits (Clock divide ratio)
#define FSMC_BTR2_CLKDIV_0 ((uint32_t)0x00100000) /* !<Bit 0 */
#define FSMC_BTR2_CLKDIV_1 ((uint32_t)0x00200000) /* !<Bit 1 */
#define FSMC_BTR2_CLKDIV_2 ((uint32_t)0x00400000) /* !<Bit 2 */
#define FSMC_BTR2_CLKDIV_3 ((uint32_t)0x00800000) /* !<Bit 3 */

#define FSMC_BTR2_DATLAT ((uint32_t)0x0F000000) /* !<DATLA[3:0] bits (Data latency) */

```

```

#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)

/****** Bit definition for FSMC_BTR3 register *****/
#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)0x00001000)
#define FSMC_BTR3_DATAST_1 ((uint32_t)0x00002000)
#define FSMC_BTR3_DATAST_2 ((uint32_t)0x00004000)
#define FSMC_BTR3_DATAST_3 ((uint32_t)0x00008000)

#define FSMC_BTR3_BUSTURN ((uint32_t)0x0000F000)
#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)

```

```

/* !<Bit 0 */ ((uint32_t)0x00100000)
/* !<Bit 1 */ ((uint32_t)0x00200000)
/* !<Bit 2 */ ((uint32_t)0x00400000)
/* !<Bit 3 */ ((uint32_t)0x00800000)

/* !<DATLA [3:0] bits (Data latency) */
/* !<Bit 0 */ ((uint32_t)0x01000000)
/* !<Bit 1 */ ((uint32_t)0x02000000)
/* !<Bit 2 */ ((uint32_t)0x04000000)
/* !<Bit 3 */ ((uint32_t)0x08000000)

/* !<ACCMOD [1:0] bits (Access mode) */
/* !<Bit 0 */ ((uint32_t)0x30000000)
/* !<Bit 1 */ ((uint32_t)0x10000000)
((uint32_t)0x20000000)

/* **** Bit definition for FSMC_BTR4 register *****/
/* !<ADDSET [3:0] bits (Address setup phase */
/* !<Bit 0 */ ((uint32_t)0x0000000F)
/* !<Bit 1 */ ((uint32_t)0x00000001)
/* !<Bit 2 */ ((uint32_t)0x00000002)
/* !<Bit 3 */ ((uint32_t)0x00000004)
((uint32_t)0x00000008)

/* !<ADDHLD [3:0] bits (Address - hold phase */
/* !<Bit 0 */ ((uint32_t)0x00000010)
/* !<Bit 1 */ ((uint32_t)0x00000020)
/* !<Bit 2 */ ((uint32_t)0x00000040)
/* !<Bit 3 */ ((uint32_t)0x00000080)

/* !<DATAST [3:0] bits (Data - phasse duration */
/* !<Bit 0 */ ((uint32_t)0x00000FF0)
((uint32_t)0x00000100)
((uint32_t)0x00000200)
((uint32_t)0x00000400)
((uint32_t)0x00000800)

/* !<BUSTURN [3:0] bits (Bus turnaround phase */
((uint32_t)0x000F0000)

```

```

#define FSMC_BTR4_BUSTURN_0 ((uint32_t)0x000010000)
#define FSMC_BTR4_BUSTURN_1 ((uint32_t)0x000020000)
#define FSMC_BTR4_BUSTURN_2 ((uint32_t)0x000040000)
#define FSMC_BTR4_BUSTURN_3 ((uint32_t)0x000080000)

#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))

/****** Bit definition for FSMC_BWTR1 register *****/
#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_4 ((uint32_t)0x00000004)

```

## Глава 21. Стандартная библиотека STM32

---

```

/* !<Bit 0 */          ((uint32_t)0x000000100)
((uint32_t)0x000000200)
((uint32_t)0x00000400)
((uint32_t)0x00000800)

/* !<Bit 1 */          ((uint32_t)0x0000F0000)
((uint32_t)0x000100000)
((uint32_t)0x000200000)
((uint32_t)0x000400000)
((uint32_t)0x000800000)

/* !<Bit 2 */          ((uint32_t)0x00F000000)
((uint32_t)0x010000000)
((uint32_t)0x020000000)
((uint32_t)0x040000000)
((uint32_t)0x080000000)

/* !<Bit 3 */          ((uint32_t)0x000F00000)
((uint32_t)0x001000000)
((uint32_t)0x002000000)
((uint32_t)0x004000000)
((uint32_t)0x008000000)

/* !<Bit 0 */          ((uint32_t)0x300000000)
((uint32_t)0x100000000)
((uint32_t)0x040000000)
((uint32_t)0x080000000)

/* !<Bit 1 */          ((uint32_t)0x300000000)
((uint32_t)0x100000000)
((uint32_t)0x040000000)
((uint32_t)0x080000000)

/* !<Bit 2 */          ((uint32_t)0x000300000)
((uint32_t)0x000100000)
((uint32_t)0x000040000)
((uint32_t)0x000080000)

/* !<Bit 3 */          ((uint32_t)0x0000000F0)
((uint32_t)0x000000010)
((uint32_t)0x000000020)
((uint32_t)0x000000040)
((uint32_t)0x000000080)

/* !<DATAST [3:0] bits (Data - phase duration) */

```

---

---

```

/* !<Bit 0 */ ((uint32_t)0x000000100)
/* !<Bit 1 */ ((uint32_t)0x000000200)
/* !<Bit 2 */ ((uint32_t)0x000000400)
/* !<Bit 3 */ ((uint32_t)0x000000800)

/* !<CLKDIV [3:0] bits (Clock divide ratio) */
/* !<Bit 0 */ ((uint32_t)0x00F00000)
/* !<Bit 1 */ ((uint32_t)0x00100000)
/* !<Bit 2 */ ((uint32_t)0x00200000)
/* !<Bit 3 */ ((uint32_t)0x00400000)
/* !<Bit 0 */ ((uint32_t)0x00800000)

/* !<DATLA [3:0] bits (Data latency) */
/* !<Bit 0 */ ((uint32_t)0x00F00000)
/* !<Bit 1 */ ((uint32_t)0x01000000)
/* !<Bit 2 */ ((uint32_t)0x02000000)
/* !<Bit 3 */ ((uint32_t)0x04000000)

/* !<ACCMOD [1:0] bits (Access mode) */
/* !<Bit 0 */ ((uint32_t)0x30000000)
/* !<Bit 1 */ ((uint32_t)0x10000000)
/* !<Bit 0 */ ((uint32_t)0x08000000)

/* !<ADDSET [3:0] bits (Address setup phase)
   Bit definition for FSMC_BWTR3 register *****/
/* !<Bit 0 */ ((uint32_t)0x000000F)
/* !<Bit 1 */ ((uint32_t)0x0000001)
/* !<Bit 2 */ ((uint32_t)0x0000002)
/* !<Bit 3 */ ((uint32_t)0x0000004)
/* !<Bit 0 */ ((uint32_t)0x0000008)

/* !<ADDHLD [3:0] bits (Address-hold phase)
   Bit definition for FSMC_BWTR3 register *****/
/* !<Bit 0 */ ((uint32_t)0x000000F0)
/* !<Bit 1 */ ((uint32_t)0x00000010)
/* !<Bit 2 */ ((uint32_t)0x00000020)
/* !<Bit 3 */ ((uint32_t)0x00000040)
/* !<Bit 0 */ ((uint32_t)0x00000080)

/* !<DATAST [3:0] bits (Data-phase duration)
   Bit definition for FSMC_BWTR3 register *****/
/* !<Bit 0 */ ((uint32_t)0x0000FF00)

```

---

```

/* !<Bit 0 */ ((uint32_t)0x000000100)
/* !<Bit 1 */ ((uint32_t)0x000000200)
/* !<Bit 2 */ ((uint32_t)0x000000400)
/* !<Bit 3 */ ((uint32_t)0x000000800)

/* !<CLKDIV [3:0] bits (Clock divide ratio) */
/* !<Bit 0 */ ((uint32_t)0x00F00000)
/* !<Bit 1 */ ((uint32_t)0x00100000)
/* !<Bit 2 */ ((uint32_t)0x00200000)
/* !<Bit 3 */ ((uint32_t)0x00400000)
/* !<Bit 0 */ ((uint32_t)0x00800000)

/* !<DATLA [3:0] bits (Data latency) */
/* !<Bit 0 */ ((uint32_t)0x00F00000)
/* !<Bit 1 */ ((uint32_t)0x01000000)
/* !<Bit 2 */ ((uint32_t)0x02000000)
/* !<Bit 3 */ ((uint32_t)0x04000000)

/* !<ACCMOD [1:0] bits (Access mode) */
/* !<Bit 0 */ ((uint32_t)0x30000000)
/* !<Bit 1 */ ((uint32_t)0x10000000)
/* !<Bit 0 */ ((uint32_t)0x08000000)

/* !<ACCMD [1:0] bits (Address setup phase) */
/* !<Bit 0 */ ((uint32_t)0x000000F)
/* !<Bit 1 */ ((uint32_t)0x0000001)
/* !<Bit 2 */ ((uint32_t)0x0000002)
/* !<Bit 3 */ ((uint32_t)0x0000004)
/* !<Bit 0 */ ((uint32_t)0x0000008)

/* !<ADDHLD [3:0] bits (Address-hold phase) */
/* !<Bit 0 */ ((uint32_t)0x00000010)
/* !<Bit 1 */ ((uint32_t)0x00000020)
/* !<Bit 2 */ ((uint32_t)0x00000040)
/* !<Bit 3 */ ((uint32_t)0x00000080)

/* !<DATAST [3:0] bits (Data-phase duration) */
/* !<Bit 0 */ ((uint32_t)0x0000FF00)

```

---

```

/* !<Bit 0 */           ((uint32_t)0x000000100)
((uint32_t)0x000000200)
((uint32_t)0x00000400)
((uint32_t)0x00000800)

/* !<Bit 1 */           ((uint32_t)0x0000F00000)
((uint32_t)0x000100000)
((uint32_t)0x000200000)
((uint32_t)0x000400000)
((uint32_t)0x000800000)

/* !<Bit 2 */           ((uint32_t)0x00F000000)
((uint32_t)0x010000000)
((uint32_t)0x020000000)
((uint32_t)0x040000000)
((uint32_t)0x080000000)

/* !<Bit 3 */           ((uint32_t)0x0F0000000)
((uint32_t)0x010000000)
((uint32_t)0x020000000)
((uint32_t)0x040000000)
((uint32_t)0x080000000)

/* !<Bit 0 */           ((uint32_t)0x300000000)
((uint32_t)0x100000000)
((uint32_t)0x040000000)
((uint32_t)0x080000000)

/* !<Bit 1 */           ((uint32_t)0x30000000)
((uint32_t)0x10000000)
((uint32_t)0x04000000)
((uint32_t)0x08000000)

/* !<Bit 2 */           ((uint32_t)0x30000000)
((uint32_t)0x10000000)
((uint32_t)0x04000000)
((uint32_t)0x08000000)

/* !<Bit 3 */           ((uint32_t)0x30000000)
((uint32_t)0x10000000)
((uint32_t)0x04000000)
((uint32_t)0x08000000)

/* !<Bit 0 */           ((uint32_t)0x00000030)
((uint32_t)0x00000010)
((uint32_t)0x00000020)
((uint32_t)0x00000040)

/* !<Bit 1 */           ((uint32_t)0x00000040)
((uint32_t)0x00000080)
((uint32_t)0x00000100)
((uint32_t)0x00000200)
((uint32_t)0x00000400)

/* !<Bit 2 */           ((uint32_t)0x00000030)
((uint32_t)0x00000010)
((uint32_t)0x00000020)
((uint32_t)0x00000040)

/* !<Bit 3 */           ((uint32_t)0x00000040)
((uint32_t)0x00000080)
((uint32_t)0x00000100)
((uint32_t)0x00000200)
((uint32_t)0x00000400)

/* !<Bit 0 */           ((uint32_t)0x00000030)
((uint32_t)0x00000010)
((uint32_t)0x00000020)
((uint32_t)0x00000040)

/* !<Bit 1 */           ((uint32_t)0x00000040)
((uint32_t)0x00000080)
((uint32_t)0x00000100)
((uint32_t)0x00000200)
((uint32_t)0x00000400)

```

---

---

```

#define FSMC_PCR2_TCLR_2 ((uint32_t)0x000000800)
#define FSMC_PCR2_TCLR_3 ((uint32_t)0x000001000)

#define FSMC_PCR2_TAR ((uint32_t)0x00001E000)
#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)

/**************** Bit definition for FSMC_PCR3 register *****/
#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)0x00001E000)
#define FSMC_PCR3_TCLR_0 ((uint32_t)0x00002000)
#define FSMC_PCR3_TCLR_1 ((uint32_t)0x00004000)
#define FSMC_PCR3_TCLR_2 ((uint32_t)0x00008000)
#define FSMC_PCR3_TCLR_3 ((uint32_t)0x00010000)

#define FSMC_PCR3_TAR ((uint32_t)0x00001E000)
#define FSMC_PCR3_TAR_0 ((uint32_t)0x00002000)
#define FSMC_PCR3_TAR_1 ((uint32_t)0x00004000)
#define FSMC_PCR3_TAR_2 ((uint32_t)0x00008000)

/* !<Bit 2 */ /* !<Bit 3 */
/* !<TAR[3:0] bits (ALE to RE delay) */
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */
/* !<Bit 3 */

/* !<ECCPS[1:0] bits (ECC page size) */
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */

/* !<Wait feature enable bit */
/* !<PC Card/NAND Flash memory bank enable */
/* !<Memory type */

/* !<PWID[1:0] bits (NAND Flash databus width) */
/* !<Bit 0 */
/* !<Bit 1 */

/* !<ECC computation logic enable bit */

/* !<TCLR[3:0] bits (CLE to RE delay) */
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */
/* !<Bit 3 */

```

---

```

#define FSMC_PCR3_TAR_3 ((uint32_t)0x00010000)
/* !<Bit 3 */

#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)

/**************** Bit definition for FSMC_PCR4 register *****/
#define FSMC_PCR4_PWAITEM ((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)0x00002000)
#define FSMC_PCR4_TCLR_1 ((uint32_t)0x00004000)
#define FSMC_PCR4_TCLR_2 ((uint32_t)0x00008000)
#define FSMC_PCR4_TCLR_3 ((uint32_t)0x00010000)

#define FSMC_PCR4_TAR ((uint32_t)0x000E0000)
#define FSMC_PCR4_TAR_0 ((uint32_t)0x00020000)
#define FSMC_PCR4_TAR_1 ((uint32_t)0x00040000)
#define FSMC_PCR4_TAR_2 ((uint32_t)0x00080000)
#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)

/* !<ECCPS [2:0] bits (ECC page size) */
/* !<Bit 0 */
/* !<PC Card/NAND Flash memory bank enable */
/* !<Memory type */

/* !<PWID [1:0] bits (NAND Flash database width) */
/* !<Bit 0 */
/* !<Bit 1 */

/* !<ECC computation logic enable bit */
/* !<TCLR [3:0] bits (CLE to RE delay) */
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */
/* !<Bit 3 */

/* !<TAR [3:0] bits (ALE to RE delay) */
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */
/* !<Bit 3 */

/* !<ECCPS [2:0] bits (ECC page size) */
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */

```

```

/*
 * Bit definition for FSMC_SR2 register *****
 *define FSMC_SR2_IRS ((uint8_t)0x01) /* !< Interrupt Rising Edge status */
 *define FSMC_SR2_ILS ((uint8_t)0x02) /* !< Interrupt Level status */
 *define FSMC_SR2_IFS ((uint8_t)0x04) /* !< Interrupt Falling Edge status */
 *define FSMC_SR2_IREN ((uint8_t)0x08) /* !< Interrupt Rising Edge detection Enab */
 *define FSMC_SR2_ILEN ((uint8_t)0x10) /* !< Interrupt Level detection Enable bit */
 *define FSMC_SR2_IFEN ((uint8_t)0x20) /* !< Interrupt Falling Edge detection Enab */
 *define FSMC_SR2_FEMPT ((uint8_t)0x40) /* !< FIFO empty */

/*
 * Bit definition for FSMC_SR3 register *****
 *define FSMC_SR3_IRS ((uint8_t)0x01) /* !< Interrupt Rising Edge status */
 *define FSMC_SR3_ILS ((uint8_t)0x02) /* !< Interrupt Level status */
 *define FSMC_SR3_IFS ((uint8_t)0x04) /* !< Interrupt Falling Edge status */
 *define FSMC_SR3_IREN ((uint8_t)0x08) /* !< Interrupt Rising Edge detection Enab */
 *define FSMC_SR3_ILEN ((uint8_t)0x10) /* !< Interrupt Level detection Enable bit */
 *define FSMC_SR3_IFEN ((uint8_t)0x20) /* !< Interrupt Falling Edge detection Enab */
 *define FSMC_SR3_FEMPT ((uint8_t)0x40) /* !< FIFO empty */

/*
 * Bit definition for FSMC_SR4 register *****
 *define FSMC_SR4_IRS ((uint8_t)0x01) /* !< Interrupt Rising Edge status */
 *define FSMC_SR4_ILS ((uint8_t)0x02) /* !< Interrupt Level status */
 *define FSMC_SR4_IFS ((uint8_t)0x04) /* !< Interrupt Falling Edge status */
 *define FSMC_SR4_IREN ((uint8_t)0x08) /* !< Interrupt Rising Edge detection Enab */
 *define FSMC_SR4_ILEN ((uint8_t)0x10) /* !< Interrupt Level detection Enable bit */
 *define FSMC_SR4_IFEN ((uint8_t)0x20) /* !< Interrupt Falling Edge detection Enab */
 *define FSMC_SR4_FEMPT ((uint8_t)0x40) /* !< FIFO empty */

/*
 * Bit definition for FSMC_PMEM2 register *****
 *define FSMC_PMEM2_MEMSET2 ((uint32_t)0x000000FF) /* !< MEMSET2[7:0] bits (Common memory 2 set)
 *define FSMC_PMEM2_MEMSET2_0 ((uint32_t)0x00000001) /* !< Bit 0 */
 *define FSMC_PMEM2_MEMSET2_1 ((uint32_t)0x00000002) /* !< Bit 1 */
 *define FSMC_PMEM2_MEMSET2_2 ((uint32_t)0x00000004) /* !< Bit 2 */
 *define FSMC_PMEM2_MEMSET2_3 ((uint32_t)0x00000008) /* !< Bit 3 */
 *define FSMC_PMEM2_MEMSET2_4 ((uint32_t)0x00000010) /* !< Bit 4 */
 *define FSMC_PMEM2_MEMSET2_5 ((uint32_t)0x00000020) /* !< Bit 5 */

```

```

#define FSMC_PMEM2_MEMSET2_6 ((uint32_t)0x000000040)
#define FSMC_PMEM2_MEMSET2_7 ((uint32_t)0x000000080)

#define FSMC_PMEM2_MEMWAIT2_0 /*!<Bit 6 */
#define FSMC_PMEM2_MEMWAIT2_1 /*!<Bit 7 */
#define FSMC_PMEM2_MEMWAIT2_2 /*!<Bit 8 */
#define FSMC_PMEM2_MEMWAIT2_3 /*!<Bit 9 */
#define FSMC_PMEM2_MEMWAIT2_4 /*!<Bit 10 */
#define FSMC_PMEM2_MEMWAIT2_5 /*!<Bit 11 */
#define FSMC_PMEM2_MEMWAIT2_6 /*!<Bit 12 */
#define FSMC_PMEM2_MEMWAIT2_7 /*!<Bit 13 */

#define FSMC_PMEM2_MEMHOLD2_0 /*!<Bit 0 */
#define FSMC_PMEM2_MEMHOLD2_1 /*!<Bit 1 */
#define FSMC_PMEM2_MEMHOLD2_2 /*!<Bit 2 */
#define FSMC_PMEM2_MEMHOLD2_3 /*!<Bit 3 */
#define FSMC_PMEM2_MEMHOLD2_4 /*!<Bit 4 */
#define FSMC_PMEM2_MEMHOLD2_5 /*!<Bit 5 */
#define FSMC_PMEM2_MEMHOLD2_6 /*!<Bit 6 */
#define FSMC_PMEM2_MEMHOLD2_7 /*!<Bit 7 */

#define FSMC_PMEM2_MEMHIZ2_0 /*!<Bit 0 */
#define FSMC_PMEM2_MEMHIZ2_1 /*!<Bit 1 */
#define FSMC_PMEM2_MEMHIZ2_2 /*!<Bit 2 */
#define FSMC_PMEM2_MEMHIZ2_3 /*!<Bit 3 */
#define FSMC_PMEM2_MEMHIZ2_4 /*!<Bit 4 */
#define FSMC_PMEM2_MEMHIZ2_5 /*!<Bit 5 */
#define FSMC_PMEM2_MEMHIZ2_6 /*!<Bit 6 */
#define FSMC_PMEM2_MEMHIZ2_7 /*!<Bit 7 */

/****** Bit definition for FSMC_PMEM3 register *****/
#define FSMC_PMEM3_MEMSET3 ((uint32_t)0x0000000FF) /*!<MEMSET3[7:0] bits (Common memory 2 set)
#define FSMC_PMEM3_REGISTER (((uint32_t)0x0000000FF) /*!<MEMSET3[7:0] bits (Common memory 2 set)

```

```

#define FSMC_PMEM3_MEMSET3_0 ((uint32_t)0x00000000)
#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)0x00001000)
#define FSMC_PMEM3_MEMWAIT3_1 ((uint32_t)0x00002000)
#define FSMC_PMEM3_MEMWAIT3_2 ((uint32_t)0x00004000)
#define FSMC_PMEM3_MEMWAIT3_3 ((uint32_t)0x00008000)
#define FSMC_PMEM3_MEMWAIT3_4 ((uint32_t)0x00010000)
#define FSMC_PMEM3_MEMWAIT3_5 ((uint32_t)0x00020000)
#define FSMC_PMEM3_MEMWAIT3_6 ((uint32_t)0x00040000)
#define FSMC_PMEM3_MEMWAIT3_7 ((uint32_t)0x00080000)

#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_MEMHZ3 ((uint32_t)0xFFFF0000)
#define FSMC_PMEM3_MEMHZ3_0 ((uint32_t)0x01000000)
#define FSMC_PMEM3_MEMHZ3_1 ((uint32_t)0x02000000)
#define FSMC_PMEM3_MEMHZ3_2 ((uint32_t)0x04000000)
#define FSMC_PMEM3_MEMHZ3_3 ((uint32_t)0x08000000)
#define FSMC_PMEM3_MEMHZ3_4 ((uint32_t)0x10000000)

/* !<Bit 0 */ /* Common memory 3 bits */
/* !<Bit 1 */ /* Common memory 3 bits */
/* !<Bit 2 */ /* Common memory 3 bits */
/* !<Bit 3 */ /* Common memory 3 bits */
/* !<Bit 4 */ /* Common memory 3 bits */
/* !<Bit 5 */ /* Common memory 3 bits */
/* !<Bit 6 */ /* Common memory 3 bits */
/* !<Bit 7 */ /* Common memory 3 bits */

/* !<MEMWAIT3 [7:0] bits (Common memory 3 bits) */
/* !<Bit 0 */ /* Common memory 3 bits */
/* !<Bit 1 */ /* Common memory 3 bits */
/* !<Bit 2 */ /* Common memory 3 bits */
/* !<Bit 3 */ /* Common memory 3 bits */
/* !<Bit 4 */ /* Common memory 3 bits */
/* !<Bit 5 */ /* Common memory 3 bits */
/* !<Bit 6 */ /* Common memory 3 bits */
/* !<Bit 7 */ /* Common memory 3 bits */

/* !<MEMHOLD3 [7:0] bits (Common memory 3 bits) */
/* !<Bit 0 */ /* Common memory 3 bits */
/* !<Bit 1 */ /* Common memory 3 bits */
/* !<Bit 2 */ /* Common memory 3 bits */
/* !<Bit 3 */ /* Common memory 3 bits */
/* !<Bit 4 */ /* Common memory 3 bits */
/* !<Bit 5 */ /* Common memory 3 bits */
/* !<Bit 6 */ /* Common memory 3 bits */
/* !<Bit 7 */ /* Common memory 3 bits */

/* !<MEMHZ3 [7:0] bits (Common memory 3 bits) */
/* !<Bit 0 */ /* Common memory 3 bits */
/* !<Bit 1 */ /* Common memory 3 bits */
/* !<Bit 2 */ /* Common memory 3 bits */
/* !<Bit 3 */ /* Common memory 3 bits */
/* !<Bit 4 */ /* Common memory 3 bits */

```

```

#define FSMC_PMEM3_MEMHZ3_5 ((uint32_t)0x20000000)
#define FSMC_PMEM3_MEMHZ3_6 ((uint32_t)0x40000000)
#define FSMC_PMEM3_MEMHZ3_7 ((uint32_t)0x80000000)

/* ***** Bit definition for FSMC_PMEM4 register *****/
#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)0x00000001)
#define FSMC_PMEM4_MEMWAIT4_1 ((uint32_t)0x00000020)
#define FSMC_PMEM4_MEMWAIT4_2 ((uint32_t)0x00000040)
#define FSMC_PMEM4_MEMWAIT4_3 ((uint32_t)0x00000080)
#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)0x000FF000)
#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)

/* !<Bit 5 */ bits (Common memory 4 words)
/* !<MEMSET4[7:0] */
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */
/* !<Bit 3 */
/* !<Bit 4 */
/* !<Bit 5 */
/* !<Bit 6 */
/* !<Bit 7 */

/* !<MEMWAIT4[7:0] */
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */
/* !<Bit 3 */
/* !<Bit 4 */
/* !<Bit 5 */
/* !<Bit 6 */
/* !<Bit 7 */

/* !<MEMHOLD4[7:0] */
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */
/* !<Bit 3 */
/* !<Bit 4 */
/* !<Bit 5 */
/* !<Bit 6 */
/* !<Bit 7 */

```

```

#define FSMC_PMEM4_MEMHZ4 ((uint32_t)0xFFFF000000)
#define FSMC_PMEM4_MEMHZ4_0 ((uint32_t)0x01000000)
#define FSMC_PMEM4_MEMHZ4_1 ((uint32_t)0x02000000)
#define FSMC_PMEM4_MEMHZ4_2 ((uint32_t)0x04000000)
#define FSMC_PMEM4_MEMHZ4_3 ((uint32_t)0x08000000)
#define FSMC_PMEM4_MEMHZ4_4 ((uint32_t)0x10000000)
#define FSMC_PMEM4_MEMHZ4_5 ((uint32_t)0x20000000)
#define FSMC_PMEM4_MEMHZ4_6 ((uint32_t)0x40000000)
#define FSMC_PMEM4_MEMHZ4_7 ((uint32_t)0x80000000)

/************************************************************************** Bit definition for FSMC_PATT2 register *****/
#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)0x000FF000)
#define FSMC_PATT2_ATTHOLD2_0 ((uint32_t)0x00010000)
#define FSMC_PATT2_ATTHOLD2_1 ((uint32_t)0x00020000)
#define FSMC_PATT2_ATTHOLD2_2 ((uint32_t)0x00040000)

/* !<MEMHZ4[7:0] bits (Common memory 2 */
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */
/* !<Bit 3 */
/* !<Bit 4 */
/* !<Bit 5 */
/* !<Bit 6 */
/* !<Bit 7 */

/* !<ATTSET2[7:0] bits (Attribute memory 2 */
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */
/* !<Bit 3 */
/* !<Bit 4 */
/* !<Bit 5 */
/* !<Bit 6 */
/* !<Bit 7 */

/* !<ATTWAIT2[7:0] bits (Attribute memory */
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */
/* !<Bit 3 */
/* !<Bit 4 */
/* !<Bit 5 */
/* !<Bit 6 */
/* !<Bit 7 */

/* !<ATTHOLD2[7:0] bits (Attribute memory */
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */

```

```

/* !<Bit 3 */
(((uint32_t)0x0000800000)
((uint32_t)0x0001000000)
((uint32_t)0x0002000000)
((uint32_t)0x0004000000)
((uint32_t)0x0008000000)

/* !<Bit 7 */
(((uint32_t)0xFF000000)
((uint32_t)0x01000000)
((uint32_t)0x02000000)
((uint32_t)0x04000000)
((uint32_t)0x08000000)
((uint32_t)0x10000000)
((uint32_t)0x20000000)
((uint32_t)0x40000000)
((uint32_t)0x80000000))

/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */
/* !<Bit 3 */
/* !<Bit 4 */
/* !<Bit 5 */
/* !<Bit 6 */
/* !<Bit 7 */

/* !<ATTHIZZ2[7:0] bits (Attribute memory 2)
(((uint32_t)0x00000000)
((uint32_t)0x01000000)
((uint32_t)0x02000000)
((uint32_t)0x04000000)
((uint32_t)0x08000000)
((uint32_t)0x10000000)
((uint32_t)0x20000000)
((uint32_t)0x40000000)
((uint32_t)0x80000000))

/* !<ATTSET3[7:0] bits (Attribute memory 3)
(((uint32_t)0x000000FF)
((uint32_t)0x00000001)
((uint32_t)0x00000002)
((uint32_t)0x00000004)
((uint32_t)0x00000008)
((uint32_t)0x00000010)
((uint32_t)0x00000020)
((uint32_t)0x00000040)
((uint32_t)0x00000080)

/* !<ATTWAIT3[7:0] bits (Attribute memory
(((uint32_t)0x00000F00)
((uint32_t)0x00000100)
((uint32_t)0x00000200)
((uint32_t)0x00000400)
((uint32_t)0x00000800)
((uint32_t)0x00001000)
((uint32_t)0x00002000)
((uint32_t)0x00004000)

/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */
/* !<Bit 3 */
/* !<Bit 4 */
/* !<Bit 5 */
/* !<Bit 6 */

/* **** Bit definition for FSMC_PATT3 register ****/
#define FSMC_PATT3_ATTSET3_0
#define FSMC_PATT3_ATTSET3_1
#define FSMC_PATT3_ATTSET3_2
#define FSMC_PATT3_ATTSET3_3
#define FSMC_PATT3_ATTSET3_4
#define FSMC_PATT3_ATTSET3_5
#define FSMC_PATT3_ATTSET3_6
#define FSMC_PATT3_ATTSET3_7

/* **** Bit definition for FSMC_PATT3_0 */
#define FSMC_PATT3_ATTWAIT3_0
#define FSMC_PATT3_ATTWAIT3_1
#define FSMC_PATT3_ATTWAIT3_2
#define FSMC_PATT3_ATTWAIT3_3
#define FSMC_PATT3_ATTWAIT3_4
#define FSMC_PATT3_ATTWAIT3_5
#define FSMC_PATT3_ATTWAIT3_6

```

```

/* !<Bit 7 */

#define FSMC_PATT3_ATTWAIT3_7 ((uint32_t)0x0000080000)

#define FSMC_PATT3_ATTHOLD3 ((uint32_t)0x000FF0000)
#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)

/**************** Bit definition for FSMC_PATT4 register *****/
#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)0x00000FF00)
#define FSMC_PATT4_ATTWAIT4_0 ((uint32_t)0x00000100)

/* !<Bit 0 */

/* !<Bit 7 */

/* !<Bit 0 */

/* !<Bit 1 */

/* !<Bit 2 */

/* !<Bit 3 */

/* !<Bit 4 */

/* !<Bit 5 */

/* !<Bit 6 */

/* !<Bit 7 */

/* !<Bit 0 */

/* !<Bit 1 */

/* !<Bit 2 */

/* !<Bit 3 */

/* !<Bit 4 */

/* !<Bit 5 */

/* !<Bit 6 */

/* !<Bit 7 */

/* !<Bit 0 */

/* !<Bit 1 */

/* !<Bit 2 */

/* !<Bit 3 */

/* !<Bit 4 */

/* !<Bit 5 */

/* !<Bit 6 */

/* !<Bit 7 */

/* !<Bit 0 */

/* !<Bit 1 */

/* !<Bit 2 */

/* !<Bit 3 */

/* !<Bit 4 */

/* !<Bit 5 */

/* !<Bit 6 */

/* !<Bit 7 */

/* !<Bit 0 */

/* !<Bit 1 */

/* !<Bit 2 */

/* !<Bit 3 */

/* !<Bit 4 */

/* !<Bit 5 */

/* !<Bit 6 */

/* !<Bit 7 */

/* !<Bit 0 */

/* !<Bit 1 */

/* !<Bit 2 */

/* !<Bit 3 */

/* !<Bit 4 */

/* !<Bit 5 */

/* !<Bit 6 */

/* !<Bit 7 */

/* !<Bit 0 */

/* !<Bit 1 */

/* !<Bit 2 */

/* !<Bit 3 */

/* !<Bit 4 */

/* !<Bit 5 */

/* !<Bit 6 */

/* !<Bit 7 */

```

```

/* !<Bit 1 */
(((uint32_t)0x000000200)
((uint32_t)0x000000400)
((uint32_t)0x000000800)
((uint32_t)0x00001000)
((uint32_t)0x00002000)
((uint32_t)0x00004000)
((uint32_t)0x00008000)

/* !<Bit 2 */
(((uint32_t)0x00000001)
((uint32_t)0x00000002)
((uint32_t)0x00000004)
((uint32_t)0x00000008)
((uint32_t)0x00000010)
((uint32_t)0x00000020)
((uint32_t)0x00000040)
((uint32_t)0x00000080)

/* !<Bit 3 */
(((uint32_t)0x00000000)
((uint32_t)0x00000001)
((uint32_t)0x00000002)
((uint32_t)0x00000004)
((uint32_t)0x00000008)
((uint32_t)0x00000010)
((uint32_t)0x00000020)
((uint32_t)0x00000040)
((uint32_t)0x00000080)

/* !<Bit 4 */
(((uint32_t)0x00000000)
((uint32_t)0x00000001)
((uint32_t)0x00000002)
((uint32_t)0x00000004)
((uint32_t)0x00000008)
((uint32_t)0x00000010)
((uint32_t)0x00000020)
((uint32_t)0x00000040)
((uint32_t)0x00000080)

/* !<Bit 5 */
(((uint32_t)0x00000000)
((uint32_t)0x00000001)
((uint32_t)0x00000002)
((uint32_t)0x00000004)
((uint32_t)0x00000008)
((uint32_t)0x00000010)
((uint32_t)0x00000020)
((uint32_t)0x00000040)
((uint32_t)0x00000080)

/* !<Bit 6 */
(((uint32_t)0x00000000)
((uint32_t)0x00000001)
((uint32_t)0x00000002)
((uint32_t)0x00000004)
((uint32_t)0x00000008)
((uint32_t)0x00000010)
((uint32_t)0x00000020)
((uint32_t)0x00000040)
((uint32_t)0x00000080)

/* !<Bit 7 */
(((uint32_t)0x00000000)
((uint32_t)0x00000001)
((uint32_t)0x00000002)
((uint32_t)0x00000004)
((uint32_t)0x00000008)
((uint32_t)0x00000010)
((uint32_t)0x00000020)
((uint32_t)0x00000040)
((uint32_t)0x00000080)

/* !<ATTHOLD4[7:0] bits (Attribute memory 4)
(((uint32_t)0x000FF0000)
((uint32_t)0x00010000)
((uint32_t)0x00020000)
((uint32_t)0x00040000)
((uint32_t)0x00080000)
((uint32_t)0x00100000)
((uint32_t)0x00200000)
((uint32_t)0x00400000)
((uint32_t)0x00800000)
((uint32_t)0x01000000)
((uint32_t)0x02000000)
((uint32_t)0x04000000)
((uint32_t)0x08000000)
((uint32_t)0x10000000)
((uint32_t)0x20000000)
((uint32_t)0x40000000)
((uint32_t)0x80000000)

/* !<ATTIIZ4[7:0] bits (Attribute memory 4)
(((uint32_t)0xFF000000)
((uint32_t)0x01000000)
((uint32_t)0x02000000)
((uint32_t)0x04000000)
((uint32_t)0x08000000)
((uint32_t)0x10000000)
((uint32_t)0x20000000)
((uint32_t)0x40000000)
((uint32_t)0x80000000)

/* !<IOSET4[7:0] bits (I/O 4 setup time) *
(((uint32_t)0x000000FF)
((uint32_t)0x00000001)
((uint32_t)0x00000002)
((uint32_t)0x00000004)
((uint32_t)0x00000008)
((uint32_t)0x00000010)

***** Bit definition for FSMC_PIO4 register *****/
#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_PATT4_ATTHOLD4 (((uint32_t)0x00000000))
#define FSMC_PATT4_ATTHOLD4_0 (((uint32_t)0x00000001))
#define FSMC_PATT4_ATTHOLD4_1 (((uint32_t)0x00000002))
#define FSMC_PATT4_ATTHOLD4_2 (((uint32_t)0x00000004))
#define FSMC_PATT4_ATTHOLD4_3 (((uint32_t)0x00000008))
#define FSMC_PATT4_ATTHOLD4_4 (((uint32_t)0x00000010))
#define FSMC_PATT4_ATTHOLD4_5 (((uint32_t)0x00000020))
#define FSMC_PATT4_ATTHOLD4_6 (((uint32_t)0x00000040))
#define FSMC_PATT4_ATTHOLD4_7 (((uint32_t)0x00000080))

#define FSMC_PATT4_ATTHIZ4 (((uint32_t)0x00000000))
#define FSMC_PATT4_ATTHIZ4_0 (((uint32_t)0x00000001))
#define FSMC_PATT4_ATTHIZ4_1 (((uint32_t)0x00000002))
#define FSMC_PATT4_ATTHIZ4_2 (((uint32_t)0x00000004))
#define FSMC_PATT4_ATTHIZ4_3 (((uint32_t)0x00000008))
#define FSMC_PATT4_ATTHIZ4_4 (((uint32_t)0x00000010))
#define FSMC_PATT4_ATTHIZ4_5 (((uint32_t)0x00000020))
#define FSMC_PATT4_ATTHIZ4_6 (((uint32_t)0x00000040))
#define FSMC_PATT4_ATTHIZ4_7 (((uint32_t)0x00000080))

```

```

/* !<Bit 5 */          bits (I/O 4 wait time) * TM32
((uint32_t)0x000000020)
((uint32_t)0x000000040)
((uint32_t)0x000000080)

/* !<Bit 6 */          bits (I/O 4 wait time) *
((uint32_t)0x000000FF00)
((uint32_t)0x00000100)
((uint32_t)0x00000200)
((uint32_t)0x00000400)
((uint32_t)0x00000800)
((uint32_t)0x00001000)
((uint32_t)0x00002000)
((uint32_t)0x00004000)
((uint32_t)0x00008000)
((uint32_t)0x0000F000)

/* !<Bit 7 */          bits (I/O 4 wait time) *
((uint32_t)0x00000000)
((uint32_t)0x0000010000)
((uint32_t)0x0000020000)
((uint32_t)0x0000040000)
((uint32_t)0x0000080000)
((uint32_t)0x0000100000)
((uint32_t)0x0000200000)
((uint32_t)0x0000400000)
((uint32_t)0x0000800000)
((uint32_t)0x0000F00000)

/* !<Bit 0 */          bits (I/O 4 hold time) *
/* !<Bit 1 */          bits (I/O 4 hold time) *
/* !<Bit 2 */          bits (I/O 4 hold time) *
/* !<Bit 3 */          bits (I/O 4 hold time) *
/* !<Bit 4 */          bits (I/O 4 hold time) *
/* !<Bit 5 */          bits (I/O 4 hold time) *
/* !<Bit 6 */          bits (I/O 4 hold time) *
/* !<Bit 7 */          bits (I/O 4 hold time) *

/* !<Bit 0 */          bits (I/O 4 databus HIZ t)
/* !<Bit 1 */          bits (I/O 4 databus HIZ t)
/* !<Bit 2 */          bits (I/O 4 databus HIZ t)
/* !<Bit 3 */          bits (I/O 4 databus HIZ t)
/* !<Bit 4 */          bits (I/O 4 databus HIZ t)
/* !<Bit 5 */          bits (I/O 4 databus HIZ t)
/* !<Bit 6 */          bits (I/O 4 databus HIZ t)
/* !<Bit 7 */          bits (I/O 4 databus HIZ t)

FSMC_PIO4_IOWAIT4_5
FSMC_PIO4_IOWAIT4_6
FSMC_PIO4_IOWAIT4_7

#define FSMC_PIO4_IOWAIT4_0
#define FSMC_PIO4_IOWAIT4_1
#define FSMC_PIO4_IOWAIT4_2
#define FSMC_PIO4_IOWAIT4_3
#define FSMC_PIO4_IOWAIT4_4
#define FSMC_PIO4_IOWAIT4_5
#define FSMC_PIO4_IOWAIT4_6
#define FSMC_PIO4_IOWAIT4_7

#define FSMC_PIO4_IOHOLD4_0
#define FSMC_PIO4_IOHOLD4_1
#define FSMC_PIO4_IOHOLD4_2
#define FSMC_PIO4_IOHOLD4_3
#define FSMC_PIO4_IOHOLD4_4
#define FSMC_PIO4_IOHOLD4_5
#define FSMC_PIO4_IOHOLD4_6
#define FSMC_PIO4_IOHOLD4_7

#define FSMC_PIO4_IOHIZ4_0
#define FSMC_PIO4_IOHIZ4_1
#define FSMC_PIO4_IOHIZ4_2
#define FSMC_PIO4_IOHIZ4_3
#define FSMC_PIO4_IOHIZ4_4
#define FSMC_PIO4_IOHIZ4_5
#define FSMC_PIO4_IOHIZ4_6
#define FSMC_PIO4_IOHIZ4_7

***** Bit definition for FSMC_ECCR2 register *****

```

```

#define FSMC_ECCR2_ECC2 ((uint32_t)0xFFFFFFFF) /* !<ECC result */

/* ***** Bit definition for FSMC_ECCR3 register *****/
#define FSMC_ECCR3_ECC3 ((uint32_t)0xFFFFFFFF) /* !<ECC result */

/* **** General Purpose I/O ****/
/* Bits definition for GPIO_MODER register *****/
#define GPIO_MODER_MODERO ((uint32_t)0x00000003)
#define GPIO_MODER_MODERO_0 ((uint32_t)0x00000001)
#define GPIO_MODER_MODERO_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)0x000000300)
#define GPIO_MODER_MODER4_0 ((uint32_t)0x00000100)
#define GPIO_MODER_MODER4_1 ((uint32_t)0x00000200)

#define GPIO_MODER_MODER5 ((uint32_t)0x000000C00)
#define GPIO_MODER_MODER5_0 ((uint32_t)0x00000400)
#define GPIO_MODER_MODER5_1 ((uint32_t)0x00000800)

```

```
#define GPIO_MODER6 (((uint32_t)0x0000030000)
#define GPIO_MODER6_0 (((uint32_t)0x000001000)
#define GPIO_MODER6_1 (((uint32_t)0x000002000))

#define GPIO_MODER7 (((uint32_t)0x00000C000)
#define GPIO_MODER7_0 (((uint32_t)0x000004000)
#define GPIO_MODER7_1 (((uint32_t)0x000008000)

#define GPIO_MODER8 (((uint32_t)0x000030000)
#define GPIO_MODER8_0 (((uint32_t)0x000010000)
#define GPIO_MODER8_1 (((uint32_t)0x000020000)

#define GPIO_MODER9 (((uint32_t)0x0000C0000)
#define GPIO_MODER9_0 (((uint32_t)0x000040000)
#define GPIO_MODER9_1 (((uint32_t)0x000080000)

#define GPIO_MODER10 (((uint32_t)0x000300000)
#define GPIO_MODER10_0 (((uint32_t)0x001000000)
#define GPIO_MODER10_1 (((uint32_t)0x002000000)

#define GPIO_MODER11 (((uint32_t)0x00C000000)
#define GPIO_MODER11_0 (((uint32_t)0x004000000)
#define GPIO_MODER11_1 (((uint32_t)0x008000000)

#define GPIO_MODER12 (((uint32_t)0x003000000)
#define GPIO_MODER12_0 (((uint32_t)0x010000000)
#define GPIO_MODER12_1 (((uint32_t)0x020000000)

#define GPIO_MODER13 (((uint32_t)0x00C000000)
#define GPIO_MODER13_0 (((uint32_t)0x040000000)
#define GPIO_MODER13_1 (((uint32_t)0x080000000)

#define GPIO_MODER14 (((uint32_t)0x300000000)
#define GPIO_MODER14_0 (((uint32_t)0x100000000)
#define GPIO_MODER14_1 (((uint32_t)0x200000000))
```

```

#define GPIO_MODER_MODER15 ((uint32_t)0xC0000000)
#define GPIO_MODER_MODER15_0 ((uint32_t)0x40000000)
#define GPIO_MODER_MODER15_1 ((uint32_t)0x80000000)

/**************** Bits definition for GPIO_OTYPER register *****/
#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)

/**************** Bits definition for GPIO_OSPEEDR register *****/
#define GPIO_OSPEEDR_OSPEEDR0 ((uint32_t)0x00000003)
#define GPIO_OSPEEDR_OSPEEDR0_0 ((uint32_t)0x00000001)
#define GPIO_OSPEEDR_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)0x000000300))
#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)0x000003000))
#define GPIO_OSPEEDER_OSPEEDR6_0 (((uint32_t)0x00001000))
#define GPIO_OSPEEDER_OSPEEDR6_1 (((uint32_t)0x00002000))

#define GPIO_OSPEEDER_OSPEEDR7 (((uint32_t)0x00000C000))
#define GPIO_OSPEEDER_OSPEEDR7_0 (((uint32_t)0x000004000))
#define GPIO_OSPEEDER_OSPEEDR7_1 (((uint32_t)0x000008000))

#define GPIO_OSPEEDER_OSPEEDR8 (((uint32_t)0x000030000))
#define GPIO_OSPEEDER_OSPEEDR8_0 (((uint32_t)0x000010000))
#define GPIO_OSPEEDER_OSPEEDR8_1 (((uint32_t)0x000020000))

#define GPIO_OSPEEDER_OSPEEDR9 (((uint32_t)0x000000000))
#define GPIO_OSPEEDER_OSPEEDR9_0 (((uint32_t)0x000040000))
#define GPIO_OSPEEDER_OSPEEDR9_1 (((uint32_t)0x000080000))

#define GPIO_OSPEEDER_OSPEEDR10 (((uint32_t)0x003000000))
#define GPIO_OSPEEDER_OSPEEDR10_0 (((uint32_t)0x001000000))
#define GPIO_OSPEEDER_OSPEEDR10_1 (((uint32_t)0x002000000))

#define GPIO_OSPEEDER_OSPEEDR11 (((uint32_t)0x00C000000))
#define GPIO_OSPEEDER_OSPEEDR11_0 (((uint32_t)0x004000000))
```

---

```

#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)

/**************** Bits definition for GPIO_PUPDR register *****/
#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)0x0000003000)
#define GPIO_PUPDR_PUPDR4_0 (((uint32_t)0x000000100)
#define GPIO_PUPDR_PUPDR4_1 (((uint32_t)0x000000200))

#define GPIO_PUPDR_PUPDR5 (((uint32_t)0x000000C000)
#define GPIO_PUPDR_PUPDR5_0 (((uint32_t)0x000000400)
#define GPIO_PUPDR_PUPDR5_1 (((uint32_t)0x000000800))

#define GPIO_PUPDR_PUPDR6 (((uint32_t)0x0000003000)
#define GPIO_PUPDR_PUPDR6_0 (((uint32_t)0x0000001000)
#define GPIO_PUPDR_PUPDR6_1 (((uint32_t)0x0000002000))

#define GPIO_PUPDR_PUPDR7 (((uint32_t)0x000000C000)
#define GPIO_PUPDR_PUPDR7_0 (((uint32_t)0x0000004000)
#define GPIO_PUPDR_PUPDR7_1 (((uint32_t)0x0000008000))

#define GPIO_PUPDR_PUPDR8 (((uint32_t)0x00000030000)
#define GPIO_PUPDR_PUPDR8_0 (((uint32_t)0x000010000)
#define GPIO_PUPDR_PUPDR8_1 (((uint32_t)0x000020000))

#define GPIO_PUPDR_PUPDR9 (((uint32_t)0x000000C0000)
#define GPIO_PUPDR_PUPDR9_0 (((uint32_t)0x000040000)
#define GPIO_PUPDR_PUPDR9_1 (((uint32_t)0x000080000))

#define GPIO_PUPDR_PUPDR10 (((uint32_t)0x0000300000)
#define GPIO_PUPDR_PUPDR10_0 (((uint32_t)0x001000000)
#define GPIO_PUPDR_PUPDR10_1 (((uint32_t)0x002000000))

#define GPIO_PUPDR_PUPDR11 (((uint32_t)0x0000C00000)
#define GPIO_PUPDR_PUPDR11_0 (((uint32_t)0x004000000)
#define GPIO_PUPDR_PUPDR11_1 (((uint32_t)0x008000000))

#define GPIO_PUPDR_PUPDR12 (((uint32_t)0x0030000000)
#define GPIO_PUPDR_PUPDR12_0 (((uint32_t)0x010000000)
#define GPIO_PUPDR_PUPDR12_1 (((uint32_t)0x020000000))
```

---

```

#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)

/****** Bits definition for GPIO_IDR register *****/
#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)

/* Old GPIO_IDR register bits definition, maintained for legacy purpose */
#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

/****** Bits definition for GPIO_ODR register *****/
#define GPIO_ODR_0              ((uint32_t)0x00000001)
#define GPIO_ODR_1              ((uint32_t)0x00000002)
#define GPIO_ODR_2              ((uint32_t)0x00000004)
#define GPIO_ODR_3              ((uint32_t)0x00000008)
#define GPIO_ODR_4              ((uint32_t)0x00000010)
#define GPIO_ODR_5              ((uint32_t)0x00000020)
#define GPIO_ODR_6              ((uint32_t)0x00000040)
#define GPIO_ODR_7              ((uint32_t)0x00000080)
#define GPIO_ODR_8              ((uint32_t)0x00000100)
#define GPIO_ODR_9              ((uint32_t)0x00000200)
#define GPIO_ODR_10             ((uint32_t)0x00000400)
#define GPIO_ODR_11             ((uint32_t)0x00000800)
#define GPIO_ODR_12             ((uint32_t)0x00001000)
#define GPIO_ODR_13             ((uint32_t)0x00002000)
#define GPIO_ODR_14             ((uint32_t)0x00004000)
#define GPIO_ODR_15             ((uint32_t)0x00008000)

/* Old GPIO_ODR register bits definition, maintained for legacy purpose */
#define GPIO_ODR_0              GPIO_ODR_ODR_0
#define GPIO_ODR_1              GPIO_ODR_ODR_1
#define GPIO_ODR_2              GPIO_ODR_ODR_2
#define GPIO_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

/**************** Bits definition for GPIO_BSRR register *****/
#define GPIO_BSRR_BS_0          ((uint32_t)0x00000000)
#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)0x00010000)
#define GPIO_BSRR_BR_0          ((uint32_t)0x00020000)
#define GPIO_BSRR_BR_1          ((uint32_t)0x00040000)
#define GPIO_BSRR_BR_2          ((uint32_t)0x00080000)
#define GPIO_BSRR_BR_3          ((uint32_t)0x00100000)
#define GPIO_BSRR_BR_4          ((uint32_t)0x00200000)

```

---

```

#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)

/***************************************************************************
 * Bits definition for HASH_CR register *****
 **************************************************************************/
#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)0x000000F00)
#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_KEY ((uint32_t)0x000010000)

/***************************************************************************
 * Bits definition for HASH_STR register *****
 **************************************************************************/
#define HASH_STR_NBW ((uint32_t)0x00000001F)

```

```

#define HASH_STR_NBW_0 ((uint32_t)0x00000000)
#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)0x000000100)

/****** Bits definition for HASH_IMR register *****/
#define HASH_SR_DINIM ((uint32_t)0x00000001)
#define HASH_SR_DCIM ((uint32_t)0x00000002)
#define HASH_SR_DMAS ((uint32_t)0x00000004)
#define HASH_SR_BUSY ((uint32_t)0x00000008)

/****** Bit definition for HASH_SR register ****/
#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)

/****** Inter-integrated Circuit Interface ****/
/****** Bit definition for I2C_CRI register ****/
#define I2C_CR1_PE ((uint16_t)0x0001) /*!< Peripheral Enable */
#define I2C_CR1_SMBUS ((uint16_t)0x0002) /*!< SMBus Mode */
#define I2C_CR1_SMBTYPE ((uint16_t)0x0008) /*!< SMBus Type */
#define I2C_CR1_ENARP ((uint16_t)0x0010) /*!< ARP Enable */
#define I2C_CR1_ENPEC ((uint16_t)0x0020) /*!< PEC Enable */
#define I2C_CR1_ENGC ((uint16_t)0x0040) /*!< General Call Enable */
#define I2C_CR1_NOSTRETCH ((uint16_t)0x0080) /*!< Clock Stretching Disable (Slave mode) */
#define I2C_CR1_START ((uint16_t)0x0100) /*!< Start Generation */
#define I2C_CR1_STOP ((uint16_t)0x0200) /*!< Stop Generation */
#define I2C_CR1_ACK ((uint16_t)0x0400) /*!< Acknowledge Enable */
#define I2C_CR1_POS ((uint16_t)0x0800) /*!< Acknowledge/PEC Position (for data receiving) */
#define I2C_CR1_PEC ((uint16_t)0x1000) /*!< Packet Error Checking */

```

```

/* !<SMBus Alert */ ((uint16_t)0x2000) /* !<Software Reset */
#define I2C_CR1_SWRST ((uint16_t)0x8000)

/* ***** Bit definition for I2C_CR2 register *****/
#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_IDEVEN ((uint16_t)0x0200)
#define I2C_CR2_ITBUFEN ((uint16_t)0x0400)
#define I2C_CR2_DMAEN ((uint16_t)0x0800)
#define I2C_CR2_LAST ((uint16_t)0x1000)

/* ***** Bit definition for I2C_OAR1 register *****/
#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)

/* !<Clock Freq bits (Peripheral ClockFreq */
/* !<FREQ [5:0] bits (Peripheral ClockFreq */
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */
/* !<Bit 3 */
/* !<Bit 4 */
/* !<Bit 5 */

/* !<Error Interrupt Enable */
/* !<Event Interrupt Enable */
/* !<Buffer Interrupt Enable */
/* !<DMA Requests Enable */
/* !<DMA Last Transfer */

/* !<Interface Address */
/* !<Interface Address */

/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */
/* !<Bit 3 */
/* !<Bit 4 */
/* !<Bit 5 */
/* !<Bit 6 */
/* !<Bit 7 */
/* !<Bit 8 */
/* !<Bit 9 */

/* !<Addressing Mode (Slave mode) */

```

```

/*
 * Bit definition for I2C_OAR2 register *****
 * #define I2C_OAR2_ENDUAL ((uint8_t)0x01) /* !< Dual addressing mode enable */
 * #define I2C_OAR2_ADD2 ((uint8_t)0xFF) /* !< Interface address */
 */

/*
 * Bit definition for I2C_DR register *****
 * #define I2C_DR_DR ((uint8_t)0xFF) /* !<8-bit Data Register */
 */

/*
 * Bit definition for I2C_SR1 register *****
 * #define I2C_SR1_SB ((uint16_t)0x0001) /* !< Start Bit (Master mode)
 * #define I2C_SR1_ADDR ((uint16_t)0x0002) /* !< Address sent (master mode)/matched
 * #define I2C_SR1_BTF ((uint16_t)0x0004) /* !< Byte Transfer Finished */
 * #define I2C_SR1_ADD10 ((uint16_t)0x0008) /* !< 10-bit header sent (Master mode)
 * #define I2C_SR1_STOPF ((uint16_t)0x0010) /* !< Stop detection (Slave mode)
 * #define I2C_SR1_RXNE ((uint16_t)0x0040) /* !< Data Register not Empty (receivers)
 * #define I2C_SR1_TXE ((uint16_t)0x0080) /* !< Data Register Empty (transmitters)
 * #define I2C_SR1_BERR ((uint16_t)0x0100) /* !< Bus Error
 * #define I2C_SR1_ARLO ((uint16_t)0x0200) /* !< Arbitration Lost (master mode)
 * #define I2C_SR1_AF ((uint16_t)0x0400) /* !< Acknowledge Failure
 * #define I2C_SR1_OVR ((uint16_t)0x0800) /* !< Overrun/Underrun
 * #define I2C_SR1_PECERR ((uint16_t)0x1000) /* !< PEC Error in reception
 * #define I2C_SR1_TIMEOUT ((uint16_t)0x4000) /* !< Timeout or Tlow Error
 * #define I2C_SR1_SMBALERT ((uint16_t)0x8000) /* !< SMBus Alert
 */

/*
 * Bit definition for I2C_SR2 register *****
 * #define I2C_SR2_MSL ((uint16_t)0x0001) /* !< Master/Slave
 * #define I2C_SR2_BUSY ((uint16_t)0x0002) /* !< Bus Busy
 * #define I2C_SR2_TRA ((uint16_t)0x0004) /* !< Transmitter/Receiver
 * #define I2C_SR2_GENCALL ((uint16_t)0x0010) /* !< General Call Address (Slave mode)
 * #define I2C_SR2_SMBDEFAULT ((uint16_t)0x0020) /* !< SMBus Device Default Address (Slave mode)
 * #define I2C_SR2_SMBHOST ((uint16_t)0x0040) /* !< SMBus Host Header (Slave mode)
 * #define I2C_SR2_DUALF ((uint16_t)0x0080) /* !< Dual Flag (Slave mode)
 * #define I2C_SR2_PEC ((uint16_t)0xFFFF00) /* !< Packet Error Checking Register
 */

/*
 * Bit definition for I2C_CCR register *****
 * #define I2C_CCR_CCR ((uint16_t)0x0FFF) /* !< Clock Control Register in Fast/SI2C
 */

```

```

#define I2C_CCR_DUTY ((uint16_t)0x4000) /* !<Fast Mode Duty Cycle */
#define I2C_CCR_FS ((uint16_t)0x8000) /* !<I2C Master Mode Selection */

/* ***** Bit definition for I2C_TRISE register *****/
#define I2C_TRISE ((uint8_t)0x3F) /* !<Maximum Rise Time in Fast / Standard mode */

/*
 */
/*
***** Bit definition for I2C_TRISE register *****/
((uint8_t)0x3F) /* !<Maximum Rise Time in Fast / Standard mode */

/*
 */
/*
***** Bit definition for I2C_KR register *****/
((uint16_t)0xFFFF) /* !<Key value (write only, read 0000) */

/*
 */
/*
***** Bit definition for IWDG_PR register *****/
((uint8_t)0x07) /* !<PR[2:0] (Prescaler divider) */
((uint8_t)0x01) /* !<Bit 0 */
((uint8_t)0x02) /* !<Bit 1 */
((uint8_t)0x04) /* !<Bit 2 */

/*
 */
/*
***** Bit definition for IWDG_RLR register *****/
((uint16_t)0x0FFF) /* !<Watchdog counter reload value */

/*
 */
/*
***** Bit definition for IWDG_SR register *****/
((uint8_t)0x01) /* !<Watchdog prescaler value update */
((uint8_t)0x02) /* !<Watchdog counter reload value update */

/*
 */
/*
***** Bit definition for PWR_CR register *****/
((uint16_t)0x0001) /* !<Low-Power Deepsleep */
((uint16_t)0x0002) /* !<Power Down Deepsleep */

/*
 */
/*
***** Bit definition for PWR_CR register *****/
((uint16_t)0x0001) /* !<Low-Power Deepsleep */
((uint16_t)0x0002) /* !<Power Down Deepsleep */

```

---

```

#define PWR_CR_CWUF ((uint16_t)0x0004) /*!< Clear Wakeup Flag */
#define PWR_CR_CSBF ((uint16_t)0x0008) /*!< Clear Standby Flag */
#define PWR_CR_PVDE ((uint16_t)0x0010) /*!< Power Voltage Detector Enable */

#define PWR_CR_PLS ((uint16_t)0x00E0) /*!< PLS [2:0] bits (PVD Level Selection)
#define PWR_CR_PLS_0 ((uint16_t)0x0020) /*!< Bit 0 */
#define PWR_CR_PLS_1 ((uint16_t)0x0040) /*!< Bit 1 */
#define PWR_CR_PLS_2 ((uint16_t)0x0080) /*!< Bit 2 */

/* !< PVD level configuration */
#define PWR_CR_PLS_LEV0 ((uint16_t)0x0000) /*!< PVD level 0 */
#define PWR_CR_PLS_LEV1 ((uint16_t)0x0020) /*!< PVD level 1 */
#define PWR_CR_PLS_LEV2 ((uint16_t)0x0040) /*!< PVD level 2 */
#define PWR_CR_PLS_LEV3 ((uint16_t)0x0060) /*!< PVD level 3 */
#define PWR_CR_PLS_LEV4 ((uint16_t)0x0080) /*!< PVD level 4 */
#define PWR_CR_PLS_LEV5 ((uint16_t)0x00A0) /*!< PVD level 5 */
#define PWR_CR_PLS_LEV6 ((uint16_t)0x00C0) /*!< PVD level 6 */
#define PWR_CR_PLS_LEV7 ((uint16_t)0x00E0) /*!< PVD level 7 */

#define PWR_CR_DBP ((uint16_t)0x0100) /*!< Disable Backup Domain write protection */
#define PWR_CR_FPDS ((uint16_t)0x0200) /*!< Flash power down in Stop mode */
#define PWR_CR_VOS ((uint16_t)0x4000) /*!< Regulator voltage scaling output selection
#define PWR_CR_PMODE PWR_CR_VOS

/* ***** Bit definition for PWR_CSR register *****/
#define PWR_CSR_WUF ((uint16_t)0x0001) /*!< Wakeup Flag */
#define PWR_CSR_SBF ((uint16_t)0x0002) /*!< Standby Flag */
#define PWR_CSR_PVDO ((uint16_t)0x0004) /*!< PVD Output */
#define PWR_CSR_BRR ((uint16_t)0x0008) /*!< Backup regulator ready */
#define PWR_CSR_EWUP ((uint16_t)0x0100) /*!< Enable WKUP pin */
#define PWR_CSR_BRE ((uint16_t)0x0200) /*!< Backup regulator enable */
#define PWR_CSR_VOSRDY ((uint16_t)0x4000) /*!< Regulator voltage scaling output selection

/* Legacy define */
/* Legacy define */

```

---

---

```

#define PWR_CSR_REGRDY
/* *****
 *      Reset and Clock Control
 * *****
 *      Bit definition for RCC_CR register *****
 */
#define RCC_CR_HSION ((uint32_t)0x00000001)
#define RCC_CR_HSIRDY ((uint32_t)0x00000002)

#define RCC_CR_HSITRIM ((uint32_t)0x000000F8)
#define RCC_CR_HSICAL_0 ((uint32_t)0x00000008)/* !<Bit 0 */
#define RCC_CR_HSICAL_1 ((uint32_t)0x00000010)/* !<Bit 1 */
#define RCC_CR_HSICAL_2 ((uint32_t)0x00000020)/* !<Bit 2 */
#define RCC_CR_HSICAL_3 ((uint32_t)0x00000040)/* !<Bit 3 */
#define RCC_CR_HSICAL_4 ((uint32_t)0x00000080)/* !<Bit 4 */

#define RCC_CR_HSICAL_0 ((uint32_t)0x00000100)/* !<Bit 0 */
#define RCC_CR_HSICAL_1 ((uint32_t)0x00000200)/* !<Bit 1 */
#define RCC_CR_HSICAL_2 ((uint32_t)0x00000400)/* !<Bit 2 */
#define RCC_CR_HSICAL_3 ((uint32_t)0x00000800)/* !<Bit 3 */
#define RCC_CR_HSICAL_4 ((uint32_t)0x00001000)/* !<Bit 4 */
#define RCC_CR_HSICAL_5 ((uint32_t)0x00002000)/* !<Bit 5 */
#define RCC_CR_HSICAL_6 ((uint32_t)0x00004000)/* !<Bit 6 */
#define RCC_CR_HSICAL_7 ((uint32_t)0x00008000)/* !<Bit 7 */

#define RCC_CR_HSEON ((uint32_t)0x000010000)
#define RCC_CR_HSERDY ((uint32_t)0x00020000)
#define RCC_CR_HSEBYP ((uint32_t)0x00040000)
#define RCC_CR_CSSEN ((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)

/* ***** Bit definition for RCC_PLLCFG register **** */
#define RCC_PLLCFG_PLLM ((uint32_t)0x0000003F)
#define RCC_PLLCFG_PLLM_0 ((uint32_t)0x00000001)
#define RCC_PLLCFG_PLLM_1 ((uint32_t)0x00000002)
#define RCC_PLLCFG_PLLM_2 ((uint32_t)0x00000004)
#define RCC_PLLCFG_PLLM_3 ((uint32_t)0x00000008)
#define RCC_PLLCFG_PLLM_4 ((uint32_t)0x00000010)
#define RCC_PLLCFG_PLLM_5 ((uint32_t)0x00000020)
#define RCC_PLLCFG_PLLN ((uint32_t)0x000007FC0)
#define RCC_PLLCFG_PLLN_0 ((uint32_t)0x00000040)
#define RCC_PLLCFG_PLLN_1 ((uint32_t)0x00000080)
#define RCC_PLLCFG_PLLN_2 ((uint32_t)0x00000100)
#define RCC_PLLCFG_PLLN_3 ((uint32_t)0x00000200)
#define RCC_PLLCFG_PLLN_4 ((uint32_t)0x00000400)
#define RCC_PLLCFG_PLLN_5 ((uint32_t)0x00000800)
#define RCC_PLLCFG_PLLN_6 ((uint32_t)0x00001000)
#define RCC_PLLCFG_PLLN_7 ((uint32_t)0x00002000)
#define RCC_PLLCFG_PLLN_8 ((uint32_t)0x00004000)

#define RCC_PLLCFG_PLLP ((uint32_t)0x00030000)
#define RCC_PLLCFG_PLLP_0 ((uint32_t)0x00010000)
#define RCC_PLLCFG_PLLP_1 ((uint32_t)0x00020000)

#define RCC_PLLCFG_PLLSRC ((uint32_t)0x00040000)
#define RCC_PLLCFG_PLLSRC_HSE ((uint32_t)0x00040000)
#define RCC_PLLCFG_PLLSRC_HSI ((uint32_t)0x00000000)

#define RCC_PLLCFG_PLLQ ((uint32_t)0x00F00000)
#define RCC_PLLCFG_PLLQ_0 ((uint32_t)0x01000000)
#define RCC_PLLCFG_PLLQ_1 ((uint32_t)0x02000000)
#define RCC_PLLCFG_PLLQ_2 ((uint32_t)0x04000000)
#define RCC_PLLCFG_PLLQ_3 ((uint32_t)0x08000000)

```

---

```

/*
 * < SW configuration */
#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)

/* !< SWS configuration */
#define RCC_CFGR_SWS ((uint32_t)0x0000000C)
#define RCC_CFGR_SWS_0 ((uint32_t)0x00000004)
#define RCC_CFGR_SWS_8 ((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)

/* !< HPRE configuration */
#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_DIV32 ((uint32_t)0x000000C0)
#define RCC_CFGR_HPRE_DIV64 ((uint32_t)0x000000D0)
#define RCC_CFGR_HPRE_DIV128 ((uint32_t)0x000000E0)
#define RCC_CFGR_HPRE_DIV256 ((uint32_t)0x000000F0)

/* !< SYSCLK not divided */
#define RCC_CFGR_HPRE_DIV2 ((uint32_t)0x00000002)
#define RCC_CFGR_HPRE_DIV4 ((uint32_t)0x00000004)
#define RCC_CFGR_HPRE_DIV8 ((uint32_t)0x00000008)
#define RCC_CFGR_HPRE_DIV16 ((uint32_t)0x00000016)
#define RCC_CFGR_HPRE_DIV32 ((uint32_t)0x00000064)
#define RCC_CFGR_HPRE_DIV64 ((uint32_t)0x000000128)
#define RCC_CFGR_HPRE_DIV256 ((uint32_t)0x000000256)

```

```

/* !< SYSCLK divided by 512 */
((uint32_t)0x0000000F0)

/* !< RCC_CFGR_HPRE_DIV12
#define RCC_CFGR_HPRE1
#define RCC_CFGR_PPREG1
#define RCC_CFGR_PPREG1_0
#define RCC_CFGR_PPREG1_1
#define RCC_CFGR_PPREG1_2

#define RCC_CFGR_PPREG1_DIV1
#define RCC_CFGR_PPREG1_DIV2
#define RCC_CFGR_PPREG1_DIV4
#define RCC_CFGR_PPREG1_DIV8
#define RCC_CFGR_PPREG1_DIV16

/* !< RCC_CFGR_PPREG2
#define RCC_CFGR_PPREG2
#define RCC_CFGR_PPREG2_0
#define RCC_CFGR_PPREG2_1
#define RCC_CFGR_PPREG2_2

#define RCC_CFGR_PPREG2_DIV1
#define RCC_CFGR_PPREG2_DIV2
#define RCC_CFGR_PPREG2_DIV4
#define RCC_CFGR_PPREG2_DIV8
#define RCC_CFGR_PPREG2_DIV16

/* !< RCC_CFGR_RTCPRE
#define RCC_CFGR_RTCPRE
#define RCC_CFGR_RTCPRE_0
#define RCC_CFGR_RTCPRE_1
#define RCC_CFGR_RTCPRE_2
#define RCC_CFGR_RTCPRE_3
#define RCC_CFGR_RTCPRE_4

/* !< MC01 configuration */
/* !< prescaler */
/* !< PRE1[2:0] bits (APB1 prescaler) */
(((uint32_t)0x000001C00)
((uint32_t)0x00000400)
((uint32_t)0x00000800)
((uint32_t)0x00001000)

((uint32_t)0x00000000)
((uint32_t)0x00001000)
((uint32_t)0x00001400)
((uint32_t)0x00001800)
((uint32_t)0x00001C00)

((uint32_t)0x0000E000)
((uint32_t)0x00002000)
((uint32_t)0x00004000)
((uint32_t)0x00008000)

((uint32_t)0x00000000)
((uint32_t)0x00008000)
((uint32_t)0x0000A000)
((uint32_t)0x0000C000)
((uint32_t)0x0000E000)

((uint32_t)0x0001F0000)
((uint32_t)0x00010000)
((uint32_t)0x00020000)
((uint32_t)0x00040000)
((uint32_t)0x00080000)
((uint32_t)0x00100000)

```

```

#define RCC_CFGR_MC01 ((uint32_t)0x00060000)
#define RCC_CFGR_MC01_0 ((uint32_t)0x00020000)
#define RCC_CFGR_MC01_1 ((uint32_t)0x00040000)

#define RCC_CFGR_I2SSRC ((uint32_t)0x00800000)

#define RCC_CFGR_MC01PRE ((uint32_t)0x07000000)
#define RCC_CFGR_MC01PRE_0 ((uint32_t)0x01000000)
#define RCC_CFGR_MC01PRE_1 ((uint32_t)0x02000000)
#define RCC_CFGR_MC01PRE_2 ((uint32_t)0x04000000)

#define RCC_CFGR_MC02PRE ((uint32_t)0x38000000)
#define RCC_CFGR_MC02PRE_0 ((uint32_t)0x08000000)
#define RCC_CFGR_MC02PRE_1 ((uint32_t)0x10000000)
#define RCC_CFGR_MC02PRE_2 ((uint32_t)0x20000000)

#define RCC_CFGR_MC02 ((uint32_t)0xC0000000)
#define RCC_CFGR_MC02_0 ((uint32_t)0x40000000)
#define RCC_CFGR_MC02_1 ((uint32_t)0x80000000)

/**************** Bit definition for RCC_CIR register *****/
#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)0x0000200000)
#define RCC_CIR_HSIRDYC ((uint32_t)0x0000400000)
#define RCC_CIR_HSERDYC ((uint32_t)0x0000800000)
#define RCC_CIR_PLLRDYC ((uint32_t)0x00100000)
#define RCC_CIR_PLLI2SRDYC ((uint32_t)0x00200000)
#define RCC_CIR_CSSC ((uint32_t)0x00800000)

/****** Bit definition for RCC_AHB1RSTR register *****/
#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)

/****** Bit definition for RCC_AHB2RSTR register *****/
#define RCC_AHB2RSTR_DCMIRST ((uint32_t)0x00000001)
#define RCC_AHB2RSTR_CRYPRST ((uint32_t)0x00000010)
#define RCC_AHB2RSTR_HASHRST ((uint32_t)0x00000020)

/* maintained for legacy purpose */
#define RCC_AHB2RSTR_HSABRST
#define RCC_AHB2RSTR_RNGRST ((uint32_t)0x00000040)
#define RCC_AHB2RSTR_OTGFSRST ((uint32_t)0x00000080)

/****** Bit definition for RCC_AHB3RSTR register *****/
#define RCC_AHB3RSTR_FSMCRST ((uint32_t)0x00000001)

```

```

/*
 * Bit definition for RCC_APB1RSTR register *****/
#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)0x00000800)
#define RCC_APB1RSTR_SPI3RST ((uint32_t)0x00010000)
#define RCC_APB1RSTR_USART2RST ((uint32_t)0x00020000)
#define RCC_APB1RSTR_USART3RST ((uint32_t)0x00040000)
#define RCC_APB1RSTR_USART4RST ((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)0x00200000)
#define RCC_APB1RSTR_CAN2RST ((uint32_t)0x00400000)
#define RCC_APB1RSTR_PWR_RST ((uint32_t)0x10000000)
#define RCC_APB1RSTR_DAC_RST ((uint32_t)0x20000000)

/*
 * Bit definition for RCC_APB2RSTR register *****/
#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_SYSCFG_RST ((uint32_t)0x00004000)
#define RCC_APB2RSTR_TIM9RST ((uint32_t)0x00010000)

```

```

#define RCC_APB2RSTR_TIM1ORST ((uint32_t)0x0000200000)
#define RCC_APB2RSTR_TIM11RST ((uint32_t)0x0000400000)
/* Old SPI1RST bit definition, maintained for legacy purpose */
#define RCC_APB2RSTR_SPI1RST

/******************* Bit definition for RCC_AHB1ENR register *****/
#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_CCMDATAARAMEN ((uint32_t)0x00100000)
#define RCC_AHB1ENR_DMA1EN ((uint32_t)0x00200000)
#define RCC_AHB1ENR_DMA2EN ((uint32_t)0x00400000)
#define RCC_AHB1ENR_ETHMACEN ((uint32_t)0x00200000)
#define RCC_AHB1ENR_ETHMACTXEN ((uint32_t)0x00400000)
#define RCC_AHB1ENR_ETHMACRXEN ((uint32_t)0x00800000)
#define RCC_AHB1ENR_ETHMACPTPEN ((uint32_t)0x10000000)
#define RCC_AHB1ENR_OTGHSEN ((uint32_t)0x20000000)
#define RCC_AHB1ENR_OTGHSULPIEN ((uint32_t)0x40000000)

/******************* Bit definition for RCC_AHB2ENR register *****/
#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)

/******************* Bit definition for RCC_AHB3ENR register *****/

```

```

#define RCC_AHB3ENR_FSMCEN ((uint32_t)0x00000001)

/* ***** Bit definition for RCC_APB1ENR register *****/
#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))

/* ***** Bit definition for RCC_APB2ENR register *****/
#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)0x000000800)
#define RCC_APB2ENR_SPI1EN ((uint32_t)0x000001000)
#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)

/* ***** Bit definition for RCC_AHB1LPEN register *****/
#define RCC_AHB1LPEN_GPIOALPEN ((uint32_t)0x00000001)
#define RCC_AHB1LPEN_GPIOBLPEN ((uint32_t)0x00000002)
#define RCC_AHB1LPEN_GPIOCLPEN ((uint32_t)0x00000004)
#define RCC_AHB1LPEN_GPIODLPEN ((uint32_t)0x00000008)
#define RCC_AHB1LPEN_GPIOELPEN ((uint32_t)0x00000010)
#define RCC_AHB1LPEN_GPIOFLPEN ((uint32_t)0x00000020)
#define RCC_AHB1LPEN_GPIOGLPEN ((uint32_t)0x00000040)
#define RCC_AHB1LPEN_GPIOHLPEN ((uint32_t)0x00000080)
#define RCC_AHB1LPEN_GPIOILPEN ((uint32_t)0x00000100)
#define RCC_AHB1LPEN_CRCLPEN ((uint32_t)0x00001000)
#define RCC_AHB1LPEN_FLITFLPEN ((uint32_t)0x00008000)
#define RCC_AHB1LPEN_SRAM1LPEN ((uint32_t)0x00010000)
#define RCC_AHB1LPEN_SRAM2LPEN ((uint32_t)0x00020000)
#define RCC_AHB1LPEN_BKPSRAMLPEN ((uint32_t)0x00040000)
#define RCC_AHB1LPEN_DMA1LPEN ((uint32_t)0x00200000)
#define RCC_AHB1LPEN_DMA2LPEN ((uint32_t)0x00400000)
#define RCC_AHB1LPEN_ETHMACLPEN ((uint32_t)0x00800000)
#define RCC_AHB1LPEN_ETHMACTXLPEN ((uint32_t)0x01000000)
#define RCC_AHB1LPEN_ETHMACRXLPEN ((uint32_t)0x02000000)
#define RCC_AHB1LPEN_ETHMACPTPLPEN ((uint32_t)0x04000000)
#define RCC_AHB1LPEN_OTGHSLIPEN ((uint32_t)0x08000000)
#define RCC_AHB1LPEN_OTGHSULPILPEN ((uint32_t)0x10000000)

/* ***** Bit definition for RCC_AHB2LPEN register *****/
#define RCC_AHB2LPEN_DCMILPEN ((uint32_t)0x00000001)
#define RCC_AHB2LPEN_CRYPLPEN ((uint32_t)0x00000010)
#define RCC_AHB2LPEN_HASHLPEN ((uint32_t)0x00000020)

```

---

```

#define RCC_AHB2LPENR_RNGLPEN ((uint32_t)0x000000040)
#define RCC_AHB2LPENR_OTGFSLPEN ((uint32_t)0x000000080)

#define RCC_AHB3LPENR_FSMCLPEN ((uint32_t)0x000000001)

/* ***** Bit definition for RCC_AHB3LPENR register *****/
#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_USART4LPEN ((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)0x00200000)
#define RCC_APB1LPENR_CAN2LPEN ((uint32_t)0x00400000)
#define RCC_APB1LPENR_PWRLPEN ((uint32_t)0x10000000)
#define RCC_APB1LPENR_DACLPEN ((uint32_t)0x20000000)

/* ***** Bit definition for RCC_APB2LPENR register *****/
#define RCC_APB2LPENR_TIM1LPEN ((uint32_t)0x00000001)
#define RCC_APB2LPENR_TIM8LPEN ((uint32_t)0x00000002)
#define RCC_APB2LPENR_USART1LPEN ((uint32_t)0x00000010)

```

---

```

#define RCC_APB2LPEN_USART6LPEN ((uint32_t)0x000000020)
#define RCC_APB2LPEN_ADC1LPEN ((uint32_t)0x000000100)
#define RCC_APB2LPEN_ADC2LPEN ((uint32_t)0x000000200)
#define RCC_APB2LPEN_ADC3LPEN ((uint32_t)0x000000400)
#define RCC_APB2LPEN_SDIOLPEN ((uint32_t)0x000000800)
#define RCC_APB2LPEN_SPI1LPEN ((uint32_t)0x00001000)
#define RCC_APB2LPEN_SPI2LPEN ((uint32_t)0x00004000)
#define RCC_APB2LPEN_SYSCFGLPEN ((uint32_t)0x000010000)
#define RCC_APB2LPEN_TIM9LPEN ((uint32_t)0x000010000)
#define RCC_APB2LPEN_TIM10LPEN ((uint32_t)0x000020000)
#define RCC_APB2LPEN_TIM11LPEN ((uint32_t)0x000040000)

/****** Bit definition for RCC_BDCR register *****/
#define RCC_BDCR_LSEON ((uint32_t)0x000000001)
#define RCC_BDCR_LSERDY ((uint32_t)0x000000002)
#define RCC_BDCR_LSEBYP ((uint32_t)0x000000004)

#define RCC_BDCR_RTCSEL ((uint32_t)0x000000300)
#define RCC_BDCR_RTCSEL_0 ((uint32_t)0x000000100)
#define RCC_BDCR_RTCSEL_1 ((uint32_t)0x000000200)

#define RCC_BDCR_RTCEN ((uint32_t)0x0000008000)
#define RCC_BDCR_BDRST ((uint32_t)0x000010000)

/****** Bit definition for RCC_CSR register *****/
#define RCC_CSR_LSION ((uint32_t)0x000000001)
#define RCC_CSR_LSIRDY ((uint32_t)0x000000002)
#define RCC_CSR_RMVF ((uint32_t)0x010000000)
#define RCC_CSR_BORRSTF ((uint32_t)0x020000000)
#define RCC_CSR_PADRSTF ((uint32_t)0x040000000)
#define RCC_CSR_PORRSTF ((uint32_t)0x080000000)
#define RCC_CSR_SFTRSTF ((uint32_t)0x100000000)
#define RCC_CSR_WDGRSTF ((uint32_t)0x200000000)
#define RCC_CSR_WWDGRSTF ((uint32_t)0x400000000)
#define RCC_CSR_LPWRSTF ((uint32_t)0x800000000)

```

```

/*
 * Bit definition for RCC_SSCGR register *****
 *define RCC_SSCGR_MODPER ((uint32_t)0x00001FFF)
 *define RCC_SSCGR_INSTEP ((uint32_t)0x0FFE000)
 *define RCC_SSCGR_SPREADSEL ((uint32_t)0x40000000)
 #define RCC_SSCGR_SSCGEN ((uint32_t)0x80000000)

/*
 * Bit definition for RCC_PLLI2SCFGR register *****
#define RCC_PLLI2SCFGR_PLLI2SN ((uint32_t)0x00007FC0)
#define RCC_PLLI2SCFGR_PLLI2SR ((uint32_t)0x70000000)

/*
 * Bits definition for RNG_CR register *****
#define RNG_CR_RNGEN ((uint32_t)0x00000004)
#define RNG_CR_IE ((uint32_t)0x00000008)

/*
 * Bits definition for RNG_SR register *****
#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)

/*
 * Bits definition for RTC_TR register *****
#define RTC_TR_PM ((uint32_t)0x00400000)
#define RTC_TR_HT ((uint32_t)0x00300000)
#define RTC_TR_HT_0 ((uint32_t)0x00100000)

/*
 * Real-Time Clock (RTC) *****
*/

```

---

```

#define RTC_TR_HT_1 ((uint32_t)0x0002000000)
#define RTC_TR_HU_0 ((uint32_t)0x000F0000)
#define RTC_TR_HU_1 ((uint32_t)0x00010000)
#define RTC_TR_HU_2 ((uint32_t)0x00020000)
#define RTC_TR_HU_3 ((uint32_t)0x00040000)
#define RTC_TR_MNT ((uint32_t)0x00007000)
#define RTC_TR_MNU_0 ((uint32_t)0x00001000)
#define RTC_TR_MNT_1 ((uint32_t)0x00002000)
#define RTC_TR_MNT_2 ((uint32_t)0x00004000)
#define RTC_TR_MNU_0 ((uint32_t)0x0000F000)
#define RTC_TR_MNU_1 ((uint32_t)0x0000100)
#define RTC_TR_MNU_2 ((uint32_t)0x0000200)
#define RTC_TR_MNU_3 ((uint32_t)0x0000400)
#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)

***** Bits definition for RTC_DR register *****
#define RTC_DR_YT ((uint32_t)0x000F0000)
#define RTC_DR_YT_0 ((uint32_t)0x00010000)
#define RTC_DR_YT_1 ((uint32_t)0x00020000)
#define RTC_DR_YT_2 ((uint32_t)0x00040000)
#define RTC_DR_YT_3 ((uint32_t)0x00080000)
#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)0x0000800000)
#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)0x0000F000)
#define RTC_DR_MU_0 ((uint32_t)0x00001000)
#define RTC_DR_MU_1 ((uint32_t)0x00002000)
#define RTC_DR_MU_2 ((uint32_t)0x00004000)
#define RTC_DR_MU_3 ((uint32_t)0x00008000)
#define RTC_DR_DT ((uint32_t)0x000030)
#define RTC_DR_DT_0 ((uint32_t)0x00000100)
#define RTC_DR_DT_1 ((uint32_t)0x00000020)
#define RTC_DR_DU ((uint32_t)0x000000F0)
#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)

/**************** Bits definition for RTC_CR register *****/
#define RTC_CR_COE ((uint32_t)0x00080000)
#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)0x000000400)
#define RTC_CR_ALRBE ((uint32_t)0x000000200)
#define RTC_CR_ALRAE ((uint32_t)0x000000100)
#define RTC_CR_DCE ((uint32_t)0x000000080)
#define RTC_CR_FMT ((uint32_t)0x000000040)
#define RTC_CR_BYPSHAD ((uint32_t)0x000000020)
#define RTC_CR_REFCKON ((uint32_t)0x000000010)
#define RTC_CR_TSEDGE ((uint32_t)0x000000008)
#define RTC_CR_WUCKSEL ((uint32_t)0x000000007)
#define RTC_CR_WUCKSEL_0 ((uint32_t)0x000000001)
#define RTC_CR_WUCKSEL_1 ((uint32_t)0x000000002)
#define RTC_CR_WUCKSEL_2 ((uint32_t)0x000000004)

/****** Bits definition for RTC_ISR register *****/
#define RTC_ISR_RECALPF ((uint32_t)0x000010000)
#define RTC_ISR_TAMP1F ((uint32_t)0x000002000)
#define RTC_ISR_TS0VF ((uint32_t)0x000001000)
#define RTC_ISR_TS0F ((uint32_t)0x000000800)
#define RTC_ISR_WUTF ((uint32_t)0x000000400)
#define RTC_ISR_ALRBFF ((uint32_t)0x000000200)
#define RTC_ISR_ALRAF ((uint32_t)0x000000100)
#define RTC_ISR_INIT ((uint32_t)0x000000080)
#define RTC_ISR_INITF ((uint32_t)0x000000040)
#define RTC_ISR_RSF ((uint32_t)0x000000020)
#define RTC_ISR_INITS ((uint32_t)0x000000010)
#define RTC_ISR_SHPF ((uint32_t)0x000000008)
#define RTC_ISR_WUTWF ((uint32_t)0x000000004)
#define RTC_ISR_ALRBWF ((uint32_t)0x000000002)
#define RTC_ISR_ALRAWF ((uint32_t)0x000000001)

/****** Bits definition for RTC_PRER register *****/
#define RTC_PRER_PREDIV_A ((uint32_t)0x0007F0000)
#define RTC_PRER_PREDIV_S ((uint32_t)0x000001FFF)

/****** Bits definition for RTC_WUTR register *****/

```

---

---

```

#define RTC_WUTR_WUT          ((uint32_t)0x00000FFF)

/* **** Bits definition for RTC_CALIBR register **** */
#define RTC_CALIBR_DCS         ((uint32_t)0x80000000)
#define RTC_CALIBR_DC          ((uint32_t)0x40000000)
#define RTC_CALIBR_WDSEL       ((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_HU_0        ((uint32_t)0x00100000)
#define RTC_ALRMAR_HU_1        ((uint32_t)0x00200000)
#define RTC_ALRMAR_HU_2        ((uint32_t)0x00400000)
#define RTC_ALRMAR_HU_3        ((uint32_t)0x00800000)
#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)0x0000F000)
#define RTC_ALRMAR_MNU_0        ((uint32_t)0x00001000)
#define RTC_ALRMAR_MNU_1        ((uint32_t)0x00002000)

```

---

---

```

#define RTC_ALRMAR_MNU_2 ((uint32_t)0x000000400)
#define RTC_ALRMAR_MNU_3 ((uint32_t)0x000000800)
#define RTC_ALRMAR_MSK1 ((uint32_t)0x00000080)
#define RTC_ALRMAR_ST ((uint32_t)0x000000070)
#define RTC_ALRMAR_ST_0 ((uint32_t)0x000000010)
#define RTC_ALRMAR_ST_1 ((uint32_t)0x000000020)
#define RTC_ALRMAR_ST_2 ((uint32_t)0x000000040)
#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)

/**************** Bits definition for RTC_ALRMBR register *****/
#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)0x00F00000)
#define RTC_ALRMBR_HU_0 ((uint32_t)0x00100000)
#define RTC_ALRMBR_HU_1 ((uint32_t)0x00200000)
#define RTC_ALRMBR_HU_2 ((uint32_t)0x00400000)
#define RTC_ALRMBR_HU_3 ((uint32_t)0x00800000)
#define RTC_ALRMBR_MSK2 ((uint32_t)0x00008000)

```

---

---

```

#define RTC_ALRMBR_MNT ((uint32_t)0x0000007000)
#define RTC_ALRMBR_MNT_0 ((uint32_t)0x000001000)
#define RTC_ALRMBR_MNT_1 ((uint32_t)0x000002000)
#define RTC_ALRMBR_MNT_2 ((uint32_t)0x000004000)
#define RTC_ALRMBR_MNU ((uint32_t)0x00000F000)
#define RTC_ALRMBR_MNU_0 ((uint32_t)0x00001000)
#define RTC_ALRMBR_MNU_1 ((uint32_t)0x00002000)
#define RTC_ALRMBR_MNU_2 ((uint32_t)0x00004000)
#define RTC_ALRMBR_MNU_3 ((uint32_t)0x00008000)
#define RTC_ALRMBR_MSK1 ((uint32_t)0x00000800)
#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)

/**************** Bits definition for RTC_WPR register *****/
#define RTC_WPR_KEY ((uint32_t)0x000000FF)

/**************** Bits definition for RTC_SSR register *****/
#define RTC_SSR_SS ((uint32_t)0x0000FFFF)

/**************** Bits definition for RTC_SHIFTR register *****/
#define RTC_SHIFTR_SUBFS ((uint32_t)0x000007FFF)
#define RTC_SHIFTR_ADD1S ((uint32_t)0x80000000)

/**************** Bits definition for RTC_TSTR register *****/
#define RTC_TSTR_PM ((uint32_t)0x000400000)
#define RTC_TSTR_HT ((uint32_t)0x00300000)
#define RTC_TSTR_HT_0 ((uint32_t)0x00100000)
#define RTC_TSTR_HT_1 ((uint32_t)0x00200000)

/**************** Bits definition for RTC_WPR register *****/
#define RTC_WPR_KEY ((uint32_t)0x000000FF)

```

---

---

```

#define RTC_TSTR_HU ((uint32_t)0x00000000)
#define RTC_TSTR_HU_0 ((uint32_t)0x00001000)
#define RTC_TSTR_HU_1 ((uint32_t)0x00002000)
#define RTC_TSTR_HU_2 ((uint32_t)0x00004000)
#define RTC_TSTR_HU_3 ((uint32_t)0x00008000)
#define RTC_TSTR_MNT ((uint32_t)0x00000100)
#define RTC_TSTR_MNT_0 ((uint32_t)0x00000200)
#define RTC_TSTR_MNT_1 ((uint32_t)0x00000700)
#define RTC_TSTR_MNT_2 ((uint32_t)0x00001000)
#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)

/****** Bits definition for RTC_TSDR register *****/
#define RTC_TSDR_WDU ((uint32_t)0x00000E00)
#define RTC_TSDR_WDU_0 ((uint32_t)0x00000200)
#define RTC_TSDR_WDU_1 ((uint32_t)0x00000400)
#define RTC_TSDR_WDU_2 ((uint32_t)0x00000800)
#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)0x000000030)
#define RTC_TSDR_DT_0 ((uint32_t)0x000000010)
#define RTC_TSDR_DT_1 ((uint32_t)0x000000020)
#define RTC_TSDR_DT_2 ((uint32_t)0x0000000F0)
#define RTC_TSDR_DT_3 ((uint32_t)0x00000000F)
#define RTC_TSDR_DU ((uint32_t)0x000000001)
#define RTC_TSDR_DU_0 ((uint32_t)0x000000001)
#define RTC_TSDR_DU_1 ((uint32_t)0x000000002)
#define RTC_TSDR_DU_2 ((uint32_t)0x000000004)
#define RTC_TSDR_DU_3 ((uint32_t)0x000000008)

/* Bits definition for RTC_TSSSR register *****/
#define RTC_TSSSR_SS

/* Bits definition for RTC_CALR register *****/
#define RTC_CALR_CALP ((uint32_t)0x000008000)
#define RTC_CALR_CALW8 ((uint32_t)0x000004000)
#define RTC_CALR_CALW16 ((uint32_t)0x000002000)
#define RTC_CALR_CALM ((uint32_t)0x0000001FF)
#define RTC_CALR_CALM_0 ((uint32_t)0x000000001)
#define RTC_CALR_CALM_1 ((uint32_t)0x000000002)
#define RTC_CALR_CALM_2 ((uint32_t)0x000000004)
#define RTC_CALR_CALM_3 ((uint32_t)0x000000008)
#define RTC_CALR_CALM_4 ((uint32_t)0x000000010)
#define RTC_CALR_CALM_5 ((uint32_t)0x000000020)
#define RTC_CALR_CALM_6 ((uint32_t)0x000000040)
#define RTC_CALR_CALM_7 ((uint32_t)0x000000080)
#define RTC_CALR_CALM_8 ((uint32_t)0x000000100)

/* Bits definition for RTC_TAFCR register *****/
#define RTC_ALARMOUTTYPE_E ((uint32_t)0x000040000)
#define RTC_TSINSEL ((uint32_t)0x000020000)
#define RTC_TAMPINSEL ((uint32_t)0x000100000)
#define RTC_TAMPPUDIS ((uint32_t)0x000008000)
#define RTC_TAMPPRCH ((uint32_t)0x000006000)
#define RTC_TAMPPRCH_0 ((uint32_t)0x000002000)
#define RTC_TAMPPRCH_1 ((uint32_t)0x000004000)

```

---

```

#define RTC_TAFCR_TAMPFLT ((uint32_t)0x000001800)
#define RTC_TAFCR_TAMPFLT_0 ((uint32_t)0x000000800)
#define RTC_TAFCR_TAMPFLT_1 ((uint32_t)0x000001000)
#define RTC_TAFCR_TAMPFLT_2 ((uint32_t)0x00000700)
#define RTC_TAFCR_TAMPFREQ ((uint32_t)0x000000100)
#define RTC_TAFCR_TAMPFREQ_0 ((uint32_t)0x000000200)
#define RTC_TAFCR_TAMPFREQ_1 ((uint32_t)0x000000400)
#define RTC_TAFCR_TAMPFREQ_2 ((uint32_t)0x000000800)
#define RTC_TAFCR_TAMPFREQ_3 ((uint32_t)0x00000004)
#define RTC_TAFCR_TAMPFREQ_4 ((uint32_t)0x00000002)
#define RTC_TAFCR_TAMPFREQ_5 ((uint32_t)0x00000001)

/* ***** Bits definition for RTC_ALRMASSR register *****/
#define RTC_ALRMASSR_MASKSS ((uint32_t)0x00F00000)
#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)0x000007FFF)

/* ***** Bits definition for RTC_ALRMBSRR register *****/
#define RTC_ALRMBSRR_MASKSS ((uint32_t)0x00F00000)
#define RTC_ALRMBSRR_MASKSS_0 ((uint32_t)0x01000000)
#define RTC_ALRMBSRR_MASKSS_1 ((uint32_t)0x02000000)
#define RTC_ALRMBSRR_MASKSS_2 ((uint32_t)0x04000000)
#define RTC_ALRMBSRR_MASKSS_3 ((uint32_t)0x08000000)
#define RTC_ALRMBSRR_SS ((uint32_t)0x000007FFF)

/* ***** Bits definition for RTC_BKPOR register *****/
#define RTC_BKPOR ((uint32_t)0xFFFFFFF)

/* ***** Bits definition for RTC_BKP1R register *****/
#define RTC_BKP1R ((uint32_t)0xFFFFFFF)

/* ***** Bits definition for RTC_BKP2R register *****/

```

```

#define RTC_BKP2R ((uint32_t)0xFFFFFFFF)
/* **** Bits definition for RTC_BKP3R register **** */
#define RTC_BKP3R ((uint32_t)0xFFFFFFFF)

/* **** Bits definition for RTC_BKP4R register **** */
#define RTC_BKP4R ((uint32_t)0xFFFFFFFF)

/* **** Bits definition for RTC_BKP5R register **** */
#define RTC_BKP5R ((uint32_t)0xFFFFFFFF)

/* **** Bits definition for RTC_BKP6R register **** */
#define RTC_BKP6R ((uint32_t)0xFFFFFFFF)

/* **** Bits definition for RTC_BKP7R register **** */
#define RTC_BKP7R ((uint32_t)0xFFFFFFFF)

/* **** Bits definition for RTC_BKP8R register **** */
#define RTC_BKP8R ((uint32_t)0xFFFFFFFF)

/* **** Bits definition for RTC_BKP9R register **** */
#define RTC_BKP9R ((uint32_t)0xFFFFFFFF)

/* **** Bits definition for RTC_BKP10R register **** */
#define RTC_BKP10R ((uint32_t)0xFFFFFFFF)

/* **** Bits definition for RTC_BKP11R register **** */
#define RTC_BKP11R ((uint32_t)0xFFFFFFFF)

/* **** Bits definition for RTC_BKP12R register **** */
#define RTC_BKP12R ((uint32_t)0xFFFFFFFF)

/* **** Bits definition for RTC_BKP13R register **** */
#define RTC_BKP13R ((uint32_t)0xFFFFFFFF)

```

```

/*
 * Bits definition for RTC_BKP14R register *****
 * #define RTC_BKP14R ((uint32_t)0xFFFFFFFF)
 */

/*
 * Bits definition for RTC_BKP15R register *****
 * #define RTC_BKP15R ((uint32_t)0xFFFFFFFF)
 */

/*
 * Bits definition for RTC_BKP16R register *****
 * #define RTC_BKP16R ((uint32_t)0xFFFFFFFF)
 */

/*
 * Bits definition for RTC_BKP17R register *****
 * #define RTC_BKP17R ((uint32_t)0xFFFFFFFF)
 */

/*
 * Bits definition for RTC_BKP18R register *****
 * #define RTC_BKP18R ((uint32_t)0xFFFFFFFF)
 */

/*
 * Bits definition for RTC_BKP19R register *****
 * #define RTC_BKP19R ((uint32_t)0xFFFFFFFF)
 */

/*
 * SD host Interface *****
 */

/*
 * Bit definition for SDIO_POWER register *****
 * #define SDIO_POWER_PWRCTRL ((uint8_t)0x03) /* !<PWRCTRL[1:0] bits (Power supply control)
 * #define SDIO_POWER_PWRCTRL_O ((uint8_t)0x01) /* !<Bit 0 */
 * #define SDIO_POWER_PWRCTRL_L ((uint8_t)0x02) /* !<Bit 1 */
 */

/*
 * Bit definition for SDIO_CLKCR register *****
 * #define SDIO_CLKCR_CLKDIV ((uint16_t)0x00FF) /* !<Clock divide factor */
 * #define SDIO_CLKCR_CLKEN ((uint16_t)0x0100) /* !<Clock enable bit */
 * #define SDIO_CLKCR_PWRSAV ((uint16_t)0x0200) /* !<Power saving configuration bit */
 * #define SDIO_CLKCR_BYPASS ((uint16_t)0x0400) /* !<Clock divider bypass enable bit */
 */

/*
 * WIDBUS[1:0] bits (Wide bus mode enable)
 * #define SDIO_CLKCR_WIDBUS ((uint16_t)0x1800) /* !<WIDBUS[1:0] bits (Wide bus mode enable)
 */

```

```

#define SDIO_CLKCR_WIDBUS_0 ((uint16_t)0x0800) /* !<Bit 0 */
#define SDIO_CLKCR_WIDBUS_1 ((uint16_t)0x1000) /* !<Bit 1 */

#define SDIO_CLKCR_NEGEDGE ((uint16_t)0x2000) /* !<SDIO_CK dephasing selection bit */
#define SDIO_CLKCR_HWFCE_N ((uint16_t)0x4000) /* !<HW Flow Control enable */

/* ***** Bit definition for SDIO_ARG register *****/
#define SDIO_ARG_CMDARG ((uint32_t)0xFFFFFFFF) /* !<Command argument */

/* ***** Bit definition for SDIO_CMD register *****/
#define SDIO_CMD_CMDINDEX ((uint16_t)0x003F) /* !<Command Index */

#define SDIO_CMD_WAITRESP ((uint16_t)0x00C0) /* !<WAITRESP[1:0] bits (Wait for response
#define SDIO_CMD_WAITRESP_0 ((uint16_t)0x0040) /* !< Bit 0 */
#define SDIO_CMD_WAITRESP_1 ((uint16_t)0x0080) /* !< Bit 1 */

#define SDIO_CMD_WAITINT ((uint16_t)0x0100) /* !<CPSM Waits for Interrupt Request */
#define SDIO_CMD_WAITPEND ((uint16_t)0x0200) /* !<CPSM Waits for ends of data transfer */
#define SDIO_CMD_CPSMEN ((uint16_t)0x0400) /* !<Command path state machine (CPSM) Enable */
#define SDIO_CMD_SDIOSUSPEND ((uint16_t)0x0800) /* !<SD I/O suspend command */
#define SDIO_CMD_ENCMDCOMPL ((uint16_t)0x1000) /* !<Enable CMD completion */
#define SDIO_CMD_NIEN ((uint16_t)0x2000) /* !<Not Interrupt Enable */
#define SDIO_CMD_CEATACMD ((uint16_t)0x4000) /* !<CE-ATA command */

/* ***** Bit definition for SDIO_RESPCMD register *****/
#define SDIO_RESPCMD_RESPCMD ((uint8_t)0x3F) /* !<Response command index */

/* ***** Bit definition for SDIO_RESP0 register *****/
#define SDIO_RESP0_CARDSTATUS0 ((uint32_t)0xFFFFFFFF) /* !<Card Status */

/* ***** Bit definition for SDIO_RESP1 register *****/
#define SDIO_RESP1_CARDSTATUS1 ((uint32_t)0xFFFFFFFF) /* !<Card Status */

/* ***** Bit definition for SDIO_RESP2 register *****/
#define SDIO_RESP2_CARDSTATUS2 ((uint32_t)0xFFFFFFFF) /* !<Card Status */

```

```

/*
***** Bit definition for SDIO_RESP3 register *****/
#define SDIO_RESP3_CARDSTATUS3 ((uint32_t)0xFFFFFFFF) /*!<Card Status */

/*
***** Bit definition for SDIO_RESP4 register *****/
#define SDIO_RESP4_CARDSTATUS4 ((uint32_t)0xFFFFFFFF) /*!<Card Status */

/*
***** Bit definition for SDIO_DTMIMER register *****/
#define SDIO_DTMIMER_DATATIME ((uint32_t)0xFFFFFFFF) /*!<Data timeout period. */

/*
***** Bit definition for SDIO_DLLEN register *****/
#define SDIO_DLLEN_DATALENGTH ((uint32_t)0x01FFFFFF) /*!<Data length value */

/*
***** Bit definition for SDIO_DCTRL register *****/
#define SDIO_DCTRL_DTEN ((uint16_t)0x0001) /*!<Data transfer enabled bit */
#define SDIO_DCTRL_DDIR ((uint16_t)0x0002) /*!<Data transfer direction selection */
#define SDIO_DCTRL_DTMODE ((uint16_t)0x0004) /*!<Data transfer mode selection */
#define SDIO_DCTRL_DMAEN ((uint16_t)0x0008) /*!<DMA enabled bit */

/*
***** Bit definition for SDIO_BLOCKSIZE[3:0] bits (Data block size)
*/
#define SDIO_DCTRL_DBLOCKSIZE_0 ((uint16_t)0x00F0) /*!<Bit 0 */
#define SDIO_DCTRL_DBLOCKSIZE_1 ((uint16_t)0x0010) /*!<Bit 1 */
#define SDIO_DCTRL_DBLOCKSIZE_2 ((uint16_t)0x0040) /*!<Bit 2 */
#define SDIO_DCTRL_DBLOCKSIZE_3 ((uint16_t)0x0080) /*!<Bit 3 */

#define SDIO_DCTRL_RWSTART ((uint16_t)0x0100) /*!<Read wait start */
#define SDIO_DCTRL_RWSSTOP ((uint16_t)0x0200) /*!<Read wait stop */
#define SDIO_DCTRL_RWMOD ((uint16_t)0x0400) /*!<Read wait mode */
#define SDIO_DCTRL_SDIOEN ((uint16_t)0x0800) /*!<SD I/O enable functions */

/*
***** Bit definition for SDIO_DCOUNT register *****/
#define SDIO_DCOUNT ((uint32_t)0x01FFFFFF) /*!<Data count value */

/*
***** Bit definition for SDIO_STA register *****/
#define SDIO_STA_CCRFAIL ((uint32_t)0x00000001) /*!<Command response received (CRC check

```

```

#define SDIO_STA_DCRCFAIL ((uint32_t)0x000000002)
#define SDIO_STA_CTIMEOUT ((uint32_t)0x00000004)
#define SDIO_STA_DTIMEOUT ((uint32_t)0x00000008)
#define SDIO_STA_TXUNDER ((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_RXACT ((uint32_t)0x00001000)
#define SDIO_STA_RXACT ((uint32_t)0x00002000)
#define SDIO_STA_RXFIFOHE ((uint32_t)0x00004000)
#define SDIO_STA_RXFIFOHF ((uint32_t)0x00008000)
#define SDIO_STA_RXFIFOFF ((uint32_t)0x00010000)
#define SDIO_STA_RXFIFOOF ((uint32_t)0x00020000)
#define SDIO_STA_RXFIFOE ((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)

/* !<Data block sent/received (CRC check for
   data 21. Counter, SDIDCOUNT, is
   required) * /
/* !<Received FIFO overrun error */
/* !<Transmit FIFO underrun error */
/* !<Command response received (CRC check */
/* !<Command sent (no response required) */
/* !<Data end (data counter, SDIDCOUNT, is
   detected on all data si */
/* !<Data block sent/received (CRC check p
   library */
/* !<Command transfer in progress */
/* !<Data transmit in progress */
/* !<Data receive in progress */
/* !<Data transmit in progress at least 8
   bytes */
/* !<Receive FIFO Half Full: there are at
   least 8 bytes available */
/* !<Transmit FIFO full */
/* !<Receive FIFO full */
/* !<Transmit FIFO empty */
/* !<Receive FIFO empty */
/* !<Transmit FIFO empty */
/* !<Data available in transmit FIFO */
/* !<Data available in receive FIFO */
/* !<SDIO interrupt received */
/* !<CE-ATA command completion signal received */

/* ***** Bit definition for SDIO_ICR register *****/
#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_TXUNDERC ((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)

/* !<CCRCFAIL flag clear bit */
/* !<DCRCFAIL flag clear bit */
/* !<CTIMEOUT flag clear bit */
/* !<DTIMEOUT flag clear bit */
/* !<TXUNDER flag clear bit */
/* !<RXOVERR flag clear bit */
/* !<CMDREND flag clear bit */
/* !<CMDSENT flag clear bit */
/* !<DATAEND flag clear bit */
/* !<STBITERR flag clear bit */

```

```

Глава 21. Стандартная библиотека SDIO

/*
 * !<DBCKEND flag clear bit */
((uint32_t)0x000000400)
/* !<SDIOIT flag clear bit */
((uint32_t)0x00400000)
/* !<CEATAEND flag clear bit */
((uint32_t)0x00800000)

/**************** Bit definition for SDIO_MASK register *****/
#define SDIO_MASK_CCRFAILIE ((uint32_t)0x00000001)
#define SDIO_MASK_DCROFAILIE ((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_RXFIFOHFIIE ((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_TXDAVIE ((uint32_t)0x00100000)
#define SDIO_MASK_RXDAVIE ((uint32_t)0x00200000)
#define SDIO_MASK_SDIOITIE ((uint32_t)0x00400000)
#define SDIO_MASK_CEATAENDIE ((uint32_t)0x00800000)

/**************** Bit definition for SDIO_FIFOCNT register *****/
#define SDIO_FIFOCNT_SDIOFIFO ((uint32_t)0x000FFFFF)
/* Remaining number of words to be written */

/**************** Bit definition for SDIO_FIFO register *****/
#define SDIO_FIFO_SDIOFIFO DATA ((uint32_t)0xFFFFFFF)
/* !<Receive and transmit FIFO data */

```

```

/*
 */
/* Serial Peripheral Interface */
/*
***** Bit definition for SPI_CR1 register *****/
#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)

/*
***** Bit definition for SPI_CR2 register *****/
#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)

/*
***** Bit definition for SPI_B�寄存器 *****/
/* !<RX Buffer DMA Enable */
/* !<Tx Buffer DMA Enable */
/* !<SS Output Enable */
/* !<Error Interrupt Enable */
/* !<RX buffer Not Empty Interrupt Enable */
/* !<Tx buffer Empty Interrupt Enable */

```

```

/*
***** Bit definition for SPI_SR register *****/
#define SPI_SR_RXNE ((uint8_t)0x01) /*!<Receive buffer Not Empty */
#define SPI_SR_TXE ((uint8_t)0x02) /*!<Transmit buffer Empty */
#define SPI_SR_CHSIDE ((uint8_t)0x04) /*!<Channel side */
#define SPI_SR_UDR ((uint8_t)0x08) /*!<Underrun flag */
#define SPI_SR_CRCERR ((uint8_t)0x10) /*!<CRC Error flag */
#define SPI_SR_MODF ((uint8_t)0x20) /*!<Mode fault */
#define SPI_SR_OVR ((uint8_t)0x40) /*!<Overrun flag */
#define SPI_SR_BSY ((uint8_t)0x80) /*!<Busy flag */

/*
***** Bit definition for SPI_DR register *****/
#define SPI_DR_DR ((uint16_t)0xFFFF) /*!<Data Register */

/*
***** Bit definition for SPI_CRCPR register *****/
#define SPI_CRCPR_CRCPOLY ((uint16_t)0xFFFF) /*!<CRC polynomial register */

/*
***** Bit definition for SPI_RXCRCR register *****/
#define SPI_RXCRCR_RXCRC ((uint16_t)0xFFFF) /*!<Rx CRC Register */

/*
***** Bit definition for SPI_TXCRCR register *****/
#define SPI_TXCRCR_TXCRC ((uint16_t)0xFFFF) /*!<Tx CRC Register */

/*
***** Bit definition for SPI_I2SCFGR register *****/
#define SPI_I2SCFGR_CHLEN ((uint16_t)0x0001) /*!<Channel length (number of bits per au*/

#define SPI_I2SCFGR_DATLEN ((uint16_t)0x0006) /*!<DATLEN[1:0] bits (Data length to be standard select
#define SPI_I2SCFGR_DATLEN_0 ((uint16_t)0x0002) /*!<Bit 0 */
#define SPI_I2SCFGR_DATLEN_1 ((uint16_t)0x0004) /*!<Bit 1 */

#define SPI_I2SCFGR_CKPOL ((uint16_t)0x0008) /*!<steady state clock polarity */

#define SPI_I2SCFGR_I2SSSTD ((uint16_t)0x0030) /*!<I2SSTD[1:0] bits (I2S standard select
#define SPI_I2SCFGR_I2SSSTD_0 ((uint16_t)0x0010) /*!<Bit 0 */
#define SPI_I2SCFGR_I2SSSTD_1 ((uint16_t)0x0020) /*!<Bit 1 */

```

```

/* !<PCM frame synchronization */
#define SPI_I2SCFGR_PCM_SYNC ((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)

/* !<I2S mode selection */
/* !<I2S bits (I2S configuration */
/* !<Bit 0 */
/* !<Bit 1 */

/* !<I2S Enable */
/* !<I2S mode selection */

/* !<I2S Linear prescaler */
/* !<Odd factor for the prescaler */
/* !<Master Clock Output Enable */

/* !<PCM register */
/* !<I2SDIV */
/* !<ODD */
/* !<MCKOE */

/* !<SYSCFG register */
/* !<SYSCFG_Memory Remap Config */

/* !<SYSCFG_PMC register */
/* !<Ethernet PHY interface selection */
/* !<Old MII_RMII_SEL bit definition, maintained for legacy purpose */
#define SYSCFG_PMC_MII_RMII_SEL

/* !<SYSCFG_EXTICR1 register */
/* !<EXTI 0 configuration */
/* !<EXTI 1 configuration */
/* !<EXTI 2 configuration */
/* !<EXTI 3 configuration */

/* !<SYSCFG_EXTICR1_EXTIO */
#define SYSCFG_EXTICR1_EXTIO
#define SYSCFG_EXTICR1_EXTI1
#define SYSCFG_EXTICR1_EXTI2
#define SYSCFG_EXTICR1_EXTI3

```

```

/*
 * @brief   EXTI0 configuration
 */
#define SYSCFG_EXTICR1_EXTI0_PA ((uint16_t)0x0000) /* !<PA[0] pin */
#define SYSCFG_EXTICR1_EXTI0_PB ((uint16_t)0x0001) /* !<PB[0] pin */
#define SYSCFG_EXTICR1_EXTI0_PC ((uint16_t)0x0002) /* !<PC[0] pin */
#define SYSCFG_EXTICR1_EXTI0_PD ((uint16_t)0x0003) /* !<PD[0] pin */
#define SYSCFG_EXTICR1_EXTI0_PE ((uint16_t)0x0004) /* !<PE[0] pin */
#define SYSCFG_EXTICR1_EXTI0_PF ((uint16_t)0x0005) /* !<PF[0] pin */
#define SYSCFG_EXTICR1_EXTI0_PG ((uint16_t)0x0006) /* !<PG[0] pin */
#define SYSCFG_EXTICR1_EXTI0_PH ((uint16_t)0x0007) /* !<PH[0] pin */
#define SYSCFG_EXTICR1_EXTI0_PI ((uint16_t)0x0008) /* !<PI[0] pin */

/*
 * @brief   EXTI1 configuration
 */
#define SYSCFG_EXTICR1_EXTI1_PA ((uint16_t)0x0000) /* !<PA[1] pin */
#define SYSCFG_EXTICR1_EXTI1_PB ((uint16_t)0x0010) /* !<PB[1] pin */
#define SYSCFG_EXTICR1_EXTI1_PC ((uint16_t)0x0020) /* !<PC[1] pin */
#define SYSCFG_EXTICR1_EXTI1_PD ((uint16_t)0x0030) /* !<PD[1] pin */
#define SYSCFG_EXTICR1_EXTI1_PE ((uint16_t)0x0040) /* !<PE[1] pin */
#define SYSCFG_EXTICR1_EXTI1_PF ((uint16_t)0x0050) /* !<PF[1] pin */
#define SYSCFG_EXTICR1_EXTI1_PG ((uint16_t)0x0060) /* !<PG[1] pin */
#define SYSCFG_EXTICR1_EXTI1_PH ((uint16_t)0x0070) /* !<PH[1] pin */
#define SYSCFG_EXTICR1_EXTI1_PI ((uint16_t)0x0080) /* !<PI[1] pin */

/*
 * @brief   EXTI2 configuration
 */
#define SYSCFG_EXTICR1_EXTI2_PA ((uint16_t)0x0000) /* !<PA[2] pin */
#define SYSCFG_EXTICR1_EXTI2_PB ((uint16_t)0x0100) /* !<PB[2] pin */
#define SYSCFG_EXTICR1_EXTI2_PC ((uint16_t)0x0200) /* !<PC[2] pin */
#define SYSCFG_EXTICR1_EXTI2_PD ((uint16_t)0x0300) /* !<PD[2] pin */
#define SYSCFG_EXTICR1_EXTI2_PE ((uint16_t)0x0400) /* !<PE[2] pin */
#define SYSCFG_EXTICR1_EXTI2_PF ((uint16_t)0x0500) /* !<PF[2] pin */
#define SYSCFG_EXTICR1_EXTI2_PG ((uint16_t)0x0600) /* !<PG[2] pin */
#define SYSCFG_EXTICR1_EXTI2_PH ((uint16_t)0x0700) /* !<PH[2] pin */
#define SYSCFG_EXTICR1_EXTI2_PI ((uint16_t)0x0800) /* !<PI[2] pin */

```

```

/** @brief EXTI3 configuration */
#define SYSCFG_EXTICR1_EXTI3_PA ((uint16_t)0x0000) /* !<PA[3] pin */
#define SYSCFG_EXTICR1_EXTI3_PB ((uint16_t)0x1000) /* !<PB[3] pin */
#define SYSCFG_EXTICR1_EXTI3_PC ((uint16_t)0x2000) /* !<PC[3] pin */
#define SYSCFG_EXTICR1_EXTI3_PD ((uint16_t)0x3000) /* !<PD[3] pin */
#define SYSCFG_EXTICR1_EXTI3_PE ((uint16_t)0x4000) /* !<PE[3] pin */
#define SYSCFG_EXTICR1_EXTI3_PF ((uint16_t)0x5000) /* !<PF[3] pin */
#define SYSCFG_EXTICR1_EXTI3_PG ((uint16_t)0x6000) /* !<PG[3] pin */
#define SYSCFG_EXTICR1_EXTI3_PH ((uint16_t)0x7000) /* !<PH[3] pin */
#define SYSCFG_EXTICR1_EXTI3_PI ((uint16_t)0x8000) /* !<PI[3] pin */

/* ***** Bit definition for SYSCFG_EXTICR2 register *****/
#define SYSCFG_EXTICR2_EXTI4 ((uint16_t)0x000F) /* !<EXTI 4 configuration */
#define SYSCFG_EXTICR2_EXTI5 ((uint16_t)0x00F0) /* !<EXTI 5 configuration */
#define SYSCFG_EXTICR2_EXTI6 ((uint16_t)0x0F00) /* !<EXTI 6 configuration */
#define SYSCFG_EXTICR2_EXTI7 ((uint16_t)0xF000) /* !<EXTI 7 configuration */

/* @brief EXTI4 configuration */
#define SYSCFG_EXTICR2_EXTI4_PA ((uint16_t)0x0000) /* !<PA[4] pin */
#define SYSCFG_EXTICR2_EXTI4_PB ((uint16_t)0x0001) /* !<PB[4] pin */
#define SYSCFG_EXTICR2_EXTI4_PC ((uint16_t)0x0002) /* !<PC[4] pin */
#define SYSCFG_EXTICR2_EXTI4_PD ((uint16_t)0x0003) /* !<PD[4] pin */
#define SYSCFG_EXTICR2_EXTI4_PE ((uint16_t)0x0004) /* !<PE[4] pin */
#define SYSCFG_EXTICR2_EXTI4_PF ((uint16_t)0x0005) /* !<PF[4] pin */
#define SYSCFG_EXTICR2_EXTI4_PG ((uint16_t)0x0006) /* !<PG[4] pin */
#define SYSCFG_EXTICR2_EXTI4_PH ((uint16_t)0x0007) /* !<PH[4] pin */
#define SYSCFG_EXTICR2_EXTI4_PI ((uint16_t)0x0008) /* !<PI[4] pin */

/* @brief EXTI5 configuration */
#define SYSCFG_EXTICR2_EXTI5_PA ((uint16_t)0x0000) /* !<PA[5] pin */
#define SYSCFG_EXTICR2_EXTI5_PB ((uint16_t)0x0010) /* !<PB[5] pin */

```

```

#define SYS_CFG_EXTICR2_EXTI5_PC ((uint16_t)0x0020) /* !<PC[5] pin */
#define SYS_CFG_EXTICR2_EXTI5_PD ((uint16_t)0x0030) /* !<PD[5] pin */
#define SYS_CFG_EXTICR2_EXTI5_PE ((uint16_t)0x0040) /* !<PE[5] pin */
#define SYS_CFG_EXTICR2_EXTI5_PF ((uint16_t)0x0050) /* !<PF[5] pin */
#define SYS_CFG_EXTICR2_EXTI5_PG ((uint16_t)0x0060) /* !<PG[5] pin */
#define SYS_CFG_EXTICR2_EXTI5_PH ((uint16_t)0x0070) /* !<PH[5] pin */
#define SYS_CFG_EXTICR2_EXTI5_PI ((uint16_t)0x0080) /* !<PI[5] pin */

/* @brief EXTI6 configuration */
#define SYS_CFG_EXTICR2_EXTI6_PA ((uint16_t)0x0000) /* !<PA[6] pin */
#define SYS_CFG_EXTICR2_EXTI6_PB ((uint16_t)0x0100) /* !<PB[6] pin */
#define SYS_CFG_EXTICR2_EXTI6_PC ((uint16_t)0x0200) /* !<PC[6] pin */
#define SYS_CFG_EXTICR2_EXTI6_PD ((uint16_t)0x0300) /* !<PD[6] pin */
#define SYS_CFG_EXTICR2_EXTI6_PE ((uint16_t)0x0400) /* !<PE[6] pin */
#define SYS_CFG_EXTICR2_EXTI6_PF ((uint16_t)0x0500) /* !<PF[6] pin */
#define SYS_CFG_EXTICR2_EXTI6_PG ((uint16_t)0x0600) /* !<PG[6] pin */
#define SYS_CFG_EXTICR2_EXTI6_PH ((uint16_t)0x0700) /* !<PH[6] pin */
#define SYS_CFG_EXTICR2_EXTI6_PI ((uint16_t)0x0800) /* !<PI[6] pin */

/* @brief EXTI7 configuration */
#define SYS_CFG_EXTICR2_EXTI7_PA ((uint16_t)0x0000) /* !<PA[7] pin */
#define SYS_CFG_EXTICR2_EXTI7_PB ((uint16_t)0x1000) /* !<PB[7] pin */
#define SYS_CFG_EXTICR2_EXTI7_PC ((uint16_t)0x2000) /* !<PC[7] pin */
#define SYS_CFG_EXTICR2_EXTI7_PD ((uint16_t)0x3000) /* !<PD[7] pin */
#define SYS_CFG_EXTICR2_EXTI7_PE ((uint16_t)0x4000) /* !<PE[7] pin */
#define SYS_CFG_EXTICR2_EXTI7_PF ((uint16_t)0x5000) /* !<PF[7] pin */
#define SYS_CFG_EXTICR2_EXTI7_PG ((uint16_t)0x6000) /* !<PG[7] pin */
#define SYS_CFG_EXTICR2_EXTI7_PH ((uint16_t)0x7000) /* !<PH[7] pin */
#define SYS_CFG_EXTICR2_EXTI7_PI ((uint16_t)0x8000) /* !<PI[7] pin */

***** Bit definition for SYSCFG_EXTICR3 register *****/
#define SYS_CFG_EXTICR3_EXTI8 ((uint16_t)0x000F) /* !<EXTI 8 configuration */
#define SYS_CFG_EXTICR3_EXTI9 ((uint16_t)0x00F0) /* !<EXTI 9 configuration */

```

```

#define SYSCFG_EXTICR3_EXTI10 ((uint16_t)0x00F00) /* !< EXTI 10 configuration */
#define SYSCFG_EXTICR3_EXTI11 ((uint16_t)0xF000) /* !< EXTI 11 configuration */

/** @brief EXTI8 configuration */
#define SYSCFG_EXTICR3_EXTI8_PA ((uint16_t)0x0000) /* !< PA[8] pin */
#define SYSCFG_EXTICR3_EXTI8_PB ((uint16_t)0x0001) /* !< PB[8] pin */
#define SYSCFG_EXTICR3_EXTI8_PC ((uint16_t)0x0002) /* !< PC[8] pin */
#define SYSCFG_EXTICR3_EXTI8_PD ((uint16_t)0x0003) /* !< PD[8] pin */
#define SYSCFG_EXTICR3_EXTI8_PE ((uint16_t)0x0004) /* !< PE[8] pin */
#define SYSCFG_EXTICR3_EXTI8_PF ((uint16_t)0x0005) /* !< PF[8] pin */
#define SYSCFG_EXTICR3_EXTI8_PG ((uint16_t)0x0006) /* !< PG[8] pin */
#define SYSCFG_EXTICR3_EXTI8_PH ((uint16_t)0x0007) /* !< PH[8] pin */
#define SYSCFG_EXTICR3_EXTI8_PI ((uint16_t)0x0008) /* !< PI[8] pin */

/** @brief EXTI9 configuration */
#define SYSCFG_EXTICR3_EXTI9_PA ((uint16_t)0x0000) /* !< PA[9] pin */
#define SYSCFG_EXTICR3_EXTI9_PB ((uint16_t)0x0010) /* !< PB[9] pin */
#define SYSCFG_EXTICR3_EXTI9_PC ((uint16_t)0x0020) /* !< PC[9] pin */
#define SYSCFG_EXTICR3_EXTI9_PD ((uint16_t)0x0030) /* !< PD[9] pin */
#define SYSCFG_EXTICR3_EXTI9_PE ((uint16_t)0x0040) /* !< PE[9] pin */
#define SYSCFG_EXTICR3_EXTI9_PF ((uint16_t)0x0050) /* !< PF[9] pin */
#define SYSCFG_EXTICR3_EXTI9_PG ((uint16_t)0x0060) /* !< PG[9] pin */
#define SYSCFG_EXTICR3_EXTI9_PH ((uint16_t)0x0070) /* !< PH[9] pin */
#define SYSCFG_EXTICR3_EXTI9_PI ((uint16_t)0x0080) /* !< PI[9] pin */

/** @brief EXTI10 configuration */
#define SYSCFG_EXTICR3_EXTI10_PA ((uint16_t)0x0000) /* !< PA[10] pin */
#define SYSCFG_EXTICR3_EXTI10_PB ((uint16_t)0x0100) /* !< PB[10] pin */
#define SYSCFG_EXTICR3_EXTI10_PC ((uint16_t)0x0200) /* !< PC[10] pin */
#define SYSCFG_EXTICR3_EXTI10_PD ((uint16_t)0x0300) /* !< PD[10] pin */
#define SYSCFG_EXTICR3_EXTI10_PE ((uint16_t)0x0400) /* !< PE[10] pin */

```

```

#define SYS_CFG_EXTICR3_EXTI10_PF ((uint16_t)0x0500) /* !<PF[10] pin */
#define SYS_CFG_EXTICR3_EXTI10_PG ((uint16_t)0x0600) /* !<PG[10] pin */
#define SYS_CFG_EXTICR3_EXTI10_PH ((uint16_t)0x0700) /* !<PH[10] pin */
#define SYS_CFG_EXTICR3_EXTI10_PI ((uint16_t)0x0800) /* !<PI[10] pin */

/* @brief EXTI11 configuration
 */
#define SYS_CFG_EXTICR3_EXTI11_PA ((uint16_t)0x0000) /* !<PA[11] pin */
#define SYS_CFG_EXTICR3_EXTI11_PB ((uint16_t)0x1000) /* !<PB[11] pin */
#define SYS_CFG_EXTICR3_EXTI11_PC ((uint16_t)0x2000) /* !<PC[11] pin */
#define SYS_CFG_EXTICR3_EXTI11_PD ((uint16_t)0x3000) /* !<PD[11] pin */
#define SYS_CFG_EXTICR3_EXTI11_PE ((uint16_t)0x4000) /* !<PE[11] pin */
#define SYS_CFG_EXTICR3_EXTI11_PF ((uint16_t)0x5000) /* !<PF[11] pin */
#define SYS_CFG_EXTICR3_EXTI11_PG ((uint16_t)0x6000) /* !<PG[11] pin */
#define SYS_CFG_EXTICR3_EXTI11_PH ((uint16_t)0x7000) /* !<PH[11] pin */
#define SYS_CFG_EXTICR3_EXTI11_PI ((uint16_t)0x8000) /* !<PI[11] pin */

/* ***** Bit definition for SYSCFG_EXTICR4 register *****/
#define SYS_CFG_EXTICR4_EXTI12 ((uint16_t)0x000F) /* !<EXTI 12 configuration */
#define SYS_CFG_EXTICR4_EXTI13 ((uint16_t)0x00F0) /* !<EXTI 13 configuration */
#define SYS_CFG_EXTICR4_EXTI14 ((uint16_t)0x0F00) /* !<EXTI 14 configuration */
#define SYS_CFG_EXTICR4_EXTI15 ((uint16_t)0xF000) /* !<EXTI 15 configuration */

/* @brief EXTI12 configuration
 */
#define SYS_CFG_EXTICR4_EXTI12_PA ((uint16_t)0x0000) /* !<PA[12] pin */
#define SYS_CFG_EXTICR4_EXTI12_PB ((uint16_t)0x0001) /* !<PB[12] pin */
#define SYS_CFG_EXTICR4_EXTI12_PC ((uint16_t)0x0002) /* !<PC[12] pin */
#define SYS_CFG_EXTICR4_EXTI12_PD ((uint16_t)0x0003) /* !<PD[12] pin */
#define SYS_CFG_EXTICR4_EXTI12_PE ((uint16_t)0x0004) /* !<PE[12] pin */
#define SYS_CFG_EXTICR4_EXTI12_PF ((uint16_t)0x0005) /* !<PF[12] pin */
#define SYS_CFG_EXTICR4_EXTI12_PG ((uint16_t)0x0006) /* !<PG[12] pin */
#define SYS_CFG_EXTICR4_EXTI12_PH ((uint16_t)0x0007) /* !<PH[12] pin */

/* @brief EXTI13 configuration
 */

```

---

```

/*
 * @brief  EXTI13 configuration
 */
#define SYSCFG_EXTICR4_EXTI13_PA ((uint16_t)0x0000) /* !<PA[13] pin */
#define SYSCFG_EXTICR4_EXTI13_PB ((uint16_t)0x0010) /* !<PB[13] pin */
#define SYSCFG_EXTICR4_EXTI13_PC ((uint16_t)0x0020) /* !<PC[13] pin */
#define SYSCFG_EXTICR4_EXTI13_PD ((uint16_t)0x0030) /* !<PD[13] pin */
#define SYSCFG_EXTICR4_EXTI13_PE ((uint16_t)0x0040) /* !<PE[13] pin */
#define SYSCFG_EXTICR4_EXTI13_PF ((uint16_t)0x0050) /* !<PF[13] pin */
#define SYSCFG_EXTICR4_EXTI13_PG ((uint16_t)0x0060) /* !<PG[13] pin */
#define SYSCFG_EXTICR3_EXTI13_PH ((uint16_t)0x0070) /* !<PH[13] pin */

/*
 * @brief  EXTI14 configuration
 */
#define SYSCFG_EXTICR4_EXTI14_PA ((uint16_t)0x0000) /* !<PA[14] pin */
#define SYSCFG_EXTICR4_EXTI14_PB ((uint16_t)0x0100) /* !<PB[14] pin */
#define SYSCFG_EXTICR4_EXTI14_PC ((uint16_t)0x0200) /* !<PC[14] pin */
#define SYSCFG_EXTICR4_EXTI14_PD ((uint16_t)0x0300) /* !<PD[14] pin */
#define SYSCFG_EXTICR4_EXTI14_PE ((uint16_t)0x0400) /* !<PE[14] pin */
#define SYSCFG_EXTICR4_EXTI14_PF ((uint16_t)0x0500) /* !<PF[14] pin */
#define SYSCFG_EXTICR4_EXTI14_PG ((uint16_t)0x0600) /* !<PG[14] pin */
#define SYSCFG_EXTICR3_EXTI14_PH ((uint16_t)0x0700) /* !<PH[14] pin */

/*
 * @brief  EXTI15 configuration
 */
#define SYSCFG_EXTICR4_EXTI15_PA ((uint16_t)0x0000) /* !<PA[15] pin */
#define SYSCFG_EXTICR4_EXTI15_PB ((uint16_t)0x0100) /* !<PB[15] pin */
#define SYSCFG_EXTICR4_EXTI15_PC ((uint16_t)0x0200) /* !<PC[15] pin */
#define SYSCFG_EXTICR4_EXTI15_PD ((uint16_t)0x0300) /* !<PD[15] pin */
#define SYSCFG_EXTICR4_EXTI15_PE ((uint16_t)0x0400) /* !<PE[15] pin */
#define SYSCFG_EXTICR4_EXTI15_PF ((uint16_t)0x0500) /* !<PF[15] pin */
#define SYSCFG_EXTICR4_EXTI15_PG ((uint16_t)0x0600) /* !<PG[15] pin */
#define SYSCFG_EXTICR3_EXTI15_PH ((uint16_t)0x0700) /* !<PH[15] pin */

***** Bit definition for SYSCFG_CMPCR register ****/
#define SYSCFG_CMPCR_CMPCR_P0 ((uint32_t)0x00000001) /* !<Compensation cell ready flag */
#define SYSCFG_CMPCR_READY ((uint32_t)0x00000010) /* !<Compensation cell power-down */

```

```

/*
 */
/*
 */
/* Bit definition for TIM_CR1 register *****/
#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_CMSS_0 ((uint16_t)0x0020)
#define TIM_CR1_CMSS_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)

/* Bit definition for TIM_CR2 register *****/
#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)

/* Mode Selection */
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */

/* Master Mode Selection */
/* !<MMS [2:0] bits (Master Mode Selection) */
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */

/* DMA Selection */
/* !<CKD [1:0] bits (clock division) */
/* !<Bit 0 */
/* !<Bit 1 */

/* Center-aligned mode */
/* !<CMS [1:0] bits (Center-aligned mode) */
/* !<Bit 0 */
/* !<Bit 1 */

/* Preloaded Control */
/* !<Capture/Compare Preloaded Control */
/* !<Capture/Compare Control Update Select */
/* !<Capture/Compare DMA Selection */

/* Selection */
/* !<TI1 Selection */
/* !<Output Idle state 1 (OC1 output) */
/* !<OC1 output */

```

```

/*
 * !<Output Idle state 1 (OC1N output)          /* !<Output Idle state 2 (OC2 output)
#define TIM_CR2_OIS1N ((uint16_t)0x0200)        /* !<Output Idle state 2 (OC2N output)
#define TIM_CR2_OIS2N ((uint16_t)0x0400)        /* !<Output Idle state 2 (OC3 output)
#define TIM_CR2_OIS2N ((uint16_t)0x0800)        /* !<Output Idle state 3 (OC3N output)
#define TIM_CR2_OIS3N ((uint16_t)0x1000)        /* !<Output Idle state 3 (OC3N output)
#define TIM_CR2_OIS3N ((uint16_t)0x2000)        /* !<Output Idle state 3 (OC4 output)
#define TIM_CR2_OIS4N ((uint16_t)0x4000)        /* !<Output Idle state 4 (OC4 output)

/* Bit definition for TIM_SMCR register *****/
#define TIM_SMCR_SMS ((uint16_t)0x0007)          /* !<SMS [2:0] bits (Slave mode selection)
#define TIM_SMCR_SMS_0 ((uint16_t)0x0001)          /* !<Bit 0 */
#define TIM_SMCR_SMS_1 ((uint16_t)0x0002)          /* !<Bit 1 */
#define TIM_SMCR_SMS_2 ((uint16_t)0x0004)          /* !<Bit 2 */

#define TIM_SMCR_TS ((uint16_t)0x0010)          /* !<TS [2:0] bits (Trigger selection)
#define TIM_SMCR_TS_0 ((uint16_t)0x0010)          /* !<Bit 0 */
#define TIM_SMCR_TS_1 ((uint16_t)0x0020)          /* !<Bit 1 */
#define TIM_SMCR_TS_2 ((uint16_t)0x0040)          /* !<Bit 2 */

#define TIM_SMCR_MSM ((uint16_t)0x0080)          /* !<Master/slave mode */

#define TIM_SMCR_ETF ((uint16_t)0x0F00)          /* !<ETF [3:0] bits (External trigger filter)
#define TIM_SMCR_ETF_0 ((uint16_t)0x0100)          /* !<Bit 0 */
#define TIM_SMCR_ETF_1 ((uint16_t)0x0200)          /* !<Bit 1 */
#define TIM_SMCR_ETF_2 ((uint16_t)0x0400)          /* !<Bit 2 */
#define TIM_SMCR_ETF_3 ((uint16_t)0x0800)          /* !<Bit 3 */

#define TIM_SMCR_ETPS ((uint16_t)0x3000)          /* !<ETPS [1:0] bits (External trigger prescaler)
#define TIM_SMCR_ETPS_0 ((uint16_t)0x1000)          /* !<Bit 0 */
#define TIM_SMCR_ETPS_1 ((uint16_t)0x2000)          /* !<Bit 1 */

#define TIM_SMCR_ECE ((uint16_t)0x4000)          /* !<External clock enable */
#define TIM_SMCR_ETP ((uint16_t)0x8000)          /* !<External trigger polarity */

/* Bit definition for TIM_DIER register *****/
#define TIM_DIER_UIE ((uint16_t)0x0001)          /* !<Update interrupt enable */

```

Глава 21. STM32 библиотека библиотека STM32

```

/*!<Capture/Compare 1 interrupt enable */          disable */
#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)

/****** Bit definition for TIM_SR register ******/
#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)

/****** Bit definition for TIM_EGR register ******/
#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)

/*!<Update Generation */          disable */
/*!<Capture/Compare 1 Generation */          enable */
/*!<Capture/Compare 2 Generation */          enable */
/*!<Capture/Compare 3 Generation */          enable */
/*!<Capture/Compare 4 Generation */          enable */

/*!<Break interrupt enable */          disable */
/*!<Update DMA request enable */          enable
/*!<Capture/Compare 1 DMA request enable */
/*!<Capture/Compare 2 DMA request enable */
/*!<Capture/Compare 3 DMA request enable */
/*!<Capture/Compare 4 DMA request enable */
/*!<COM DMA request enable */
/*!<Trigger DMA request enable */

/*!<Update interrupt Flag */          Flag */
/*!<Capture/Compare 1 interrupt Flag */          Flag
/*!<Capture/Compare 2 interrupt Flag */          Flag
/*!<Capture/Compare 3 interrupt Flag */          Flag
/*!<Capture/Compare 4 interrupt Flag */          Flag
/*!<COM interrupt Flag */          Flag
/*!<Trigger interrupt Flag */          Flag
/*!<Break interrupt Flag */          Flag
/*!<Capture/Compare 1 Overcapture Flag */          Flag
/*!<Capture/Compare 2 Overcapture Flag */          Flag
/*!<Capture/Compare 3 Overcapture Flag */          Flag
/*!<Capture/Compare 4 Overcapture Flag */          Flag

/*!<Update Generation */          disable */
/*!<Capture/Compare 1 Generation */          enable */
/*!<Capture/Compare 2 Generation */          enable */
/*!<Capture/Compare 3 Generation */          enable */
/*!<Capture/Compare 4 Generation */          enable

```

```

/*!<Capture/Compare Control Update General
((uint8_t)0x20)          /*!<Trigger Generation */
((uint8_t)0x40)          /*!<Break Generation */
((uint8_t)0x80)          /*!<Capture/Compare Control Update General Sel

/****** Bit definition for TIM_CCMR1 register *****/
#define TIM_CCMR1_CC1S      ((uint16_t)0x0003)          /*!<CC1S[1:0] bits (Capture/Compare 1 Mode
#define TIM_CCMR1_CC1S_0     ((uint16_t)0x0001)          /*!<Bit 0 */
#define TIM_CCMR1_CC1S_1     ((uint16_t)0x0002)          /*!<Bit 1 */

#define TIM_CCMR1_OC1FE     ((uint16_t)0x0004)          /*!<Output Compare 1 Fast enable */
#define TIM_CCMR1_OC1PE     ((uint16_t)0x0008)          /*!<Output Compare 1 Preload enable */

#define TIM_CCMR1_OC1M      ((uint16_t)0x0070)          /*!<OC1M[2:0] bits (Output Compare 1 Mode
#define TIM_CCMR1_OC1M_0     ((uint16_t)0x0010)          /*!<Bit 0 */
#define TIM_CCMR1_OC1M_1     ((uint16_t)0x0020)          /*!<Bit 1 */
#define TIM_CCMR1_OC1M_2     ((uint16_t)0x0040)          /*!<Bit 2 */

#define TIM_CCMR1_OC1CE     ((uint16_t)0x0080)          /*!<Output Compare 1Clear Enable */

#define TIM_CCMR1_CC2S      ((uint16_t)0x0300)          /*!<CC2S[1:0] bits (Capture/Compare 2 Mode
#define TIM_CCMR1_CC2S_0     ((uint16_t)0x0100)          /*!<Bit 0 */
#define TIM_CCMR1_CC2S_1     ((uint16_t)0x0200)          /*!<Bit 1 */

#define TIM_CCMR1_OC2FE     ((uint16_t)0x0400)          /*!<Output Compare 2 Fast enable */
#define TIM_CCMR1_OC2PE     ((uint16_t)0x0800)          /*!<Output Compare 2 Preload enable */

#define TIM_CCMR1_OC2M      ((uint16_t)0x7000)          /*!<OC2M[2:0] bits (Output Compare 2 Mode
#define TIM_CCMR1_OC2M_0     ((uint16_t)0x1000)          /*!<Bit 0 */
#define TIM_CCMR1_OC2M_1     ((uint16_t)0x2000)          /*!<Bit 1 */
#define TIM_CCMR1_OC2M_2     ((uint16_t)0x4000)          /*!<Bit 2 */

#define TIM_CCMR1_OC2CE     ((uint16_t)0x8000)          /*!<Output Compare 2 Clear Enable */

/*--*/
```

```

/* !< IC1PSC[1:0] bits (Input Capture 1 Filter)
((uint16_t)0x000C)
((uint16_t)0x0004)
((uint16_t)0x0008)

/* !< IC1F[3:0] bits (Input Capture 1 Filter)
((uint16_t)0x00F0)
((uint16_t)0x0010)
((uint16_t)0x0020)
((uint16_t)0x0040)
((uint16_t)0x0080)

/* !< IC2PSC[1:0] bits (Input Capture 2 Filter)
((uint16_t)0x0C00)
((uint16_t)0x0400)
((uint16_t)0x0800)

/* !< IC2F[3:0] bits (Input Capture 2 Filter)
((uint16_t)0xF000)
((uint16_t)0x1000)
((uint16_t)0x2000)
((uint16_t)0x4000)
((uint16_t)0x8000)

/* **** Bit definition for TIM_CCMR2 register *****/
/* !< CC3S[1:0] bits (Capture/Compare 3 Selection)
((uint16_t)0x0003)
((uint16_t)0x0001)
((uint16_t)0x0002)

/* !< Output Compare 3 Fast enable */
/* !< Output Compare 3 Preload enable */
/* !< OC3M[2:0] bits (Output Compare 3 Mode)
((uint16_t)0x0070)
((uint16_t)0x0010)
((uint16_t)0x0020)
((uint16_t)0x0040)

/* !< OC3E[1:0] bits (Output Compare 3 Enable */
((uint16_t)0x0080)

#define TIM_CCMR1_IC1PSC
#define TIM_CCMR1_IC1F_0
#define TIM_CCMR1_IC1PSC_0
#define TIM_CCMR1_IC1F_1
#define TIM_CCMR1_IC1PSC_1
#define TIM_CCMR1_IC1F_2
#define TIM_CCMR1_IC1PSC_2
#define TIM_CCMR1_IC1F_3
#define TIM_CCMR1_IC1PSC_3

#define TIM_CCMR1_IC2PSC
#define TIM_CCMR1_IC2F_0
#define TIM_CCMR1_IC2PSC_0
#define TIM_CCMR1_IC2F_1
#define TIM_CCMR1_IC2PSC_1
#define TIM_CCMR1_IC2F_2
#define TIM_CCMR1_IC2PSC_2
#define TIM_CCMR1_IC2F_3
#define TIM_CCMR1_IC2PSC_3

#define TIM_CCMR2_CC3S
#define TIM_CCMR2_CC3S_0
#define TIM_CCMR2_CC3S_1

#define TIM_CCMR2_OC3FE
#define TIM_CCMR2_OC3PE

#define TIM_CCMR2_OC3M
#define TIM_CCMR2_OC3M_0
#define TIM_CCMR2_OC3M_1
#define TIM_CCMR2_OC3M_2
#define TIM_CCMR2_OC3M_3

```

```

/* !<CC4S [1:0] bits (Capture/Compare
#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)0xF000)
#define TIM_CCMR2_IC4PSC_0 ((uint16_t)0x0400)
#define TIM_CCMR2_IC4PSC_1 ((uint16_t)0x0800)

#define TIM_CCMR2_IC4F ((uint16_t)0x0000)
#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)

/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */
/* !<Bit 3 */

/* !<Output Compare 4 Fast enable */
/* !<Output Compare 4 Preload enable */
/* !<Output Compare 4 Compare */
/* !<Output Compare 4 Compare */
/* !<Output Compare 4 Clear Enable */

/* !<OC4M [2:0] bits (Output Compare 4 Mode */
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */

/* !<IC3PSC [1:0] bits (Input Capture 3 */
/* !<Bit 0 */
/* !<Bit 1 */

/* !<IC3F [3:0] bits (Input Capture 3 Filter */
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */
/* !<Bit 3 */

/* !<IC4PSC [1:0] bits (Input Capture 4 Filter */
/* !<Bit 0 */
/* !<Bit 1 */

/* !<IC4F [3:0] bits (Input Capture 4 Filter */
/* !<Bit 0 */
/* !<Bit 1 */
/* !<Bit 2 */
/* !<Bit 3 */

```

```

/*
***** Bit definition for TIM_CCER register *****/
#define TIM_CCER_CC1E ((uint16_t)0x0001) /*!< Capture/Compare 1 output enable */
#define TIM_CCER_CC1P ((uint16_t)0x0002) /*!< Capture/Compare 1 output Polarity */
#define TIM_CCER_CC1NE ((uint16_t)0x0004) /*!< Capture/Compare 1 Complementary output */
#define TIM_CCER_CC1NP ((uint16_t)0x0008) /*!< Capture/Compare 1 Complementary output enable */
#define TIM_CCER_CC2E ((uint16_t)0x0010) /*!< Capture/Compare 2 output enable */
#define TIM_CCER_CC2P ((uint16_t)0x0020) /*!< Capture/Compare 2 output Polarity */
#define TIM_CCER_CC2NE ((uint16_t)0x0040) /*!< Capture/Compare 2 Complementary output */
#define TIM_CCER_CC2NP ((uint16_t)0x0080) /*!< Capture/Compare 2 Complementary output enable */
#define TIM_CCER_CC3E ((uint16_t)0x0100) /*!< Capture/Compare 3 output enable */
#define TIM_CCER_CC3P ((uint16_t)0x0200) /*!< Capture/Compare 3 output Polarity */
#define TIM_CCER_CC3NE ((uint16_t)0x0400) /*!< Capture/Compare 3 Complementary output */
#define TIM_CCER_CC3NP ((uint16_t)0x0800) /*!< Capture/Compare 3 Complementary output enable */
#define TIM_CCER_CC4E ((uint16_t)0x1000) /*!< Capture/Compare 4 output enable */
#define TIM_CCER_CC4P ((uint16_t)0x2000) /*!< Capture/Compare 4 output Polarity */
#define TIM_CCER_CC4NP ((uint16_t)0x8000) /*!< Capture/Compare 4 Complementary output */

/*
***** Bit definition for TIM_CNT register *****/
#define TIM_CNT_CNT ((uint16_t)0xFFFF) /*!< Counter Value */

/*
***** Bit definition for TIM_PSC register *****/
#define TIM_PSC_PSC ((uint16_t)0xFFFF) /*!< Prescaler Value */

/*
***** Bit definition for TIM_ARR register *****/
#define TIM_ARR_ARR ((uint16_t)0xFF) /*!< actual auto-reload Value */

/*
***** Bit definition for TIM_RCR register *****/
#define TIM_RCR_RCR ((uint8_t)0xFF) /*!< Repetition Counter Value */

/*
***** Bit definition for TIM_CCR1 register *****/
#define TIM_CCR1_CCR1 ((uint16_t)0xFFFF) /*!< Capture/Compare 1 Value */

/*
***** Bit definition for TIM_CCR2 register *****/
#define TIM_CCR2_CCR2 ((uint16_t)0xFFFF) /*!< Capture/Compare 2 Value */

```

```

/*
***** Bit definition for TIM_CCR3 register *****/
#define TIM_CCR3_CCR3 ((uint16_t)0xFFFF) /*!<Capture/Compare 3 Value */

/*
***** Bit definition for TIM_CCR4 register *****/
#define TIM_CCR4_CCR4 ((uint16_t)0xFFFF) /*!<Capture/Compare 4 Value */

/*
***** Bit definition for TIM_BDTR register *****/
#define TIM_BDTR_DTG ((uint16_t)0x00FF) /*!<DTG [0:7] bits (Dead-Time Generator Selection)
#define TIM_BDTR_DTG_0 ((uint16_t)0x0001) /*!<Bit 0 */
#define TIM_BDTR_DTG_1 ((uint16_t)0x0002) /*!<Bit 1 */
#define TIM_BDTR_DTG_2 ((uint16_t)0x0004) /*!<Bit 2 */
#define TIM_BDTR_DTG_3 ((uint16_t)0x0008) /*!<Bit 3 */
#define TIM_BDTR_DTG_4 ((uint16_t)0x0010) /*!<Bit 4 */
#define TIM_BDTR_DTG_5 ((uint16_t)0x0020) /*!<Bit 5 */
#define TIM_BDTR_DTG_6 ((uint16_t)0x0040) /*!<Bit 6 */
#define TIM_BDTR_DTG_7 ((uint16_t)0x0080) /*!<Bit 7 */

#define TIM_BDTR_LOCK ((uint16_t)0x0300) /*!<LOCK [1:0] bits (Lock Configuration)
#define TIM_BDTR_LOCK_0 ((uint16_t)0x0100) /*!<Bit 0 */
#define TIM_BDTR_LOCK_1 ((uint16_t)0x0200) /*!<Bit 1 */

#define TIM_BDTR_OSSI ((uint16_t)0x0400) /*!<Off-State Selection for Idle mode */
#define TIM_BDTR_OSSR ((uint16_t)0x0800) /*!<Off-State Selection for Run mode */
#define TIM_BDTR_BKE ((uint16_t)0x1000) /*!<Break enable */
#define TIM_BDTR_BKP ((uint16_t)0x2000) /*!<Break Polarity */
#define TIM_BDTR_AOE ((uint16_t)0x4000) /*!<Automatic Output enable */
#define TIM_BDTR_MOE ((uint16_t)0x8000) /*!<Main Output enable */

/*
***** Bit definition for TIM_DCR register *****/
#define TIM_DCR_DBA ((uint16_t)0x001F) /*!<DBA [4:0] bits (DMA Base Address) */
#define TIM_DCR_DBA_0 ((uint16_t)0x0001) /*!<Bit 0 */
#define TIM_DCR_DBA_1 ((uint16_t)0x0002) /*!<Bit 1 */
#define TIM_DCR_DBA_2 ((uint16_t)0x0004) /*!<Bit 2 */
#define TIM_DCR_DBA_3 ((uint16_t)0x0008) /*!<Bit 3 */

```

```

#define TIM_DCR_DBA_4 ((uint16_t)0x0010) /* !<Bit 4 */

#define TIM_DCR_DBL ((uint16_t)0x1F00) /* !<DBL [4:0] bits (DMA Burst Length) */
#define TIM_DCR_DBL_0 ((uint16_t)0x0100) /* !<Bit 0 */
#define TIM_DCR_DBL_1 ((uint16_t)0x0200) /* !<Bit 1 */
#define TIM_DCR_DBL_2 ((uint16_t)0x0400) /* !<Bit 2 */
#define TIM_DCR_DBL_3 ((uint16_t)0x0800) /* !<Bit 3 */
#define TIM_DCR_DBL_4 ((uint16_t)0x1000) /* !<Bit 4 */

/* Bit definition for TIM_DMAR register *****/
#define TIM_DMAR_DMAB ((uint16_t)0xFFFF) /* !<DMA register for burst accesses */

/* Bit definition for TIM_OR register *****/
#define TIM_OR_TI4_RMP ((uint16_t)0x00C0) /* !<TI4_RMP [1:0] bits (TIM5 Input 4 remap */
#define TIM_OR_TI4_RMP_0 ((uint16_t)0x0040) /* !<Bit 0 */
#define TIM_OR_TI4_RMP_1 ((uint16_t)0x0080) /* !<Bit 1 */
#define TIM_OR_ITR1_RMP ((uint16_t)0x0C00) /* !<ITR1_RMP [1:0] bits (TIM2 Interrupt */
#define TIM_OR_ITR1_RMP_0 ((uint16_t)0x0400) /* !<Bit 0 */
#define TIM_OR_ITR1_RMP_1 ((uint16_t)0x0800) /* !<Bit 1 */

/* Bit definition for USART_SR register *****/
/* Universal Synchronous Asynchronous Receiver Transmitter */
/* Bit definition for USART_SR register *****/
#define USART_SR_PE ((uint16_t)0x0001) /* !<Parity Error */
#define USART_SR_FE ((uint16_t)0x0002) /* !<Framing Error */
#define USART_SR_NE ((uint16_t)0x0004) /* !<Noise Error Flag */
#define USART_SR_ORE ((uint16_t)0x0008) /* !<OverRun Error */
#define USART_SR_IDLE ((uint16_t)0x0010) /* !<IDLE line detected */
#define USART_SR_RXNE ((uint16_t)0x0020) /* !<Read Data Register Not Empty */
#define USART_SR_TC ((uint16_t)0x0040) /* !<Transmission Complete */
#define USART_SR_TXE ((uint16_t)0x0080) /* !<Transmit Data Register Empty */

```

```

#define USART_SR_LBD ((uint16_t)0x0100) /* !<LIN Break Detection Flag */
#define USART_SR_CTS ((uint16_t)0x0200) /* !<CTS Flag */

/******************* Bit definition for USART_DR register *****/
#define USART_BRR_DIV_Fraction ((uint16_t)0x000F) /* !<Data value */
#define USART_BRR_DIV_Mantissa ((uint16_t)0xFFFF) /* !<Data value */

/******************* Bit definition for USART_BRR register *****/
#define USART_CR1_SBK ((uint16_t)0x0001) /* !<Send Break */
#define USART_CR1_RWU ((uint16_t)0x0002) /* !<Receiver wakeup */
#define USART_CR1_RE ((uint16_t)0x0004) /* !<Receiver Enable */
#define USART_CR1_TE ((uint16_t)0x0008) /* !<Transmitter Enable */
#define USART_CR1_IDLEIE ((uint16_t)0x0010) /* !<IDLE Interrupt Enable */
#define USART_CR1_RXNEIE ((uint16_t)0x0020) /* !<RXNE Interrupt Enable */
#define USART_CR1_TCIE ((uint16_t)0x0040) /* !<Transmission Complete Interrupt Enable */
#define USART_CR1_TXEIE ((uint16_t)0x0080) /* !<PE Interrupt Enable */
#define USART_CR1_PEIE ((uint16_t)0x0100) /* !<PE Interrupt Enable */
#define USART_CR1_PS ((uint16_t)0x0200) /* !<Parity Selection */
#define USART_CR1_PCE ((uint16_t)0x0400) /* !<Parity Control Enable */
#define USART_CR1_WAKE ((uint16_t)0x0800) /* !<Wake up method */
#define USART_CR1_M ((uint16_t)0x1000) /* !<Word length */
#define USART_CR1 UE ((uint16_t)0x2000) /* !<USART Enable */
#define USART_CR1_OVER8 ((uint16_t)0x8000) /* !<USART Oversampling by 8 enable */

/******************* Bit definition for USART_CR1 register *****/
#define USART_CR2_ADD ((uint16_t)0x000F) /* !<Address of the USART node */
#define USART_CR2_LBDL ((uint16_t)0x0020) /* !<LIN Break Detection Length */
#define USART_CR2_LBDIE ((uint16_t)0x0040) /* !<LIN Break Detection Interrupt Enable */
#define USART_CR2_LBCL ((uint16_t)0x0100) /* !<Last Bit Clock pulse */
#define USART_CR2_CPHA ((uint16_t)0x0200) /* !<Clock Phase */
#define USART_CR2_CPOL ((uint16_t)0x0400) /* !<Clock Polarity */
#define USART_CR2_CLKEN ((uint16_t)0x0800) /* !<Clock Enable */

```

```

#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)
/* !<LIN mode enable */

/* ***** Bit definition for USART_CR3 register *****/
#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_CTSIE ((uint16_t)0x0200)
#define USART_CR3_CTSIE ((uint16_t)0x0400)
#define USART_CR3_ONEBIT ((uint16_t)0x0800)
/* !<USART One bit method enable */

/* ***** Bit definition for USART_GTPR register *****/
#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)
/* !<Guard time value */

/* ***** Bit definition for USART_GT register *****/
#define USART_GT ((uint16_t)0xFFFF)
/* !<Guard time value */

```

```

/*
 */
/* Window WATCHDOG */

/* Bit definition for WWDG_CR register **** LS
   * !<T[6:0] bits (7-Bit counter (MSB
   * !<Bit 0 */
   * !<Bit 1 */
   * !<Bit 2 */
   * !<Bit 3 */
   * !<Bit 4 */
   * !<Bit 5 */
   * !<Bit 6 */

/* Bit definition for WWDG_CR register **** LS
   * !<W[6:0] bits (7-bit window value) */
   * !<Bit 0 */
   * !<Bit 1 */
   * !<Bit 2 */
   * !<Bit 3 */
   * !<Bit 4 */
   * !<Bit 5 */
   * !<Bit 6 */

/* Bit definition for WWDG_CFR register **** LS
   * !<WDGTB[1:0] bits (Timer Base) */
   * !<Bit 0 */
   * !<Bit 1 */

/* Bit definition for WWDG_SR register **** LS
   * !<Early Wakeup Interrupt */
   * !<Early Wakeup Flag */

/*
 */
/* Bit definition for WWDG_CR register **** LS
#define WWDG_CR_T ((uint8_t)0x7F)
#define WWDG_CR_TO ((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)

/* Bit definition for WWDG_CFR register **** LS
#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)

/* Bit definition for WWDG_SR register **** LS
#define WWDG_SR_EWIF ((uint8_t)0x01)

```

```

/*
 */
/* ***** Bit definition for DBGMCU_IDCODE register *****/
#define DBGMCU_IDCODE_DEV_ID ((uint32_t)0x00000FFF)
#define DBGMCU_IDCODE_REV_ID ((uint32_t)0xFFFFF000)

/* ***** Bit definition for DBGMCU_CR register *****/
#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) /* !<Bit 0 */
#define DBGMCU_CR_TRACE_MODE_1 ((uint32_t)0x00000080) /* !<Bit 1 */

/* ***** Bit definition for DBGMCU_APB1_FZ register *****/
#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_IWDG_STOP ((uint32_t)0x00001000)
#define DBGMCU_APB1_FZ_I2C1_SMBUS_TIMEOUT ((uint32_t)0x00200000)
#define DBGMCU_APB1_FZ_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)
/* Old IWDGSTOP bit definition, maintained for legacy purpose */
#define DBGMCU_APB1_FZ_DBG_IWDEG_STOP DBGMCU_APB1_FZ_DBG_IWDG_STOP

/******************* Bit definition for DBGMCU_APB2_FZ register *****/
#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)

/******************* Ethernet MAC Registers bits definitions *****/
/*
 * Bit definition for Ethernet MAC Control Register register */
#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)0x00008000) /* Duplex mode */

```

```

/* IP Checksum offload */
#define ETH_MACCR_IPCO ((uint32_t)0x000000400) /* IP Checksum offload */
#define ETH_MACCR_RD ((uint32_t)0x000000200) /* Retry disable */
#define ETH_MACCR_APICS ((uint32_t)0x00000080) /* Automatic Pad/CRC stripping */
#define ETH_MACCR_BL ((uint32_t)0x00000060) /* Back-off limit: random integer number (r) of slot time delays before a transmission attempt during retries after a collision: 0 < r <= 10 */

#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) /* Deferal check */
#define ETH_MACCR_TE ((uint32_t)0x00000008) /* Transmitter enable */
#define ETH_MACCR_RE ((uint32_t)0x00000004) /* Receiver enable */

/* Bit definition for Ethernet MAC Frame Filter Register */
#define ETH_MACFFR_RA ((uint32_t)0x80000000) /* Receive all */
#define ETH_MACFFR_HPF ((uint32_t)0x000000400) /* Hash or perfect filter */
#define ETH_MACFFR_SAF ((uint32_t)0x000000200) /* Source address filter enable */
#define ETH_MACFFR_SAIF ((uint32_t)0x000000100) /* SA inverse filtering */
#define ETH_MACFFR_PCF ((uint32_t)0x0000000C0) /* Pass control frames: 3 cases */
#define ETH_MACFFR_BLOCKALL ((uint32_t)0x000000040) /* MAC filters all control frames from receiver */
#define ETH_MACFFR_FORWARDALL ((uint32_t)0x000000080) /* MAC forwards all control frames to application */
#define ETH_MACFFR_FORWARDPASSEDADDRFILTER ((uint32_t)0x0000000C0) /* MAC forwards control frames that pass through MAC filters */
#define ETH_MACFFR_BFD ((uint32_t)0x000000020) /* Broadcast frame disable */
#define ETH_MACFFR_PAM ((uint32_t)0x00000010) /* Pass all multicast */
#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 */

/* Bit definition for Ethernet MAC Hash Table High Register */
#define ETH_MACHTHR_HTH ((uint32-t)0xFFFFFFF) /* Hash table high */

/* Bit definition for Ethernet MAC Hash Table Low Register */
#define ETH_MACHTLR_HTL ((uint32-t)0xFFFFFFF) /* Hash table low */

```

```

/* Bit definition for Ethernet MAC MII Address Register */
#define ETH_MACMIAR_PA ((uint32_t)0x00000F800) /* Physical layer address */
#define ETH_MACMIAR_MR ((uint32_t)0x0000007C0) /* MII register in the selected PHY */
#define ETH_MACMIAR_CR ((uint32_t)0x00000001C) /* CR clock range: 6 cases */
#define ETH_MACMIAR_CR_Div42 ((uint32_t)0x00000000) /* HCLK:60-100 MHz; MDC clock= HCLK/42 */
#define ETH_MACMIAR_CR_Div62 ((uint32_t)0x0000004) /* HCLK :100-150 MHz; MDC clock= HCLK/62 */
#define ETH_MACMIAR_CR_Div16 ((uint32_t)0x0000008) /* HCLK :20-35 MHz; MDC clock= HCLK/16 */
#define ETH_MACMIAR_CR_Div26 ((uint32_t)0x000000C) /* HCLK :35-60 MHz; MDC clock= HCLK/26 */
#define ETH_MACMIAR_CR_Div102 ((uint32_t)0x0000010) /* HCLK :150-168 MHz; MDC clock= HCLK/102 */

#define ETH_MACMIAR_MW ((uint32_t)0x000000002) /* MII write */
#define ETH_MACMIAR_MB ((uint32_t)0x000000001) /* MII busy */

/* Bit definition for Ethernet MAC MII Data Register */
#define ETH_MACMIDR_MD ((uint32_t)0x00000FFF) /* MII data: read/write data from/to PHY */

/* Bit definition for Ethernet MAC Flow Control Register */
#define ETH_MACFCR_PT ((uint32_t)0xFFFFF0000) /* 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_FCBPAA ((uint32_t)0x00000001) /* Flow control busy/backpressure activate */

/* Bit definition for Ethernet MAC VLAN Tag Register */
#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) */

/* Bit definition for Ethernet Wake-UpFrame Filter Register */
#define ETH_MACRMUFFR_D ((uint32_t)0xFFFFFFF) /* Wake-up frame filter register data */
/* Eight sequential Writes to this address (offset 0x28) will write all Wake-UpFrame Filter Registers.

```

```

/*
 * Eight sequential Reads from this address (offset 0x28) will read all Wake-UpFrame Filter Registers. */
/* Wake-UpFrame Filter Reg0 : Filter 0 Byte Mask
 * Wake-UpFrame Filter Reg1 : Filter 1 Byte Mask
 * Wake-UpFrame Filter Reg2 : Filter 2 Byte Mask
 * Wake-UpFrame Filter Reg3 : Filter 3 Byte Mask
 * Wake-UpFrame Filter Reg4 : RSVD - Filter3 Command - RSVD - Filter2 Command -
 *                           RSVD - Filter1 Command - RSVD - Filter0 Command
 * Wake-UpFrame Filter Reg5 : Filter3 Offset - Filter2 Offset - Filter1 Offset - Filter0 Offset
 * Wake-UpFrame Filter Reg6 : Filter1 CRC16 - Filter0 CRC16
 * Wake-UpFrame Filter Reg7 : Filter3 CRC16 - Filter2 CRC16 */

/* Bit definition for Ethernet MAC PMT Control and Status Register */
#define ETH_MACPMTCSR_WFFRPR ((uint32_t)0x80000000) /* Wake-Up Frame Filter Register Pointer Reset */
#define ETH_MACPMTCSR_GU ((uint32_t)0x000000200) /* Global Unicast */
#define ETH_MACPMTCSR_WFR ((uint32_t)0x000000040) /* Wake-Up Frame Received */
#define ETH_MACPMTCSR_MPR ((uint32_t)0x000000020) /* Magic Packet Received */
#define ETH_MACPMTCSR_WFE ((uint32_t)0x000000004) /* Wake-Up Frame Enable */
#define ETH_MACPMTCSR_MPE ((uint32_t)0x000000002) /* Magic Packet Enable */
#define ETH_MACPMTCSR_PD ((uint32_t)0x000000001) /* Power Down */

/* Bit definition for Ethernet MAC Status Register */
#define ETH_MACSR_TSTS ((uint32_t)0x000000200) /* Time stamp trigger status */
#define ETH_MACSR_MMCTS ((uint32_t)0x000000040) /* MMC transmit status */
#define ETH_MACSR_MMCRS ((uint32_t)0x000000020) /* MMC receive status */
#define ETH_MACSR_MMCS ((uint32_t)0x000000010) /* MMC status */
#define ETH_MACSR_PMTS ((uint32_t)0x000000008) /* PMT status */

/* Bit definition for Ethernet MAC Interrupt Mask Register */
#define ETH_MACIMR_TSTIM ((uint32_t)0x000000200) /* Time stamp trigger interrupt mask */
#define ETH_MACIMR_PMTIM ((uint32_t)0x000000008) /* PMT interrupt mask */

/* Bit definition for Ethernet MAC Address0 High Register */
#define ETH_MACAOHR_MACAOH ((uint32_t)0x00000FFF) /* MAC address0 high */
/* Bit definition for Ethernet MAC Address0 Low Register */

```

```

/* Bit definition for Ethernet MAC Address1 High Register */
#define ETH_MACA0LR_MACA0L ((uint32_t)0xFFFFFFFF) /* MAC address0 low */

/* Bit definition for Ethernet MAC Address1 Low Register */
#define ETH_MACA1LR_MACA1L ((uint32_t)0x0000FFFF) /* MAC address1 low */

/* Bit definition for Ethernet MAC Address2 High Register */
#define ETH_MACA2HR_MACA2H ((uint32_t)0x80000000) /* Address enable */
#define ETH_MACA2HR_SA ((uint32_t)0x40000000) /* Source address */
#define ETH_MACA2HR_MBC ((uint32_t)0x3F000000) /* Mask byte control: bits to mask for comparison of the MAC Address high reg bits [15:8] */
#define ETH_MACA1HR_MACA1H ((uint32_t)0x20000000) /* Mask MAC Address high reg bits [7:0] */
#define ETH_MACA1HR_HBIts7_0 ((uint32_t)0x10000000) /* Mask MAC Address high reg bits [31:24] */
#define ETH_MACA1HR_HBIts31_24 ((uint32_t)0x08000000) /* Mask MAC Address low reg bits [23:16] */
#define ETH_MACA1HR_HBIts23_16 ((uint32_t)0x04000000) /* Mask MAC Address low reg bits [7:0] */
#define ETH_MACA1HR_MACA1H ((uint32_t)0x02000000) /* Mask MAC Address low reg bits [15:8] */
#define ETH_MACA1HR_MACA1L ((uint32_t)0x01000000) /* Mask MAC Address low reg bits [15:8] */
#define ETH_MACA1HR_MACA1H ((uint32_t)0x00000FFF) /* MAC address1 high */

/* Bit definition for Ethernet MAC Address2 Low Register */
#define ETH_MACA2LR_MACA2L ((uint32_t)0xFFFFFFFF) /* MAC address2 low */

/* Bit definition for Ethernet MAC Address3 High Register */
#define ETH_MACA3HR_MACA3H ((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_HBits15_8 (((uint32_t)0x20000000) /* Mask MAC Address high reg bits [15:8] */
#define ETH_MACA3HR_HBits7_0 (((uint32_t)0x10000000) /* Mask MAC Address high reg bits [7:0] */
#define ETH_MACA3HR_LBits31_24 (((uint32_t)0x08000000) /* Mask MAC Address low reg bits [31:24] */
#define ETH_MACA3HR_LBits23_16 (((uint32_t)0x04000000) /* Mask MAC Address low reg bits [23:16] */
#define ETH_MACA3HR_LBits15_8 (((uint32_t)0x02000000) /* Mask MAC Address low reg bits [15:8] */
#define ETH_MACA3HR_LBits7_0 (((uint32_t)0x01000000) /* Mask MAC Address low reg bits [7:0] */

/* Bit definition for Ethernet MAC Address3 Low Register */
#define ETH_MACA3LR_MACA3L ((uint32_t)0x0000FFFF) /* MAC address3 low */

/******************* Ethernet MMC Registers bits definition */
/******************* Ethernet MMC Control Register */
#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 */

/* Bit definition for Ethernet MMC Receive Interrupt Register */
#define ETH_MMCRIR_RGUFS ((uint32_t)0x00002000) /* 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 */

/* Bit definition for Ethernet MMC Transmit Interrupt Register */
#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 */

/* Bit definition for Ethernet MMC Receive Interrupt Mask Register */

```

Each frame reaches half the maximum error counter value when Rx good unicast frames counter reaches half the maximum error counter value. STM32 library provides a header file `STM32F4xx.h` which contains definitions for all the registers and their bit fields.

```
#define ETH_MMCRIMR_RGUFM ((uint32_t)0x0000200000) /* Mask the interrupt when Rx good unicast frames counter reaches half the maximum error counter value*/
#define ETH_MMCRIMR_RFAEM ((uint32_t)0x00000040) /* Mask the interrupt when Rx alignment error counter reaches half the maximum error counter value*/
#define ETH_MMCRIMR_RFCEM ((uint32_t)0x00000020) /* Mask the interrupt when Rx crc error counter reaches half the maximum error counter value*/

/* Bit definition for Ethernet MMC Transmit Interrupt Mask Register */
#define ETH_MMCTIMR_TGFM ((uint32_t)0x00200000) /* Mask the interrupt when Tx good frame count counter reaches half the maximum error counter value*/
#define ETH_MMCTIMR_TGFMSCM ((uint32_t)0x00008000) /* Mask the interrupt when Tx good multi col counter reaches half the maximum error counter value*/
#define ETH_MMCTIMR_TGFSCM ((uint32_t)0x00004000) /* Mask the interrupt when Tx good single col counter reaches half the maximum error counter value*/

/* Bit definition for Ethernet MMC Transmitted Good Frames after Single Collision Counter Register */
#define ETH_MMCTGFSSCCR_TGFSCC ((uint32_t)0xFFFFFFF0) /* Number of successfully transmitted frames after a single collision */

/* Bit definition for Ethernet MMC Transmitted Good Frames after More than a Single Collision Counter Register */
#define ETH_MMCTGFMSCCR_TGFMSCC ((uint32_t)0xFFFFFFF0) /* Number of successfully transmitted frames after more than a single collision */

/* Bit definition for Ethernet MMC Transmitted Good Frames Counter Register */
#define ETH_MMCTGFCR_TGFC ((uint32_t)0xFFFFFFFF) /* Number of good frames transmitted */

/* Bit definition for Ethernet MMC Received Frames with CRC Error Counter Register */
#define ETH_MMCRFCECR_RFCEC ((uint32_t)0xFFFFFFFF) /* Number of frames received with CRC error */

/* Bit definition for Ethernet MMC Received Frames with Alignment Error Counter Register */
#define ETH_MMCRFAECR_RFAEC ((uint32_t)0xFFFFFFFF) /* Number of frames received with alignment error */

/* Bit definition for Ethernet MMC Received Good Unicast Frames Counter Register */
#define ETH_MMCRGUFCR_RGUFC ((uint32_t)0xFFFFFFFF) /* Number of good unicast frames received */

/* **** Ethernet PTP Registers bits definition */
/* **** Ethernet PTP Time Stamp Control Register */
#define ETH_PPTSCR_TS_CNT ((uint32_t)0x000030000) /* Time stamp clock node type */
#define ETH_PPTSSR_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 */

/* Bit definition for Ethernet PTP Time Stamp Control Register */
#define ETH_PPTSCR_TS_CNT ((uint32_t)0x000030000) /* Time stamp clock node type */
#define ETH_PPTSSR_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_PPTSSR_TSSIPV4FE ((uint32_t)0x0000020000) /* Time stamp snapshot for IPv4 frames enable */
#define ETH_PPTSSR_TSSIPV6FE ((uint32_t)0x0000010000) /* Time stamp snapshot for IPv6 frames enable */
#define ETH_PPTSSR_TSSTPOEE ((uint32_t)0x00000800) /* Time stamp snapshot for PTP over ethernet frames enable */
#define ETH_PPTSSR_TSPTPPSV2E ((uint32_t)0x00000400) /* Time stamp PTP packet snooping for version2 format enable */
#define ETH_PPTSSR_TSSSR ((uint32_t)0x00000200) /* Time stamp Sub-seconds rollover */
#define ETH_PPTSSR_TSSARFE ((uint32_t)0x00000100) /* Time stamp snapshot for all received frames enable */

#define ETH_PPTSCR_TSARU ((uint32_t)0x00000020) /* Addend register update */
#define ETH_PPTSCR_TSITE ((uint32_t)0x00000010) /* Time stamp interrupt trigger enable */
#define ETH_PPTSCR_TSSTU ((uint32_t)0x00000008) /* Time stamp update */
#define ETH_PPTSCR_TSSTI ((uint32_t)0x00000004) /* Time stamp initialize */
#define ETH_PPTSCR_TSFCU ((uint32_t)0x00000002) /* Time stamp fine or coarse update */
#define ETH_PPTSCR_TSE ((uint32_t)0x00000001) /* Time stamp enable */

/* Bit definition for Ethernet PTP Sub-Second Increment Register */
#define ETH_PTPSSR_STSSI ((uint32_t)0x000000FF) /* System time Sub-second increment value */

/* Bit definition for Ethernet PTP Time Stamp High Register */
#define ETH_PPTSHR_STS ((uint32_t)0xFFFFFFF) /* System Time second */

/* Bit definition for Ethernet PTP Time Stamp Low Register */
#define ETH_PPTSLR_STPNS ((uint32_t)0x80000000) /* System Time Positive or negative time */
#define ETH_PPTSLR_STSS ((uint32_t)0x7FFFFFFF) /* System Time sub-seconds */

/* Bit definition for Ethernet PTP Time Stamp High Update Register */
#define ETH_PPTSHUR_TSUS ((uint32_t)0xFFFFFFFF) /* Time stamp update seconds */

/* Bit definition for Ethernet PTP Time Stamp Low Update Register */
#define ETH_PPTSLUR_TSUPNS ((uint32_t)0x80000000) /* Time stamp update Positive or negative time */
#define ETH_PPTSLUR_TSUSS ((uint32_t)0x7FFFFFFF) /* Time stamp update sub-seconds */

/* Bit definition for Ethernet PTP Time Stamp Addend Register */
#define ETH_PPTTSAR_TSA ((uint32_t)0xFFFFFFFF) /* Time stamp addend */

/* Bit definition for Ethernet PTP Target Time High Register */

```

```

#define ETH_PPTTHR_TTSH ((uint32_t)0xFFFFFFFF) /* Target time stamp high */

/* Bit definition for Ethernet PTP Target Time Low Register */
#define ETH_PPTTLR_TTSL ((uint32_t)0xFFFFFFFF) /* Target time stamp low */

/* Bit definition for Ethernet PTP Time Stamp Status Register */
#define ETH_PPTSSR_TSTR ((uint32_t)0x00000020) /* Time stamp target time reached */
#define ETH_PPTSSR_TSS0 ((uint32_t)0x00000010) /* Time stamp seconds overflow */

/********************* Ethernet DMA Registers bits definition *******/

/* Bit definition for Ethernet DMA Mode Register */
#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 */
#define ETH_DMABMR_RDP_2Beat ((uint32_t)0x00040000) /* maximum number of beats to be transferred in one RxDMA */
#define ETH_DMABMR_RDP_4Beat ((uint32_t)0x00080000) /* maximum number of beats to be transferred in one RxDMA */
#define ETH_DMABMR_RDP_8Beat ((uint32_t)0x00100000) /* maximum number of beats to be transferred in one RxDMA */
#define ETH_DMABMR_RDP_16Beat ((uint32_t)0x00200000) /* maximum number of beats to be transferred in one RxDMA */
#define ETH_DMABMR_RDP_32Beat ((uint32_t)0x00400000) /* maximum number of beats to be transferred in one RxDMA */
#define ETH_DMABMR_RDP_4xPBL_4Beat ((uint32_t)0x01020000) /* maximum number of beats to be transferred in one RxDMA */
#define ETH_DMABMR_RDP_4xPBL_8Beat ((uint32_t)0x01040000) /* maximum number of beats to be transferred in one RxDMA */
#define ETH_DMABMR_RDP_4xPBL_16Beat ((uint32_t)0x01080000) /* maximum number of beats to be transferred in one RxDMA */
#define ETH_DMABMR_RDP_4xPBL_32Beat ((uint32_t)0x01100000) /* maximum number of beats to be transferred in one RxDMA */
#define ETH_DMABMR_RDP_4xPBL_64Beat ((uint32_t)0x01200000) /* maximum number of beats to be transferred in one RxDMA */
#define ETH_DMABMR_RDP_4xPBL_128Beat ((uint32_t)0x01400000) /* maximum number of beats to be transferred in one RxDMA */
#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_PBL ((uint32_t)0x000003F00) /* Rx Tx priority ratio */
#define ETH_DMABMR_PBL_1Beat ((uint32_t)0x00000100) /* Programmable burst length */
#define ETH_DMABMR_PBL_2Beat ((uint32_t)0x00000200) /* maximum number of beats to be transferred in one TxDMA */
#define ETH_DMABMR_PBL_4Beat ((uint32_t)0x00000400) /* maximum number of beats to be transferred in one TxDMA */
#define ETH_DMABMR_PBL_8Beat ((uint32_t)0x00000800) /* maximum number of beats to be transferred in one TxDMA */
#define ETH_DMABMR_PBL_16Beat ((uint32_t)0x00001000) /* maximum number of beats to be transferred in one TxDMA */
#define ETH_DMABMR_PBL_32Beat ((uint32_t)0x00002000) /* maximum number of beats to be transferred in one TxDMA */
#define ETH_DMABMR_PBL_4xPBL_4Beat ((uint32_t)0x01000100) /* maximum number of beats to be transferred in one TxDMA */
#define ETH_DMABMR_PBL_4xPBL_8Beat ((uint32_t)0x01000200) /* maximum number of beats to be transferred in one TxDMA */
#define ETH_DMABMR_PBL_4xPBL_16Beat ((uint32_t)0x01000400) /* maximum number of beats to be transferred in one TxDMA */
#define ETH_DMABMR_PBL_4xPBL_32Beat ((uint32_t)0x01000800) /* maximum number of beats to be transferred in one TxDMA */
#define ETH_DMABMR_PBL_4xPBL_64Beat ((uint32_t)0x01001000) /* maximum number of beats to be transferred in one TxDMA */
#define ETH_DMABMR_PBL_128Beat ((uint32_t)0x01002000) /* maximum number of beats to be transferred in one TxDMA */
#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 */

/* Bit definition for Ethernet DMA Transmit Poll Demand Register */
#define ETH_DMATPDR_TPD ((uint32_t)0xFFFFFFF) /* Transmit poll demand */

/* Bit definition for Ethernet DMA Receive Poll Demand Register */
#define ETH_DMARPDR_RPD ((uint32_t)0x0FFFFFFF) /* Receive poll demand */

/* Bit definition for Ethernet DMA Receive Descriptor List Address Register */
#define ETH_DMARDLAR_SRL ((uint32_t)0x0FFFFFFF) /* Start of receive list */

/* Bit definition for Ethernet DMA Transmit Descriptor List Address Register */
#define ETH_DMATDLAR_STL ((uint32_t)0x08000000) /* Start of transmit list */

/* Bit definition for Ethernet DMA Status Register */
#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 */
/* combination with EBS [2:0] for GetFlagStatus function */
#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 transf , 1-read transf */
#define ETH_DMASR_EBS_DataTransfTX ((uint32_t)0x00800000) /* Error bits O-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 unavailable */
#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_Queueing ((uint32_t)0x000E0000) /* Running - queuing the recieve frame into host mem */

#define ETH_DMASR_NIS ((uint32_t)0x00001000) /* Normal interrupt summary */
#define ETH_DMASR_AIS ((uint32_t)0x00008000) /* Abnormal interrupt summary */
#define ETH_DMASR_ERRORS ((uint32_t)0x00004000) /* Early receive status */
#define ETH_DMASR_FBES ((uint32_t)0x00002000) /* Fatal bus error status */
#define ETH_DMASRETS ((uint32_t)0x0000400) /* Early transmit status */
#define ETH_DMASR_RWITS ((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_TJITS ((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 */

```

```

/* Bit definition for Ethernet DMA Operation Mode Register */
#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)0x00010000) /* Transmit threshold control */

#define ETH_DMAOMR_TTC_64Bytes ((uint32_t)0x00000000) /* threshold level of the MTL Transmit FIFO is 64 Byte
#define ETH_DMAOMR_TTC_128Bytes ((uint32_t)0x00004000) /* threshold level of the MTL Transmit FIFO is 128 Byte
#define ETH_DMAOMR_TTC_192Bytes ((uint32_t)0x00008000) /* threshold level of the MTL Transmit FIFO is 192 Byte
#define ETH_DMAOMR_TTC_256Bytes ((uint32_t)0x0000C000) /* threshold level of the MTL Transmit FIFO is 256 Byte
#define ETH_DMAOMR_TTC_40Bytes ((uint32_t)0x00010000) /* threshold level of the MTL Transmit FIFO is 40 Byte
#define ETH_DMAOMR_TTC_32Bytes ((uint32_t)0x00014000) /* threshold level of the MTL Transmit FIFO is 32 Byte
#define ETH_DMAOMR_TTC_24Bytes ((uint32_t)0x00018000) /* threshold level of the MTL Transmit FIFO is 24 Byte
#define ETH_DMAOMR_TTC_16Bytes ((uint32_t)0x0001C000) /* threshold level of the MTL Transmit FIFO is 16 Byte
#define ETH_DMAOMR_ST ((uint32_t)0x00002000) /* Start/stop transmission command */
#define ETH_DMAOMR_FEF ((uint32_t)0x00000800) /* Forward error frames */
#define ETH_DMAOMR_FUGF ((uint32_t)0x00000400) /* 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 Byte
#define ETH_DMAOMR_RTC_32Bytes ((uint32_t)0x00000008) /* threshold level of the MTL Receive FIFO is 32 Byte
#define ETH_DMAOMR_RTC_96Bytes ((uint32_t)0x00000010) /* threshold level of the MTL Receive FIFO is 96 Byte
#define ETH_DMAOMR_RTC_128Bytes ((uint32_t)0x00000018) /* threshold level of the MTL Receive FIFO is 128 Byte

#define ETH_DMAOMR_OSF ((uint32_t)0x00000004) /* operate on second frame */
#define ETH_DMAOMR_SR ((uint32_t)0x00000002) /* Start/stop receive */

/* Bit definition for Ethernet DMA Interrupt Enable Register */
#define ETH_DMAIER_NISE ((uint32_t)0x00001000) /* 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)0x0000400) /* Early transmit interrupt enable */
#define ETH_DMAIER_RWIE ((uint32_t)0x0000200) /* Receive watchdog timeout interrupt enable */
#define ETH_DMAIER_RPSIE ((uint32_t)0x0000100) /* Receive process stopped interrupt enable */

```

```

#define ETH_DMAIER_RBUIE ((uint32_t)0x000000080) /* Receive buffer unavailable interrupt enable */
#define ETH_DMAIER_RIE ((uint32_t)0x000000040) /* Receive interrupt enable */
#define ETH_DMAIER_TUIE ((uint32_t)0x000000020) /* Transmit Underflow interrupt enable */
#define ETH_DMAIER_ROIE ((uint32_t)0x000000010) /* 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)0x000000002) /* Transmit process stopped interrupt enable */
#define ETH_DMAIER_TIE ((uint32_t)0x000000001) /* Transmit interrupt enable */

/* Bit definition for Ethernet DMA Missed Frame and Buffer Overflow Counter Register */
#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_0MFC ((uint32_t)0x00010000) /* Overflow bit for missed frame counter */
#define ETH_DMAMFBOCR_MFC ((uint32_t)0x00000FFF) /* Number of frames missed by the controller */

/* Bit definition for Ethernet DMA Current Host Transmit Descriptor Register */
#define ETH_DMACHTDR_HTDAP ((uint32_t)0xFFFFFFFF) /* Host transmit descriptor address pointer */

/* Bit definition for Ethernet DMA Current Host Receive Descriptor Register */
#define ETH_DMACHRDR_HRDAP ((uint32_t)0xFFFFFFFF) /* Host receive descriptor address pointer */

/* Bit definition for Ethernet DMA Current Host Transmit Buffer Address Register */
#define ETH_DMACHTBAR_HTBAP ((uint32_t)0xFFFFFFFF) /* Host transmit buffer address pointer */

/* Bit definition for Ethernet DMA Current Host Receive Buffer Address Register */
#define ETH_DMACHRBAR_HRBAP ((uint32_t)0xFFFFFFFF) /* Host receive buffer address pointer */

*/
* @}
*/
/*
* @}
*/

```

```
#ifdef USE_STDPERIPH_DRIVER
#include "stm32f4xx_conf.h"
#endif /* USE_STDPERIPH_DRIVER */

/** @addtogroup Exported_macro
 * @{
 */

#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))) |

/*
 * @}
 */

#ifndef __cplusplus
#endif /* __cplusplus */

#endif /* __STM32F4xx_H */
```

```
/*
 * @}
 */
***** (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
```

## Глава 22

### USB client/host

# **Часть VIII**

## **Ядро Cortex-Mx**

## Глава 23

### Режимы ARM и Thumb

## **Глава 24**

### **DMA**

# Глава 25

## DSP /Cortex-M3/

## Глава 26

### FPU /Cortex-M4F/

# **Часть IX**

## **Интерфейсы**

## Глава 27

### USB

# Глава 28

## UART

Глава 29

SPI

# Глава 30

## I2C

# Глава 31

## CAN

# **Часть X**

## **Операционные системы ОСРВ**

## Глава 32

### Keil RTX

# Глава 33

## FreeRTOS

## Глава 34

eCos

# Глава 35

## Linux

подробно рассмотрен в отдельном разделе XIII

# **Часть XI**

## **Стек TCP/IP**

## Глава 36

### Ethernet

# Глава 37

PPP

## **Часть XII**

### **Типовые применения**

# Глава 38

## GPS

### 38.1 Протокол NMEA 0183

NMEA 0183— текстовый протокол связи морского и навигационного оборудования.

[http://www8.garmin.com/support/pdf/NMEA\\_0183.pdf](http://www8.garmin.com/support/pdf/NMEA_0183.pdf)  
[http://www8.garmin.com/support/pdf/NMEA\\_0183.pdf](http://www8.garmin.com/support/pdf/NMEA_0183.pdf)  
[http://ru.wikipedia.org/wiki/NMEA\\_0183](http://ru.wikipedia.org/wiki/NMEA_0183)  
<http://www.robosoft.info/ru/technologies/knowledgebase/nmea0183>

Датум *WGS<sup>84</sup>*

[http://ru.wikipedia.org/wiki/WGS\\_84](http://ru.wikipedia.org/wiki/WGS_84)

Большинство GPS приемников отдают данные по NMEA 0183, но вместо режима последовательного порта 4800 8N1 прописанного в протоколе, могут использовать другие режимы, характерные для портов RS232 (9600, 115200). Координаты передаются в датуме *WGS<sup>84</sup>*.

Формат сообщения NMEA<sup>1</sup>:

\\$[A-Z]5(,<data>)+\\*[0-9A-F]2<CR><LF>

- символ \$
- 5-буквенный идентификатор сообщения. Первые две буквы — идентификатор источника сообщения, следующие три буквы — идентификатор формата сообщения
  - GPGGA — данные о последнем определении местоположения
  - GPGLL — координаты, широта/долгота
  - GPGSA — DOP (GPS) и активные спутники
  - GPGSV — наблюдаемые спутники
  - GPWPL — параметры заданной точки
  - GPBOD — азимут одной точки относительно другой
  - GPRMB — рекомендуемый минимум навигационных данных для достижения заданной точки

---

<sup>1</sup>регулярные выражения [http://en.wikipedia.org/wiki/Regular\\_expression](http://en.wikipedia.org/wiki/Regular_expression)

- GPRMC — рекомендуемый минимум навигационных данных: информацию о времени, местоположении (*WGS<sup>84</sup>*), курсе и скорости, передаваемые навигационным GPS приёмником. Контрольная сумма обязательна для этого сообщения, интервалы передачи не должны превышать 2 секунды.
- GPRTE — маршруты
- HCHDG — данные от компаса
- блоки данных, разделённых запятыми. Пустые данные, не заданные в середине строки, оставляют запятыю. Пустые данные в конце строки могут отбрасываться вместе с запятой.
- символ \*
- двузначное hex число — контрольная XOR-сумма всех байт в строке между \$ и \*.
- конец строки <CR><LF> = 0x0D 0x0A = \r\n

## 38.2 \$GPRMC — рекомендуемый минимум навигационных данных

\$GPRMC, hhmmss.ss, [AV], GGMM.MM, [NS], gggmm.mm, [EW], v.v, b.b, ddmmyy, x.x, n, m\*hh<CR><LF>

- GP — приём сигналов GPS, GN — ГЛОНАСС
- RMC — Recommended Minimum sentence C
- hhmmss.ss — время фиксации местоположения по времени UTC
- A — данные достоверны, V — данные недостоверны
- GGMM.MM — широта, 2 цифры градусов, 2 цифры целых минут, точка и дробная часть минут;
- N/S — северная/южная широта
- gggmm.mm — долгота, 3 цифры градусов, 2 цифры целых минут, точка и дробная часть минут;
- E/W — восточная/западная долгота
- v.v — горизонтальная составляющая скорости в узлах
- b.b — путевой угол (направление скорости) в градусах: 0° север, 90° восток, 180° юг, 270° запад
- ddmmyy — дата
- x.x — магнитное склонение в градусах (часто отсутствует)
- n — направление магнитного склонения: для получения магнитного курса магнитное склонение необходимо вычесть (E) или прибавить (W) к истинному курсу
- m — индикатор режима: A — автономный, D — дифференциальный, E — аппроксимация, N — недостоверные данные (часто отсутствует, данное поле включая запятую отсутствует в старых версиях NMEA)

### 38.3 Системы координат (датум)

<http://ne-grusti.narod.ru/Glossary/datums.html>

Системы координат (datums) можно разделить на геоцентрические и топоцентрические.

В геоцентрической системе размеры эллипсоида, ориентация и положение его центра выбираются следующим образом:

- объем эллипсоида предполагается равным объему геоида;
- большая полуось эллипсоида лежит в плоскости экватора геоида;
- малая полуось направлена по оси вращения Земли;
- среднеквадратичное отклонение поверхности эллипсоида от поверхности геоида
- минимально по всей территории земного шара.

*WGS<sup>72</sup>* и сменившая ее *WGS<sup>84</sup>*, а также российская *SGS<sup>85</sup>* являются геоцентрическими системами координат на эллипсоидах *WGS72*, *GRS80* и *SGS85* соответственно. В системе GPS/NAVSTAR используется *WGS<sup>84</sup>*, а в системе GLONASS — *SGS<sup>85</sup>*.

Топоцентрическая (национальная) система координат появляется так: вы берете некоторый эллипсоид и располагаете его таким образом, чтобы для заданной территории среднеквадратичное отклонение поверхности эллипсоида от поверхности геоида было минимальным. При этом остальная часть мира вас не интересует: отклонения на другой стороне Земли может быть сколь угодно велико.

В России используются несколько геодезических систем координат: Пулково<sup>1942</sup> Пулково 1942 г., 1963 г. и 1991 г. Система координат 1963 г. используется военными и ее параметры преобразования засекречены. Обычно мы пользуемся картами, составленными в системе Пулково<sup>1942</sup>. Она базируется на эллипсоиде Красовского.

Параметры преобразования для Пулково<sup>1942</sup>:

#### Преобразование Bursa-Wolf (Position Vector Transformation)

<http://ne-grusti.narod.ru/Glossary/transformations.html#pvt>

направление	dX	dY	dZ	rX	rY	rZ	M	источник
<i>WGS84</i> → 1942	-27.0	+135.0	+84.5	0.0	0.0	0.554	-0.2263	Data +
1942 → <i>WGS84</i>	+25.0	-141.0	-78.5	0.0	0.35	0.736	0.0	Stefan A. Voser

#### Преобразование Молоденского

<http://ne-grusti.narod.ru/Glossary/transformations.html#molodensky>

направление	dX	dY	dZ	da	df	источник
1942 → <i>WGS84</i>	+28.0	-130.0	-95.0	-108.0	+0.00480795	Vladimir Zh. Boston-PC Forum
1942 → <i>WGS84</i>	+28.0	-130.0	-95.0	-108.0	+0.00480795	MADTRAN, Peter H. Dana
1942 → <i>WGS84</i>	+24.0	-123.0	-94.0	-108.0	+0.00480795	Stefan A. Voser

## 38.4 Методы преобразования систем координат

<http://ne-grusti.narod.ru/Glossary/transformations.html>

Часто требуется перейти от одной системы координат (datum) к другой. Например, вы хотите данные с GPS в системе координат  $WGS^{84}$  наложить на карту в системе координат Пулково<sup>1942</sup>.

Произодить преобразование проще (математически), если сначала перевести координаты из географических (широта и долгота) в прямоугольные (XYZ).

Чтобы получить точные прямоугольные координаты точки на поверхности Земли, необходимо знать не только широту и долготу, но и ее высоту над поверхностью эллипсоида. GPS показывает высоту над эллипсоидом  $WGS^{84}$ . Можно вычислить прямоугольные координаты только по широте и долготе, считая, что точка лежит на поверхности эллипса, но при этом точность понизится.

Высоту точки на другими эллипсоядами получить сложнее. Обычно мы знаем высоту в национальной системе высот. За нулевую высоту, как правило, принимается среднее значение уровня моря в определенной точке побережья по результатам многолетних наблюдений. Высоты в этой системе измеряются геодезическими методами относительно поверхности геоида. Поэтому, чтобы узнать высоту точки над эллипсоядом, нужно знать возвышение геоида над эллипсоядом в данном месте, которое трудно вычислить с хорошей точностью. Существуют различные математические модели геоида для разных территорий и для всего земного шара, которые постоянно уточняются.

На заре спутниковой геодезии, когда взаимосвязи между системами координат не были четко определены, а данные исследований были не очень точны, применяли простой сдвиг начала координат  $dX$ ,  $dY$ ,  $dZ$  для перехода от одной системы координат к другой. Это предполагало, что направления осей двух эллипсоядов параллельны (что во многих случаях не соответствует действительности). Для работ на небольшой территории погрешности, вносимые этим предположением, были меньше, чем точность самих данных. Однако, по мере накопления и уточнения данных и повышения точности измерений, стало очевидно, что преобразование по трем параметрам не подходит для больших территорий и глобального использования, если требуется максимальная точность и единый набор параметров преобразования.

Простейший метод — сдвиг центра координат прямоугольной системы, предполагая, что оси исходной и целевой систем координат параллельны.

$$\begin{pmatrix} X_B \\ Y_B \\ Z_B \end{pmatrix} = \begin{pmatrix} X_A \\ Y_A \\ Z_A \end{pmatrix} + \begin{pmatrix} dX \\ dY \\ dZ \end{pmatrix} \quad (38.1)$$

Молоденский разработал формулы для применения параметров сдвига к географическим координатам.

$$\Delta\varphi'' = \frac{206265}{\rho} (-dX \sin \varphi \cos \lambda - dY \sin \varphi \sin \lambda + dZ \cos \varphi + [a\Delta f + f\Delta a] \sin 2\varphi) \quad (38.2)$$

$$\Delta\lambda'' = \frac{206265}{\nabla \cos \varphi} (-dX \sin \lambda + dY \cos \lambda) \quad (38.3)$$

$$\Delta h = dX \cos \varphi \cos \lambda + dY \cos \varphi \sin \lambda + dZ \sin \varphi + (a\Delta f + f\Delta a) \sin^2 \varphi - \Delta a \quad (38.4)$$

здесь  $dX$ ,  $dY$ ,  $dZ$  — сдвиг по осям, м;  $a$  и  $f$  — большая полуось и сжатие исходного эллипса;  $da$  и  $df$  — разности между большой полуосью и сжатием исходного эллипса и целевого эллипса.

Повышенная точность достигается преобразованием Хелмерта с семью параметрами. Есть две его разновидности, различающиеся присвоением знака для параметров поворота.

### 1. Position Vector Transformation (Bursa-Wolf)

$$\begin{pmatrix} X_B \\ Y_B \\ Z_B \end{pmatrix} = M \cdot \begin{pmatrix} 1 & -Rz & +Ry \\ +Rz & 1 & -Rx \\ -Ry & +Rx & 1 \end{pmatrix} \begin{pmatrix} X_A \\ Y_A \\ Z_A \end{pmatrix} + \begin{pmatrix} dX \\ dY \\ dZ \end{pmatrix} \quad (38.5)$$

### 2. Coordinate Frame Transformation

$$\begin{pmatrix} X_B \\ Y_B \\ Z_B \end{pmatrix} = M \cdot \begin{pmatrix} 1 & +Rz & -Ry \\ -Rz & 1 & +Rx \\ +Ry & -Rx & 1 \end{pmatrix} \begin{pmatrix} X_A \\ Y_A \\ Z_A \end{pmatrix} + \begin{pmatrix} dX \\ dY \\ dZ \end{pmatrix} \quad (38.6)$$

Здесь  $M$  — это масштаб (scale), параметры берутся из описания системы координат.

## 38.5 Геоид и эллипсоиды

<http://ne-grusti.narod.ru/Glossary/ellipsoid-geoid.html>

**Геоид** — фигура сложной формы, образованная поверхностью уровня вод Мирового океана, продолженной под материками. Эта поверхность во всех точках перпендикулярна (нормальна) вектору силы тяжести. Отвес направлен перпендикулярно поверхности геоида, а не к центру Земли! Это связано с тем, что плотность Земли распределена неравномерно.

**Эллипсоид** — тело, полученное вращением эллипса вокруг его малой оси. Размеры подбирают так, чтобы среднеквадратичное отклонение от поверхности геоида было минимально либо по всей поверхности Земли, либо для заданной территории.

Параметры некоторых эллипсоидов

эллипсоид	использование	большая полуось a, м	малая полуось b, м	сжатие $f = (a - b)/a$
Красовского (1940)	Россия и др. Пулково <sup>1942</sup>	6378245	6356863	1/298.3
GRS80	международный <i>WGS<sup>84</sup></i>	6378137	6356752.31425	1/298.25722356
SGS85	ГЛОНАСС SGS85			
Бесселя	СССР до 1942 г.			

В таблицах эллипсоидов часто указывается не полярное сжатие  $f$ , а обратная величина  $1/f$ , например, для эллипсоида Красовского  $1/f = 298.3$ .

Отклонения эллипсоида Красовского от геоида на территории СНГ не превышают 150 м.

## 38.6 Tistar15



Цена: 250 руб. <http://www.voltmaster.ru/cgi-bin/qquery.pl?id=127000056703&group=700000>

## 38.7 WISMO228



Цена: 1070 руб. <http://www.voltmaster.ru/cgi-bin/qquery.pl?id=127000881980&group=700000>

# Глава 39

## GSM

### 39.1 WISMO228



Цена: 1070 руб. <http://www.voltmaster.ru/cgi-bin/qwery.pl?id=127000881980&group=700000>

# Глава 40

## шина Dallas 1Wire

40.1 RTC

40.2 Датчики температуры DS18x20

# **Часть XIII**

## **Встраиваемый Linux**

## **Часть XIV**

### **QA**

**Где перевод пунктов меню** Если вы собираетесь заниматься разработкой embedded ПО, знание английского на уровне пользования online словарями и Google Translate обязательно. Кому это не нравится, могут и дальше ждать перевод datasheetов и programmer's manualов на снимаемые с производства компоненты — перевод как раз появляются к этому времени.

# **Часть XV**

## **Приложения**

# Глава 41

## Сводная таблица процессоров

	ядро Cortex-	MHz	Flash	SRAM	корпус LQFP	USB	UART	SPI	CAN
STM32F100C4T6B	M3	24	16K	4K	48		2	1	
STM32F100RBT	M3	24	128K	8K	100		1		
STM32F103C8T	M3	72	64K	20K	48	1	3	2	1
STM32F103CBT	M3	72	128K	20K	48	1	3	2	1
STM32F407VGT	M4F	168	1M	192K	144	2	6		2
STM32F407IGT	M4F	168	1M	192K	176	2	8		2
STM32F427IIT	M4F	168	2M	256K	176	2	8		2

### 41.1 STM32F10x

#### STM32F103C8T

#### STM32F103CBT

	Cortex-M3
Flash	64K/128K
SRAM	20K
	LQFP48
WDG	2
Timers	2x16b
	RTC
ADC	10x12b
GPIO	36
DAC	no
SPI	2
I2C	2
USART	3
USB	1
CAN	1

**STM32F100C4T6B**

Ядро	Cortex-M3
Flash	16K
SRAM	4K
16-битные таймеры	6
таймеры ШИМ	3
RTC	да
UART	2
SPI	1
I2C	1
DMA	1 канал
АЦП	10x12 бит
ЦАП	2x12 бит
корпус	LQFP48