My Project

Generated by Doxygen 1.9.1

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

File Index

3.1 File List

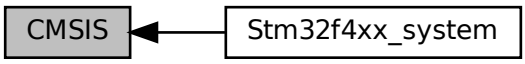
Here is a list of all documented files with brief descriptions:

Chapter 4

Module Documentation

4.1 CMSIS

Collaboration diagram for CMSIS:



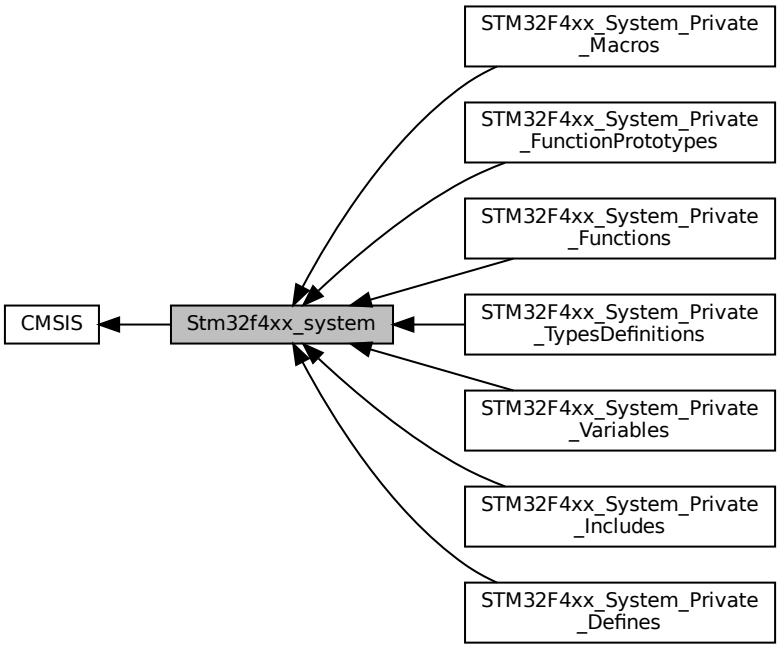
Modules

- Stm32f4xx_system

4.1.1 Detailed Description

4.2 Stm32f4xx_system

Collaboration diagram for Stm32f4xx_system:



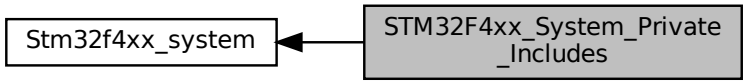
Modules

- STM32F4xx_System_Private_Includes
- STM32F4xx_System_Private_TypesDefinitions
- STM32F4xx_System_Private_Defines
- STM32F4xx_System_Private_Macros
- STM32F4xx_System_Private_Variables
- STM32F4xx_System_Private_FunctionPrototypes
- STM32F4xx_System_Private_Functions

4.2.1 Detailed Description

4.3 STM32F4xx_System_Private_Includes

Collaboration diagram for STM32F4xx_System_Private_Includes:



Macros

- #define HSE_VALUE ((uint32_t)25000000)
- #define HSI_VALUE ((uint32_t)16000000)

4.3.1 Detailed Description

4.3.2 Macro Definition Documentation

4.3.2.1 HSE_VALUE

```
#define HSE_VALUE ((uint32_t)25000000)
```

Default value of the External oscillator in Hz

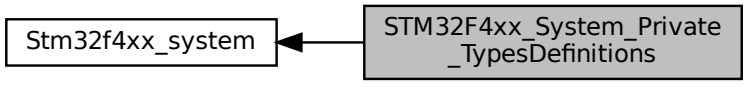
4.3.2.2 HSI_VALUE

```
#define HSI_VALUE ((uint32_t)16000000)
```

Value of the Internal oscillator in Hz

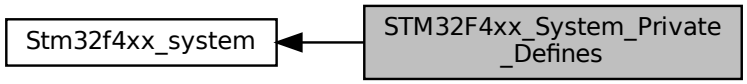
4.4 STM32F4xx_System_Private_TypesDefinitions

Collaboration diagram for STM32F4xx_System_Private_TypesDefinitions:



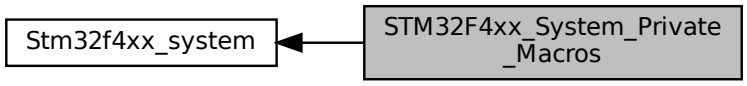
4.5 STM32F4xx_System_Private_Defines

Collaboration diagram for STM32F4xx_System_Private_Defines:



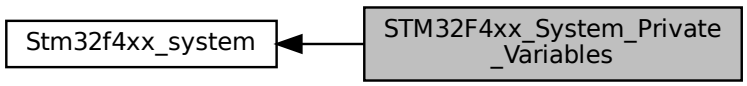
4.6 STM32F4xx_System_Private_Macros

Collaboration diagram for STM32F4xx_System_Private_Macros:



4.7 STM32F4xx_System_Private_Variables

Collaboration diagram for STM32F4xx_System_Private_Variables:



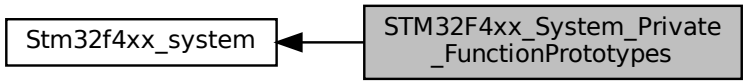
Variables

- uint32_t **SystemCoreClock** = 16000000
- const uint8_t **AHBPrescTable** [16] = {0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 3, 4, 6, 7, 8, 9}
- const uint8_t **APBPrescTable** [8] = {0, 0, 0, 0, 1, 2, 3, 4}

4.7.1 Detailed Description

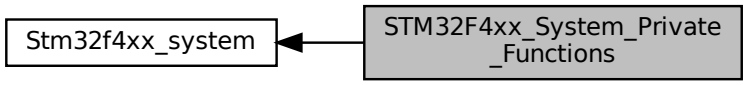
4.8 STM32F4xx_System_Private_FunctionPrototypes

Collaboration diagram for STM32F4xx_System_Private_FunctionPrototypes:



4.9 STM32F4xx_System_Private_Functions

Collaboration diagram for STM32F4xx_System_Private_Functions:



Functions

- void SystemInit (void)

  Setup the microcontroller system Initialize the FPU setting, vector table location and External memory configuration.
- void SystemCoreClockUpdate (void)

  Update SystemCoreClock variable according to Clock Register Values. The SystemCoreClock variable contains the core clock (HCLK), it can be used by the user application to setup the SysTick timer or configure other parameters.

4.9.1 Detailed Description

4.9.2 Function Documentation

4.9.2.1 SystemCoreClockUpdate()

```
void SystemCoreClockUpdate (
            void  )
```

Update SystemCoreClock variable according to Clock Register Values. The SystemCoreClock variable contains the core clock (HCLK), it can be used by the user application to setup the SysTick timer or configure other parameters.

Note

Each time the core clock (HCLK) changes, this function must be called to update SystemCoreClock variable value. Otherwise, any configuration based on this variable will be incorrect.

- The system frequency computed by this function is not the real frequency in the chip. It is calculated based on the predefined constant and the selected clock source:

- If SYSCLK source is HSI, SystemCoreClock will contain the HSI_VALUE(∗)
- If SYSCLK source is HSE, SystemCoreClock will contain the HSE_VALUE(∗∗)
- If SYSCLK source is PLL, SystemCoreClock will contain the HSE_VALUE(∗∗) or HSI_VALUE(∗) multiplied/divided by the PLL factors.

(∗) HSI_VALUE is a constant defined in stm32f4xx_hal_conf.h file (default value 16 MHz) but the real value may vary depending on the variations in voltage and temperature.

(∗∗) HSE_VALUE is a constant defined in stm32f4xx_hal_conf.h file (its value depends on the application requirements), user has to ensure that HSE_VALUE is same as the real frequency of the crystal used. Otherwise, this function may have wrong result.

- The result of this function could be not correct when using fractional value for HSE crystal.

Parameters

None	

Return values

None	

4.9.2.2 SystemInit()

```
void SystemInit (
            void  )
```

Setup the microcontroller system Initialize the FPU setting, vector table location and External memory configuration.

Parameters

None	

Return values

None	

Chapter 5

Class Documentation

5.1 gps Struct Reference

Public Attributes

- char **gps_buffer** [GPS_BUFFER]
- uint8_t **_longitude_attitude**
- float **_longitude**
- uint8_t **_latitude_attitude**
- float **_latitude**

The documentation for this struct was generated from the following file:

- Core/Inc/gps.h

Chapter 6

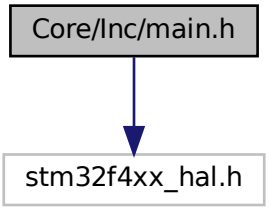
File Documentation

6.1 Core/Inc/main.h File Reference

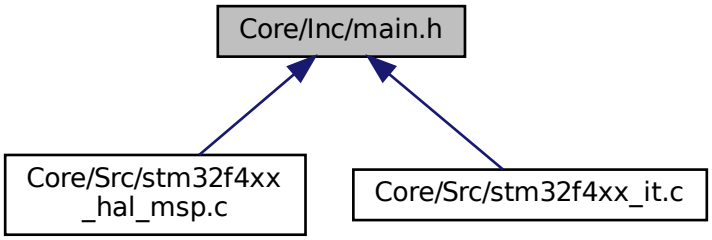
: Header for main.c file. This file contains the common defines of the application.

```
#include "stm32f4xx_hal.h"
```

Include dependency graph for main.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define **USER_Btn_Pin** GPIO_PIN_13
- #define **USER_Btn_GPIO_Port** GPIOC
- #define **MCO_Pin** GPIO_PIN_0
- #define **MCO_GPIO_Port** GPIOH
- #define **RMII_MDC_Pin** GPIO_PIN_1
- #define **RMII_MDC_GPIO_Port** GPIOC
- #define **RMII_REF_CLK_Pin** GPIO_PIN_1
- #define **RMII_REF_CLK_GPIO_Port** GPIOA
- #define **RMII_MDIO_Pin** GPIO_PIN_2
- #define **RMII_MDIO_GPIO_Port** GPIOA
- #define **RMII_CRS_DV_Pin** GPIO_PIN_7
- #define **RMII_CRS_DV_GPIO_Port** GPIOA
- #define **RMII_RXD0_Pin** GPIO_PIN_4
- #define **RMII_RXD0_GPIO_Port** GPIOC
- #define **RMII_RXD1_Pin** GPIO_PIN_5
- #define **RMII_RXD1_GPIO_Port** GPIOC
- #define **LD1_Pin** GPIO_PIN_0
- #define **LD1_GPIO_Port** GPIOB
- #define **RMII_TXD1_Pin** GPIO_PIN_13
- #define **RMII_TXD1_GPIO_Port** GPIOB
- #define **LD3_Pin** GPIO_PIN_14
- #define **LD3_GPIO_Port** GPIOB
- #define **STLK_RX_Pin** GPIO_PIN_8
- #define **STLK_RX_GPIO_Port** GPIOD
- #define **STLK_TX_Pin** GPIO_PIN_9
- #define **STLK_TX_GPIO_Port** GPIOD
- #define **USB_PowerSwitchOn_Pin** GPIO_PIN_6
- #define **USB_PowerSwitchOn_GPIO_Port** GPIOG
- #define **USB_OverCurrent_Pin** GPIO_PIN_7
- #define **USB_OverCurrent_GPIO_Port** GPIOG
- #define **USB_SOF_Pin** GPIO_PIN_8
- #define **USB_SOF_GPIO_Port** GPIOA
- #define **USB_VBUS_Pin** GPIO_PIN_9
- #define **USB_VBUS_GPIO_Port** GPIOA
- #define **USB_ID_Pin** GPIO_PIN_10
- #define **USB_ID_GPIO_Port** GPIOA
- #define **USB_DM_Pin** GPIO_PIN_11
- #define **USB_DM_GPIO_Port** GPIOA
- #define **USB_DP_Pin** GPIO_PIN_12
- #define **USB_DP_GPIO_Port** GPIOA
- #define **TMS_Pin** GPIO_PIN_13
- #define **TMS_GPIO_Port** GPIOA
- #define **TCK_Pin** GPIO_PIN_14
- #define **TCK_GPIO_Port** GPIOA
- #define **RMII_TX_EN_Pin** GPIO_PIN_11
- #define **RMII_TX_EN_GPIO_Port** GPIOG
- #define **RMII_TXD0_Pin** GPIO_PIN_13
- #define **RMII_TXD0_GPIO_Port** GPIOG
- #define **LD2_Pin** GPIO_PIN_7
- #define **LD2_GPIO_Port** GPIOB

Functions

- void Error_Handler (void)

  This function is executed in case of error occurrence.

6.1.1 Detailed Description

: Header for main.c file. This file contains the common defines of the application.

Attention

6.1.2 Function Documentation

6.1.2.1 Error_Handler()

```
void Error_Handler (
             void  )
```

This function is executed in case of error occurrence.

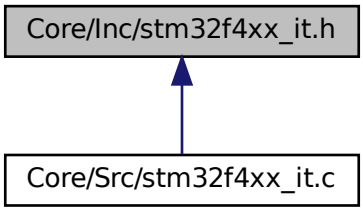
Return values

None	

6.2 Core/Inc/stm32f4xx_it.h File Reference

This file contains the headers of the interrupt handlers.

This graph shows which files directly or indirectly include this file:



Functions

- void NMI_Handler (void)

  This function handles Non maskable interrupt.
- void HardFault_Handler (void)

  This function handles Hard fault interrupt.
- void MemManage_Handler (void)

  This function handles Memory management fault.
- void BusFault_Handler (void)

  This function handles Pre-fetch fault, memory access fault.
- void UsageFault_Handler (void)

  This function handles Undefined instruction or illegal state.
- void DebugMon_Handler (void)

  This function handles Debug monitor.
- void TIM1_UP_TIM10_IRQHandler (void)

  This function handles TIM1 update interrupt and TIM10 global interrupt.

6.2.1 Detailed Description

This file contains the headers of the interrupt handlers.

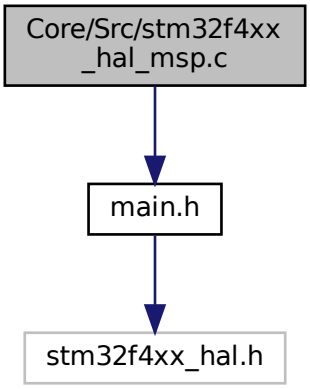
Attention

6.3 Core/Src/stm32f4xx_hal_msp.c File Reference

This file provides code for the MSP Initialization and de-Initialization codes.

```
#include "main.h"
```

Include dependency graph for stm32f4xx_hal_msp.c:



Functions

- void HAL_MspInit (void)
- void HAL_ETH_MspInit (ETH_HandleTypeDef ∗heth)

  ETH MSP Initialization This function configures the hardware resources used in this example.
- void HAL_ETH_MspDeInit (ETH_HandleTypeDef ∗heth)

  ETH MSP De-Initialization This function freeze the hardware resources used in this example.
- void HAL_UART_MspInit (UART_HandleTypeDef ∗huart)

  UART MSP Initialization This function configures the hardware resources used in this example.
- void HAL_UART_MspDeInit (UART_HandleTypeDef ∗huart)

  UART MSP De-Initialization This function freeze the hardware resources used in this example.
- void HAL_PCD_MspInit (PCD_HandleTypeDef ∗hpcd)

  PCD MSP Initialization This function configures the hardware resources used in this example.
- void HAL_PCD_MspDeInit (PCD_HandleTypeDef ∗hpcd)

  PCD MSP De-Initialization This function freeze the hardware resources used in this example.

6.3.1 Detailed Description

This file provides code for the MSP Initialization and de-Initialization codes.

Attention

6.3.2 Function Documentation

6.3.2.1 HAL_ETH_MspDeInit()

```
void HAL_ETH_MspDeInit (
            ETH_HandleTypeDef * heth )
```

ETH MSP De-Initialization This function freeze the hardware resources used in this example.

Parameters

heth	ETH handle pointer

Return values

None	

ETH GPIO Configuration PC1 ------> ETH_MDC PA1 ------> ETH_REF_CLK PA2 ------> ETH_MDIO PA7 ------> ETH_CRS_DV PC4 ------> ETH_RXD0 PC5 ------> ETH_RXD1 PB13 ------> ETH_TXD1 PG11 ------> ETH_TX↩ _EN PG13 ------> ETH_TXD0

6.3.2.2 HAL_ETH_MspInit()

```
void HAL_ETH_MspInit (
            ETH_HandleTypeDef * heth )
```

ETH MSP Initialization This function configures the hardware resources used in this example.

Parameters

heth	ETH handle pointer

Return values

None	

ETH GPIO Configuration PC1 ------> ETH_MDC PA1 ------> ETH_REF_CLK PA2 ------> ETH_MDIO PA7 ------> ETH_CRS_DV PC4 ------> ETH_RXD0 PC5 ------> ETH_RXD1 PB13 ------> ETH_TXD1 PG11 ------> ETH_TX↩ _EN PG13 ------> ETH_TXD0

6.3.2.3 HAL_MspInit()

```
void HAL_MspInit (
            void  )
```

Initializes the Global MSP.

6.3.2.4 HAL_PCD_MspDeInit()

```
void HAL_PCD_MspDeInit (
            PCD_HandleTypeDef * hpcd )
```

PCD MSP De-Initialization This function freeze the hardware resources used in this example.

Parameters

hpcd	PCD handle pointer

Return values

None	

USB_OTG_FS GPIO Configuration PA8 ----—> USB_OTG_FS_SOF PA9 ----—> USB_OTG_FS_VBUS PA10 ----—> USB_OTG_FS_ID PA11 ----—> USB_OTG_FS_DM PA12 ----—> USB_OTG_FS_DP

6.3.2.5 HAL_PCD_MspInit()

```
void HAL_PCD_MspInit (
            PCD_HandleTypeDef * hpcd )
```

PCD MSP Initialization This function configures the hardware resources used in this example.

Parameters

hpcd	PCD handle pointer

Return values

None	

USB_OTG_FS GPIO Configuration PA8 ----—> USB_OTG_FS_SOF PA9 ----—> USB_OTG_FS_VBUS PA10 ----—> USB_OTG_FS_ID PA11 ----—> USB_OTG_FS_DM PA12 ----—> USB_OTG_FS_DP

6.3.2.6 HAL_UART_MspDeInit()

```
void HAL_UART_MspDeInit (
            UART_HandleTypeDef * huart )
```

UART MSP De-Initialization This function freeze the hardware resources used in this example.

Parameters

huart	UART handle pointer

Return values

None	

USART3 GPIO Configuration PD8 ----—> USART3_TX PD9 ----—> USART3_RX

6.3.2.7 HAL_UART_MspInit()

```
void HAL_UART_MspInit (
            UART_HandleTypeDef * huart )
```

UART MSP Initialization This function configures the hardware resources used in this example.

Parameters

huart	UART handle pointer

Return values

None	

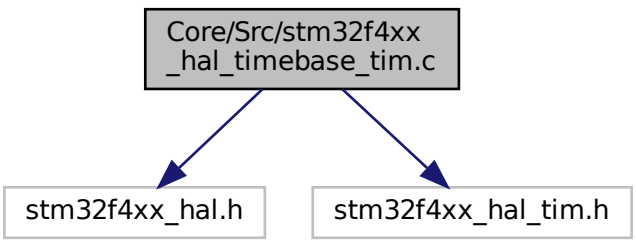
USART3 GPIO Configuration PD8 ----—> USART3_TX PD9 ----—> USART3_RX

6.4 Core/Src/stm32f4xx_hal_timebase_tim.c File Reference

HAL time base based on the hardware TIM.

```
#include "stm32f4xx_hal.h"
#include "stm32f4xx_hal_tim.h"
```

Include dependency graph for stm32f4xx_hal_timebase_tim.c:



Functions

- HAL_StatusTypeDef HAL_InitTick (uint32_t TickPriority)

  This function configures the TIM1 as a time base source. The time source is configured to have 1ms time base with a dedicated Tick interrupt priority.
- void HAL_SuspendTick (void)

  Suspend Tick increment.
- void HAL_ResumeTick (void)

  Resume Tick increment.

Variables

- TIM_HandleTypeDef **htim1**

6.4.1 Detailed Description

HAL time base based on the hardware TIM.

Attention

6.4.2 Function Documentation

6.4.2.1 HAL_InitTick()

```
HAL_StatusTypeDef HAL_InitTick (
            uint32_t TickPriority )
```

This function configures the TIM1 as a time base source. The time source is configured to have 1ms time base with a dedicated Tick interrupt priority.

Note

This function is called automatically at the beginning of program after reset by HAL_Init() or at any time when clock is configured, by HAL_RCC_ClockConfig().

Parameters

TickPriority	Tick interrupt priority.

Return values

HAL	status

6.4.2.2 HAL_ResumeTick()

```
void HAL_ResumeTick (
             void  )
```

Resume Tick increment.

Note

Enable the tick increment by Enabling TIM1 update interrupt.

Parameters

None	

Return values

None	

6.4.2.3 HAL_SuspendTick()

```
void HAL_SuspendTick (
             void  )
```

Suspend Tick increment.

Note

Disable the tick increment by disabling TIM1 update interrupt.

Parameters

None	

Return values

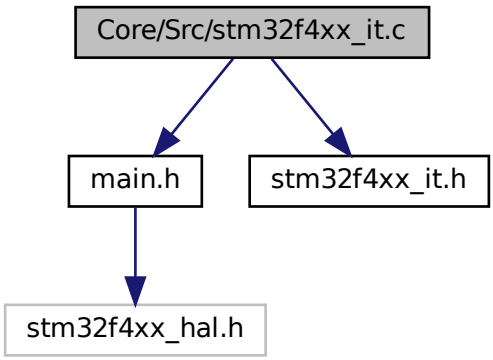
None	

6.5 Core/Src/stm32f4xx_it.c File Reference

Interrupt Service Routines.

```
#include "main.h"
#include "stm32f4xx_it.h"
```

Include dependency graph for stm32f4xx_it.c:



Functions

- void NMI_Handler (void)

  This function handles Non maskable interrupt.
- void HardFault_Handler (void)

  This function handles Hard fault interrupt.
- void MemManage_Handler (void)

  This function handles Memory management fault.
- void BusFault_Handler (void)

  This function handles Pre-fetch fault, memory access fault.
- void UsageFault_Handler (void)

  This function handles Undefined instruction or illegal state.
- void DebugMon_Handler (void)

  This function handles Debug monitor.
- void TIM1_UP_TIM10_IRQHandler (void)

  This function handles TIM1 update interrupt and TIM10 global interrupt.

Variables

- TIM_HandleTypeDef **htim1**

6.5.1 Detailed Description

Interrupt Service Routines.

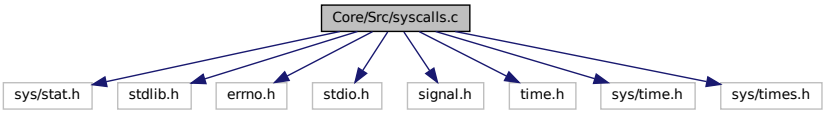
Attention

6.6 Core/Src/syscalls.c File Reference

STM32CubeIDE Minimal System calls file.

```
#include <sys/stat.h>
#include <stdlib.h>
#include <errno.h>
#include <stdio.h>
#include <signal.h>
#include <time.h>
#include <sys/time.h>
#include <sys/times.h>
```

Include dependency graph for syscalls.c:



Functions

- int **__io_putchar** (int ch) __attribute__((weak))
- int **__io_getchar** (void)
- void **initialise_monitor_handles** ()
- int **_getpid** (void)
- int **_kill** (int pid, int sig)
- void **_exit** (int status)
- **__attribute__** ((weak))
- int **_close** (int file)
- int **_fstat** (int file, struct stat ∗st)
- int **_isatty** (int file)
- int **_lseek** (int file, int ptr, int dir)
- int **_open** (char ∗path, int flags,...)
- int **_wait** (int ∗status)
- int **_unlink** (char ∗name)
- int **_times** (struct tms ∗buf)
- int **_stat** (char ∗file, struct stat ∗st)
- int **_link** (char ∗old, char ∗new)
- int **_fork** (void)
- int **_execve** (char ∗name, char ∗∗argv, char ∗∗env)

Variables

- char ** **environ** = __env

6.6.1 Detailed Description

STM32CubeIDE Minimal System calls file.

Author

Auto-generated by STM32CubeIDE

```
For more information about which c-functions
need which of these lowlevel functions
please consult the Newlib libc-manual
```

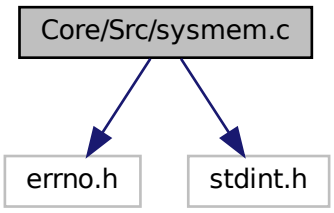
Attention

6.7 Core/Src/sysmem.c File Reference

STM32CubeIDE System Memory calls file.

```
#include <errno.h>
#include <stdint.h>
```

Include dependency graph for sysmem.c:



Functions

- void * _sbrk (ptrdiff_t incr)

  _sbrk() allocates memory to the newlib heap and is used by malloc and others from the C library

6.7.1 Detailed Description

STM32CubeIDE System Memory calls file.

Author

Generated by STM32CubeIDE

```
For more information about which C functions
need which of these lowlevel functions
please consult the newlib libc manual
```

Attention

6.7.2 Function Documentation

6.7.2.1 _sbrk()

```
void* _sbrk (
             ptrdiff_t incr )
```

_sbrk() allocates memory to the newlib heap and is used by malloc and others from the C library

```
* ##########################################################################
* #  .data  #  .bss  #       newlib heap       #          MSP stack          #
* #         #        #                         # Reserved by _Min_Stack_Size #
* ##########################################################################
* ^-- RAM start      ^-- _end                             _estack, RAM end --^
*
```

This implementation starts allocating at the '_end' linker symbol The '_Min_Stack_Size' linker symbol reserves a memory for the MSP stack The implementation considers '_estack' linker symbol to be RAM end NOTE: If the MSP stack, at any point during execution, grows larger than the reserved size, please increase the '_Min_Stack_Size'.

Parameters

incr	Memory size

Returns

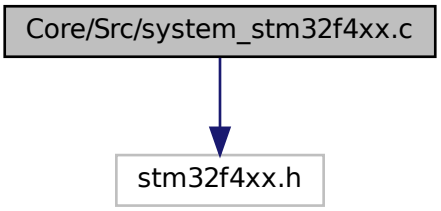
Pointer to allocated memory

6.8 Core/Src/system_stm32f4xx.c File Reference

CMSIS Cortex-M4 Device Peripheral Access Layer System Source File.

```
#include "stm32f4xx.h"
```

Include dependency graph for system_stm32f4xx.c:



Macros

- #define HSE_VALUE ((uint32_t)25000000)
- #define HSI_VALUE ((uint32_t)16000000)

Functions

- void SystemInit (void)

  Setup the microcontroller system Initialize the FPU setting, vector table location and External memory configuration.
- void SystemCoreClockUpdate (void)

  Update SystemCoreClock variable according to Clock Register Values. The SystemCoreClock variable contains the core clock (HCLK), it can be used by the user application to setup the SysTick timer or configure other parameters.

Variables

- uint32_t **SystemCoreClock** = 16000000
- const uint8_t **AHBPrescTable** [16] = {0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 3, 4, 6, 7, 8, 9}
- const uint8_t **APBPrescTable** [8] = {0, 0, 0, 0, 1, 2, 3, 4}

6.8.1 Detailed Description

CMSIS Cortex-M4 Device Peripheral Access Layer System Source File.

Author

MCD Application Team

This file provides two functions and one global variable to be called from user application:

- SystemInit(): This function is called at startup just after reset and before branch to main program. This call is made inside the "startup_stm32f4xx.s" file.
- SystemCoreClock variable: Contains the core clock (HCLK), it can be used by the user application to setup the SysTick timer or configure other parameters.
- SystemCoreClockUpdate(): Updates the variable SystemCoreClock and must be called whenever the core clock is changed during program execution.

Attention

Index