

TP MCC - CHAPUIS & SUTRA

Generated by Doxygen 1.9.5



<b>1 Module Index</b>	<b>1</b>
1.1 Modules . . . . .	1
<b>2 File Index</b>	<b>3</b>
2.1 File List . . . . .	3
<b>3 Module Documentation</b>	<b>5</b>
3.1 CMSIS . . . . .	5
3.1.1 Detailed Description . . . . .	5
3.2 Stm32g4xx_system . . . . .	5
3.2.1 Detailed Description . . . . .	5
3.3 STM32G4xx_System_Private_Includes . . . . .	5
3.3.1 Detailed Description . . . . .	6
3.3.2 Macro Definition Documentation . . . . .	6
3.3.2.1 HSE_VALUE . . . . .	6
3.3.2.2 HSI_VALUE . . . . .	6
3.4 STM32G4xx_System_Private_TypesDefinitions . . . . .	6
3.5 STM32G4xx_System_Private_Defines . . . . .	6
3.6 STM32G4xx_System_Private_Macros . . . . .	6
3.7 STM32G4xx_System_Private_Variables . . . . .	6
3.7.1 Detailed Description . . . . .	6
3.8 STM32G4xx_System_Private_FunctionPrototypes . . . . .	6
3.9 STM32G4xx_System_Private_Functions . . . . .	6
3.9.1 Detailed Description . . . . .	7
3.9.2 Function Documentation . . . . .	7
3.9.2.1 SystemCoreClockUpdate() . . . . .	7
3.9.2.2 SystemInit() . . . . .	7
<b>4 File Documentation</b>	<b>9</b>
4.1 main.c File Reference . . . . .	9
4.1.1 Detailed Description . . . . .	11
4.1.2 Function Documentation . . . . .	11
4.1.2.1 data() . . . . .	11
4.1.2.2 delete() . . . . .	11
4.1.2.3 Error_Handler() . . . . .	13
4.1.2.4 get() . . . . .	13
4.1.2.5 get_current() . . . . .	13
4.1.2.6 HAL_TIM_PeriodElapsedCallback() . . . . .	14
4.1.2.7 help() . . . . .	14
4.1.2.8 main() . . . . .	14
4.1.2.9 new_carac() . . . . .	15
4.1.2.10 newcom() . . . . .	15
4.1.2.11 pinout() . . . . .	15

4.1.2.12 read_enc()	16
4.1.2.13 read_speed()	16
4.1.2.14 set()	17
4.1.2.15 set_current()	17
4.1.2.16 set_speed()	17
4.1.2.17 start()	18
4.1.2.18 stop()	18
4.1.2.19 SystemClock_Config()	18
4.1.3 Variable Documentation	19
4.1.3.1 started	19
4.2 stm32g4xx_hal_msp.c File Reference	19
4.2.1 Detailed Description	20
4.2.2 Function Documentation	20
4.2.2.1 HAL_ADC_MspDeInit()	20
4.2.2.2 HAL_ADC_MspInit()	20
4.2.2.3 HAL_MspInit()	21
4.2.2.4 HAL_TIM_Base_MspDeInit()	21
4.2.2.5 HAL_TIM_Base_MspInit()	21
4.2.2.6 HAL_TIM_Encoder_MspDeInit()	22
4.2.2.7 HAL_TIM_Encoder_MspInit()	22
4.2.2.8 HAL_TIM_MspPostInit()	22
4.2.2.9 HAL_UART_MspDeInit()	23
4.2.2.10 HAL_UART_MspInit()	23
4.3 stm32g4xx_it.c File Reference	23
4.3.1 Detailed Description	24
4.4 syscalls.c File Reference	25
4.4.1 Detailed Description	25
4.5 systemem.c File Reference	26
4.5.1 Detailed Description	26
4.5.2 Function Documentation	26
4.5.2.1 _sbrk()	26
4.6 system_stm32g4xx.c File Reference	27
4.6.1 Detailed Description	27
4.6.2 This file configures the system clock as follows:	28
4.6.2.1 System Clock source   HSI	28
4.6.2.2 SYSCLK(Hz)   16000000	28
4.6.2.3 HCLK(Hz)   16000000	28
4.6.2.4 AHB Prescaler   1	28
4.6.2.5 APB1 Prescaler   1	28
4.6.2.6 APB2 Prescaler   1	28
4.6.2.7 PLL_M   1	28
4.6.2.8 PLL_N   16	28

---

4.6.2.9 PLL_P   7 . . . . .	28
4.6.2.10 PLL_Q   2 . . . . .	28
4.6.2.11 PLL_R   2 . . . . .	28
4.6.2.12 Require 48MHz for RNG   Disabled . . . . .	28

<b>Index</b>	<b>29</b>
--------------	-----------



# Chapter 1

## Module Index

### 1.1 Modules

Here is a list of all modules:

CMSIS . . . . .	5
Stm32g4xx_system . . . . .	5
STM32G4xx_System_Private_Includes . . . . .	5
STM32G4xx_System_Private_TypesDefinitions . . . . .	6
STM32G4xx_System_Private_Defines . . . . .	6
STM32G4xx_System_Private_Macros . . . . .	6
STM32G4xx_System_Private_Variables . . . . .	6
STM32G4xx_System_Private_FunctionPrototypes . . . . .	6
STM32G4xx_System_Private_Functions . . . . .	6





## Chapter 2

# File Index

### 2.1 File List

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

<b>main.c</b>	
: Main program body . . . . .	9
<b>stm32g4xx_hal_msp.c</b>	
This file provides code for the MSP Initialization and de-Initialization codes . . . . .	19
<b>stm32g4xx_it.c</b>	
Interrupt Service Routines . . . . .	23
<b>syscalls.c</b>	
STM32CubeIDE Minimal System calls file . . . . .	25
<b>sysmem.c</b>	
STM32CubeIDE System Memory calls file . . . . .	26
<b>system_stm32g4xx.c</b>	
CMSIS Cortex-M4 Device Peripheral Access Layer System Source File . . . . .	27



## Chapter 3

# Module Documentation

### 3.1 CMSIS

#### Modules

- `Stm32g4xx_system`

#### 3.1.1 Detailed Description

### 3.2 Stm32g4xx\_system

#### Modules

- `STM32G4xx_System_Private_Includes`
- `STM32G4xx_System_Private_TypesDefinitions`
- `STM32G4xx_System_Private_Defines`
- `STM32G4xx_System_Private_Macros`
- `STM32G4xx_System_Private_Variables`
- `STM32G4xx_System_Private_FunctionPrototypes`
- `STM32G4xx_System_Private_Functions`

#### 3.2.1 Detailed Description

### 3.3 STM32G4xx\_System\_Private\_Includes

#### Macros

- `#define HSE_VALUE 24000000U`
- `#define HSI_VALUE 16000000U`

### 3.3.1 Detailed Description

### 3.3.2 Macro Definition Documentation

#### 3.3.2.1 HSE\_VALUE

```
#define HSE_VALUE 24000000U
```

Value of the External oscillator in Hz

#### 3.3.2.2 HSI\_VALUE

```
#define HSI_VALUE 16000000U
```

Value of the Internal oscillator in Hz

## 3.4 STM32G4xx\_System\_Private\_TypeDefinitions

## 3.5 STM32G4xx\_System\_Private\_Defines

## 3.6 STM32G4xx\_System\_Private\_Macros

## 3.7 STM32G4xx\_System\_Private\_Variables

### Variables

- uint32\_t **SystemCoreClock** = **HSI\_VALUE**
- const uint8\_t **AHBPrescTable** [16] = {0U, 0U, 0U, 0U, 0U, 0U, 0U, 0U, 1U, 2U, 3U, 4U, 6U, 7U, 8U, 9U}
- const uint8\_t **APBPrescTable** [8] = {0U, 0U, 0U, 0U, 1U, 2U, 3U, 4U}

### 3.7.1 Detailed Description

## 3.8 STM32G4xx\_System\_Private\_FunctionPrototypes

## 3.9 STM32G4xx\_System\_Private\_Functions

### Functions

- void **SystemInit** (void)  
*Setup the microcontroller system.*
- 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.*

### 3.9.1 Detailed Description

### 3.9.2 Function Documentation

#### 3.9.2.1 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(\*\*)** (p. 6)
- If SYSCLK source is HSE, SystemCoreClock will contain the **HSE\_VALUE(\*\*\*)** (p. 6)
- If SYSCLK source is PLL, SystemCoreClock will contain the **HSE\_VALUE(\*\*\*)** (p. 6) or **HSI\_VALUE(\*)** (p. 6) multiplied/divided by the PLL factors.

(\*\*) HSI\_VALUE is a constant defined in stm32g4xx\_hal.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 stm32g4xx\_hal.h file (default value 24 MHz), user has to ensure that HSE\_VALUE is same as the real frequency of the crystal used. Otherwise, this function may have wrong result.

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

##### Parameters

None	
------	--

##### Return values

None	
------	--

#### 3.9.2.2 SystemInit()

```
void SystemInit (
```

```
void  )
```

Setup the microcontroller system.

#### Parameters

<i>None</i>	
-------------	--

#### Return values

<i>None</i>	
-------------	--

## Chapter 4

# File Documentation

### 4.1 main.c File Reference

: Main program body

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

#### Functions

- void **SystemClock\_Config** (void)  
*System Clock Configuration.*
- void **newcom** (void)  
*This function makes the initialization for the PI controller.*
- void **delete** (void)  
*This function allows us the delete letters in the shell.*
- void **new\_carac** (void)  
*This function allows us to write a new letter in the shell.*
- void **get** (void)  
*This function writes a message in the shell in the case the command is not found.*
- void **set** (void)  
*This function switch on/off the led if the user writes PA5.*
- void **help** (void)  
*This function displays all the usable functions for the user.*
- void **pinout** (void)  
*This function shows all the connections for the pinouts.*
- void **start** (void)  
*This function makes the initialization for the chopper, the PWM for the motor and the initialization for the encoder.*
- void **stop** (void)  
*This function makes the initialization for the PI controller.*
- void **set\_speed** (char \*vit)  
*This function allows the user to set the motor speed.*
- void **read\_enc** (void)  
*This function read the encoder value.*
- void **read\_speed** (void)

*This function read the value of the speed (measured by the encoder).*

- void **set\_current** (char \*set\_cur)

*This function makes the conversion for the current value for the motor and its enslavement.*

- float **get\_current** ()

*This function make the measure of the current for the current enslavement.*

- void **data** (void)

*This function shows all the data needed from the motor for this project: current value, encoder value and speed value.*

- void **HAL\_TIM\_PeriodElapsedCallback** (TIM\_HandleTypeDef \*htim)

*The function which generate the interruptions.*

- int **main** (void)

*The application entry point.*

- void **HAL\_UART\_RxCpltCallback** (UART\_HandleTypeDef \*huart)

- void **HAL\_ADC\_ConvCpltCallback** (ADC\_HandleTypeDef \*hadc)

- void **Error\_Handler** (void)

*This function is executed in case of error occurrence.*

## Variables

- ADC\_HandleTypeDef **hadc1**
- DMA\_HandleTypeDef **hdma\_adc1**
- TIM\_HandleTypeDef **htim1**
- TIM\_HandleTypeDef **htim3**
- TIM\_HandleTypeDef **htim4**
- UART\_HandleTypeDef **huart2**
- uint8\_t **prompt** [] ="user@Nucleo-STM32G474>>"
- uint8\_t **started** []
- uint8\_t **newline** [] ="\\n"
- uint8\_t **cmdNotFound** [] ="Command not found\\n"
- uint32\_t **uartRxReceived**
- uint8\_t **uartRxBuffer** [UART\_RX\_BUFFER\_SIZE]
- uint8\_t **uartTxBuffer** [UART\_TX\_BUFFER\_SIZE]
- uint8\_t **powerOn** [] ="Allumage du moteur\\n"
- uint8\_t **powerOff** [] ="Extinction du moteur\\n"
- uint32\_t **adcBuffer** [ADC\_BUFFER\_SIZE]
- char **cmdBuffer** [CMD\_BUFFER\_SIZE]
- int **idx\_cmd**
- char \* **argv** [MAX\_ARGS]
- int **argc** = 0
- char \* **token**
- int **newCmdReady** = 0
- int **abs\_speed** = 50
- int **adcDMAFlag** = 0
- int **it\_tim3** = 0
- int **it\_tim1** = 0
- int **enc** = 0
- int **speed** = 0
- int **inc** = 0
- float **targetcur**
- char **pi\_buffer** [256]
- PIController **pi** = {PI\_KP, PI\_KI, PI\_LIM\_MIN, PI\_LIM\_MAX, PI\_LIM\_MIN\_INT, PI\_LIM\_MAX\_INT, SAMPLE\_TIME\_S}



### 4.1.1 Detailed Description

: Main program body

#### Author

: Clément Chapuis & Aurélien Sutra

#### Date

: Date du dernier commit&push : 03/01/2023

#### Attention

Copyright (c) 2022 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

### 4.1.2 Function Documentation

#### 4.1.2.1 data()

```
void data (
    void )
```

This function shows all the data needed from the motor for this project: current value, encoder value and speed value.

#### Parameters

in	None	
----	------	--

#### Return values

None	
------	--

#### 4.1.2.2 delete()

```
void delete (
    void )
```

This function allows us the delete letters in the shell.

**Parameters**

in	None	
----	------	--

**Return values**

None	
------	--

**4.1.2.3 Error\_Handler()**

```
void Error_Handler (
    void )
```

This function is executed in case of error occurrence.

**Return values**

None	
------	--

**4.1.2.4 get()**

```
void get (
    void )
```

This function writes a message in the shell in the case the command is not found.

**Parameters**

in	None	
----	------	--

**Return values**

None	
------	--

**4.1.2.5 get\_current()**

```
float get_current ( )
```

This function make the measure of the current for the current enslavement.

**Parameters**

in	<i>None</i>	
----	-------------	--

**Return values**

<i>None</i>	
-------------	--

**4.1.2.6 HAL\_TIM\_PeriodElapsedCallback()**

```
void HAL_TIM_PeriodElapsedCallback (
    TIM_HandleTypeDef * htim )
```

The function which generate the interruptions.

**Parameters**

in	<i>htim</i>	timer adress
----	-------------	--------------

**Return values**

<i>None</i>	
-------------	--

**4.1.2.7 help()**

```
void help (
    void )
```

This function displays all the usable functions for the user.

**Parameters**

in	<i>None</i>	
----	-------------	--

**Return values**

<i>None</i>	
-------------	--

**4.1.2.8 main()**

```
int main (
```

```
void )
```

The application entry point.

#### Return values

<i>int</i>	
------------	--

#### 4.1.2.9 new\_carac()

```
void new_carac (  
    void )
```

This function allows us to write a new letter in the shell.

#### Parameters

<i>in</i>	<i>None</i>	
-----------	-------------	--

#### Return values

<i>None</i>	
-------------	--

#### 4.1.2.10 newcom()

```
void newcom (  
    void )
```

This function makes the initialization for the PI controller.

#### Parameters

<i>in</i>	<i>PIController</i>	The structure for all the PI parameters
-----------	---------------------	---

#### Return values

<i>None</i>	
-------------	--

#### 4.1.2.11 pinout()

```
void pinout (  
    void )
```

This function shows all the connections for the pinouts.

**Parameters**

in	<i>None</i>	
----	-------------	--

**Return values**

<i>None</i>	
-------------	--

**4.1.2.12 read\_enc()**

```
void read_enc (
    void )
```

This function read the encoder value.

**Parameters**

in	<i>None</i>	
----	-------------	--

**Return values**

<i>None</i>	
-------------	--

**4.1.2.13 read\_speed()**

```
void read_speed (
    void )
```

This function read the value of the speed (measured by the encoder).

**Parameters**

in	<i>None</i>	
----	-------------	--

**Return values**

<i>None</i>	
-------------	--

#### 4.1.2.14 set()

```
void set (
    void )
```

This function switch on/off the led if the user writes PA5.

##### Parameters

in	None	
----	------	--

##### Return values

None	
------	--

#### 4.1.2.15 set\_current()

```
void set_current (
    char * set_cur )
```

This function makes the conversion for the current value for the motor and its enslavement.

##### Parameters

in	set_cur	The value of current the user wants for the motor enslavement
----	---------	---

##### Return values

None	
------	--

#### 4.1.2.16 set\_speed()

```
void set_speed (
    char * vit )
```

This function allows the user to set the motor speed.

##### Parameters

in	vit	The speed wanted by the user (0-49 in one direction, 51-100 in the other, 50 the motor doesn't turn)
----	-----	--

##### Return values

None	
------	--

#### 4.1.2.17 start()

```
void start (  
           void )
```

This function makes the initialization for the chopper, the PWM for the motor and the initialization for the encoder.

##### Parameters

in	None	
----	------	--

##### Return values

None	
------	--

#### 4.1.2.18 stop()

```
void stop (  
           void )
```

This function makes the initialization for the PI controller.

##### Parameters

in	None	
----	------	--

##### Return values

None	
------	--

#### 4.1.2.19 SystemClock\_Config()

```
void SystemClock_Config (  
                          void )
```

System Clock Configuration.

##### Return values

None	
------	--



Configure the main internal regulator output voltage

Initializes the RCC Oscillators according to the specified parameters in the RCC\_OscInitTypeDef structure.

Initializes the CPU, AHB and APB buses clocks

### 4.1.3 Variable Documentation

#### 4.1.3.1 started

```
uint8_t started[]
```

**Initial value:**

```
=
    "\r\n*-----*"
    "\r\n| Welcome on Nucleo-STM32G474 |"
    "\r\n*-----*"
    "\r\n"
```

## 4.2 stm32g4xx\_hal\_msp.c File Reference

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

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

### Functions

- void **HAL\_TIM\_MspPostInit** (TIM\_HandleTypeDef \*htim)
- void **HAL\_MspInit** (void)
- void **HAL\_ADC\_MspInit** (ADC\_HandleTypeDef \*hadc)
 

*ADC MSP Initialization This function configures the hardware resources used in this example.*
- void **HAL\_ADC\_MspDeInit** (ADC\_HandleTypeDef \*hadc)
 

*ADC MSP De-Initialization This function freeze the hardware resources used in this example.*
- void **HAL\_TIM\_Base\_MspInit** (TIM\_HandleTypeDef \*htim\_base)
 

*TIM\_Base MSP Initialization This function configures the hardware resources used in this example.*
- void **HAL\_TIM\_Encoder\_MspInit** (TIM\_HandleTypeDef \*htim\_encoder)
 

*TIM\_Encoder MSP Initialization This function configures the hardware resources used in this example.*
- void **HAL\_TIM\_Base\_MspDeInit** (TIM\_HandleTypeDef \*htim\_base)
 

*TIM\_Base MSP De-Initialization This function freeze the hardware resources used in this example.*
- void **HAL\_TIM\_Encoder\_MspDeInit** (TIM\_HandleTypeDef \*htim\_encoder)
 

*TIM\_Encoder 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.*

## Variables

- DMA\_HandleTypeDef **hdma\_adc1**

### 4.2.1 Detailed Description

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

#### Attention

Copyright (c) 2022 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

### 4.2.2 Function Documentation

#### 4.2.2.1 HAL\_ADC\_MspDeInit()

```
void HAL_ADC_MspDeInit (
    ADC_HandleTypeDef * hadc )
```

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

#### Parameters

<i>hadc</i>	ADC handle pointer
-------------	--------------------

#### Return values

<i>None</i>	
-------------	--

ADC1 GPIO Configuration PA0 ----> ADC1\_IN1

#### 4.2.2.2 HAL\_ADC\_MspInit()

```
void HAL_ADC_MspInit (
    ADC_HandleTypeDef * hadc )
```

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

## Parameters

<i>hadc</i>	ADC handle pointer
-------------	--------------------

## Return values

<i>None</i>	
-------------	--

Initializes the peripherals clocks

ADC1 GPIO Configuration PA0 ----> ADC1\_IN1

**4.2.2.3 HAL\_MspInit()**

```
void HAL_MspInit (
    void )
```

Initializes the Global MSP. Disable the internal Pull-Up in Dead Battery pins of UCPD peripheral

**4.2.2.4 HAL\_TIM\_Base\_MspDeInit()**

```
void HAL_TIM_Base_MspDeInit (
    TIM_HandleTypeDef * htim_base )
```

TIM\_Base MSP De-Initialization This function freeze the hardware resources used in this example.

## Parameters

<i>htim_base</i>	TIM_Base handle pointer
------------------	-------------------------

## Return values

<i>None</i>	
-------------	--

**4.2.2.5 HAL\_TIM\_Base\_MspInit()**

```
void HAL_TIM_Base_MspInit (
    TIM_HandleTypeDef * htim_base )
```

TIM\_Base MSP Initialization This function configures the hardware resources used in this example.

## Parameters

<i>htim_base</i>	TIM_Base handle pointer
------------------	-------------------------

## Return values

<i>None</i>	
-------------	--

**4.2.2.6 HAL\_TIM\_Encoder\_MspDeInit()**

```
void HAL_TIM_Encoder_MspDeInit (
    TIM_HandleTypeDef * htim_encoder )
```

TIM\_Encoder MSP De-Initialization This function freeze the hardware resources used in this example.

## Parameters

<i>htim_encoder</i>	TIM_Encoder handle pointer
---------------------	----------------------------

## Return values

<i>None</i>	
-------------	--

TIM4 GPIO Configuration PB6 ----> TIM4\_CH1 PB7 ----> TIM4\_CH2

**4.2.2.7 HAL\_TIM\_Encoder\_MspInit()**

```
void HAL_TIM_Encoder_MspInit (
    TIM_HandleTypeDef * htim_encoder )
```

TIM\_Encoder MSP Initialization This function configures the hardware resources used in this example.

## Parameters

<i>htim_encoder</i>	TIM_Encoder handle pointer
---------------------	----------------------------

## Return values

<i>None</i>	
-------------	--

TIM4 GPIO Configuration PB6 ----> TIM4\_CH1 PB7 ----> TIM4\_CH2

**4.2.2.8 HAL\_TIM\_MspPostInit()**

```
void HAL_TIM_MspPostInit (
    TIM_HandleTypeDef * htim )
```

TIM1 GPIO Configuration PA8 ----> TIM1\_CH1 PA9 ----> TIM1\_CH2 PA11 ----> TIM1\_CH1N PA12 ----> TIM1\_CH2N

#### 4.2.2.9 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

<i>huart</i>	UART handle pointer
--------------	---------------------

##### Return values

<i>None</i>	
-------------	--

USART2 GPIO Configuration PA2 ----> USART2\_TX PA3 ----> USART2\_RX

#### 4.2.2.10 HAL\_UART\_MspInit()

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

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

##### Parameters

<i>huart</i>	UART handle pointer
--------------	---------------------

##### Return values

<i>None</i>	
-------------	--

Initializes the peripherals clocks

USART2 GPIO Configuration PA2 ----> USART2\_TX PA3 ----> USART2\_RX

## 4.3 stm32g4xx\_it.c File Reference

Interrupt Service Routines.

```
#include "main.h"
#include "stm32g4xx_it.h"
```

## 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 Prefetch fault, memory access fault.*
- void **UsageFault\_Handler** (void)  
*This function handles Undefined instruction or illegal state.*
- void **SVC\_Handler** (void)  
*This function handles System service call via SWI instruction.*
- void **DebugMon\_Handler** (void)  
*This function handles Debug monitor.*
- void **PendSV\_Handler** (void)  
*This function handles Pendable request for system service.*
- void **SysTick\_Handler** (void)  
*This function handles System tick timer.*
- void **EXTI1\_IRQHandler** (void)  
*This function handles EXTI line1 interrupt.*
- void **DMA1\_Channel1\_IRQHandler** (void)  
*This function handles DMA1 channel1 global interrupt.*
- void **TIM1\_UP\_TIM16\_IRQHandler** (void)  
*This function handles TIM1 update interrupt and TIM16 global interrupt.*
- void **TIM3\_IRQHandler** (void)  
*This function handles TIM3 global interrupt.*
- void **USART2\_IRQHandler** (void)  
*This function handles USART2 global interrupt / USART2 wake-up interrupt through EXTI line 26.*
- void **EXTI15\_10\_IRQHandler** (void)  
*This function handles EXTI line[15:10] interrupts.*

## Variables

- DMA\_HandleTypeDef **hdma\_adc1**
- TIM\_HandleTypeDef **htim1**
- TIM\_HandleTypeDef **htim3**
- UART\_HandleTypeDef **huart2**

### 4.3.1 Detailed Description

Interrupt Service Routines.

Attention

Copyright (c) 2022 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

## 4.4 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>
```

### 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**

#### 4.4.1 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

Copyright (c) 2022 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

## 4.5 sysmem.c File Reference

STM32CubeIDE System Memory calls file.

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

### Functions

- void \* **\_sbrk** (ptrdiff\_t incr)  
**\_sbrk()** (p. 26) allocates memory to the newlib heap and is used by malloc and others from the C library

#### 4.5.1 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

Copyright (c) 2022 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

#### 4.5.2 Function Documentation

##### 4.5.2.1 \_sbrk()

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

**\_sbrk()** (p. 26) 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

<i>incr</i>	Memory size
-------------	-------------

## Returns

Pointer to allocated memory

## 4.6 system\_stm32g4xx.c File Reference

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

```
#include "stm32g4xx.h"
```

### Macros

- `#define HSE_VALUE 24000000U`
- `#define HSI_VALUE 16000000U`

### Functions

- void **SystemInit** (void)  
*Setup the microcontroller system.*
- 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 = HSI_VALUE`
- `const uint8_t AHBPrescTable [16] = {0U, 0U, 0U, 0U, 0U, 0U, 0U, 0U, 1U, 2U, 3U, 4U, 6U, 7U, 8U, 9U}`
- `const uint8_t APBPrescTable [8] = {0U, 0U, 0U, 0U, 1U, 2U, 3U, 4U}`

#### 4.6.1 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()** (p. 7): This function is called at startup just after reset and before branch to main program. This call is made inside the "startup\_stm32g4xx.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()** (p. 7): Updates the variable SystemCoreClock and must be called whenever the core clock is changed during program execution.

After each device reset the HSI (16 MHz) is used as system clock source. Then **SystemInit()** (p. 7) function is called, in "startup\_stm32g4xx.s" file, to configure the system clock before to branch to main program.

## 4.6.2 This file configures the system clock as follows:

4.6.2.1 System Clock source | HSI

4.6.2.2 SYSClk(Hz) | 16000000

4.6.2.3 HCLK(Hz) | 16000000

4.6.2.4 AHB Prescaler | 1

4.6.2.5 APB1 Prescaler | 1

4.6.2.6 APB2 Prescaler | 1

4.6.2.7 PLL\_M | 1

4.6.2.8 PLL\_N | 16

4.6.2.9 PLL\_P | 7

4.6.2.10 PLL\_Q | 2

4.6.2.11 PLL\_R | 2

4.6.2.12 Require 48MHz for RNG | Disabled

=====  
Attention

Copyright (c) 2019 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

# Index

- \_sbrk
    - sysmem.c, 26
- CMSIS, 5
- data
  - main.c, 11
- delete
  - main.c, 11
- Error\_Handler
  - main.c, 13
- get
  - main.c, 13
- get\_current
  - main.c, 13
- HAL\_ADC\_MspDeInit
  - stm32g4xx\_hal\_msp.c, 20
- HAL\_ADC\_MspInit
  - stm32g4xx\_hal\_msp.c, 20
- HAL\_MspInit
  - stm32g4xx\_hal\_msp.c, 21
- HAL\_TIM\_Base\_MspDeInit
  - stm32g4xx\_hal\_msp.c, 21
- HAL\_TIM\_Base\_MspInit
  - stm32g4xx\_hal\_msp.c, 21
- HAL\_TIM\_Encoder\_MspDeInit
  - stm32g4xx\_hal\_msp.c, 22
- HAL\_TIM\_Encoder\_MspInit
  - stm32g4xx\_hal\_msp.c, 22
- HAL\_TIM\_MspPostInit
  - stm32g4xx\_hal\_msp.c, 22
- HAL\_TIM\_PeriodElapsedCallback
  - main.c, 14
- HAL\_UART\_MspDeInit
  - stm32g4xx\_hal\_msp.c, 22
- HAL\_UART\_MspInit
  - stm32g4xx\_hal\_msp.c, 23
- help
  - main.c, 14
- HSE\_VALUE
  - STM32G4xx\_System\_Private\_Includes, 6
- HSI\_VALUE
  - STM32G4xx\_System\_Private\_Includes, 6
- main
  - main.c, 14
- main.c, 9
  - data, 11
- delete, 11
- Error\_Handler, 13
- get, 13
- get\_current, 13
- HAL\_TIM\_PeriodElapsedCallback, 14
- help, 14
- main, 14
- new\_carac, 15
- newcom, 15
- pinout, 15
- read\_enc, 16
- read\_speed, 16
- set, 16
- set\_current, 17
- set\_speed, 17
- start, 18
- started, 19
- stop, 18
- SystemClock\_Config, 18
- new\_carac
  - main.c, 15
- newcom
  - main.c, 15
- pinout
  - main.c, 15
- read\_enc
  - main.c, 16
- read\_speed
  - main.c, 16
- set
  - main.c, 16
- set\_current
  - main.c, 17
- set\_speed
  - main.c, 17
- start
  - main.c, 18
- started
  - main.c, 19
- stm32g4xx\_hal\_msp.c, 19
  - HAL\_ADC\_MspDeInit, 20
  - HAL\_ADC\_MspInit, 20
  - HAL\_MspInit, 21
  - HAL\_TIM\_Base\_MspDeInit, 21
  - HAL\_TIM\_Base\_MspInit, 21
  - HAL\_TIM\_Encoder\_MspDeInit, 22

- HAL\_TIM\_Encoder\_MspInit, 22
- HAL\_TIM\_MspPostInit, 22
- HAL\_UART\_MspDeInit, 22
- HAL\_UART\_MspInit, 23
- stm32g4xx\_it.c, 23
- Stm32g4xx\_system, 5
- STM32G4xx\_System\_Private\_Defines, 6
- STM32G4xx\_System\_Private\_FunctionPrototypes, 6
- STM32G4xx\_System\_Private\_Functions, 6
  - SystemCoreClockUpdate, 7
  - SystemInit, 7
- STM32G4xx\_System\_Private\_Includes, 5
  - HSE\_VALUE, 6
  - HSI\_VALUE, 6
- STM32G4xx\_System\_Private\_Macros, 6
- STM32G4xx\_System\_Private\_TypesDefinitions, 6
- STM32G4xx\_System\_Private\_Variables, 6
- stop
  - main.c, 18
- syscalls.c, 25
- sysmem.c, 26
  - \_sbrk, 26
- system\_stm32g4xx.c, 27
- SystemClock\_Config
  - main.c, 18
- SystemCoreClockUpdate
  - STM32G4xx\_System\_Private\_Functions, 7
- SystemInit
  - STM32G4xx\_System\_Private\_Functions, 7