Learning Report – Embedded C

L&T Technology Services

Document History

Ver. Rel. No.	Release Date	Prepared. By	Reviewed By	Approved By	Remarks/Revision Details
		Katherapalle Rama Subba Reddy		Bhargav N	

Contents

Activity 1 – COMPILATION APROACH

This is the complete compilation process of the sample program for ARM Cortex Mx processor based boards.
Following are the compilation stages of a C program:

1. Preprocessor stage
2. Compilation stage
3. Assembly stage
4. Linking stage

1.1- MAKE FILE

Below is the make file for the sample program:

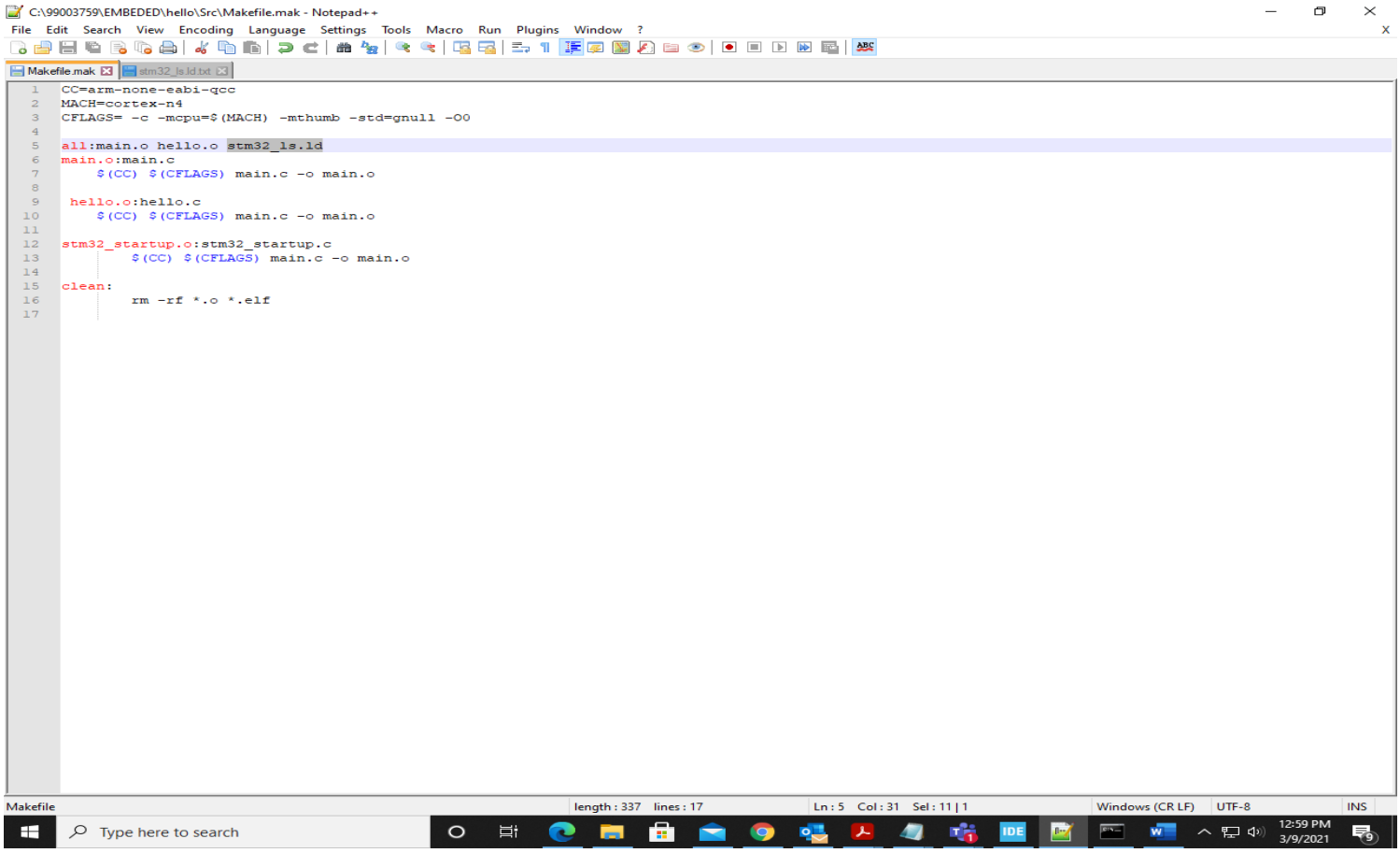


Fig 1.1.1 make file

The command to run this make file in the command prompt is:

Fig 1.1.2 Make command

- **-mcpu=cortex-m4** is used to select our cortex-m4 processor which is used
- **-mthumb** is used to generate the code that executes in ARM state
- **main.o** is the target file
- **main.c** is the dependency

1.2- STARTUP CODE

- The startup file is responsible for setting up the right environments to run the code in main.c file.
- Some part of the startup code is target (processor) dependent.
- Role of startup file:
  1. Create a MCU specific vector table for microcontroller.
  2. To write a startup code which initializes .data and .bss section in SRAM.
  3. Call main()

```
*
*******************************************************************************/
  .section  .isr_vector,"a",%progbits
  .type  g_pfnVectors, %object
  .size  g_pfnVectors, .-g_pfnVectors


g_pfnVectors:
  .word  _estack
  .word  Reset_Handler
  .word  NMI_Handler
  .word  HardFault_Handler
  .word  MemManage_Handler
  .word  BusFault_Handler
  .word  UsageFault_Handler
  .word  0
  .word  0
  .word  0
  .word  0
  .word  SVC_Handler
  .word  DebugMon_Handler
  .word  0
  .word  PendSV_Handler
  .word  SysTick_Handler

  /* External Interrupts */
  .word     WWDG_IRQHandler                   /* Window WatchDog                      */
  .word     PVD_IRQHandler                    /* PVD through EXTI Line detection */
  .word     TAMP_STAMP_IRQHandler             /* Tamper and TimeStamps through the EXTI line */
  .word     RTC_WKUP_IRQHandler               /* RTC Wakeup through the EXTI line */
  .word     FLASH_IRQHandler                  /* FLASH                                */
  .word     RCC_IRQHandler                    /* RCC                                  */
  .word     EXTI0_IRQHandler                  /* EXTI Line0                           */
  .word     EXTI1_IRQHandler                  /* EXTI Line1                           */
  .word     EXTI2_IRQHandler                  /* EXTI Line2                           */
  .word     EXTI3_IRQHandler                  /* EXTI Line3                           */
  .word     EXTI4_IRQHandler                  /* EXTI Line4                           */
  .word     DMA1_Stream0_IRQHandler           /* DMA1 Stream 0                        */
  .word     DMA1_Stream1_IRQHandler           /* DMA1 Stream 1                        */
```

```
.word     EXTI1_IRQHandler                  /* EXTI Line1                    */
.word     EXTI2_IRQHandler                  /* EXTI Line2                    */
.word     EXTI3_IRQHandler                  /* EXTI Line3                    */
.word     EXTI4_IRQHandler                  /* EXTI Line4                    */
.word     DMA1_Stream0_IRQHandler           /* DMA1 Stream 0                 */
.word     DMA1_Stream1_IRQHandler           /* DMA1 Stream 1                 */
.word     DMA1_Stream2_IRQHandler           /* DMA1 Stream 2                 */
.word     DMA1_Stream3_IRQHandler           /* DMA1 Stream 3                 */
.word     DMA1_Stream4_IRQHandler           /* DMA1 Stream 4                 */
.word     DMA1_Stream5_IRQHandler           /* DMA1 Stream 5                 */
.word     DMA1_Stream6_IRQHandler           /* DMA1 Stream 6                 */
.word     ADC_IRQHandler                    /* ADC1, ADC2 and ADC3s          */
.word     CAN1_TX_IRQHandler                /* CAN1 TX                       */
.word     CAN1_RX0_IRQHandler               /* CAN1 RX0                      */
.word     CAN1_RX1_IRQHandler               /* CAN1 RX1                      */
.word     CAN1_SCE_IRQHandler               /* CAN1 SCE                      */
.word     EXTI9_5_IRQHandler                /* External Line[9:5]s           */
.word     TIM1_BRK_TIM9_IRQHandler          /* TIM1 Break and TIM9           */
.word     TIM1_UP_TIM10_IRQHandler          /* TIM1 Update and TIM10         */
.word     TIM1_TRG_COM_TIM11_IRQHandler     /* TIM1 Trigger and Commutation and TIM11 */
.word     TIM1_CC_IRQHandler                /* TIM1 Capture Compare          */
.word     TIM2_IRQHandler                   /* TIM2                          */
.word     TIM3_IRQHandler                   /* TIM3                          */
.word     TIM4_IRQHandler                   /* TIM4                          */
.word     I2C1_EV_IRQHandler                /* I2C1 Event                    */
.word     I2C1_ER_IRQHandler                /* I2C1 Error                    */
.word     I2C2_EV_IRQHandler                /* I2C2 Event                    */
.word     I2C2_ER_IRQHandler                /* I2C2 Error                    */
.word     SPI1_IRQHandler                   /* SPI1                          */
.word     SPI2_IRQHandler                   /* SPI2                          */
.word     USART1_IRQHandler                 /* USART1                        */
.word     USART2_IRQHandler                 /* USART2                        */
.word     USART3_IRQHandler                 /* USART3                        */
.word     EXTI15_10_IRQHandler              /* External Line[15:10]s         */
.word     RTC_Alarm_IRQHandler              /* RTC Alarm (A and B) through EXTI Line */
.word     OTG_FS_WKUP_IRQHandler            /* USB OTG FS Wakeup through EXTI line */
.word     TIM8_BRK_TIM12_IRQHandler         /* TIM8 Break and TIM12          */
.word     TIM8_UP_TIM13_IRQHandler          /* TIM8 Update and TIM13         */
.word     TIM8_TRG_COM_TIM14_IRQHandler     /* TIM8 Trigger and Commutation and TIM14 */
.word     TIM8_CC_IRQHandler                /* TIM8 Capture Compare          */
.word     DMA1_Stream7_IRQHandler           /* DMA1 Stream7                  */
.word     FSMC_IRQHandler                   /* FSMC                          */
.word     SDIO_IRQHandler                   /* SDIO                          */
.word     TIM5_IRQHandler                   /* TIM5                          */
```

Fig 1.2.1 Startup code

In startup code we use variable attributes to store some variables in the user defined function.
Function attributes:

- Weak: Lets programmer override already defined weak function (dummy function) with the same function name.
- Alias: Lets programmer give any alias name for same function.

The startup.o file generated is of elf executable format, various sections of which are shown below:

```
C:\Windows\System32\cmd.exe

C:\Users\Training\STM32CubeIDE\workspace_1.5.0\new\Src>make -f Makefile.mak
arm-none-eabi-gcc -c -mcpu=cortex-m4 -mthumb -std=gnu11 -Wall -O0  -o stm32_startup.o stm32_startup.c

C:\Users\Training\STM32CubeIDE\workspace_1.5.0\new\Src>arm-none-eabi-objdump.exe -h stm32_startup.o

stm32_startup.o:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         00000090  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000000  00000000  00000000  000000c4  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  00000000  00000000  000000c4  2**0
                  ALLOC
  3 .isr_vector   00000188  00000000  00000000  000000c4  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, DATA
  4 .comment      0000004e  00000000  00000000  0000024c  2**0
                  CONTENTS, READONLY
  5 .ARM.attributes 0000002e  00000000  00000000  0000029a  2**0
                  CONTENTS, READONLY

C:\Users\Training\STM32CubeIDE\workspace_1.5.0\new\Src>
```

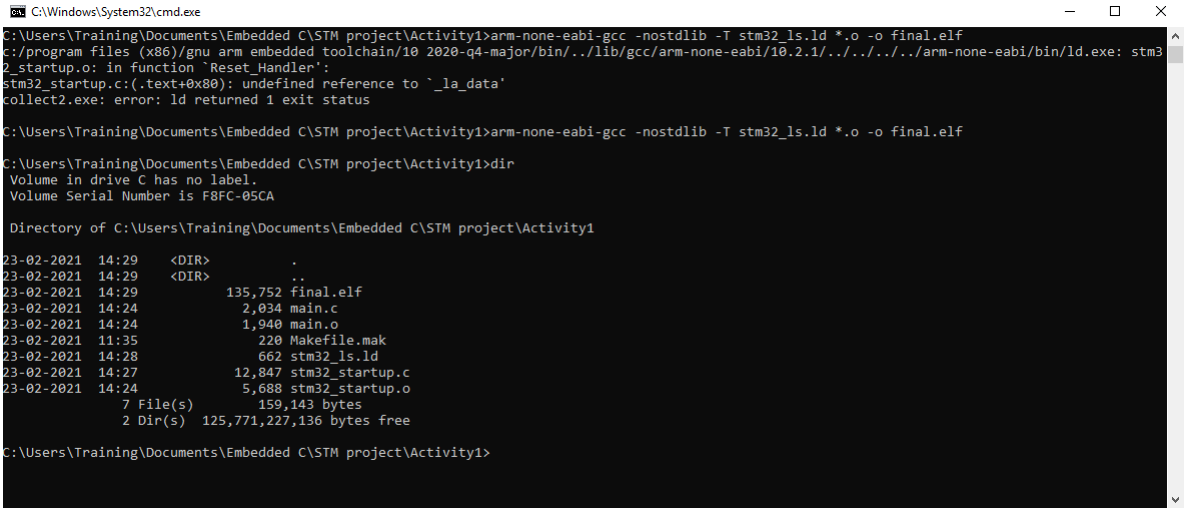
Fig 1.2.2: Startup command

1.3- LINKER SCRIPT

- Linkers take one or more object files or libraries as input and combines them to create a single executable file as output.
- Linker scripts decide how different sections of object file should be merged to create an output file.
- Reset handler is the entry point to the application
- Entry command is used to set the "Entry point address" information in the header of final elf file generated.
  Syntax: Entry(symbol_name)
  Entry(Reset_Handler)

Makefile.mak | stm32_ls.ld.txt

```
ENTRY(Reset_Handler)

MEMORY
{
  FLASH(rx):ORIGIN = 0x08000000,LENGTH=1024K
  SRAM(rwx):ORIGIN = 0x02000000,LENGTH=128K
}

__max_heap_size = 0x400;
__max_stack_size = 0x200;

SECTIONS
{
  .text :
  {
    *(.isr_vecrtor)
    *(.text)
    *(.rodata)
    . = ALIGN(4);
    __etext = .
    end_of_text = .;
  }> FLASH

  .data :
  {
  __sdata = .;
   start_of_data = 0x02000000;
   *(.data)
  }> SRAM AT> FLASH

  .bss :
  {
  *(.bss)
  . = ALIGN(4);
  }> SRAM
}
```



```
C:\Windows\System32\cmd.exe
C:\Users\Training\Documents\Embedded C\STM project\Activity1>arm-none-eabi-gcc -nostdlib -T stm32_ls.ld *.o -o final.elf
c:/program files (x86)/gnu arm embedded toolchain/10 2020-q4-major/bin/../lib/gcc/arm-none-eabi/10.2.1/../../../../arm-none-eabi/bin/ld.exe: stm3
2_startup.o: in function `Reset_Handler':
stm32_startup.c:(.text+0x80): undefined reference to `_la_data'
collect2.exe: error: ld returned 1 exit status

C:\Users\Training\Documents\Embedded C\STM project\Activity1>arm-none-eabi-gcc -nostdlib -T stm32_ls.ld *.o -o final.elf

C:\Users\Training\Documents\Embedded C\STM project\Activity1>dir
 Volume in drive C has no label.
 Volume Serial Number is F8FC-05CA

 Directory of C:\Users\Training\Documents\Embedded C\STM project\Activity1

23-02-2021  14:29    <DIR>          .
23-02-2021  14:29    <DIR>          ..
23-02-2021  14:29           135,752 final.elf
23-02-2021  14:24             2,034 main.c
23-02-2021  14:24             1,940 main.o
23-02-2021  11:35               220 Makefile.mak
23-02-2021  14:28               662 stm32_ls.ld
23-02-2021  14:27            12,847 stm32_startup.c
23-02-2021  14:24             5,688 stm32_startup.o
               7 File(s)        159,143 bytes
               2 Dir(s)  125,771,227,136 bytes free

C:\Users\Training\Documents\Embedded C\STM project\Activity1>
```

Fig 1.3.1 command to generate final.elf file

1.4- DEBUGGING TECHNIQUES

- The STM32F407VG is embedded with on chip debugger for debugging the code.
- The OCD ON-Chip Debugger aims to provide debugging, in system programming and boundary scan testing for embedded target devices.
- OCD is a free and opensource host application allows you to program, debug, and analyze your applications using GDB.
- It supports various target boards based on different processor architecture.

Activity 2 – IMPLEMENTATION OF PROTOCOLS USING STM IDE

Implementation of protocols for STM32F407VG microcontroller featuring ARM32 bit ARM-cortex - M4 with FPU core using HAL library.

2.1 GPIO:

Toggling LED at pin PD12 at GREEN_LED_GPIO_PORT. Serial wire is enabled at pin PA13.

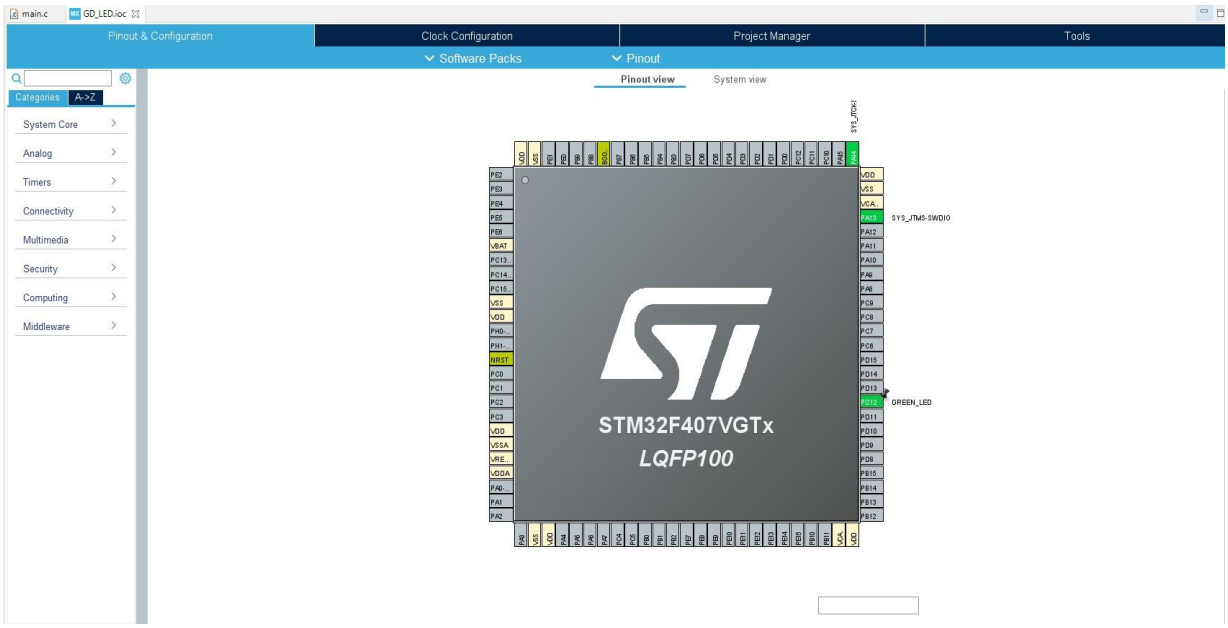


Fig: 2.1.1 GPIO pin configuration

```
  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  /* USER CODE BEGIN 2 */

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
      HAL_GPIO_TogglePin(GREEN_LED_GPIO_Port, GREEN_LED_Pin);
      HAL_Delay(500);
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}
```

Fig: 2.1.2 GPIO configuration code

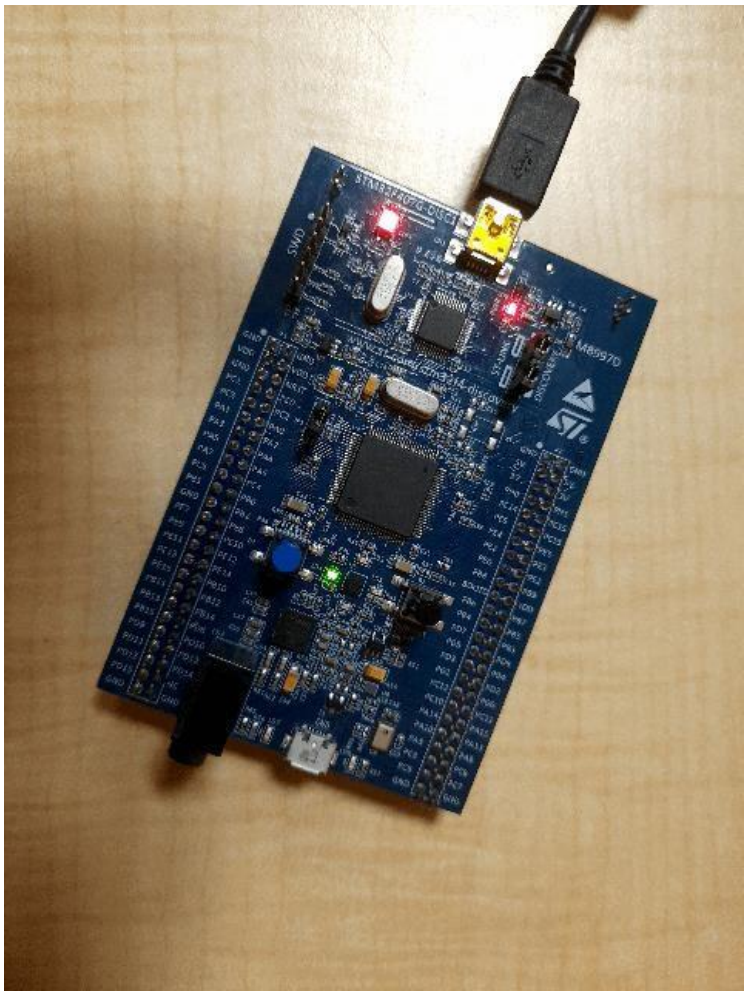


Fig: 2.1.2 LED toggling

2.2 EXTI:

Blue button at PA0 works as an external interrupt.
When the blue button is pressed the Green LED at pin PD12 toggles.

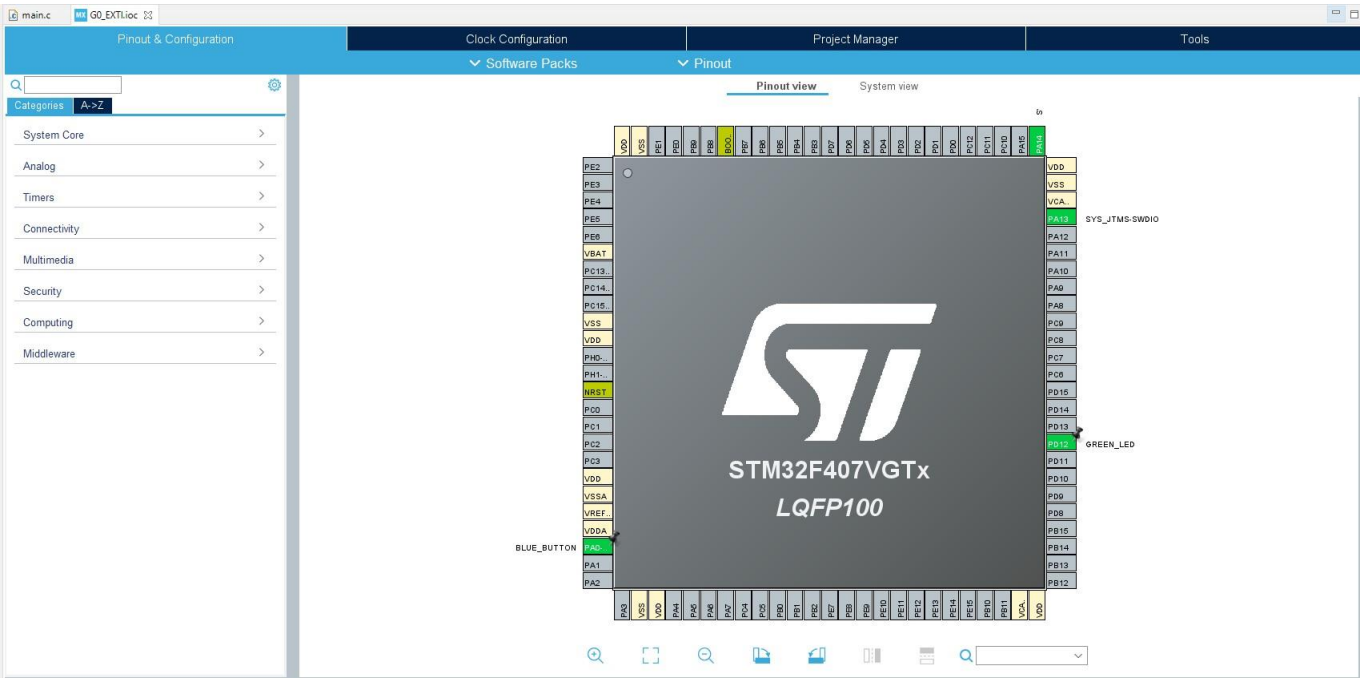


Fig: 2.2.1 EXTI pin configuration

In the main.c file a flag is initialized and if the flag == 1, the condition under the if loop executed to toggle the LED at PD12.

```c
/* USER CODE END PV */

/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
/* USER CODE BEGIN PFP */

/* USER CODE END PFP */

/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */

/* USER CODE END 0 */

/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  /* USER CODE BEGIN 2 */

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
int read_pin=0;
  while (1)
  {

      read_pin=HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_3);
                        if(read_pin==1)
                        {
```

Fig: 2.2.1 EXTI configuration code

2.3 ADC

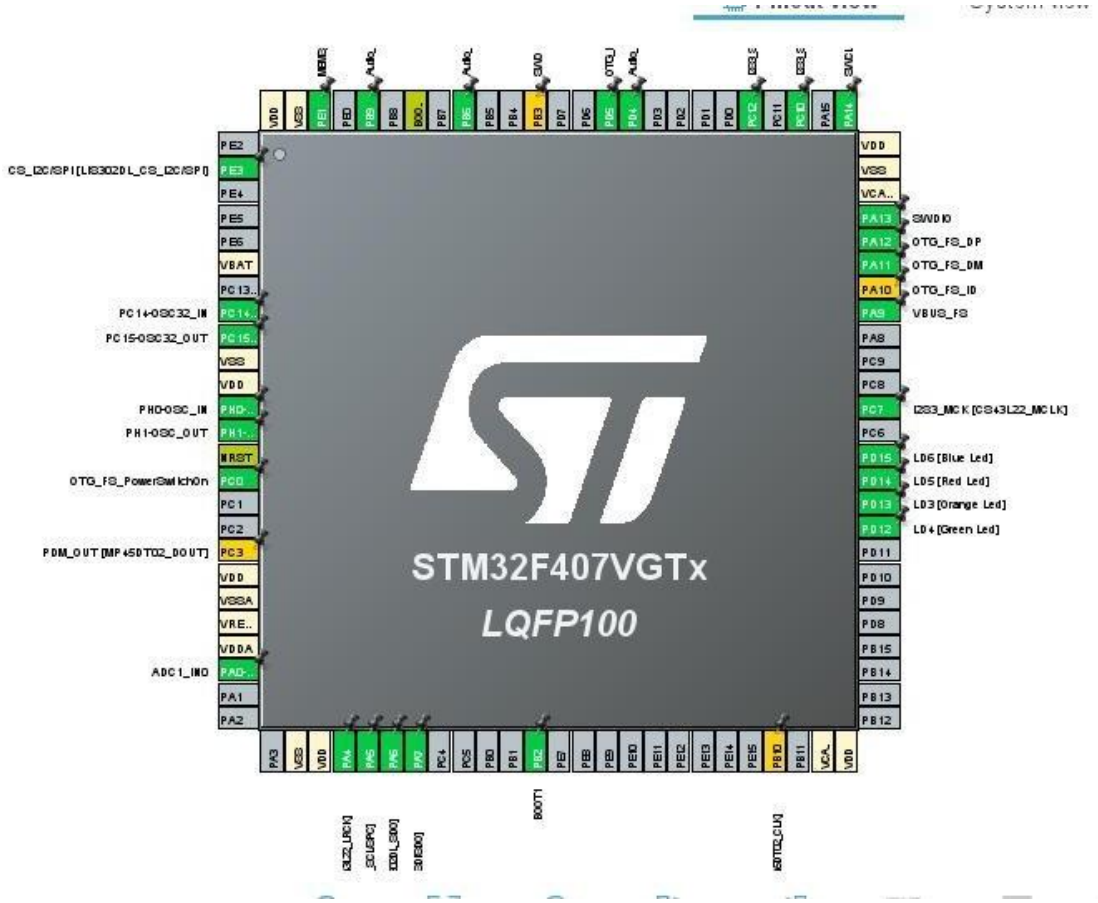


Fig: 2.3.1 ADC pin configuration

*main.c

```c
/* USER CODE END Header */
/* Includes ------------------------------------------------------------------*/
#include "main.h"
#include "usb_host.h"


ADC_HandleTypeDef hadc1;

I2C_HandleTypeDef hi2c1;

I2S_HandleTypeDef hi2s3;

SPI_HandleTypeDef hspi1;


/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_I2C1_Init(void);
static void MX_I2S3_Init(void);
static void MX_SPI1_Init(void);
static void MX_ADC1_Init(void);
void MX_USB_HOST_Process(void);


uint32_t adc_value;

int main(void)
{

  HAL_Init();

  SystemClock_Config();

  MX_GPIO_Init();
  MX_I2C1_Init();
  MX_I2S3_Init();
```

*main.c

```c

  MX_GPIO_Init();
  MX_I2C1_Init();
  MX_I2S3_Init();
  MX_SPI1_Init();
  MX_USB_HOST_Init();
  MX_ADC1_Init();

  while (1)
  {

    MX_USB_HOST_Process();


    HAL_ADC_Start(&hadc1);


    if(HAL_ADC_PollForConversion(&hadc1, 5)== HAL_OK){

        adc_value= HAL_ADC_GetValue(&hadc1);

    }


    HAL_ADC_Stop(&hadc1);
    HAL_Delay(100);

  }

}


void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
  RCC_PeriphCLKInitTypeDef PeriphClkInitStruct = {0};
```

Fig: 2.3.2 ADC configuration code

2.4 SPI

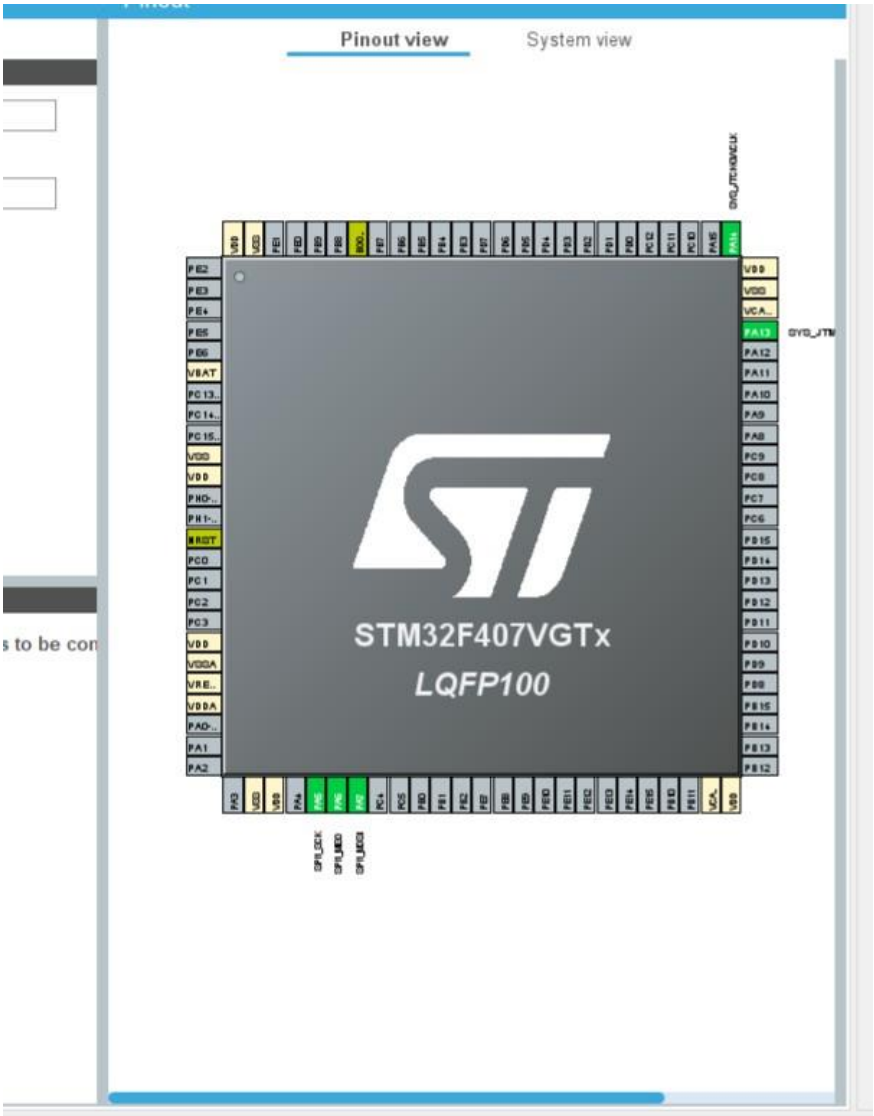
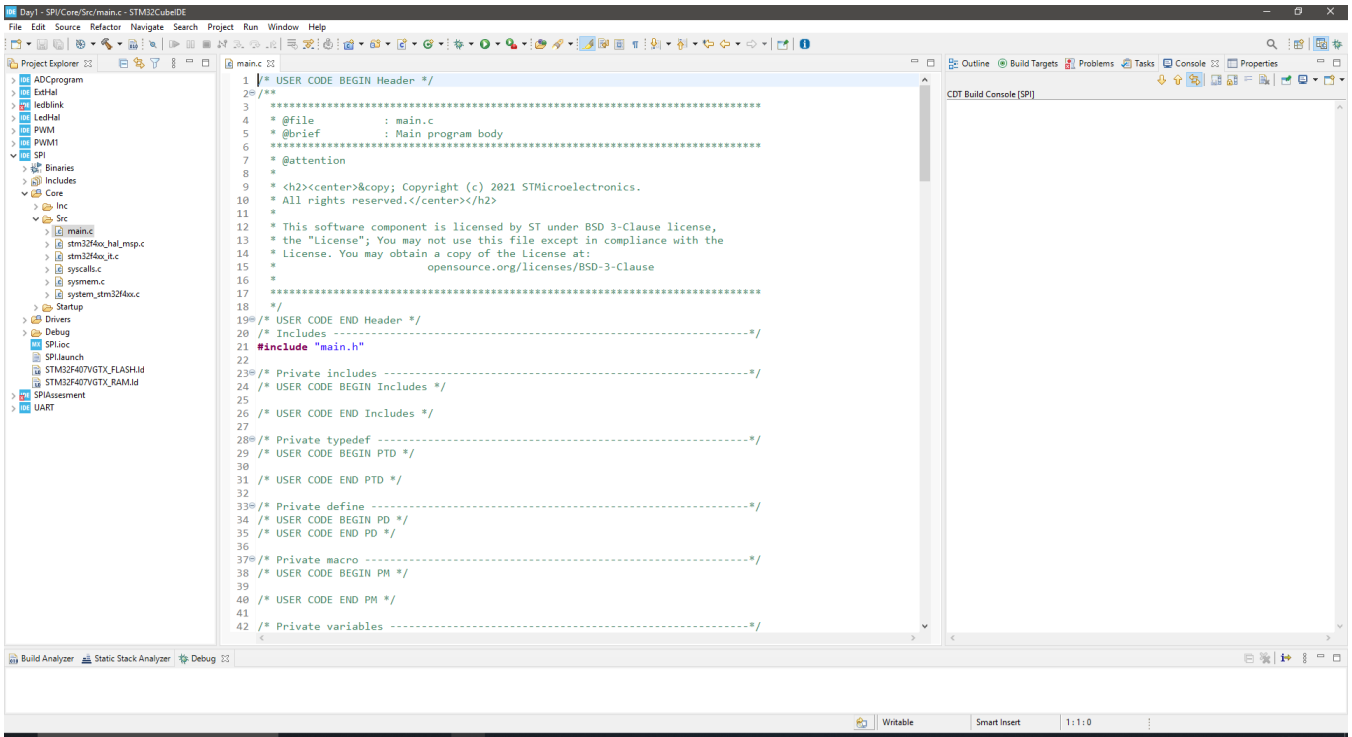
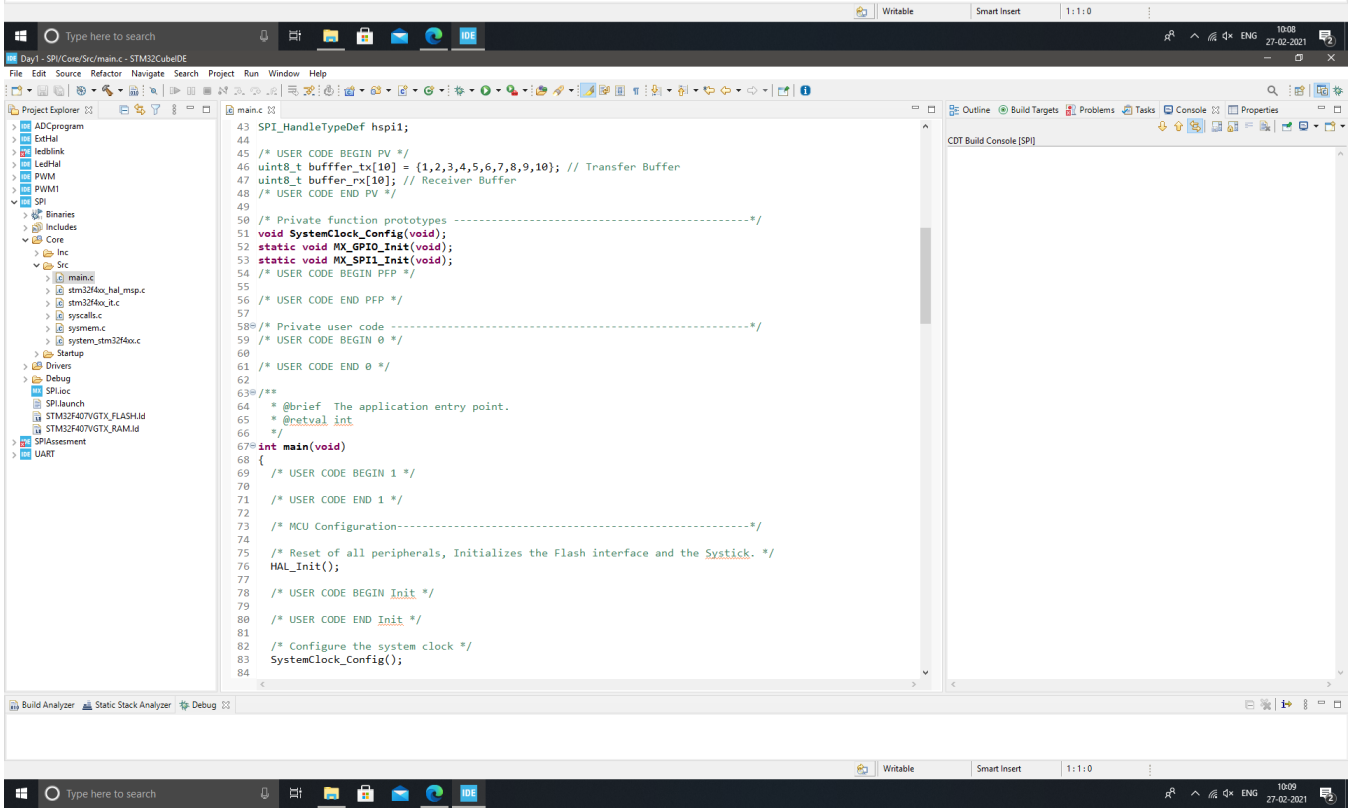


Fig: 2.4.1 SPI Pin configuration


Day1 - SPI/Core/Src/main.c - STM32CubeIDE
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
ADCprogram
ExtHal
ledblink
LedHal
PWM
PWM1
SPI
Binaries
Includes
Core
Inc
Src
main.c
stm32f4xx_hal_msp.c
stm32f4xx_it.c
syscalls.c
sysmem.c
system_stm32f4xx.c
Startup
Drivers
Debug
SPI.ioc
SPI.launch
STM32F407VGTX_FLASH.ld
STM32F407VGTX_RAM.ld
SPIAssesment
UART
main.c
/* USER CODE BEGIN Header */
/**
* @file : main.c
* @brief : Main program body
* @attention
* <h2> © Copyright (c) 2021 STMicroelectronics.
* All rights reserved. </h2>
* This software component is licensed by ST under BSD 3-Clause license,
* the "License"; You may not use this file except in compliance with the
* License. You may obtain a copy of the License at:
opensource.org/licenses/BSD-3-Clause
*/
/* USER CODE END Header */
/* Includes */
#include "main.h"
/* Private includes */
/* USER CODE BEGIN Includes */
/* USER CODE END Includes */
/* Private typedef */
/* USER CODE BEGIN PTD */
/* USER CODE END PTD */
/* Private define */
/* USER CODE BEGIN PD */
/* USER CODE END PD */
/* Private macro */
/* USER CODE BEGIN PM */
/* USER CODE END PM */
/* Private variables */
Outline
Build Targets
Problems
Tasks
Console
Properties
CDT Build Console [SPI]
Build Analyzer
Static Stack Analyzer
Debug
Writable
Smart Insert
1 : 1 : 0


Day1 - SPI/Core/Src/main.c - STM32CubeIDE
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
main.c
SPI_HandleTypeDef hspi1;
/* USER CODE BEGIN PV */
uint8_t bufffer_tx[10] = {1,2,3,4,5,6,7,8,9,10}; // Transfer Buffer
uint8_t buffer_rx[10]; // Receiver Buffer
/* USER CODE END PV */
/* Private function prototypes */
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_SPI1_Init(void);
/* USER CODE BEGIN PFP */
/* USER CODE END PFP */
/* Private user code */
/* USER CODE BEGIN 0 */
/* USER CODE END 0 */
/**
* @brief The application entry point.
* @retval int
*/
int main(void)
{
/* USER CODE BEGIN 1 */
/* USER CODE END 1 */
/* MCU Configuration */
/* Reset of all peripherals, Initializes the Flash interface and the Systick. */
HAL_Init();
/* USER CODE BEGIN Init */
/* USER CODE END Init */
/* Configure the system clock */
SystemClock_Config();
Outline
Build Targets
Problems
Tasks
Console
Properties
CDT Build Console [SPI]
Build Analyzer
Static Stack Analyzer
Debug
Writable
Smart Insert
1 : 1 : 0

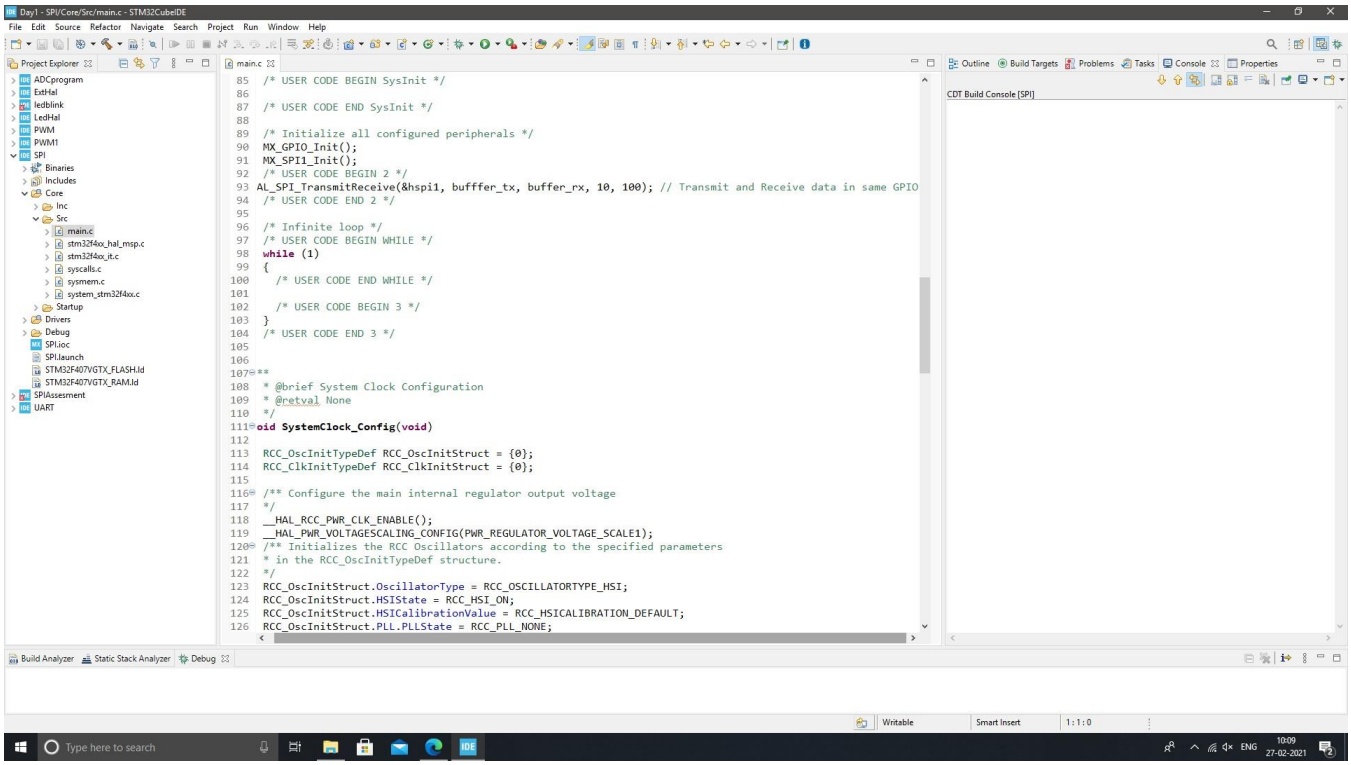


Fig: 2.4.2 SPI configuration code

2.5 UART

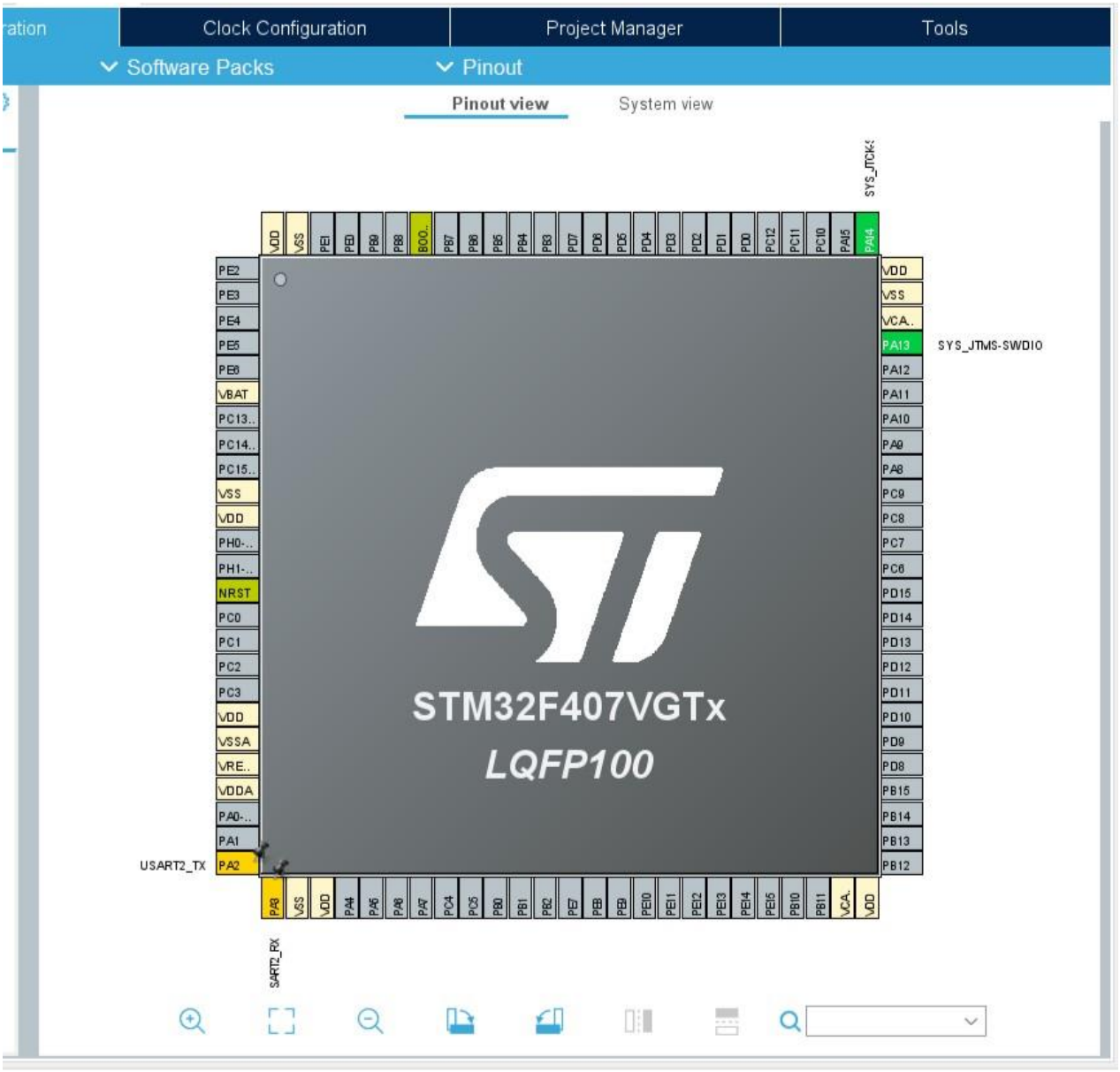
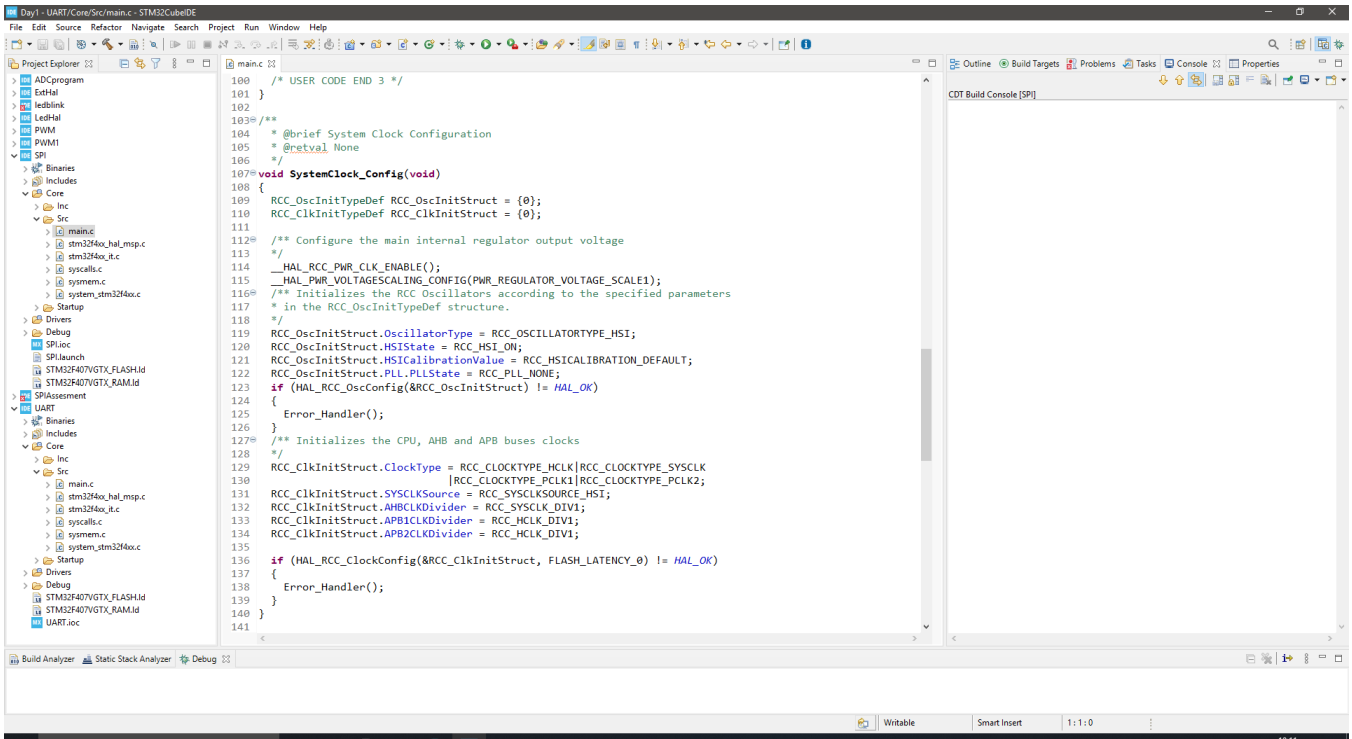
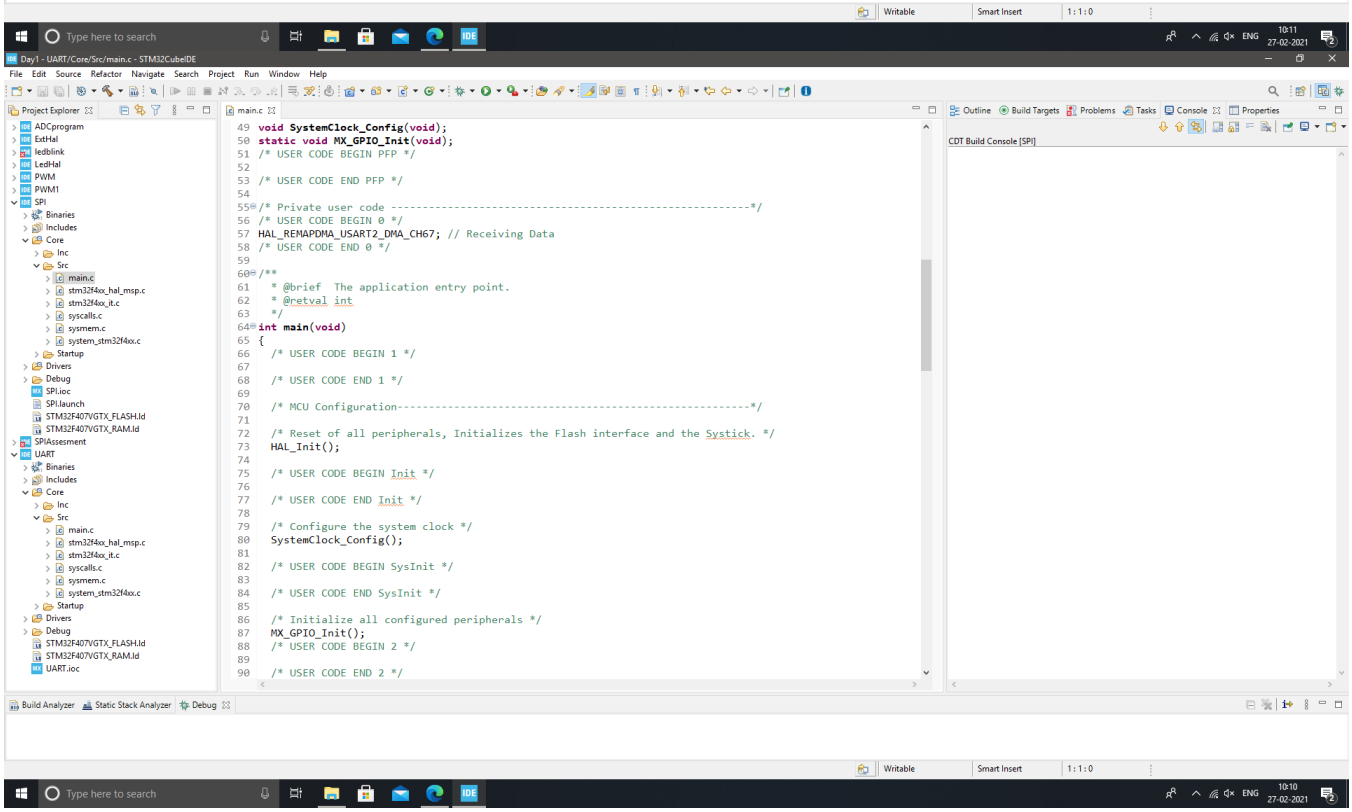


Fig: 2.5.1 UART Pin configuration


Day1 - UART/Core/Src/main.c - STM32CubeIDE
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
main.c
Outline Build Targets Problems Tasks Console Properties
CDT Build Console [SPI]
/* USER CODE END 3 */
}
/**
* @brief System Clock Configuration
* @retval None
*/
void SystemClock_Config(void)
{
RCC_OscInitTypeDef RCC_OscInitStruct = {0};
RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
/** Configure the main internal regulator output voltage
*/
__HAL_RCC_PWR_CLK_ENABLE();
__HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE1);
/** Initializes the RCC Oscillators according to the specified parameters
* in the RCC_OscInitTypeDef structure.
*/
RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSI;
RCC_OscInitStruct.HSIState = RCC_HSI_ON;
RCC_OscInitStruct.HSICalibrationValue = RCC_HSICALIBRATION_DEFAULT;
RCC_OscInitStruct.PLL.PLLState = RCC_PLL_NONE;
if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
{
Error_Handler();
}
/** Initializes the CPU, AHB and APB buses clocks
*/
RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
|RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_HSI;
RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV1;
RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;
if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_0) != HAL_OK)
{
Error_Handler();
}
}
Build Analyzer Static Stack Analyzer Debug
Writable Smart Insert 1 : 1 : 0


Day1 - UART/Core/Src/main.c - STM32CubeIDE
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
main.c
Outline Build Targets Problems Tasks Console Properties
CDT Build Console [SPI]
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
/* USER CODE BEGIN PFP */
/* USER CODE END PFP */
/* Private user code ---*/
/* USER CODE BEGIN 0 */
HAL_REMAPDMA_USART2_DMA_CH67; // Receiving Data
/* USER CODE END 0 */
/**
* @brief The application entry point.
* @retval int
*/
int main(void)
{
/* USER CODE BEGIN 1 */
/* USER CODE END 1 */
/* MCU Configuration--*/
/* Reset of all peripherals, Initializes the Flash interface and the Systick. */
HAL_Init();
/* USER CODE BEGIN Init */
/* USER CODE END Init */
/* Configure the system clock */
SystemClock_Config();
/* USER CODE BEGIN SysInit */
/* USER CODE END SysInit */
/* Initialize all configured peripherals */
MX_GPIO_Init();
/* USER CODE BEGIN 2 */
/* USER CODE END 2 */
Build Analyzer Static Stack Analyzer Debug
Writable Smart Insert 1 : 1 : 0