

# GENESIS Learning Report - Embedded C: Hardware + Programming + Testing (99003161)



## Document History

Ver. Rel. No.	Release Date	Prepared. By	Reviewed By	Approved By	Remarks/Revision Details
1.0	24-12-20	DEEPAK KUMAR SAHOO		Dr. Vivek K and Bhargav N	
1.1	28-12-20	DEEPAK KUMAR SAHOO		Dr. Vivek K and Bhargav N	
1.2	29-12-20	DEEPAK KUMAR SAHOO		Dr. Vivek K and Bhargav N	

## Contents

<b>1. ACTIVITY1 .....</b>	<b>5</b>
<b>1.1 LINKER SCRIPT.....</b>	<b>5</b>
<b>1.2 MAKE FILE.....</b>	<b>6</b>
<i>1.2.1 Main code.....</i>	<i>6</i>
<i>1.2.2 Make file execution.....</i>	<i>6</i>
<i>1.2.3 Make file code.....</i>	<i>7</i>
<b>1.3 STARTUP .....</b>	<b>7</b>
<b>1.4 OUTPUT FILES .....</b>	<b>8</b>
<b>1.5 GITHUB LINK OF CODE FILES .....</b>	<b>8</b>
<b>1.6 DEBUGGING TECHNIQUES.....</b>	<b>8</b>
<i>1.6.1 Step in, step over and step out .....</i>	<i>8</i>
<i>1.6.2 Disassembly.....</i>	<i>9</i>
<i>1.6.3 Break points .....</i>	<i>9</i>
<b>2. ACTIVITY 2 (DRIVER CODE DEVELOPMENT) .....</b>	<b>10</b>
<b>2.1 MCU SPECIFIC HEADER FILE .....</b>	<b>10</b>
<b>2.2 GPIO DRIVER FILE.....</b>	<b>13</b>
<b>2.3 SOURCE FILE .....</b>	<b>15</b>
<b>2.4 GITHUB LINK TO THESE FILES: .....</b>	<b>20</b>
<b>3. ACTIVITY 3 (MINI PROJECT) .....</b>	<b>21</b>
<b>3.1 MAIN LOGIC .....</b>	<b>21</b>
<b>3.2 ARDUINO CODE .....</b>	<b>21</b>
<b>3.3 GITHUB LINK TO THE PROJECT .....</b>	<b>22</b>
<b>4. REFERENCES.....</b>	<b>23</b>

## Table of Figures

Figure 1.Linker Script.....	5
Figure 2. Main code .....	6
Figure 3. Make file execution .....	6
Figure 4. Make file code .....	7
Figure 5. Startup .....	7
Figure 6. Output files .....	8
Figure 7. Main Logic function .....	21

## List of Tables

No table of figures entries found.

# 1. Activity1

## 1.1 Linker Script

```

1 ENTRY(Reset_Handler)
2 MEMORY
3 {
4     FLASH(rx):ORIGIN =0x08000000,LENGTH =1024K
5     SRAM(rwx):ORIGIN =0x20000000,LENGTH =128K
6 }
7 SECTIONS
8 {
9     .text :
10    {
11        *(.isr_vector)
12        *(.text)
13        *(.text.*)
14        *(.init)
15        *(.fini)
16        *(.rodata)
17        *(.rodata.*)
18        . = ALIGN(4);
19        _etext = .;
20    }> FLASH
21    _la_data = LOADADDR(.data);
22    .data :
23    {
24        _sdata = .;
25        *(.data)
26        *(.data.*)
27        . = ALIGN(4);
28        _edata = .;
29    }> SRAM AT> FLASH
30    .bss :
31    {
32        _sbss = .;
33        __bss_start__ = _sbss;
34        *(.bss)
35        *(.bss.*)
36        *(COMMON)
37        . = ALIGN(4);
38        _ebss = .;
39        __bss_end__ = _ebss;
40        . = ALIGN(4);
41        end = .;
42        __end__ = .;
43    }> SRAM
44 }
```

Figure 1.Linker Script

## 1.2 Make file

### 1.2.1 Main code

```

22 lines (19 sloc) | 559 Bytes
1  #include <stdio.h>
2  int main() {
3      int n, reversedN = 0, remainder, originalN;
4      printf("Enter an integer: ");
5      scanf("%d", &n);
6      originalN = n;
7
8      // reversed integer is stored in reversedN
9      while (n != 0) {
10         remainder = n % 10;
11         reversedN = reversedN * 10 + remainder;
12         n /= 10;
13     }
14
15     // palindrome if originalN and reversedN are equal
16     if (originalN == reversedN)
17         printf("%d is a palindrome.", originalN);
18     else
19         printf("%d is not a palindrome.", originalN);
20
21     return 0;
22 }

```

Figure 2. Main code

### 1.2.2 Make file execution

```

MINGW64:/c:/Users/99003161/desktop/embedded
99003161@EESBLRW365 MINGW64 ~
$ cd desktop
99003161@EESBLRW365 MINGW64 ~/desktop
$ cd embedded
99003161@EESBLRW365 MINGW64 ~/desktop/embedded
$ ls
main.c  Makefile          stm32_ls.ld      syscalls.c
main.h  mcu_exception_handlr_prototypes.txt  stm32_startup.c
99003161@EESBLRW365 MINGW64 ~/desktop/embedded
$ make
arm-none-eabi-gcc -c -mcpu=cortex-m4 -mthumb -mfloat-abi=soft -std=gnu11 -Wall -
00 -o main.o main.c
arm-none-eabi-gcc -c -mcpu=cortex-m4 -mthumb -mfloat-abi=soft -std=gnu11 -Wall -00 -o stm32_startup.o stm32_startup.c
arm-none-eabi-gcc -c -mcpu=cortex-m4 -mthumb -mfloat-abi=soft -std=gnu11 -Wall -00 -o syscalls.o syscalls.c
arm-none-eabi-gcc -mcpu=cortex-m4 -mthumb -mfloat-abi=soft --specs=nano.specs -T stm32_ls.ld -Wl,-Map=final.map -o final.elf main.o stm32_startup.o syscalls.o
99003161@EESBLRW365 MINGW64 ~/desktop/embedded
$

```

Figure 3. Make file execution

### 1.2.3 Make file code

```
32 lines (21 sloc) | 813 Bytes
1 CC=arm-none-eabi-gcc
2 MACH=cortex-m4
3 CFLAGS= -c -mcpu=$(MACH) -mthumb -mfloat-abi=soft -std=gnu11 -Wall -O0
4 LDFLAGS= -mcpu=$(MACH) -mthumb -mfloat-abi=soft --specs=nano.specs -T stm32_ls.ld -Wl,-Map=final.map
5 LDFLAGS_SH= -mcpu=$(MACH) -mthumb -mfloat-abi=soft --specs=rdimon.specs -T stm32_ls.ld -Wl,-Map=final.map
6
7 all:main.o stm32_startup.o syscalls.o final.elf
8
9 semi:main.o stm32_startup.o syscalls.o final_sh.elf
10
11 main.o:main.c
12     $(CC) $(CFLAGS) -o $@ $^
13
14
15 stm32_startup.o:stm32_startup.c
16     $(CC) $(CFLAGS) -o $@ $^
17
18 syscalls.o:syscalls.c
19     $(CC) $(CFLAGS) -o $@ $^
20
21 final.elf: main.o stm32_startup.o syscalls.o
22     $(CC) $(LDFLAGS) -o $@ $^
23
24 final_sh.elf: main.o stm32_startup.o
25     $(CC) $(LDFLAGS_SH) -o $@ $^
26
27 clean:
28     rm -rf *.o *.elf
29
```

Figure 4. Make file code

### 1.3 Startup

```
99003161@EESBLRW365 MINGW64 ~/desktop/embedded
$ 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

99003161@EESBLRW365 MINGW64 ~/desktop/embedded
$ |
```

Figure 5. Startup

## 1.4 Output Files

> embedded				
Name	Date modified	Type	Size	
final.elf	1/2/2021 2:53 PM	ELF File	136 KB	
final.map	1/2/2021 2:53 PM	MAP File	27 KB	
main	1/2/2021 4:22 PM	C File	1 KB	
main.h	12/26/2020 3:16 PM	H File	2 KB	
main.o	1/2/2021 2:53 PM	O File	1 KB	
Makefile	12/26/2020 2:38 PM	File	1 KB	
mcu_exception_handlr_prototypes	12/26/2020 3:10 PM	Text Document	9 KB	
stm32_ls.ld	12/26/2020 1:04 PM	LD File	1 KB	
stm32_startup	12/26/2020 12:00 PM	C File	13 KB	
stm32_startup.o	1/2/2021 2:53 PM	O File	6 KB	
syscalls	12/26/2020 2:26 PM	C File	5 KB	
syscalls.o	1/2/2021 2:53 PM	O File	3 KB	

Figure 6. Output files

## 1.5 GitHub Link of code files

[embeddedC/embedded at master · 99003161/embeddedC \(github.com\)](https://github.com/99003161/embeddedC)

## 1.6 Debugging Techniques

### 1.6.1 Step in, step over and step out

**Step over** – An action to take in the debugger that will step over a given line. If the line contains a function, the function will be executed and the result returned without debugging each line. If we have a break point in the program and if we press step over button, then the line where the program was paused, that line will be executed. Then the program would pause at the next line.

**Step into** – An action to take in the debugger. If the current program line contains a function or a method, we can shift the debugging control into the function by pressing ‘step in’ button.

**Step out** - Once ‘step in’ action is performed, step return will be enabled. The debugging control will return from the method when step return is pressed. When execution is paused inside a function, you can click the Step Out button on the Debug toolbar or select Debug: Step Out to step out of the function. The debugger executes the rest of the function without pausing, and then returns to the line after the function call and pauses.



### **1.6.2 Disassembly**

The Debug Disassembly Window gives the user access to debugging in assembly language for project written in C. The Debug Disassembly Window allows the user to perform all the normal debug operations including single stepping and setting breakpoints on the individual assembly instructions generated from C code.

### **1.6.3 Break points**

Setting breakpoints while debugging code for an embedded system is a common and familiar task. Each Cortex M-series device supports some number of hardware breakpoints. These are comparators in the CPU core that pauses the core when a designated match condition occurs (e.g. the program counter matches the value that corresponds to the address of a specific instruction).

## 2. Activity 2 (Driver code development)

### 2.1 MCU Specific Header File

```
#include <stdint.h>
#ifndef INC_STM32F407XX_H_
#define INC_STM32F407XX_H_

//To find base addresses of the memories
#define FLASH_BASEADDR 0x08000000U
#define SRAM1_BASEADDR 0x20000000U
#define SRAM2_BASEADDR 0x2001C000U
#define SRAM 0x20000000U
#define ROM_BASEADDR 0x1FFF0000U

// To find bus address
#define APB1_BASEADDR 0x40000000U
#define APB2_BASEADDR 0x40010000U
#define AHB1_BASEADDR 0x40020000U
#define AHB2_BASEADDR 0x50000000U

//To find the base address of the peripheral hanging on AHB1
#define GPIOA_BASEADDR (AHB1_BASEADDR + 0x0000U)
#define GPIOB_BASEADDR (AHB1_BASEADDR + 0x0400U)
#define GPIOC_BASEADDR (AHB1_BASEADDR + 0x0800U)
#define GPIOD_BASEADDR (AHB1_BASEADDR + 0x0C00U)
#define GPIOE_BASEADDR (AHB1_BASEADDR + 0x1000U)
#define GPIOF_BASEADDR (AHB1_BASEADDR + 0x1400U)
#define GPIOG_BASEADDR (AHB1_BASEADDR + 0x1800U)
#define GPIOH_BASEADDR (AHB1_BASEADDR + 0x1C00U)
#define GPIOI_BASEADDR (AHB1_BASEADDR + 0x2000U)
#define GPIOJ_BASEADDR (AHB1_BASEADDR + 0x2400U)
#define GPIOK_BASEADDR (AHB1_BASEADDR + 0x2800U)
#define RCC_BASEADDR (AHB1_BASEADDR + 0x3800U)

//To find the base address of the peripheral hanging on APB1
#define I2C1 (APB1_BASEADDR + 0x5400U)
#define I2C2 (APB1_BASEADDR + 0x5800U)
#define I2C3 (APB1_BASEADDR + 0x5C00U)
#define SPI2 (APB1_BASEADDR + 0x3800U)
#define SPI3 (APB1_BASEADDR + 0x3C00U)
#define USART2 (APB1_BASEADDR + 0x4400U)
#define USART3 (APB1_BASEADDR + 0x4800U)

//To find the base address of the peripheral hanging on APB2
#define SPI1 (APB2_BASEADDR + 0x3000U)
#define USART1 (APB2_BASEADDR + 0x1000U)
#define USART6 (APB2_BASEADDR + 0x1400U)
#define EXTI (APB2_BASEADDR + 0x3C00U)
```

```
#define SYSCFG                (APB2_BASEADDR + 0x3800U)

                                //Registers or Reg structures of GPIOA peripherals

typedef struct
{
    volatile uint32_t MODER;
    volatile uint32_t OTYPER;
    volatile uint32_t OSPEEDR;
    volatile uint32_t PUPDR;
    volatile uint32_t IDR;
    volatile uint32_t ODR;
    volatile uint32_t BSRR;
    volatile uint32_t LCKR;
    volatile uint32_t AFRL;
    volatile uint32_t AFRH;
}GPIO_REGDEF_t;

                                //Registers or Reg structure of RCC

typedef struct
{
    volatile uint32_t CR;
    volatile uint32_t PLLCFGR;
    volatile uint32_t CFGR;
    volatile uint32_t CIR;
    volatile uint32_t AHB1RSTR;
    volatile uint32_t AHB2RSTR;
    volatile uint32_t AHB3RSTR;
    volatile uint32_t RESERVED0;
    volatile uint32_t APB1RSTR;
    volatile uint32_t APB2RSTR;
    volatile uint32_t RESERVED1[0];
    volatile uint32_t RESERVED1[1];
    volatile uint32_t AHB1ENR;
    volatile uint32_t AHB2ENR;
    volatile uint32_t AHB3ENR;
    volatile uint32_t RESERVED2;
    volatile uint32_t APB1ENR;
    volatile uint32_t APB2ENR;
    volatile uint32_t RESERVED3[0];
    volatile uint32_t RESERVED3[1];
    volatile uint32_t AHB1LPENR;
    volatile uint32_t AHB2LPENR;
    volatile uint32_t AHB3LPENR;
    volatile uint32_t RESERVED4;
    volatile uint32_t APB1LPENR;
    volatile uint32_t APB2LPENR;
    volatile uint32_t RESERVED5[0];
    volatile uint32_t RESERVED5[1];
    volatile uint32_t BDCR;
    volatile uint32_t CSR;
    volatile uint32_t RESERVED6[0];
    volatile uint32_t RESERVED6[1];
}
```

```

    volatile uint32_t SSCGR;
    volatile uint32_t PLLI2SCFGR;
    volatile uint32_t PLLSAICFGR;
    volatile uint32_t DCKCFGR;
}RCC_REGDEF_t;

//Peripheral definition for GPIO
#define GPIOA ((GPIO_REGDEF_t*) GPIOA_BASEADDR)
#define GPIOB ((GPIO_REGDEF_t*) GPIOB_BASEADDR)
#define GPIOC ((GPIO_REGDEF_t*) GPIOC_BASEADDR)
#define GPIOD ((GPIO_REGDEF_t*) GPIOD_BASEADDR)
#define GPIOE ((GPIO_REGDEF_t*) GPIOE_BASEADDR)
#define GPIOF ((GPIO_REGDEF_t*) GPIOF_BASEADDR)
#define GPIOG ((GPIO_REGDEF_t*) GPIOG_BASEADDR)
#define GPIOH ((GPIO_REGDEF_t*) GPIOH_BASEADDR)
#define GPIOI ((GPIO_REGDEF_t*) GPIOI_BASEADDR)

// Peripheral definition for RCC
#define RCC ((RCC_REGDEF_t*) RCC_BASEADDR)

//Clock enable macros for GPIOx
#define GPIOA_PCLK_EN() (RCC->AHB1ENR |= (1<<0))
#define GPIOB_PCLK_EN() (RCC->AHB1ENR |= (1<<1))
#define GPIOC_PCLK_EN() (RCC->AHB1ENR |= (1<<2))
#define GPIOD_PCLK_EN() (RCC->AHB1ENR |= (1<<3))
#define GPIOE_PCLK_EN() (RCC->AHB1ENR |= (1<<4))
#define GPIOF_PCLK_EN() (RCC->AHB1ENR |= (1<<5))
#define GPIOG_PCLK_EN() (RCC->AHB1ENR |= (1<<6))
#define GPIOH_PCLK_EN() (RCC->AHB1ENR |= (1<<7))
#define GPIOI_PCLK_EN() (RCC->AHB1ENR |= (1<<8))

//Clock Disable macros for GPIOx
#define GPIOA_PCLK_DI() (RCC->AHB1ENR &= ~(1<<0))
#define GPIOB_PCLK_DI() (RCC->AHB1ENR &= ~(1<<1))
#define GPIOC_PCLK_DI() (RCC->AHB1ENR &= ~(1<<2))
#define GPIOD_PCLK_DI() (RCC->AHB1ENR &= ~(1<<3))
#define GPIOE_PCLK_DI() (RCC->AHB1ENR &= ~(1<<4))
#define GPIOF_PCLK_DI() (RCC->AHB1ENR &= ~(1<<5))
#define GPIOG_PCLK_DI() (RCC->AHB1ENR &= ~(1<<6))
#define GPIOH_PCLK_DI() (RCC->AHB1ENR &= ~(1<<7))
#define GPIOI_PCLK_DI() (RCC->AHB1ENR &= ~(1<<8))

// some important macros
#define ENABLE 1
#define DISABLE 0
#define SET ENABLE
#define RESET DISABLE
#define GPIO_PIN_SET SET
#define GPIO_PIN_RESET RESET

```

```

// GPIO Reset
#define GPIOA_REG_RESET() do{(RCC->AHB1RSTR |=1<<0); (RCC->AHB1RSTR &=~(1<<0));}while(0)
#define GPIOB_REG_RESET() do{(RCC->AHB1RSTR |=1<<1); (RCC->AHB1RSTR &=~(1<<1));}while(0)
#define GPIOC_REG_RESET() do{(RCC->AHB1RSTR |=1<<2); (RCC->AHB1RSTR &=~(1<<2));}while(0)
#define GPIOD_REG_RESET() do{(RCC->AHB1RSTR |=1<<3); (RCC->AHB1RSTR &=~(1<<3));}while(0)
#define GPIOE_REG_RESET() do{(RCC->AHB1RSTR |=1<<4); (RCC->AHB1RSTR &=~(1<<4));}while(0)
#define GPIOF_REG_RESET() do{(RCC->AHB1RSTR |=1<<5); (RCC->AHB1RSTR &=~(1<<5));}while(0)
#define GPIOG_REG_RESET() do{(RCC->AHB1RSTR |=1<<6); (RCC->AHB1RSTR &=~(1<<6));}while(0)
#define GPIOH_REG_RESET() do{(RCC->AHB1RSTR |=1<<7); (RCC->AHB1RSTR &=~(1<<7));}while(0)
#define GPIOI_REG_RESET() do{(RCC->AHB1RSTR |=1<<8); (RCC->AHB1RSTR &=~(1<<8));}while(0)

#endif /* INC_STM32F407XX_H_ */

```

## 2.2 GPIO Driver File

```

/*
 * stm32f407xx_gpio_driver.h
 */

#ifndef INC_STM32F407XX_GPIO_DRIVER_H_
#define INC_STM32F407XX_GPIO_DRIVER_H_
#include "STM32F407xx.h"
#include <stdint.h>

typedef struct
{
    uint8_t GPIO_PINNUMBER;
    uint8_t GPIO_PINNAME;
    uint8_t GPIO_PINMODE;
    uint8_t GPIO_PINSPEED;
    uint8_t GPIO_PINOPTYPE;
    uint8_t GPIO_PINPUPDCONTROL;
    uint8_t GPIO_PINALTFUNMODE;
}GPIO_PINCONFIG_t;

typedef struct
{
    GPIO_REGDEF_t *pGPIOx;
    GPIO_PINCONFIG_t GPIO_PINCONFIG;
}GPIO_HANDLE_t;

//GPIO_PinNumber of GPIOx
#define GPIO_PINNUMBER_0      0
#define GPIO_PINNUMBER_1      1
#define GPIO_PINNUMBER_2      2
#define GPIO_PINNUMBER_3      3
#define GPIO_PINNUMBER_4      4

```

```
#define GPIO_PINNUMBER_5          5
#define GPIO_PINNUMBER_6          6
#define GPIO_PINNUMBER_7          7
#define GPIO_PINNUMBER_8          8
#define GPIO_PINNUMBER_9          9
#define GPIO_PINNUMBER_10         10
#define GPIO_PINNUMBER_11         11
#define GPIO_PINNUMBER_12         12
#define GPIO_PINNUMBER_13         13
#define GPIO_PINNUMBER_14         14
#define GPIO_PINNUMBER_15         15

//Possible Modes
#define GPIO_MODE_IN               0
#define GPIO_MODE_OUT              1
#define GPIO_MODE_AFN              2
#define GPIO_MODE_ANALOG           3

//GPIO_PinSpeed
#define GPIO_SPEED_LOW             0
#define GPIO_SPEED_MEDIUM          1
#define GPIO_SPEED_HIGH            2
#define GPIO_SPEED_VERY_HIGH       3

//GPIO_PinOPType
#define GPIO_OP_TYPE_PP            0
#define GPIO_OP_TYPE_OD            1

//GPIO_PinPuPdControl
#define GPIO_NO_PU_PD              0
#define GPIO_PU                    1
#define GPIO_PD                    2
#define GPIO_RESERVED              3

void GPIO_INIT(GPIO_HANDLE_t *GPIOHANDLE);

void GPIO_DINIT(GPIO_REGDEF_t *pGPIOx);

void GPIO_PERICLKCTRL(GPIO_REGDEF_t *pGPIOx, uint8_t EorDi);

uint8_t GPIO_READFROMIPPIN(GPIO_REGDEF_t *pGPIOx, uint8_t PINNO);
uint16_t GPIO_READFROMIPPORT(GPIO_REGDEF_t *pGPIOx);

void GPIO_WRITETOOUTPUTPIN(GPIO_REGDEF_t *pGPIOx, uint8_t PINNO, uint8_t VALUE);
void GPIO_WRITETOOUTPUTPORT(GPIO_REGDEF_t *pGPIOx, uint16_t VALUE);

void GPIO_TOGGLEOUTPUTPIN(GPIO_REGDEF_t *pGPIOx, uint8_t PINNO);
#endif /* INC_STM32F407XX_GPIO_DRIVER_H_ */
```

## 2.3 Source File

```
/*
 * stm32f407xx_gpio_driver.c
 */
#include "stm32f407xx_gpio_driver.h"
#include <stdint.h>

/*
 *
 * @Brief description: function to enable RCC clock
 *
 * @Function-GPIOX_PCLK_EN(),where x=a..i
 * @Param1-GPIO_REGDEF_t *pGPIOx
 * @Param2-uint8_t EorDi
 * @Definition-GPIO Clock enable and disable-
 */

void GPIO_PERICLKCTRL(GPIO_REGDEF_t *pGPIOx, uint8_t EorDi)
{
    if(EorDi==ENABLE)
    {
        if(pGPIOx == GPIOA)
        {
            GPIOA_PCLK_EN();
        }
        else if(pGPIOx == GPIOB)
        {
            GPIOB_PCLK_EN();
        }
        else if(pGPIOx == GPIOC)
        {
            GPIOC_PCLK_EN();
        }
        else if(pGPIOx == GPIOD)
        {
            GPIOD_PCLK_EN();
        }
        else if(pGPIOx == GPIOE)
        {
            GPIOE_PCLK_EN();
        }
        else if(pGPIOx == GPIOF)
        {
            GPIOF_PCLK_EN();
        }
        else if(pGPIOx == GPIOG)
        {
            GPIOG_PCLK_EN();
        }
    }
}
```

```

        else if(pGPIOx == GPIOH)
        {
            GPIOH_PCLK_EN();
        }
        else if(pGPIOx == GPIOI)
        {
            GPIOI_PCLK_EN();
        }
    }
    else
    {
        if(pGPIOx == GPIOA)
        {
            GPIOA_PCLK_DI();
        }
        else if(pGPIOx == GPIOB)
        {
            GPIOB_PCLK_DI();
        }
        else if(pGPIOx == GPIOC)
        {
            GPIOC_PCLK_DI();
        }
        else if(pGPIOx == GPIOD)
        {
            GPIOD_PCLK_DI();
        }
        else if(pGPIOx == GPIOE)
        {
            GPIOE_PCLK_DI();
        }
        else if(pGPIOx == GPIOF)
        {
            GPIOF_PCLK_DI();
        }
        else if(pGPIOx == GPIOG)
        {
            GPIOG_PCLK_DI();
        }
        else if(pGPIOx == GPIOH)
        {
            GPIOH_PCLK_DI();
        }
        else if(pGPIOx == GPIOI)
        {
            GPIOI_PCLK_DI();
        }
    }
}

/*
*
```



```
* @Brief description:GPIO Clock enable and disable
* @Function-gPIO Port intzatzation
* @Param1-GPIO_Handle_t *
*/
```

```
void GPIO_INIT(GPIO_HANDLE_t *GPIOHANDLE)
{
    //Init Mode
    uint32_t temp=0;
    temp = (GPIOHANDLE->GPIO_PINCONFIG.GPIO_PINMODE << (2 * (GPIOHANDLE-
>GPIO_PINCONFIG.GPIO_PINNUMBER)));
    (GPIOHANDLE->pGPIOx->MODER) &= ~(0x03<<2*GPIOHANDLE->GPIO_PINCONFIG.GPIO_PINNUMBER);
    //First Reset the same pins and then set the values
    GPIOHANDLE->pGPIOx->MODER |=temp;
    //Config Speed
    temp=0;
    temp = (GPIOHANDLE->GPIO_PINCONFIG.GPIO_PINSPEED <<(2 * (GPIOHANDLE-
>GPIO_PINCONFIG.GPIO_PINNUMBER)));
    (GPIOHANDLE->pGPIOx->OSPEEDR) &= ~(0x03<<2*GPIOHANDLE-
>GPIO_PINCONFIG.GPIO_PINNUMBER);
    GPIOHANDLE->pGPIOx->OSPEEDR |=temp;

    //Config Pull up and pull down
    temp=0;
    temp = (GPIOHANDLE->GPIO_PINCONFIG.GPIO_PINPUPDCONTROL << (2*(GPIOHANDLE-
>GPIO_PINCONFIG.GPIO_PINNUMBER)));
    (GPIOHANDLE->pGPIOx->PUPDR) &= ~(0x03<<2*GPIOHANDLE->GPIO_PINCONFIG.GPIO_PINNUMBER);
    GPIOHANDLE->pGPIOx->PUPDR |=temp;

    //Config Output type
    temp=0;
    temp = (GPIOHANDLE->GPIO_PINCONFIG.GPIO_PINOPTHYPE << (GPIOHANDLE-
>GPIO_PINCONFIG.GPIO_PINNUMBER));
    (GPIOHANDLE->pGPIOx->OTYPER) &= ~(0x01<<GPIOHANDLE->GPIO_PINCONFIG.GPIO_PINNUMBER);
    GPIOHANDLE->pGPIOx->OTYPER |=temp;

    //Config Alternating Function

    if(GPIOHANDLE->GPIO_PINCONFIG.GPIO_PINMODE == GPIO_MODE_AFN)
    {
        uint8_t temp1=0, temp2=0;
        temp1 = GPIOHANDLE->GPIO_PINCONFIG.GPIO_PINNUMBER/8;
        temp2 = GPIOHANDLE->GPIO_PINCONFIG.GPIO_PINNUMBER%8;
        GPIOHANDLE->pGPIOx->AFR[temp1] &= ~(0x0F<<4*temp2);
        GPIOHANDLE->pGPIOx->AFR[temp1]= GPIOHANDLE-
>GPIO_PINCONFIG.GPIO_PINALTFUNMODE<<(4*temp2);
    }
}
/*
*
*
*/
```

```
* @Brief description:reset of GPIO port
* @Function-GPIO_DInit
* @Param1-GPIO_REGDEF_t*
*/
```

```
void GPIO_DINIT(GPIO_REGDEF_t *pGPIOx)
{
```

```
    if(pGPIOx == GPIOA)
    {
        GPIOA_REG_RESET();
    }
    else if(pGPIOx == GPIOB)
    {
        GPIOB_REG_RESET();
    }
    else if(pGPIOx == GPIOC)
    {
        GPIOC_REG_RESET();
    }
    else if(pGPIOx == GPIOD)
    {
        GPIOD_REG_RESET();
    }
    else if(pGPIOx == GPIOE)
    {
        GPIOE_REG_RESET();
    }
    else if(pGPIOx == GPIOF)
    {
        GPIOF_REG_RESET();
    }
    else if(pGPIOx == GPIOG)
    {
        GPIOG_REG_RESET();
    }
    else if(pGPIOx == GPIOH)
    {
        GPIOH_REG_RESET();
    }
    else if(pGPIOx == GPIOI)
    {
        GPIOI_REG_RESET();
    }
}
```

```
/*
*
*
* @Brief description:GPIO Input from pin
* @Function-GPIO_ReadFromInptPin
* @Param1-GPIO_REGDEF_t *
*
*/
```

```
uint8_t GPIO_READFROMIPPIN(GPIO_REGDEF_t *pGPIOx, uint8_t PINNO)
{
    uint8_t value=0;
    value = (uint8_t)(pGPIOx->IDR>>PINNO)&(0x00000001);
    return value;
}

/*
 *
 *
 * * @Brief description:Input from port input
 * @Function-GPIO_ReadFromInptPort
 * @Param1-GPIO_REGDEF_t *
 *
 */

uint16_t GPIO_READFROMIPPOROT(GPIO_REGDEF_t *pGPIOx)
{
    uint16_t value=0;
    value = (uint16_t)(pGPIOx->IDR);
    return value;
}

/*
 *
 *
 * @Brief description:output port pin set or reset
 * @Function-GPIO_WriteToOutputPin
 * @Param1-GPIO_REGDEF_t *
 *
 */

void GPIO_WRITETOOUTPUTPIN(GPIO_REGDEF_t *pGPIOx, uint8_t PINNO, uint8_t VALUE)
{
    if(VALUE == GPIO_PIN_SET)
    {
        pGPIOx->ODR|=(1<<PINNO);
    }
    else
    {
        pGPIOx->ODR&=~(1<<PINNO);
    }
}

/*
 *
 * @Brief description:to write to output port
 * @Function-GPIO_WriteToOutputPort
 * @Param1-GPIO_REGDEF_t *
```

```
*  
*/  
  
void GPIO_WRITETOOUTPUTPORT(GPIO_REGDEF_t *pGPIOx, uint16_t VALUE)  
{  
    pGPIOx->ODR = VALUE;  
}  
  
/*  
 *  
 * @Function-to call toggle function  
 * @Param1-GPIO_REGDEF_t *  
 *  
 */  
  
void GPIO_TOGGLEOUTPUTPIN(GPIO_REGDEF_t *pGPIOx, uint8_t PINNO)  
{  
    pGPIOx->ODR^=(1<<PINNO);  
}
```

## 2.4 Github link to these files:

[embeddedC/driver\\_development 99003161 at master · 99003161/embeddedC \(github.com\)](https://github.com/99003161/embeddedC/tree/master/driver_development)

## 3. Activity 3 (Mini Project)

### 3.1 Main Logic

22 lines (19 sloc) | 559 Bytes

```
1  #include <stdio.h>
2  int main() {
3      int n, reversedN = 0, remainder, originalN;
4      printf("Enter an integer: ");
5      scanf("%d", &n);
6      originalN = n;
7
8      // reversed integer is stored in reversedN
9      while (n != 0) {
10         remainder = n % 10;
11         reversedN = reversedN * 10 + remainder;
12         n /= 10;
13     }
14
15     // palindrome if originalN and reversedN are equal
16     if (originalN == reversedN)
17         printf("%d is a palindrome.", originalN);
18     else
19         printf("%d is not a palindrome.", originalN);
20
21     return 0;
22 }
```

Figure 7. Main Logic function

### 3.2 Arduino Code

```
#include<SPI.h>

volatile boolean info;
volatile int Slave_data;

void setup()

{
    Serial.begin(9600);
```

```
pinMode(MISO, OUTPUT);

SPCR |= _BV(SPE);          //Turn on SPI in Slave Mode
info = false;

SPI.attachInterrupt();      //Interuupt ON is set for SPI commnucation
}

ISR (SPI_STC_vect)          //Inerrrrput routine function
{
  Slave_data = SPDR;        // Value received from master
  info = true;              //Sets received as True
}

void loop()
{ if(info)
  {
    delay(500);
    Serial.println(Slave_data);
    if(Slave_data==0)
    {
      Serial.println("Please maintain social distancing \n");
    }
    else if(Slave_data==1)
    {
      Serial.println("Way clear\n");
    }
    else
    {
      Serial.println("Sensor value is less than 500\n");
    }
  }
}
```

### 3.3 Github link to the project

[embeddedC/Embedded\\_C\\_Mini\\_Project\\_at\\_master · 99003161/embeddedC \(github.com\)](https://github.com/99003161/embeddedC)

## 4. References

- [1] [https://www.youtube.com/watch?v=5aafG5mjZ\\_Y&list=PLERTijJOmYrDiiWd10iRHY0VRHdJwUH4g&index=5](https://www.youtube.com/watch?v=5aafG5mjZ_Y&list=PLERTijJOmYrDiiWd10iRHY0VRHdJwUH4g&index=5)
- [2] <https://youtu.be/B7oKdUvRhQQ>
- [3] [https://youtu.be/5aafG5mjZ\\_Y](https://youtu.be/5aafG5mjZ_Y)
- [4] <https://youtu.be/Bsq6P1B8Jql>
- [5] <https://youtu.be/2Hm8eEHsgls>