**Assignment 3**

a. What is Morse Code? Answer: **Morse code** is a method used in telecommunication to encode text characters as standardized sequences of two different signal durations, called dots and dashes. The duration of a dash is three times the duration of a dot.

b. Who developed Morse Code? Answer: **The Morse code** was first developed by Samuel **Morse.**

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

/////////////////////////////////////////////////////////////////////////////

/////////////////////Home work for LED1 PA5 ///////////////////////////////

/////////////////////////////////////////////////////////////////////////////

#if 1

#define ODR5 (1<<5)

#define HELLO WORLD AND WELCOME TO EMBEDDED SYSTEM

#define RCC\_BASE 0x40021000

#define RCC\_AHB2ENR (\*((unsigned int\*)(RCC\_BASE + 0x4C)))

#define GPIOA\_BASE 0x48000000

#define GPIOA\_MODER (\*((unsigned int\*)(GPIOA\_BASE + 0x00)))

#define GPIOA\_ODR (\*((unsigned int\*)(GPIOA\_BASE + 0x14)))

void delayMs(unsigned int n);

int main() {

//RCC Base Address:0x40021000

//RCC AHB2 peripheral clock enable register (RCC\_AHB2ENR)

//Address offset

//1. Enable clock to peripheral

RCC\_AHB2ENR |=0x1; //Enable Clock for GPIOA -> BIT 0 -> 0x1

//GPIOA Base Address:0x4800 0000

//GPIO port mode register (GPIOx\_MODER) (x = A..E and H)

//Address offset: 0x00

//Set bit[11:10] to 0x01 so --> 0x20 enable PA5 as output

GPIOA\_MODER &=0xFFFFF7FF;

//GPIOA Base Address:0x48000000

//GPIO port output data register (GPIOx\_ODR) (x = A..E and H)

//Address offset: 0x14

//Set bit[5] to 1 so --> turn LED to ON

//Set bit[5] to 0 so --> turn LED to OFF

while(1) {

delayMs(100);//delay 1 second

GPIOA\_ODR ^= ODR5;// =0x0000;//GPIO\_ODR: Turn LED1 (PA5) OFF

}

return 0;

}

//Delays n milli seconds for (80 MHZ CPU clock

void delayMs(unsigned int n){

unsigned int i,j;

for(i = 0; i < n; i++){

for(j = 0; j < 3180; j++){

// do nothing

}

}

}

#endif

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

/////////////////////////////////////////////////////////////////////////////

////////////// The main.c file for the “Morse Code” problem///////////////

/////////////////////////////////////////////////////////////////////////////

/\*

//Morse code

//The following program uses LED2 to show morse code of first name

// Name - RETTA

R . - .

E .

T -

T -

A . -

one unit is defined 1 ms

The length of a dot is one unit --> 1 ms

The length of a dash is three unit --> 3 ms

Space between units of the same letter is one unit --> 1ms

Space between letters is three units --> 3ms

Space between words is 7 units --> 7 ms

\*/

#include <stdio.h>

#include <string.h>

void delayMs(unsigned int n); // Prototype for delay function

#if 1

#define R ".-."

#define E "."

#define T "-"

#define T "-"

#define A "-."

#define dotLen 100 //delay in milliseconds

#define dashLen 3\*dotLen // 3X longer than the dot

#define wordLen 7\*dotLen // space between words.

#define ODR14 (1<<14)

#define HELLO WORLD AND WELCOME TO EMBEDDED SYSTEM

#define RCC\_BASE 0x40021000

#define RCC\_AHB2ENR (\*((unsigned int\*)(RCC\_BASE + 0x4C)))

#define GPIOB\_BASE 0x48000400

#define GPIOB\_MODER (\*((unsigned int\*)(GPIOB\_BASE + 0x00)))

#define GPIOB\_ODR (\*((unsigned int\*)(GPIOB\_BASE + 0x14)))

char ch\_arr[5][6] = { R,E,T,T,A}; //First Name ---> RETTA

int counter;

int main() {

//RCC Base Address:0x40021000

//RCC AHB2 peripheral clock enable register (RCC\_AHB2ENR)

//Address offset

//1. Enable clock to peripheral

RCC\_AHB2ENR |=0x2; //Enable Clock for GPIOB -> BIT 1 -> 0x2

//GPIOB Base Address:0x48000400

//GPIO port mode register (GPIOx\_MODER) (x = A..E and H)

//Address offset: 0x00

//Set bit[29:28] to 0x01 so --> 0x4000 enable PB14 as output

GPIOB\_MODER &=0xDFFFFFFF;

//GPIOB Base Address:0x48000400

//GPIO port output data register (GPIOx\_ODR) (x = A..E and H)

//Address offset: 0x14

//Set bit[14] to 1 so --> turn LED to ON

//Set bit[14] to 0 so --> turn LED to OFF

while(1)

{

delayMs(wordLen);// Reset to the begining of the word -->First Name

int i = 0;

char c = 0;

while(i < 5){

int j = 0;

while(j < 5) {

c = ch\_arr[i][j];

if(c == '\0') {

break;

}

else {

if(c =='.') {

//delayMs(dotLen); //delay 1 second

GPIOB\_ODR |= ODR14;// =0x0000;//GPIO\_ODR: Turn LED2 (PB14) ON

delayMs(dotLen); //delay 1 second

GPIOB\_ODR &= ~ODR14;// =0x0000;//GPIO\_ODR: Turn LED2 (PB14) OFF

delayMs(dotLen);//between

}

else if(ch\_arr[i][j]=='-') {

//delayMs(dotLen); //delay 1 second

GPIOB\_ODR |= ODR14;// =0x0000;//GPIO\_ODR: Turn LED2 (PB14) ON

delayMs(dashLen); //delay 1 second

GPIOB\_ODR &= ~ODR14;// =0x0000;//GPIO\_ODR: Turn LED2 (PB14) OFF

delayMs(dotLen);

}

}

j++;

}

i++;

}

}

return 0;

}

#endif

//Delays n milli seconds for (80 MHZ CPU clock)

void delayMs(unsigned int n){

unsigned int i,j;

for(i = 0; i < n; i++){

for(j = 0; j < 3180; j++){

// do nothing

}

}

}

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

STM32MXCube tool. - Submit a link to your GitHub repo assignment:

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

LED1 and LED2 blinking using STM32MXCube tool code generation

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

/\* USER CODE BEGIN Header \*/

/\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* @file : main.c

\* @brief : Main program body

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* @attention

\*

\* <h2><center>&copy; Copyright (c) 2020 STMicroelectronics.

\* All rights reserved.</center></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 ---------------------------------------------------------\*/

/\* USER CODE BEGIN PV \*/

/\* 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 \*/

while (1)

{

/\* USER CODE END WHILE \*/

HAL\_GPIO\_TogglePin(GPIOA, GPIO\_PIN\_5);

HAL\_GPIO\_TogglePin(GPIOB, GPIO\_PIN\_14);

HAL\_Delay(1000);

/\* USER CODE BEGIN 3 \*/

}

/\* USER CODE END 3 \*/

}

/\*\*

\* @brief System Clock Configuration

\* @retval None

\*/

void SystemClock\_Config(void)

{

RCC\_OscInitTypeDef RCC\_OscInitStruct = {0};

RCC\_ClkInitTypeDef RCC\_ClkInitStruct = {0};

/\*\* Initializes the RCC Oscillators according to the specified parameters

\* in the RCC\_OscInitTypeDef structure.

\*/

RCC\_OscInitStruct.OscillatorType = RCC\_OSCILLATORTYPE\_MSI;

RCC\_OscInitStruct.MSIState = RCC\_MSI\_ON;

RCC\_OscInitStruct.MSICalibrationValue = 0;

RCC\_OscInitStruct.MSIClockRange = RCC\_MSIRANGE\_6;

RCC\_OscInitStruct.PLL.PLLState = RCC\_PLL\_ON;

RCC\_OscInitStruct.PLL.PLLSource = RCC\_PLLSOURCE\_MSI;

RCC\_OscInitStruct.PLL.PLLM = 1;

RCC\_OscInitStruct.PLL.PLLN = 40;

RCC\_OscInitStruct.PLL.PLLP = RCC\_PLLP\_DIV7;

RCC\_OscInitStruct.PLL.PLLQ = RCC\_PLLQ\_DIV2;

RCC\_OscInitStruct.PLL.PLLR = RCC\_PLLR\_DIV2;

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\_PLLCLK;

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\_4) != HAL\_OK)

{

Error\_Handler();

}

/\*\* Configure the main internal regulator output voltage

\*/

if (HAL\_PWREx\_ControlVoltageScaling(PWR\_REGULATOR\_VOLTAGE\_SCALE1) != HAL\_OK)

{

Error\_Handler();

}

}

/\*\*

\* @brief GPIO Initialization Function

\* @param None

\* @retval None

\*/

static void MX\_GPIO\_Init(void)

{

GPIO\_InitTypeDef GPIO\_InitStruct = {0};

/\* GPIO Ports Clock Enable \*/

\_\_HAL\_RCC\_GPIOA\_CLK\_ENABLE();

\_\_HAL\_RCC\_GPIOB\_CLK\_ENABLE();

/\*Configure GPIO pin Output Level \*/

HAL\_GPIO\_WritePin(LED1\_GPIO\_Port, LED1\_Pin, GPIO\_PIN\_RESET);

/\*Configure GPIO pin Output Level \*/

HAL\_GPIO\_WritePin(LED2\_GPIO\_Port, LED2\_Pin, GPIO\_PIN\_RESET);

/\*Configure GPIO pin : LED1\_Pin \*/

GPIO\_InitStruct.Pin = LED1\_Pin;

GPIO\_InitStruct.Mode = GPIO\_MODE\_OUTPUT\_PP;

GPIO\_InitStruct.Pull = GPIO\_NOPULL;

GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_LOW;

HAL\_GPIO\_Init(LED1\_GPIO\_Port, &GPIO\_InitStruct);

/\*Configure GPIO pin : LED2\_Pin \*/

GPIO\_InitStruct.Pin = LED2\_Pin;

GPIO\_InitStruct.Mode = GPIO\_MODE\_OUTPUT\_PP;

GPIO\_InitStruct.Pull = GPIO\_NOPULL;

GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_LOW;

HAL\_GPIO\_Init(LED2\_GPIO\_Port, &GPIO\_InitStruct);

}

/\* USER CODE BEGIN 4 \*/

/\* USER CODE END 4 \*/

/\*\*

\* @brief This function is executed in case of error occurrence.

\* @retval None

\*/

void Error\_Handler(void)

{

/\* USER CODE BEGIN Error\_Handler\_Debug \*/

/\* User can add his own implementation to report the HAL error return state \*/

/\* USER CODE END Error\_Handler\_Debug \*/

}

#ifdef USE\_FULL\_ASSERT

/\*\*

\* @brief Reports the name of the source file and the source line number

\* where the assert\_param error has occurred.

\* @param file: pointer to the source file name

\* @param line: assert\_param error line source number

\* @retval None

\*/

void assert\_failed(uint8\_t \*file, uint32\_t line)

{

/\* USER CODE BEGIN 6 \*/

/\* User can add his own implementation to report the file name and line number,

tex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) \*/

/\* USER CODE END 6 \*/

}

#endif /\* USE\_FULL\_ASSERT \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* (C) COPYRIGHT STMicroelectronics \*\*\*\*\*END OF FILE\*\*\*\*/

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

o Ex: “https://github.com//embsys310/assignment03”