УО «Белорусский государственный университет информатики и радиоэлектроники»

Кафедра программного обеспечения информационных технологий

Отчет по лабораторной работе №3

по дисциплине

«Программное обеспечение встроенных систем»

Выполнил: Студент гр. 051003

Кукреш А.С.

Проверил: Деменковец Д.В.

Минск 2023

**Настройки STM32CubeMX**:

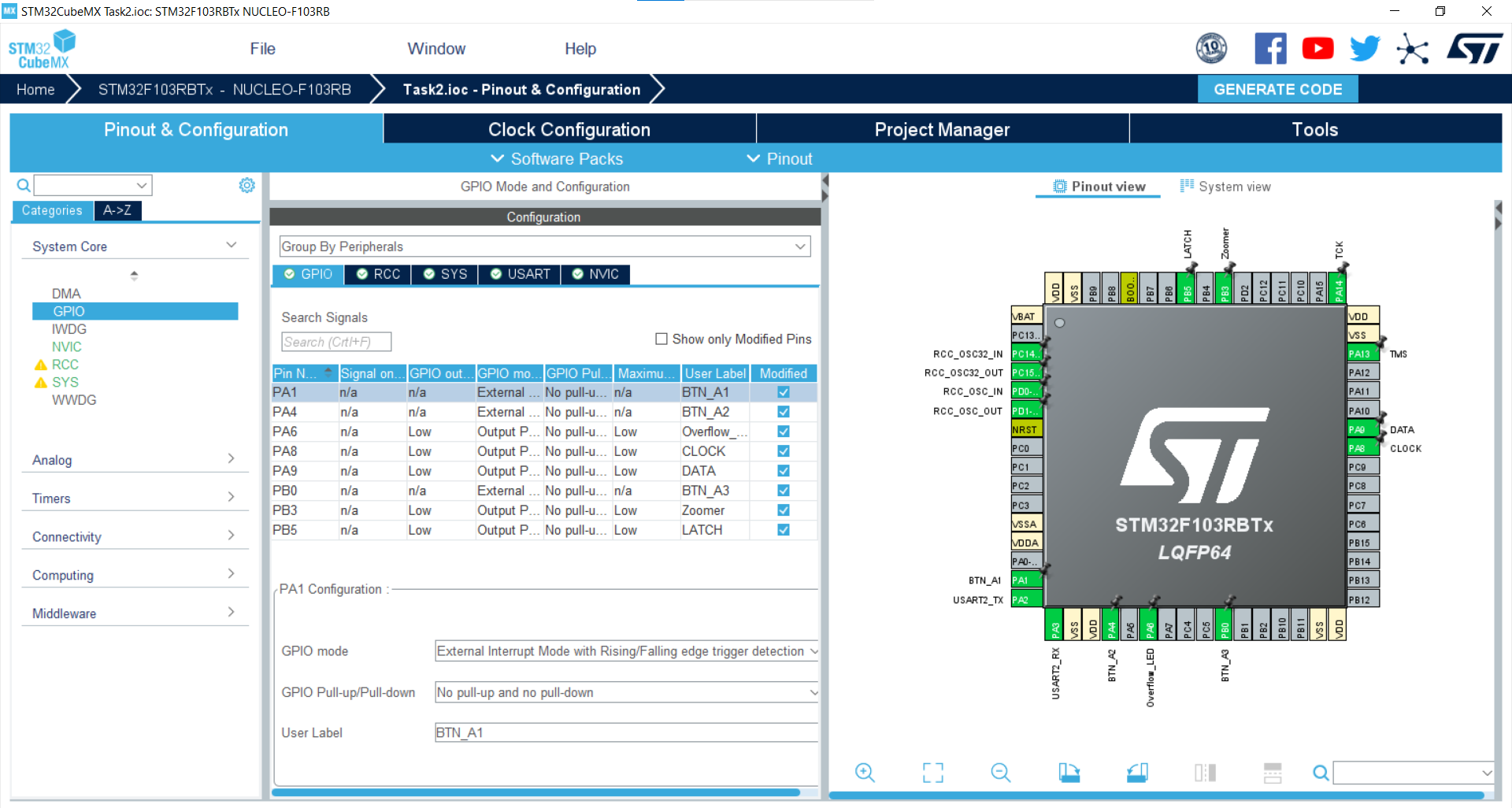


Рисунок – Настройки в CubeMX

***Код main.c****:*

/\* USER CODE BEGIN Header \*/

/\*\*

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

\* @file : main.c

\* @brief : Main program body

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

\* @attention

\*

\* <h2><center>&copy; Copyright (c) 2023 STMicroelectronics.

\* All rights reserved.</center></h2>

\*/

/\* USER CODE END Header \*/

/\* Includes ------------------------------------------------------------------\*/

#include "main.h"

#include "tim.h"

#include "usart.h"

#include "gpio.h"

/\* Private includes ----------------------------------------------------------\*/

/\* USER CODE BEGIN Includes \*/

#include <stdio.h>

#include <string.h>

/\* 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 \*/

/\*

a

-----

f| |b

| g |

-----

e| |c

| |

----- .h

d

\*/

// a b c d e f g h invert

// 0 1 1 1 1 1 1 0 0 0000 0011

// 1 0 1 1 0 0 0 0 0 1001 1111

// 2 1 1 0 1 1 0 1 0 0010 0101

// 3 1 1 1 1 0 0 1 0 0000 1101

// 4 0 1 1 0 0 1 1 0 1001 1001

// 5 1 0 1 1 0 1 1 0 0100 1001

// 6 1 0 1 1 1 1 1 0 0100 0001

// 7 1 1 1 0 0 0 0 0 0001 1111

// 8 1 1 1 1 1 1 1 0 0000 0001

// 9 1 1 1 1 0 1 1 0 0000 1001

// 0 1 2 3 4 5 6 7 8 9

const uint8\_t DIGITS[] = {0x03, 0x9F, 0x25, 0x0D, 0x99, 0x49, 0x41, 0x1F, 0x01, 0x09};

// 0 1 2 3

const uint8\_t SEGMENTS[] = {0x10, 0x20, 0x40, 0x80};

/\* USER CODE END PV \*/

/\* Private function prototypes -----------------------------------------------\*/

void SystemClock\_Config(void);

/\* USER CODE BEGIN PFP \*/

void ZoomerInc(void);

void ZoomerDec(void);

void ZoomerReset(void);

void ZoomerPlaySound(int);

void BeginDisplayWrite(void);

void EndDisplayWrite(void);

void WriteDataToRegister(uint8\_t dataBits);

void WriteDataToSegment(uint8\_t, uint8\_t);

void WriteNumberToDisplay(uint32\_t);

void setPWM(uint16\_t pwmValue);

/\* USER CODE END PFP \*/

/\* Private user code ---------------------------------------------------------\*/

/\* USER CODE BEGIN 0 \*/

volatile int zoomerValue = 0;

const int zoomerDefaultValue = 0;

const int BUTTON\_LONG\_CLICK\_MILLISEC = 3000;

const int LONG\_CLICK\_UPDATE\_MILLISEC = 1000;

volatile int Btn1Click = 0;

volatile int Btn1ClickTime = 0;

volatile int Btn1LongClick = 0;

volatile int Btn2Click = 0;

volatile int Btn2ClickTime = 0;

volatile int Overflow = 0;

volatile int Btn3Click = 0;

volatile int Btn3ClickTime = 0;

volatile int Btn3LongClick = 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();

MX\_TIM3\_Init();

MX\_TIM2\_Init();

MX\_USART2\_UART\_Init();

/\* USER CODE BEGIN 2 \*/

int currentTime = 0;

/\* USER CODE END 2 \*/

/\* Infinite loop \*/

/\* USER CODE BEGIN WHILE \*/

int const BUFFER\_LEN = 256;

char buffer[BUFFER\_LEN];

while (1)

{

/\* USER CODE END WHILE \*/

/\* USER CODE BEGIN 3 \*/

currentTime = HAL\_GetTick();

if (Btn1Click && currentTime - Btn1ClickTime > BUTTON\_LONG\_CLICK\_MILLISEC)

{

Btn1LongClick = 1;

Btn1ClickTime += BUTTON\_LONG\_CLICK\_MILLISEC;

ZoomerDec();

}

if (Btn1LongClick && currentTime - Btn1ClickTime > LONG\_CLICK\_UPDATE\_MILLISEC)

{

Btn1ClickTime += LONG\_CLICK\_UPDATE\_MILLISEC;

ZoomerDec();

}

if (Btn3Click && currentTime - Btn3ClickTime > BUTTON\_LONG\_CLICK\_MILLISEC)

{

Btn3LongClick = 1;

Btn3ClickTime += BUTTON\_LONG\_CLICK\_MILLISEC;

ZoomerInc();

}

if (Btn3LongClick && currentTime - Btn3ClickTime > LONG\_CLICK\_UPDATE\_MILLISEC)

{

Btn3ClickTime += LONG\_CLICK\_UPDATE\_MILLISEC;

ZoomerInc();

}

if (Overflow)

{

HAL\_GPIO\_WritePin(GPIOA, Overflow\_LED\_Pin, GPIO\_PIN\_RESET);

}

else

{

HAL\_GPIO\_WritePin(GPIOA, Overflow\_LED\_Pin, GPIO\_PIN\_SET);

}

snprintf(buffer, BUFFER\_LEN, "X=%04d\n", (int)zoomerValue);

HAL\_UART\_Transmit(&huart2, (uint8\_t \*)buffer, strlen(buffer), HAL\_MAX\_DELAY);

ZoomerPlaySound(zoomerValue);

}

/\* 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\_HSI;

RCC\_OscInitStruct.HSIState = RCC\_HSI\_ON;

RCC\_OscInitStruct.HSICalibrationValue = RCC\_HSICALIBRATION\_DEFAULT;

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

RCC\_OscInitStruct.PLL.PLLSource = RCC\_PLLSOURCE\_HSI\_DIV2;

RCC\_OscInitStruct.PLL.PLLMUL = RCC\_PLL\_MUL2;

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\_DIV2;

RCC\_ClkInitStruct.APB2CLKDivider = RCC\_HCLK\_DIV1;

if (HAL\_RCC\_ClockConfig(&RCC\_ClkInitStruct, FLASH\_LATENCY\_0) != HAL\_OK)

{

Error\_Handler();

}

}

/\* USER CODE BEGIN 4 \*/

void BeginDisplayWrite()

{

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

}

void EndDisplayWrite()

{

HAL\_GPIO\_WritePin(LATCH\_GPIO\_Port, LATCH\_Pin, GPIO\_PIN\_SET);

}

void WriteDataToRegister(uint8\_t dataBits)

{

for (int i = 0; i < 8; i++)

{

HAL\_GPIO\_WritePin(DATA\_GPIO\_Port, DATA\_Pin, dataBits % 2);

dataBits >>= 1;

HAL\_GPIO\_WritePin(CLOCK\_GPIO\_Port, CLOCK\_Pin, GPIO\_PIN\_SET);

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

}

}

void WriteDataToSegment(uint8\_t dataBits, uint8\_t segmentBits)

{

BeginDisplayWrite();

WriteDataToRegister(dataBits);

WriteDataToRegister(segmentBits);

EndDisplayWrite();

}

void WriteNumberToDisplay(uint32\_t number)

{

for (int i = 0; i < 4; i++)

{

WriteDataToSegment(DIGITS[number % 10], SEGMENTS[i]);

number /= 10;

}

}

const int BUTTON\_STEP = 1;

const int BUTTON\_DEBOUNCE\_MILLISEC = 150;

void HAL\_GPIO\_EXTI\_Callback(uint16\_t GPIO\_Pin)

{

int currentTime = HAL\_GetTick();

switch (GPIO\_Pin)

{

case BTN\_A1\_Pin:

if (HAL\_GPIO\_ReadPin(GPIOA, BTN\_A1\_Pin) == 0)

{

if (currentTime - Btn1ClickTime > BUTTON\_DEBOUNCE\_MILLISEC)

{

Btn1ClickTime = currentTime;

Btn1Click = 1;

Btn1LongClick = 0;

ZoomerDec();

}

}

else

{

Btn1Click = 0;

Btn1LongClick = 0;

}

break;

case BTN\_A2\_Pin:

if (HAL\_GPIO\_ReadPin(GPIOA, BTN\_A2\_Pin) == 0)

{

if (currentTime - Btn2ClickTime > BUTTON\_DEBOUNCE\_MILLISEC)

{

ZoomerReset();

Overflow = 1;

}

}

else

{

Btn2ClickTime = currentTime;

Btn2Click = 1;

Btn2Click = 0;

Overflow = 0;

}

break;

case BTN\_A3\_Pin:

if (HAL\_GPIO\_ReadPin(GPIOB, BTN\_A3\_Pin) == 0)

{

if (currentTime - Btn3ClickTime > BUTTON\_DEBOUNCE\_MILLISEC)

{

Btn3ClickTime = currentTime;

Btn3Click = 1;

Btn3LongClick = 0;

ZoomerInc();

}

}

else

{

Btn3Click = 0;

Btn3LongClick = 0;

}

break;

}

}

void ZoomerInc()

{

Overflow = 0;

zoomerValue += BUTTON\_STEP;

if (zoomerValue > 254)

{

ZoomerReset();

Overflow = 1;

}

}

void ZoomerDec()

{

Overflow = 0;

zoomerValue -= BUTTON\_STEP;

if (zoomerValue < 0) zoomerValue = 0;

}

void ZoomerReset()

{

Overflow = 0;

zoomerValue = zoomerDefaultValue;

}

void ZoomerPlaySound(int zoomerValue)

{

HAL\_GPIO\_TogglePin(GPIOB, Zoomer\_Pin);

WriteNumberToDisplay(zoomerValue);

HAL\_Delay(zoomerValue);

WriteNumberToDisplay(zoomerValue);

HAL\_GPIO\_TogglePin(GPIOB, Zoomer\_Pin);

WriteNumberToDisplay(zoomerValue);

HAL\_Delay(2);

WriteNumberToDisplay(zoomerValue);

}

/\* 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 \*/

\_\_disable\_irq();

while (1)

{

}

/\* 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,

ex: 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\*\*\*\*/