

LAB NOTE

Subject: Hardware/Software Interfacing

Lab 2: GPIO

Student: Minh Quan Tran

Sep 19th, 2024

Table of Contents

- 1. Objectives 4
- 2. Problems and Solutions..... 5
 - 2.1 Problems 5
 - 2.2 Solutions 5
- 3. Software Design 6
 - 3.1 List of function 6
 - 3.2 While loop 7
- 4. Result 7

TABLE OF FIGURES

Figure 4-1: Lab2's result.....	7
--------------------------------	---

1. Objectives

1. Objectives

- Code's requirement:
 - Uses 3 GPIO for inputs and makes use of Interrupt Handler to detect state changes
 - Uses 3 GPIO for outputs and displays the state in an interesting way (not just on/off or 1/0)
 - Obtains input from the user to change the state of the GPIO outputs but does not wait forever for user input
 - Displays the current state of the GPIO pins when not interrupted to change GPIO outputs

2. Problems and Solutions

2. Problems and Solutions

2.1 Problems

- No problem.

2.2 Solutions

- No problem.

3. Software Design

3. Software Design

3.1 List of function

This call back is to start the timer for the debounce time for all GPIO Input

```
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
    btnPressed = GPIO_Pin;

    // Start a timer to handle debounce
    HAL_TIM_Base_Start_IT(&htim2);
}
```

When Timer callback if button still pressed then it will change the btnState or else it will be counted as bounce and not updated the btnState.

```
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{
    if (htim->Instance == TIM2)
    {
        // Stop the timer
        HAL_TIM_Base_Stop_IT(&htim2);

        // Check the button state after 20ms
        if ((btnPressed == LEFT_PIN) && (HAL_GPIO_ReadPin(GPIOC, LEFT_PIN) ==
GPIO_PIN_RESET))
        {
            btnState = LEFT;
        }
        else if ((btnPressed == RIGHT_PIN) && (HAL_GPIO_ReadPin(GPIOC, RIGHT_PIN) ==
GPIO_PIN_RESET))
        {
            btnState = RIGHT;
        }
        else if ((btnPressed == TOGGLE_PIN) && (HAL_GPIO_ReadPin(GPIOC, TOGGLE_PIN) ==
GPIO_PIN_RESET))
        {
            btnState = TOGGLE;
        }
    }
}
```

setLed is used to set Led base on the parameter and used that function to create a sequence of how the LED will blink like shift left LED or shift right LED.

```
void setLed(bool led1Status, bool led2Status, bool led3Status)
{
    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, led1Status);
    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, led2Status);
    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, led3Status);
}

void shiftLeftLed(void)
{
    setLed(0,0,1);
    HAL_Delay(100);
    setLed(0,1,0);
}
```

3. Result

```
    HAL_Delay(100);
    setLed(1,0,0);
    HAL_Delay(100);
    setLed(0,0,0);
    HAL_Delay(100);
}
void shiftRightLed(void)
{
    setLed(1,0,0);
    HAL_Delay(100);
    setLed(0,1,0);
    HAL_Delay(100);
    setLed(0,0,1);
    HAL_Delay(100);
    setLed(0,0,0);
    HAL_Delay(100);
}
```

3.2 While loop

```
while (1)
{
    switch(controlState)
    {
        case BTN:
            switch(btnState)
            {
                case LEFT:
                    shiftLeftLed();
                    break;
                case RIGHT:
                    shiftRightLed();
                    break;
                case TOGGLE:
                    controlState = PROMPT;
                    btnState = NONE;
                    break;
                default:
                    break;
            }
            break;
        case PROMPT:
            // Toggle a button again to enter how the LED lit
            if (btnState == TOGGLE)
            {
                printf("Insert L/R to control LED\r\n");
                switch(getCharFromUart2())
                {
                    case 'L':
                        rcvChar = 'L';
                        printf("Shift Left\r\n");
                        break;
                    case 'R':
                        rcvChar = 'R';
                        printf("Shift Right\r\n");
```

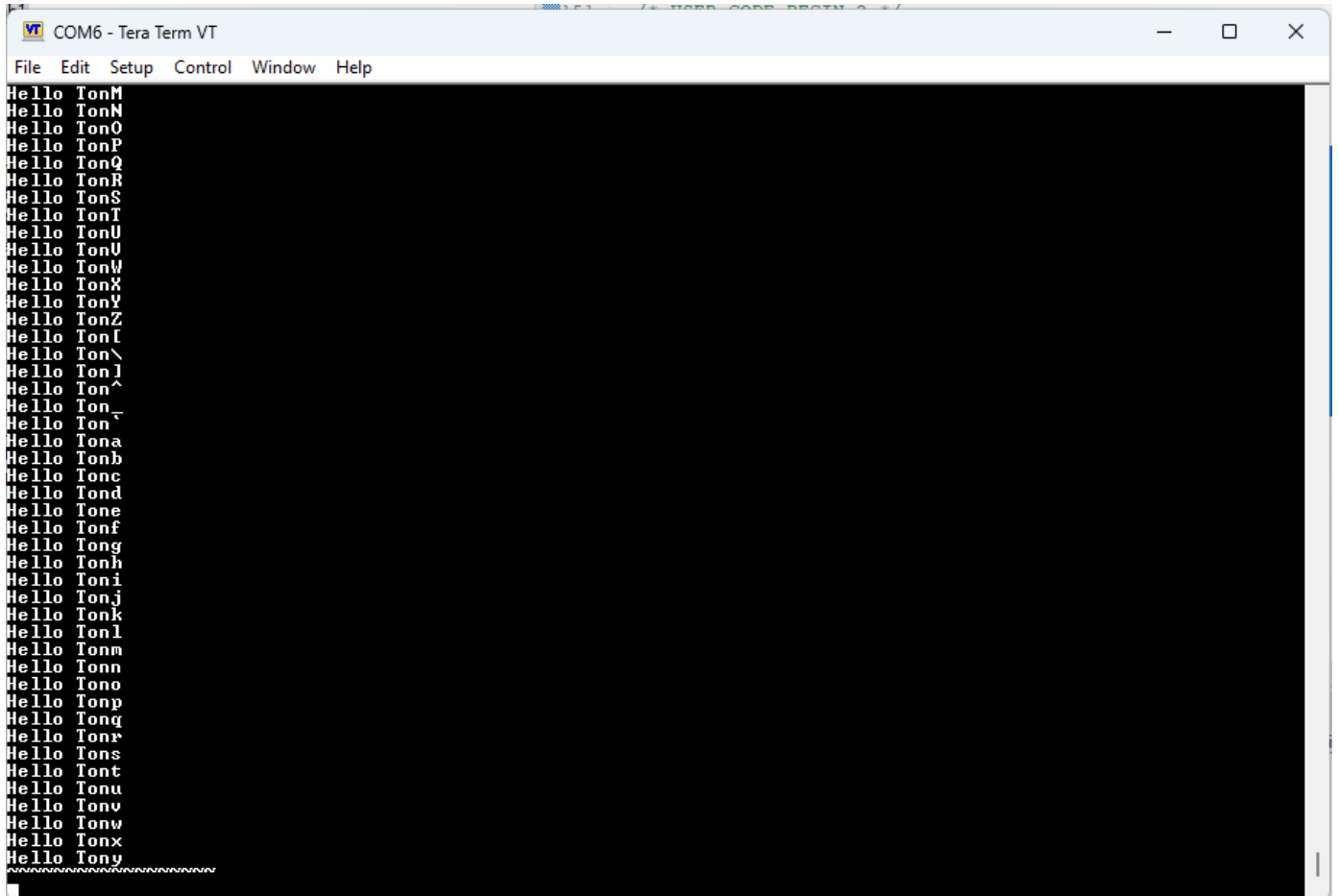
3. Result

```
                break;
            case UART_TIMEOUT_ERROR:
                printf("Prompt Timeout\r\n");
                printf("Switching to BTN control state\r\n");
                controlState = BTN;
                rcvChar = ' ';
            default:
                break;
        }
        btnState = NONE;
    }

    switch(rcvChar)
    {
        case 'L':
            shiftLeftLed();
            break;
        case 'R':
            shiftRightLed();
            break;
        default:
            break;
    }
}
```


4. Result

4. Result



The screenshot shows a Tera Term VT window titled "COM6 - Tera Term VT". The window has a menu bar with "File", "Edit", "Setup", "Control", "Window", and "Help". The main display area is black with white text. The text consists of a list of "Hello" messages, each followed by a character name. The characters listed are: TonM, TonN, TonO, TonP, TonQ, TonR, TonS, TonT, TonU, TonV, TonW, TonX, TonY, TonZ, TonI, Ton\, Tonl, Ton^, Ton~, Tona, Tonb, Tonc, Tond, Tone, Tonf, Tong, Tonh, Toni, Tonj, Tonk, Tonl, Tonm, Tonn, Tono, Tonp, Tonq, Tonr, Tons, Tent, Tonu, Tonv, Tonw, Tonx, and Tony. The list ends with a line of asterisks.

```
COM6 - Tera Term VT
File Edit Setup Control Window Help
Hello TonM
Hello TonN
Hello TonO
Hello TonP
Hello TonQ
Hello TonR
Hello TonS
Hello TonT
Hello TonU
Hello TonV
Hello TonW
Hello TonX
Hello TonY
Hello TonZ
Hello TonI
Hello Ton\
Hello Tonl
Hello Ton^
Hello Ton~
Hello Tona
Hello Tonb
Hello Tonc
Hello Tond
Hello Tone
Hello Tonf
Hello Tong
Hello Tonh
Hello Toni
Hello Tonj
Hello Tonk
Hello Tonl
Hello Tonm
Hello Tonn
Hello Tono
Hello Tonp
Hello Tonq
Hello Tonr
Hello Tons
Hello Tent
Hello Tonu
Hello Tonv
Hello Tonw
Hello Tonx
Hello Tony
*****
```

Figure 4-1: Lab2's result

REFERENCES