ASSIGNMENT NOTE

Subject: Embedded Programming Principle

Topic: LED Flashing

Student: Minh Quan Tran

Jun 14th, 2024

Table of Contents

TABLE OF FIGURES

1. Objectives

- Build a circuit that blink LED.
- Write a C program in STM32 CUBE IDE that blink LED from initials.
- Flash code and debug project.

2. Requirement

2.1 Requirement

Wire and code to blink led based on binary equivalent of your initials.

- Example for “AK”
  A’s binary equivalent is 01000001
  B’s binary equivalent is 01001011
- Turn on led for every 1 in your initials.
- Turn off your led for every 0 in your initials.
- Delay required for every single 1 or 0 is 300 milliseconds. For consecutive 1/0s add up the delay.
- Delay between two letters must be 1 second.
- Delay between two initials must be 3 seconds.

Possible above and Beyond Features:

- Add anything that increases the complexity of the task or adds to the task in an interesting novel way.
- Find a way to make your code work for any set of initials.

Be sure to follow ESD Programming Standards.

3. Hardware and Software design

3.1 Hardware Design

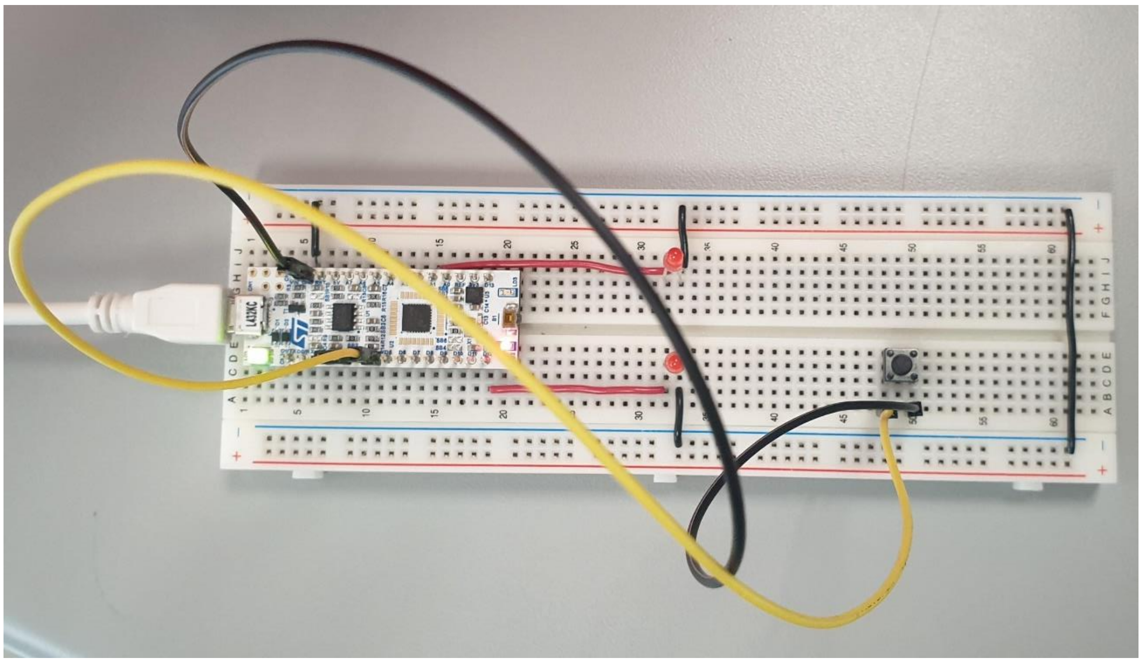
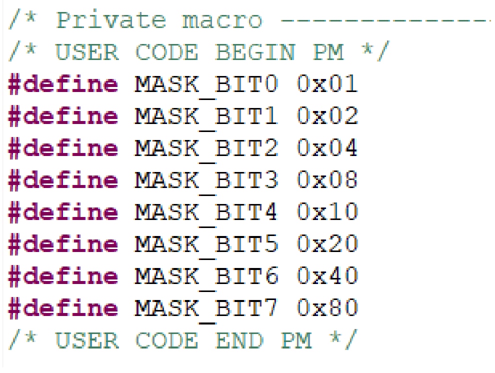


Figure 3-1: Breadboard's circuit

3.2 Software Design

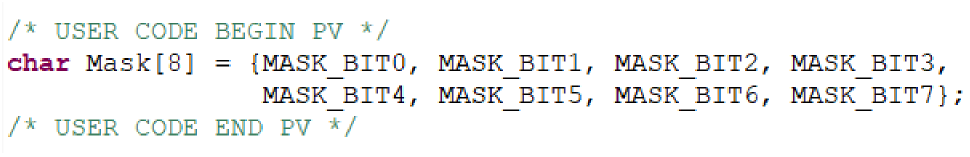
3.2.1 Assignment design

Create an array Mask[8] that had Mask from the LSB to MSB



```
/* Private macro -------------
/* USER CODE BEGIN PM */
#define MASK_BIT0 0x01
#define MASK_BIT1 0x02
#define MASK_BIT2 0x04
#define MASK_BIT3 0x08
#define MASK_BIT4 0x10
#define MASK_BIT5 0x20
#define MASK_BIT6 0x40
#define MASK_BIT7 0x80
/* USER CODE END PM */
```

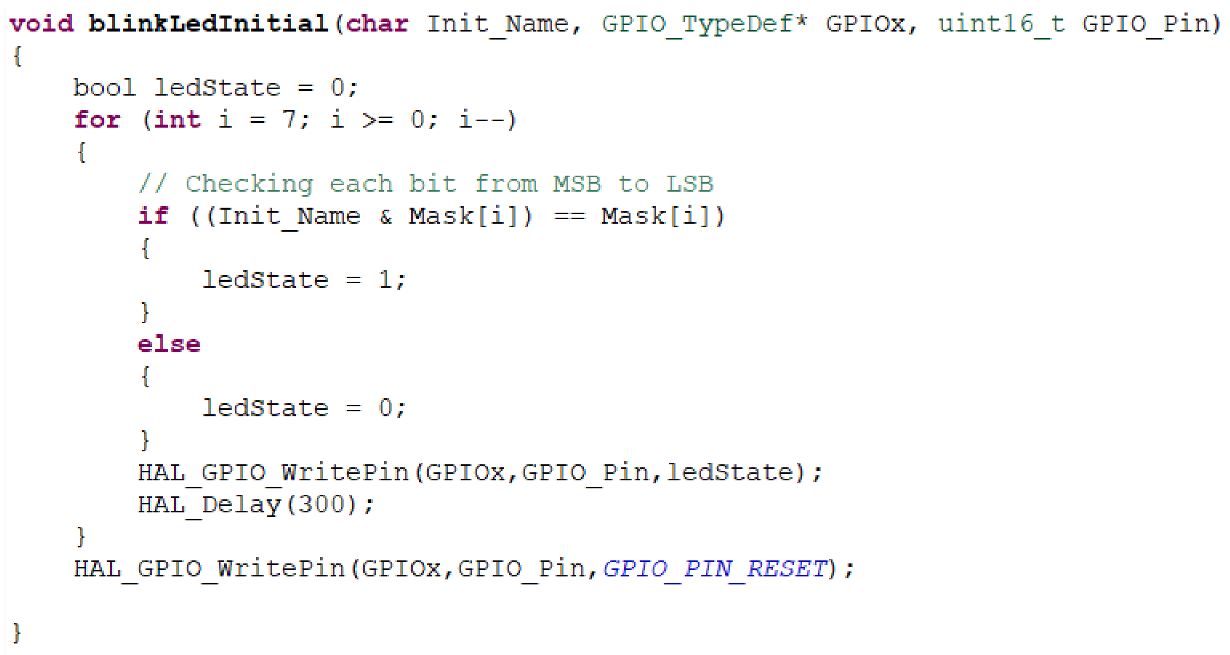
Figure 3-2: Define Mask



```
/* USER CODE BEGIN PV */
char Mask[8] = {MASK_BIT0, MASK_BIT1, MASK_BIT2, MASK_BIT3,
                MASK_BIT4, MASK_BIT5, MASK_BIT6, MASK_BIT7};
/* USER CODE END PV */
```

Figure 3-3: Creating Array from MASK

Then use an AND operation for the initials and each MASK to get the respected value from each bit and use those bit to define will the LED be ON or OFF.



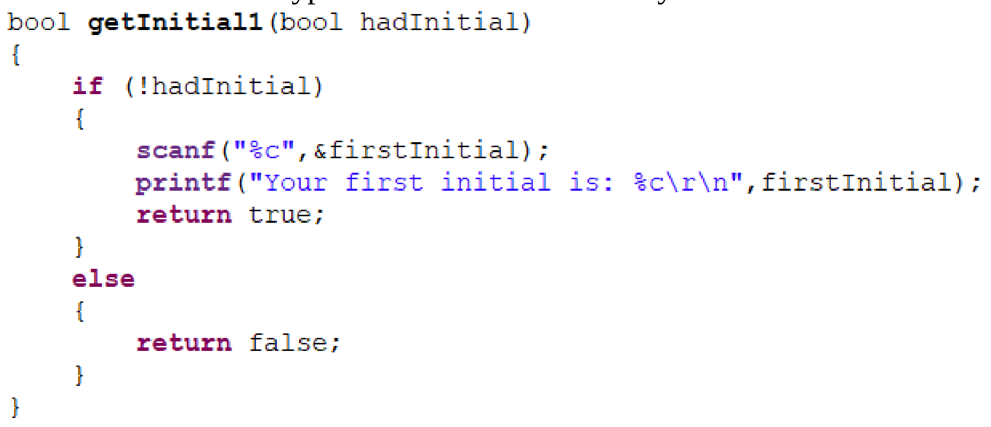
```
void blinkLedInitial(char Init_Name, GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin)
{
    bool ledState = 0;
    for (int i = 7; i >= 0; i--)
    {
        // Checking each bit from MSB to LSB
        if ((Init_Name & Mask[i]) == Mask[i])
        {
            ledState = 1;
        }
        else
        {
            ledState = 0;
        }
        HAL_GPIO_WritePin(GPIOx,GPIO_Pin,ledState);
        HAL_Delay(300);
    }
    HAL_GPIO_WritePin(GPIOx,GPIO_Pin,GPIO_PIN_RESET);

}
```

Figure 3-4: blinkLedInitial funciton block

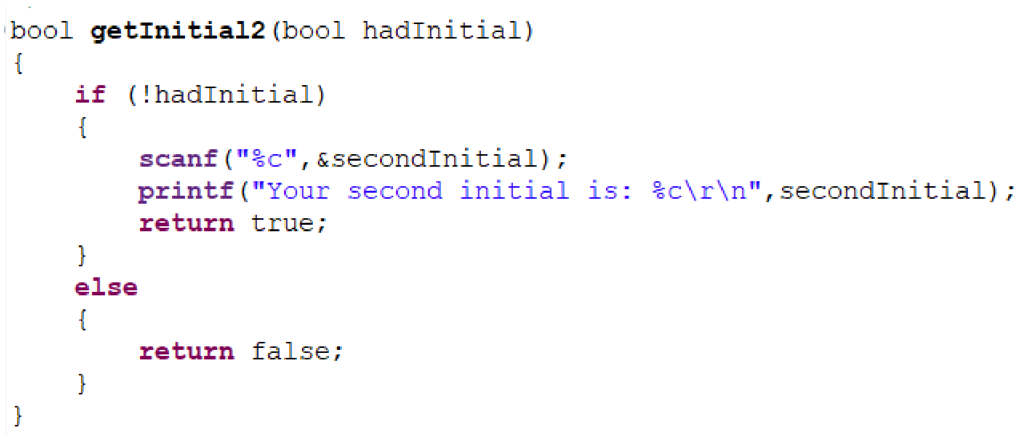
3.2.2 Above and beyond

Use a button to reset the initials and type in new initials from Putty



```
bool getInitial1(bool hadInitial)
{
    if (!hadInitial)
    {
        scanf("%c",&firstInitial);
        printf("Your first initial is: %c\r\n",firstInitial);
        return true;
    }
    else
    {
        return false;
    }
}
```

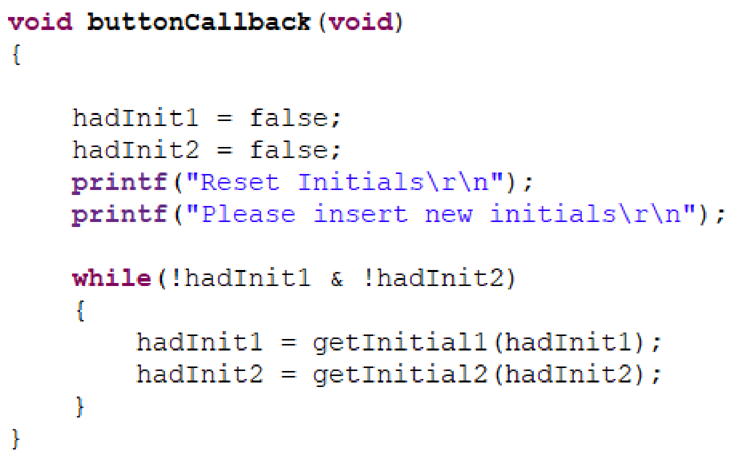
Figure 3-5: Get Initial 1 funciton block



```
bool getInitial2(bool hadInitial)
{
    if (!hadInitial)
    {
        scanf("%c",&secondInitial);
        printf("Your second initial is: %c\r\n",secondInitial);
        return true;
    }
    else
    {
        return false;
    }
}
```

Figure 3-6:Get Initial 2 function block

Also use external interrupt for the button so that the button won’t interfere with the main code.



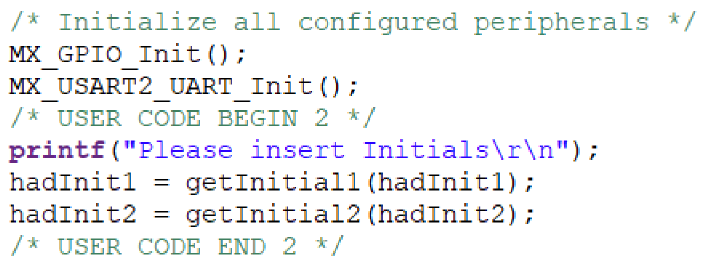
```
void buttonCallback(void)
{

    hadInit1 = false;
    hadInit2 = false;
    printf("Reset Initials\r\n");
    printf("Please insert new initials\r\n");

    while(!hadInit1 & !hadInit2)
    {
        hadInit1 = getInitial1(hadInit1);
        hadInit2 = getInitial2(hadInit2);
    }
}
```

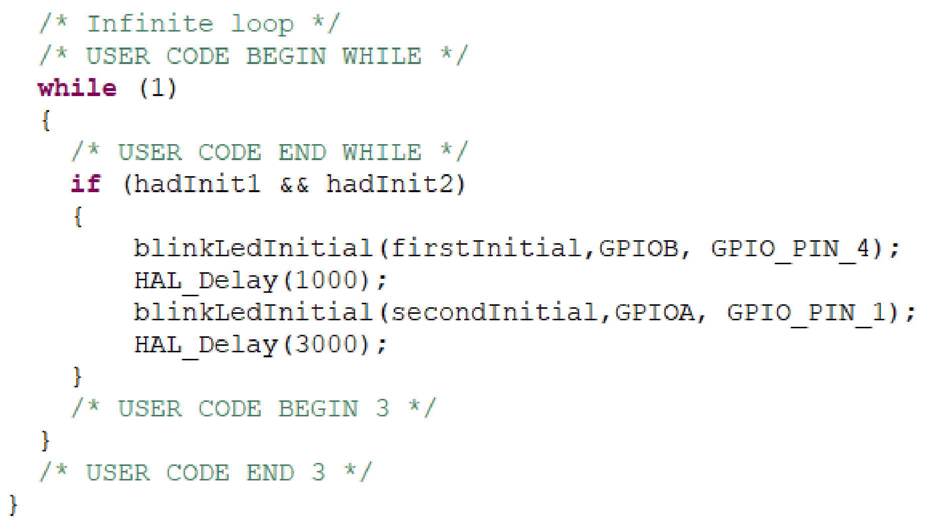
Figure 3-7: Button Callback function

Main code:



```
/* Initialize all configured peripherals */
MX_GPIO_Init();
MX_USART2_UART_Init();
/* USER CODE BEGIN 2 */
printf("Please insert Initials\r\n");
hadInit1 = getInitial1(hadInit1);
hadInit2 = getInitial2(hadInit2);
/* USER CODE END 2 */
```

Figure 3-8: Initialize both initials

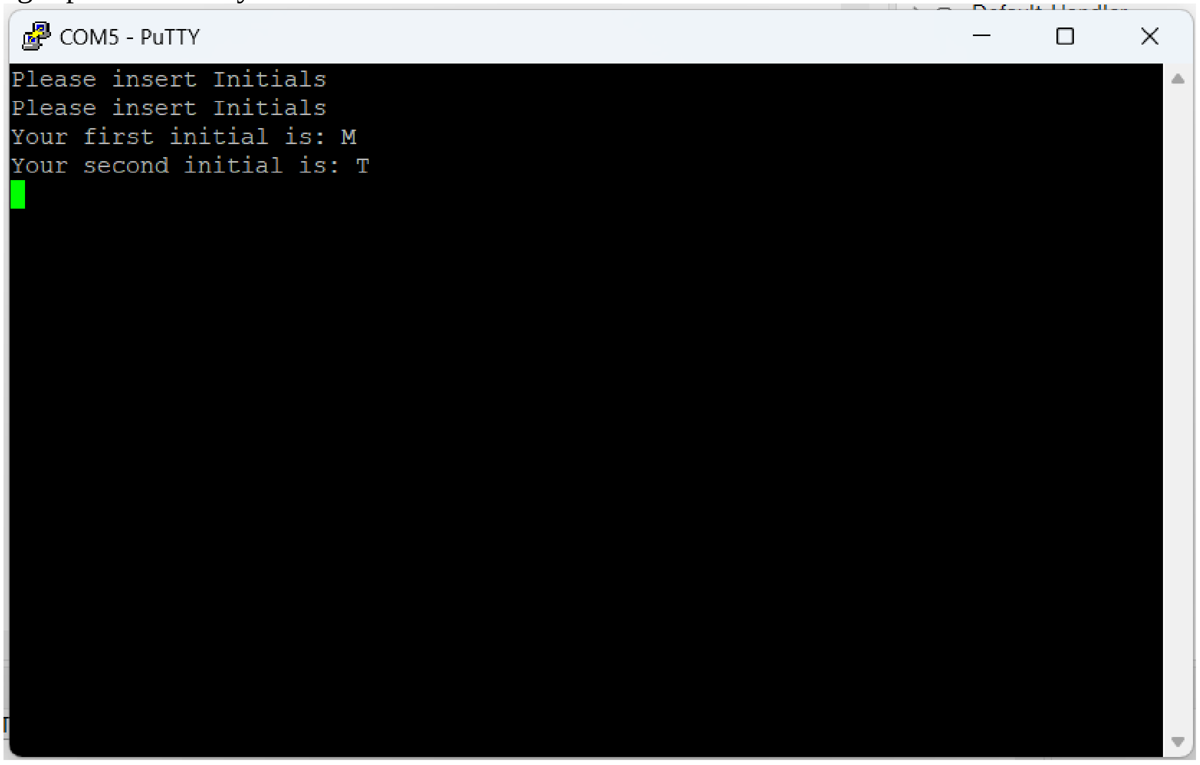


```
  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */
    if (hadInit1 && hadInit2)
    {
        blinkLedInitial(firstInitial,GPIOB, GPIO_PIN_4);
        HAL_Delay(1000);
        blinkLedInitial(secondInitial,GPIOA, GPIO_PIN_1);
        HAL_Delay(3000);
    }
    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}
```

Figure 3-9: Loop function

4. Result

Choosing input from Putty:



```
Please insert Initials
Please insert Initials
Your first initial is: M
Your second initial is: T
```

Figure 4-1: Typing initials in Putty

With input is ‘M’ and ‘T’, result should be:

- ‘M’: 0 1 0 0 1 1 0 1

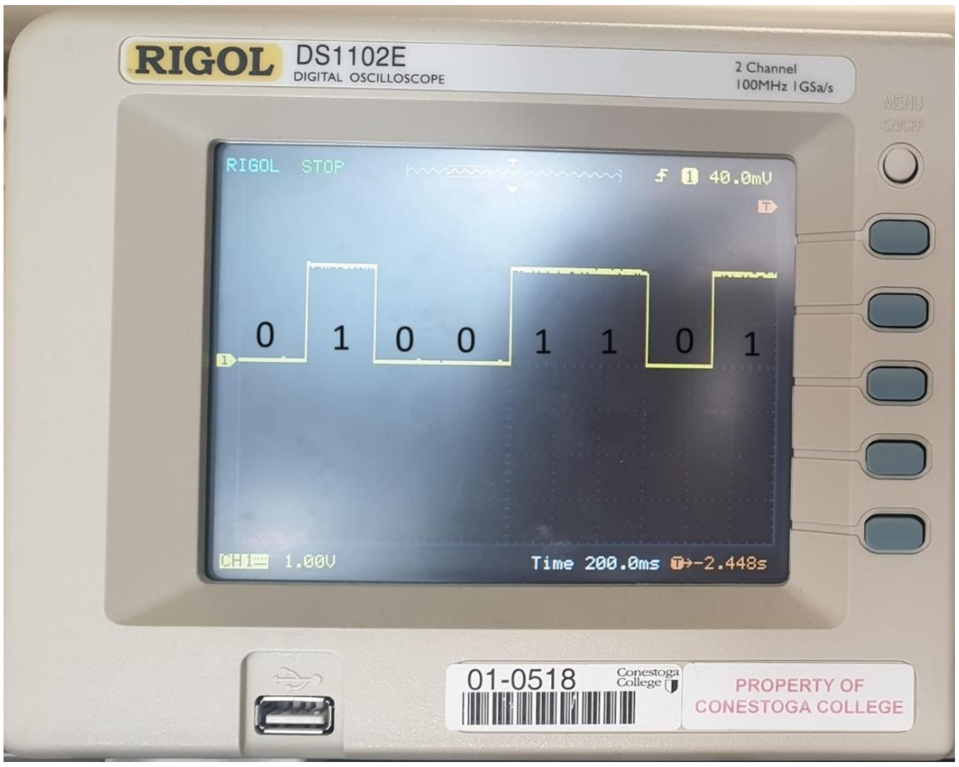


Figure 4-2: Initials 1 = ‘M’.

4. Result

- ‘T’: 0 1 0 1 0 1 0 0

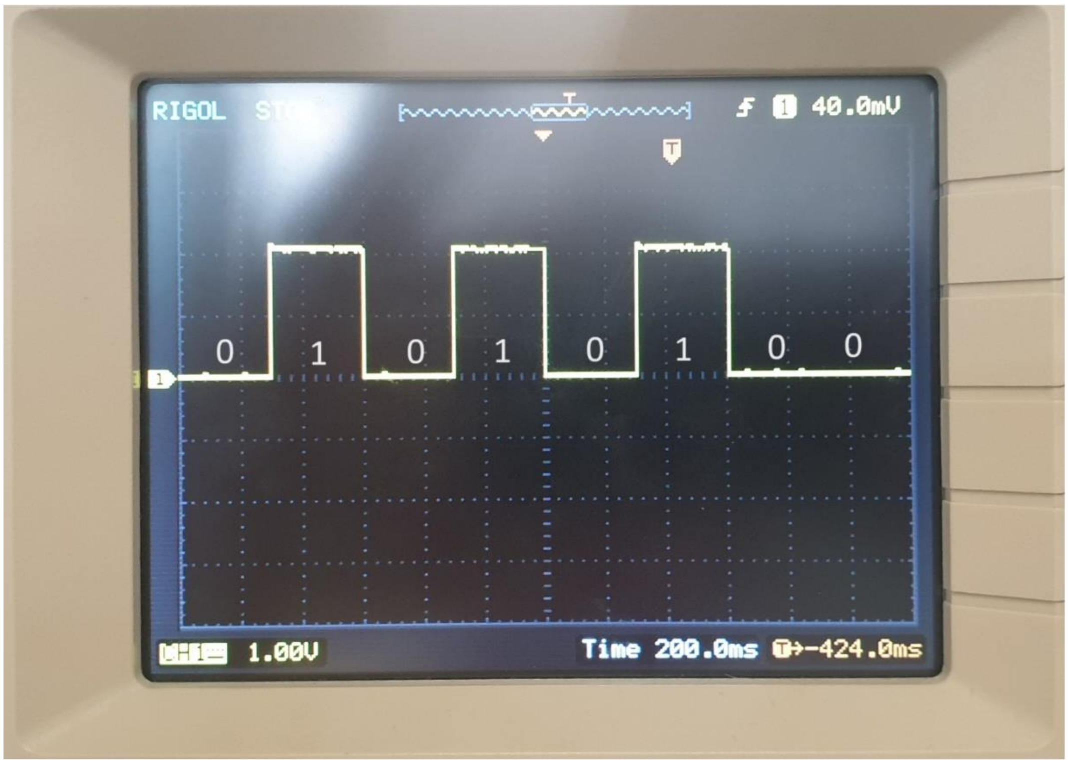


Figure 4-3: Initials 2 = ‘T’

REFERENCES