

# 임베디드시스템 설계 및 실험 보고서

[002 분반 - 2 조 - 5 주차]



조원	202055531 김후겸 202055584 이태경 202155540 김채현 202255535 김진우
----	--

## 1. 실험 주제

- Clock Tree 와 UART 통신

## 2. 실험 목적

- Clock Tree 의 이해 및 사용자 Clock 설정
- UART 통신의 원리를 배우고, 실제 설정 방법 파악

## 3. 세부 목표

- Datasheet 및 Reference Manual 을 참고하여 해당 레지스터 및 주소에 대한 설정 이해
- 예제 코드에서 설정되는 Clock 값을 파악하고, 지정된 Clock 으로 설정
- 예제 설정 항목에 따라 UART 를 설정하고, 지정된 Baud rate 로 설정
- User S1 버튼을 누르는 동안 터미널 프로그램(Putty)을 통해 "Hello Team02 출력 후 줄 바꿈
- MCO 를 통해 나오는 System Clock 을 오실로스코프로 수치 확인

## 4. 실험 장비

- STM32F107VCT6
- IAR Embedded Workbench (EW)
- oscilloscope

## 5. 실험 과정(Todo 코드설명)

### 1) 프로젝트 생성 및 파일 구조 설정

4 차와 동일

### 2) 코드 작성 및 설명

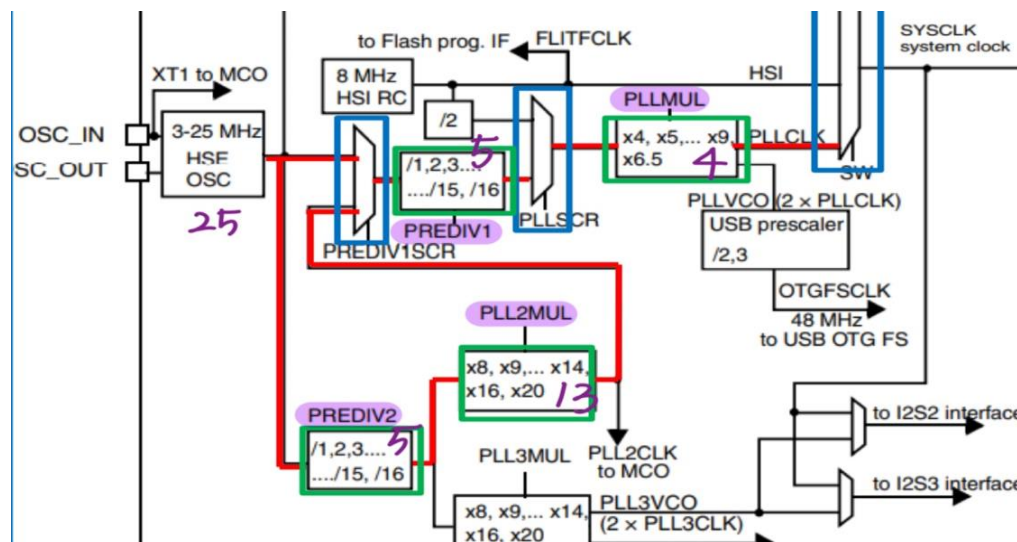
<TODO 1-1>

```
/* HCLK = SYSCLK */
RCC->CFGR |= (uint32_t)RCC_CFGR_HPRE_DIV1;
/* PCLK2 = HCLK / 2, use PPRE2 */
RCC->CFGR |= (uint32_t)RCC_CFGR_PPRE2_DIV2; //TODO 1-1
```

TODO 1-1 에서 SYSCLK 와 PCLK2 값을 설정한다. PCLK2 는 26MHz 이고, SYSCLK 는 52MHz 이므로 DIV2 를 해야한다.

#### <TODO 1-2 & 1-3>

```
/* Configure PLLs -----*/
RCC->CFGR &= (uint32_t)~(RCC_CFGR_PLLSRC | RCC_CFGR_PLLMULL);
RCC->CFGR |= (uint32_t)(RCC_CFGR_PLLSRC_PREDIV1 | RCC_CFGR_PLLMULL4); // TODO 1-2
RCC->CFGR2 &= (uint32_t)~(RCC_CFGR2_PREDIV2 | RCCs_CFGR2_PLL2MUL |
RCC_CFGR2_PREDIV1 | RCC_CFGR2_PREDIV1SRC);
RCC->CFGR2 |= (uint32_t)(RCC_CFGR2_PREDIV2_DIV5 | RCC_CFGR2_PLL2MUL13 |
RCC_CFGR2_PREDIV1SRC_PLL2 | RCC_CFGR2_PREDIV1_DIV5); // TODO 1-3
```



[그림 1]

[그림 1]에 따라, HSE 오실레이터에 25MHz 가 생성되었고, PREDIV2 = 5, PLL2MUL = 13, PREDIV1 = 5, PLLMUL = 4 로 설정해서 52Mhz 를 만들었다.

$$(52 = 25 / 5 * 13 / 5 * 4)$$

#### <TODO 2>

//Set the MCO port for system clock output

```
RCC->CFGR &= ~(uint32_t)RCC_CFGR_MCO;
RCC->CFGR |= (uint32_t)RCC_CFGR_MCO_SYSCLK; // TODO 2
```

```
#define RCC_CFGR_MCO_NOCLOCK ((uint32_t)0x00000000) /*!< No clock */
#define RCC_CFGR_MCO_SYSCLK ((uint32_t)0x04000000) /*!< System clock selected as MCO source */
#define RCC_CFGR_MCO_HSI ((uint32_t)0x00000000) /*!< HSI clock selected as MCO source */
```

[그림 2]

[그림 2]와 같이, stm32f10x.h 파일에서 MCO port 를 설정하는 변수를 찾는다.

### <TODO 3-1>

```
//RCC Setting
/*----- RCC Configuration -----*/
/* GPIO RCC Enable */
/* UART Tx, Rx, MCO port */
RCC->APB2ENR |= (uint32_t)RCC_APB2ENR_IOPAEN; // TODO 3-1
```

D9	41	67	PA8	I/O FT	PA8	USART1_CK/OTG_FS_SOF / TIM1_CH1 <sup>(8)</sup> /MCO	-
C9	42	68	PA9	I/O FT	PA9	USART1_TX <sup>(7)</sup> /TIM1_CH2 <sup>(7)</sup> / OTG_FS_VBUS	-
D10	43	69	PA10	I/O FT	PA10	USART1_RX <sup>(7)</sup> / TIM1_CH3 <sup>(7)</sup> /OTG_FS_ID	-

[그림 3]

```
#define RCC_APB2ENR_AFIOEN ((uint32_t)0x00000001) /*!< Alternate Function I/O clock enable */
#define RCC_APB2ENR_IOPAEN ((uint32_t)0x00000004) /*!< I/O port A clock enable */
#define RCC_APB2ENR_IOPBEN ((uint32_t)0x00000008) /*!< I/O port B clock enable */
```

[그림 4]

[그림 3]에 따라, UART Tx, RX, MCO 는 port A 에 있으므로, [그림 4]와 같이 stm32f10x.h 파일에서 port A 를 키는 변수를 찾는다.

### <TODO 3-2>

```
/* USART RCC Enable */
RCC->APB2ENR |= (uint32_t)RCC_APB2ENR_USART1EN; // TODO 3-2
```

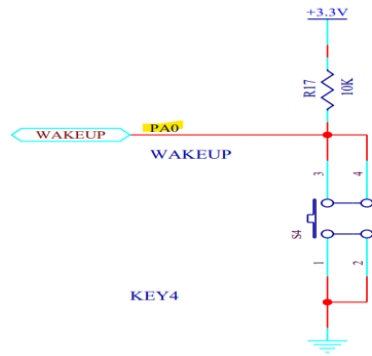
```
#define RCC_APB2ENR_SPI1EN ((uint32_t)0x00001000) /*!< SPI 1 clock enable */
#define RCC_APB2ENR_USART1EN ((uint32_t)0x00004000) /*!< USART1 clock enable */
```

[그림 5]

[그림 5]와 같이 USART clock 을 활성화하기 위해 stm32f10x.h 파일에서 USART1 clock 을 활성화하는 변수를 찾는다.

### <TODO 3-3>

```
/* User key4 Button RCC Enable */
RCC->APB2ENR |= (uint32_t)RCC_APB2ENR_IOPAEN; // TODO 3-3
```



[그림 6]

[그림 6]에 따라, Key 4 는 port A 에 있으므로, <TODO 3-1>과 동일한 변수를 사용한다.

#### <TODO 4-1>

```
//@TODO - 4 GPIO Configuration
/* Reset(Clear) Port A CRH - MCO, USART1 TX,RX*/
GPIOA->CRH &= ~(
    (GPIO_CRH_CNF8 | GPIO_CRH_MODE8) |
    (GPIO_CRH_CNF9 | GPIO_CRH_MODE9) |
    (GPIO_CRH_CNF10 | GPIO_CRH_MODE10)
);
/* MCO Pin Configuration */
GPIOA->CRH |= (GPIO_CRH_CNF8_1 | GPIO_CRH_MODE8_1 | GPIO_CRH_MODE8_0); // TODO
4-1
```

## 9.2.2 Port configuration register high (GPIOx\_CRH) (x=A..G)

Address offset: 0x04

Reset value: 0x4444 4444

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNF15[1:0]	MODE15[1:0]	CNF14[1:0]	MODE14[1:0]	CNF13[1:0]	MODE13[1:0]	CNF12[1:0]	MODE12[1:0]								
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNF11[1:0]	MODE11[1:0]	CNF10[1:0]	MODE10[1:0]	CNF9[1:0]	MODE9[1:0]	CNF8[1:0]	MODE8[1:0]								
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 31:30, 27:26, **CNFy[1:0]**: Port x configuration bits (y= 8 .. 15)  
 23:22, 19:18, 15:14, These bits are written by software to configure the corresponding I/O port.  
 11:10, 7:6, 3:2 Refer to [Table 20: Port bit configuration table on page 161](#).  
**In input mode (MODE[1:0]=00):**  
 00: Analog mode  
 01: Floating input (reset state)  
 10: Input with pull-up / pull-down  
 11: Reserved  
**In output mode (MODE[1:0] > 00):**  
 00: General purpose output push-pull  
 01: General purpose output Open-drain  
**10: Alternate function output Push-pull**  
 11: Alternate function output Open-drain

Bits 29:28, 25:24, **MODEy[1:0]**: Port x mode bits (y= 8 .. 15)  
 21:20, 17:16, 13:12, These bits are written by software to configure the corresponding I/O port.  
 9:8, 5:4, 1:0 Refer to [Table 20: Port bit configuration table on page 161](#).  
 00: Input mode (reset state)  
 01: Output mode, max speed 10 MHz.  
 10: Output mode, max speed 2 MHz.  
**11: Output mode, max speed 50 MHz.**

[그림 7]

[그림 3]에 따르면 MCO 는 port 8 에 존재한다. 강의자료에 따르면, Alternate function output Push-pull 으로 설정해야하므로 [그림 7]과 같이 CNF8 비트를 10 으로 설정한다. 그리고 output mode 는 최대 50MHz 로 출력하기 위해 [그림 7]과 같이 MODE8 비트를 11 으로 설정한다.

<TODO 4-2>

```
/* USART Pin Configuration */
GPIOA->CRH |= (GPIO_CRH_CNF9_1 | GPIO_CRH_MODE9_1 | GPIO_CRH_MODE9_0
| GPIO_CRH_CNF10_1); // TODO 4-2
```

### 9.2.2 Port configuration register high (GPIOx\_CRH) (x=A..G)

Address offset: 0x04

Reset value: 0x4444 4444

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNF15[1:0]	CNF15[1:0]	MODE15[1:0]	MODE15[1:0]	CNF14[1:0]	CNF14[1:0]	MODE14[1:0]	MODE14[1:0]	CNF13[1:0]	CNF13[1:0]	MODE13[1:0]	MODE13[1:0]	CNF12[1:0]	CNF12[1:0]	MODE12[1:0]	MODE12[1:0]
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNF11[1:0]	CNF11[1:0]	MODE11[1:0]	MODE11[1:0]	CNF10[1:0]	CNF10[1:0]	MODE10[1:0]	MODE10[1:0]	CNF9[1:0]	CNF9[1:0]	MODE9[1:0]	MODE9[1:0]	CNF8[1:0]	CNF8[1:0]	MODE8[1:0]	MODE8[1:0]
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Bits 31:30, 27:26, 23:22, 19:18, 15:14, 11:10, 7:6, 3:2

**CNFy[1:0]:** Port x configuration bits (y= 8 .. 15)  
These bits are written by software to configure the corresponding I/O port.  
Refer to [Table 20: Port bit configuration table on page 161](#).

**In input mode (MODE[1:0]=00):**

00: Analog mode

01: Floating input (reset state)

10: Input with pull-up / pull-down

11: Reserved

**In output mode (MODE[1:0] > 00):**

00: General purpose output push-pull

01: General purpose output Open-drain

10: Alternate function output Push-pull

11: Alternate function output Open-drain

Bits 29:28, 25:24, 21:20, 17:16, 13:12, 9:8, 5:4, 1:0

**MODEy[1:0]:** Port x mode bits (y= 8 .. 15)  
These bits are written by software to configure the corresponding I/O port.  
Refer to [Table 20: Port bit configuration table on page 161](#).

00: Input mode (reset state)

01: Output mode, max speed 10 MHz.

10: Output mode, max speed 2 MHz.

11: Output mode, max speed 50 MHz.

[그림 8]

[그림 3]에 따르면 TX 는 port 9 에, RX 는 port10 존재한다. 강의자료에 따르면, TX 는 Alternate function output Push-pull 으로 설정해야하므로 [그림 8]과 같이 CNF9 비트를 10 으로 설정한다. 그리고, 강의자료에 따르면 RX 는 Input with pull-up / pull- down 으로 설정해야하므로, [그림 8]과 같이 CNF10 비트를 10 으로 설정한다. output mode 는 최대 50MHz 로 출력하기 위해 [그림 8]과 같이 MODE8 비트를 11 으로 설정한다.

<TODO 4-3 & TODO 4-4>

```
/* Reset(Clear) Port A CRL - User key4 Button */
GPIOC->CRL &= ~(GPIO_CRL_CNF0 | GPIO_CRL_MODE0); // TODO 4-3
/* User key4 Button Configuration */
GPIOC->CRL |= (GPIO_CRL_CNF0_1); // TODO 4-4
```

### 9.2.1 Port configuration register low (GPIOx\_CRL) (x=A..G)

Address offset: 0x00

Reset value: 0x4444 4444

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNF7[1:0]	MODE7[1:0]	CNF6[1:0]	MODE6[1:0]	CNF5[1:0]	MODE5[1:0]	CNF4[1:0]	MODE4[1:0]	CNF3[1:0]	MODE3[1:0]	CNF2[1:0]	MODE2[1:0]	CNF1[1:0]	MODE1[1:0]	CNF0[1:0]	MODE0[1:0]
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNF3[1:0]	MODE3[1:0]	CNF2[1:0]	MODE2[1:0]	CNF1[1:0]	MODE1[1:0]	CNF0[1:0]	MODE0[1:0]	CNF0[1:0]	MODE0[1:0]	CNF0[1:0]	MODE0[1:0]	CNF0[1:0]	MODE0[1:0]	CNF0[1:0]	MODE0[1:0]
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Bits 31:30, 27:26, 23:22, 19:18, 15:14, 11:10, 7:6, 3:2

**CNFy[1:0]:** Port x configuration bits (y= 0 .. 7)  
These bits are written by software to configure the corresponding I/O port.  
Refer to [Table 20: Port bit configuration table on page 161](#).

**In input mode (MODE[1:0]=00):**

00: Analog mode

01: Floating input (reset state)

**10: Input with pull-up / pull-down**

11: Reserved

**In output mode (MODE[1:0] > 00):**

00: General purpose output push-pull

01: General purpose output Open-drain

10: Alternate function output Push-pull

11: Alternate function output Open-drain

Bits 29:28, 25:24, 21:20, 17:16, 13:12, 9:8, 5:4, 1:0

**MODEy[1:0]:** Port x mode bits (y= 0 .. 7)  
These bits are written by software to configure the corresponding I/O port.  
Refer to [Table 20: Port bit configuration table on page 161](#).

00: Input mode (reset state)

01: Output mode, max speed 10 MHz.

10: Output mode, max speed 2 MHz.

11: Output mode, max speed 50 MHz.

[그림 9]

[그림 6]에 따르면 key4 는 PA0 에 있으므로, <TODO 4-3>에서 CNF0 와 MODE0 bit 를 초기화한다.  
그리고, <TODO 4-4>에서 key4 를 Input with pull-up/pull down 으로 설정하기 위해 [그림 9]와 같이 CNF0 bit 를 10 으로 설정한다.

#### <TODO 6>

// WordLength : 8bit

USART1->CR1 &= ~(uint32\_t)( USART\_CR1\_M); // TODO 6

Bit 12 **M**: Word length

This bit determines the word length. It is set or cleared by software.

**0: 1 Start bit, 8 Data bits, n Stop bit**

**1: 1 Start bit, 9 Data bits, n Stop bit**

Note: The M bit must not be modified during a data transfer (both transmission and reception)

[그림 10]

```
#define USART_CR1_WAKE ((uint16_t)0x0800) /*!< Wakeup method */
#define USART_CR1_M ((uint16_t)0x1000) /*!< Word length */
#define USART_CR1_UE ((uint16_t)0x0000) /*!< USART Enable */
```

[그림 10]에 따르면, word length 를 8bit 로 만들기 위해서는 0 으로 설정해야한다. [그림 11]에 따라, word length 를 설정하는 변수를 찾는다. USART\_CR1\_M 변수에 not 을 하면, bit12 가 0 이 된다.



## <TODO 7>

// Parity : None

```
USART1->CR1 &= ~(uint32_t)(USART_CR1_PCE); // TODO 7
```

Bit 10 **PCE**: Parity control enable

This bit selects the hardware parity control (generation and detection). When the parity control is enabled, the computed parity is inserted at the MSB position (9th bit if M=1; 8th bit if M=0) and parity is checked on the received data. This bit is set and cleared by software. Once it is set, PCE is active after the current byte (in reception and in transmission).

0: Parity control disabled

1: Parity control enabled

[그림 12]

```

#define USART_CR1_PS ((uint16_t)0x0200) /*!< Parity Selection */
#define USART_CR1_PCE ((uint16_t)0x0400) /*!< Parity Control Enable */

```

[그림 13]

[그림 12]에 따르면, parity 를 none 으로 만들기 위해서는 0 으로 설정해야한다. [그림 13]에 따라, parity control 을 활성화하는 변수를 찾는다. USART\_CR1\_PCE 변수에 not 을 하면, bit10 가 0 이 된다.

## <TODO 8>

// Enable Tx and Rx

```
USART1->CR1 |= (uint32_t)(USART_CR1_TE | USART_CR1_RE); // TODO 8
```

Bit 3 **TE**: Transmitter enable

This bit enables the transmitter. It is set and cleared by software.

0: Transmitter is disabled

1: Transmitter is enabled

Note: 1: During transmission, a "0" pulse on the TE bit ("0" followed by "1") sends a preamble (idle line) after the current word, except in smartcard mode.

2: When TE is set there is a 1 bit-time delay before the transmission starts.

Bit 2 **RE**: Receiver enable

This bit enables the receiver. It is set and cleared by software.

0: Receiver is disabled

1: Receiver is enabled and begins searching for a start bit

[그림 14]

```

#define USART_CR1_RE ((uint16_t)0x0004) /*!< Receiver Enable */
#define USART_CR1_TE ((uint16_t)0x0008) /*!< Transmitter Enable */

```

[그림 15]

[그림 14]에 따르면, USART 송신 및 수신을 활성화하기 위해서는 1 으로 설정해야한다. [그림 15]에 따라, TE 와 RE 를 활성화하는 변수를 찾는다. OR 연산자를 활용하여 bit 2 와 3 을 1 로 설정한다.

## <TODO 9>

// Stop bit : 1bit

```
USART1->CR2 &= ~(uint32_t)(USART_CR2_STOP); // TODO 9
```

Bits 13:12 **STOP**: STOP bits  
 These bits are used for programming the stop bits.  
 00: 1 Stop bit  
 01: 0.5 Stop bit  
 10: 2 Stop bits  
 11: 1.5 Stop bit  
 The 0.5 Stop bit and 1.5 Stop bit are not available for UART4 & UART5.

[그림 16]

```
#define USART_CR2_STOP ((uint16_t)0x3000) /*!< STOP[1:0] bits (STOP bits) */
```

[그림 17]

[그림 16]에 따르면, stop bit 를 1bit 로 만들기 위해서는 00 으로 설정해야한다. [그림 17]에 따라, stop bit 에 관한 변수를 찾는다. USART\_CR2\_STOP 변수에 not 을 하면, bit12 와 13 이 0 이 된다.

#### <TODO 10>

```
// CTS, RTS : disable
USART1->CR3 &= ~(uint32_t)(USART_CR3_CTSE | USART_CR3_RTSE); // TODO 10
```

#### Bit 9 CTSE: CTS enable

0: CTS hardware flow control disabled  
 1: CTS mode enabled, data is only transmitted when the nCTS input is asserted (tied to 0). If the nCTS input is deasserted while a data is being transmitted, then the transmission is completed before stopping. If a data is written into the data register while nCTS is deasserted, the transmission is postponed until nCTS is asserted.  
 This bit is not available for UART4 & UART5.

#### Bit 8 RTSE: RTS enable

0: RTS hardware flow control disabled  
 1: RTS interrupt enabled, data is only requested when there is space in the receive buffer. The transmission of data is expected to cease after the current character has been transmitted. The nRTS output is asserted (tied to 0) when a data can be received.  
 This bit is not available for UART4 & UART5.

[그림 18]

```
#define USART_CR3_RTSE ((uint16_t)0x0100) /*!< RTS Enable */
#define USART_CR3_CTSE ((uint16_t)0x0200) /*!< CTS Enable */
```

[그림 19]

[그림 18]에 따르면, CTS 와 RTS 를 비활성화하기 위해서는 0 으로 설정해야 한다. [그림 19]에 따라, CTS 와 RTS 를 활성화하는 변수를 찾는다. USART\_CR3\_CTSE | USART\_CR3\_RTSE 에 not 연산을 하면, bit8 과 9 가 0 이 된다.

#### <TODO 11>

```
// Calculate & configure BRR
USART1->BRR |= 0xA94; // TODO 11
```

$$T_x/R_x \text{ baud} = \frac{f_{clk}}{16 \times \text{USARTDIV}}$$

$$\Rightarrow 9600 = \frac{26 \times 10^6}{16 \times \text{USARTDIV}}$$

$$\textcircled{+} \text{USARTDIV} = \frac{26 \times 10^6}{16 \times 9600} = 169.2908777 \dots$$

$$= 0x169.2908777 \dots$$

$\downarrow$   $\swarrow \times 16 = 0x169.2908777 \dots$   
 $0x169$   $\swarrow$   
 $\therefore 0x169$

[그림 20]

강의자료 p23 에 따라 [그림 20]과 같이 BRR 을 계산한다.

### <TODO 12>

```
// Enable UART (UE)
USART1->CR1 |= USART_CR1_UE; // TODO 12
```

Bit 13 **UE**: USART enable

When this bit is cleared the USART prescalers and outputs are stopped and the end of the current

byte transfer in order to reduce power consumption. This bit is set and cleared by software.

0: USART prescaler and outputs disabled

1: USART enabled

[그림 21]

```
#define USART_CR1_UE ((uint16_t)0x1000) /* USART Enable */
#define USART_CR1_UE ((uint16_t)0x2000) /* USART Enable */
```

[그림 22]

[그림 21]에 따르면, UART 를 활성화하기 위해서는 1 로 설정해야 한다. [그림 22]에 따라, USART 를 활성화하는 변수를 찾는다. OR 연산을 하면, bit13 이 1 이 된다.

### <TODO 13>

```
// Send the message when button is pressed
if ((GPIOA->IDR & GPIO_IDR_IDR0) == 0)
{
    i = 0;
    char* message = &msg[i];
    while (*message != '\0') {
        SendData(*message);
        message++;
    }
}
```

```

        delay();
    } // TODO 13
// ~~~~~ BIT definition for GPIO_IDR register ~~~~~
#define GPIO_IDR_IDR0 ((uint16_t)0x0001) /*!< Port input data, bit 0 */

```

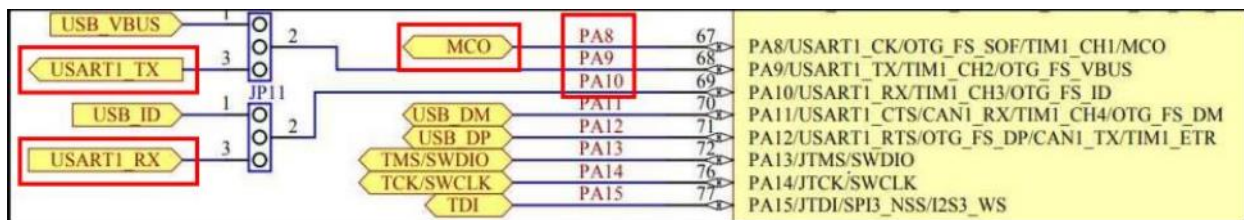
[그림 22]

[그림 6]에 따르면 key4 는 PA0 에 존재하므로 [그림 22]와 같이 0x0001 에 해당하는 변수를 찾는다. Key4 를 눌렀을 때 "Hello Team02"를 한 줄씩 출력하도록 위와 같이 코드를 짠다.

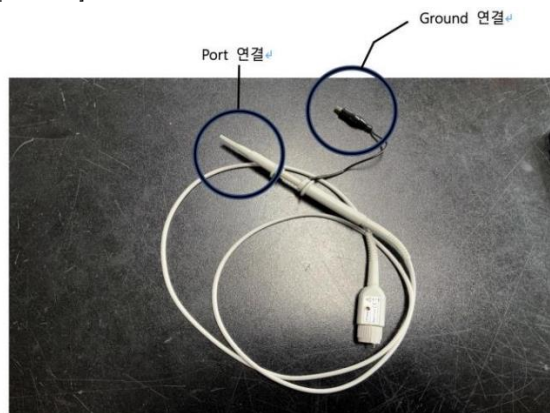
### 3) 시리얼 통신 확인

강의자료 p25 에 따라 putty 를 접속한다.

### 4) 오실로스코프 연결



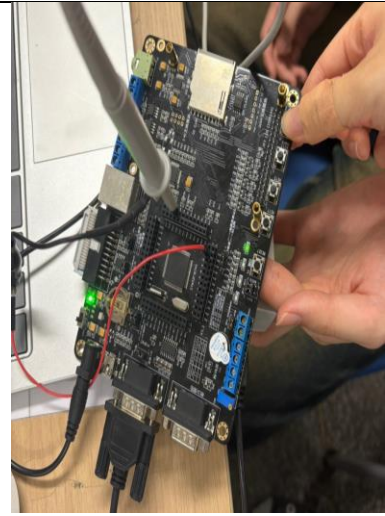
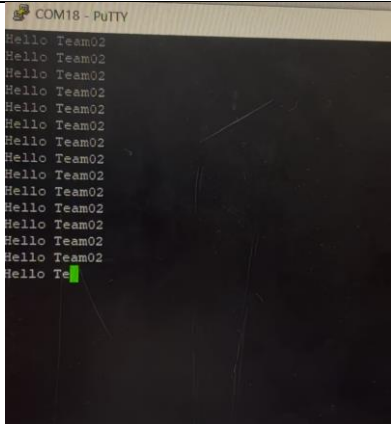
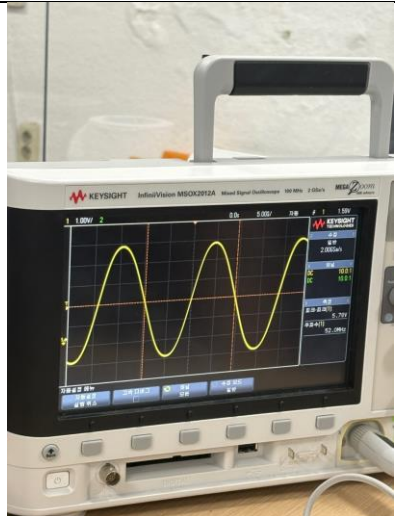
[그림 23]



[그림 23]

[그림 23]에 따르면, MCO 가 PA8 으로 나오므로 [그림 24]의 오실로스코프에서 나오는 선 중 Gound 연결 선은 Ground 에 Port 연결 선은 보드의 PA8 에 연결한다. 강의자료 p26 에 따라 순서대로 오실로스코프를 조작하면 그래프가 그려진다.

## 6. 실험 결과

Key4 눌렀을 때	PUTTY	오실로스코프
		

## 7. 분석 및 결론

- 5 주차 실험은 5 주차 실험은 clock tree 를 이해하여, 원하는 system clock 을 도출해내는 시간을 가졌다. 또한, 주어진 조건에 맞춰 baud rate 를 계산하였고, 오실로스코프와 보드를 연결하였다. 그리고, UART 통신에 대해 공부하고, 관련된 라이브러리 변수에 찾을 수 있게 되었다.

## 8. 참고자료

- stm32\_Datasheet.pdf
- stm32\_ReferenceManual.pdf
- STM32107VCT6\_Schematic.pdf