

“본 강의 동영상 및 자료는 대한민국 저작권법을 준수합니다. 본 강의 동영상 및 자료는 상명대학교 재학생들의 수업목적으로 제작·배포되는 것이므로, 수업목적으로 내려받은 강의 동영상 및 자료는 수업목적 이외에 다른 용도로 사용할 수 없으며, 다른 장소 및 타인에게 복제, 전송하여 공유할 수 없습니다. 이를 위반해서 발생하는 모든 법적 책임은 행위 주체인 본인에게 있습니다.”



# 피지컬 컴퓨팅

## Lec. 9. USART

### Universal(Serial) Synchronous Asynchronous Receiver Transmitter

Heenam Yoon

Department of  
Human-Centered Artificial Intelligence

E-mail) [h-yoon@smu.ac.kr](mailto:h-yoon@smu.ac.kr)  
Room) 0112



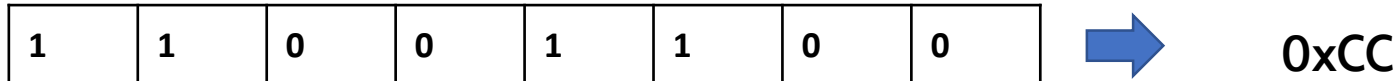
## 정리

- 데이터 송/수신 장비 간의 커넥션이 필요함
- 두 장비는 약속된 규칙에 의해 데이터를 주고 받아야 함
- 데이터를 어떻게 보내고 받을 것인지, 어떤 속도로 전송할 것인지 약속해야 함

# I USART

## 정리

- 데이터를 보내면 받은 장비에서는 수신한 데이터로 하나의 값을 만들게 됨



- 데이터를 어떻게 보내고 받을 것인지 (몇 비트를 보낼 것인지)
  - 송신 쪽에서는 8비트를 보내는데, 수신 쪽에서 7비트씩 계산하면, 다른 결과가 만들어 질 것

- 데이터를 어떻게 보내고 받을 것인지 (어떤 비트를 먼저 보낼 것인지)
  - 아래 두 결과는 다름 (11001100 vs. 00110011)

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

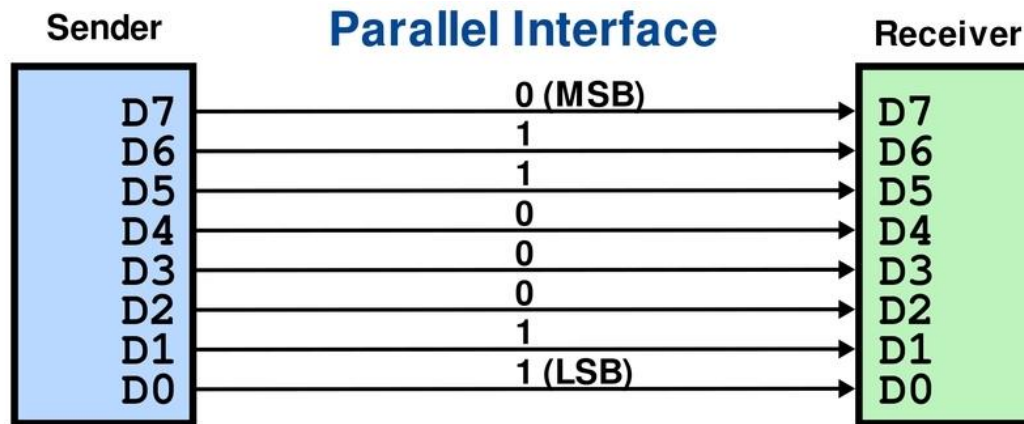


## 정리

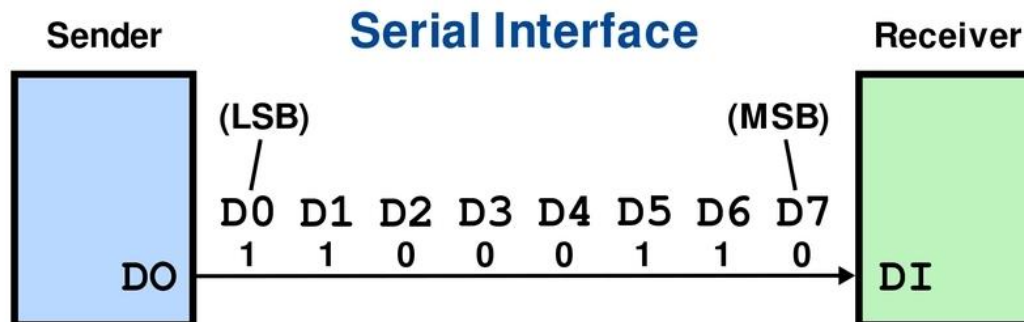
- 어떤 속도로 전송할 것 인지
  - 송수신 단에서 같은 속도를 인지하고 있어야 함
  - 여기서 단위는 보통 bps를 씀 (bits per second)
  - 그리고, 이를 baud rate이라고 함
  - Baud rate이 115200bps라면 초당 115,200개의 비트를 전송할 수 있다는 것

# Digital Communication

- Parallel interface



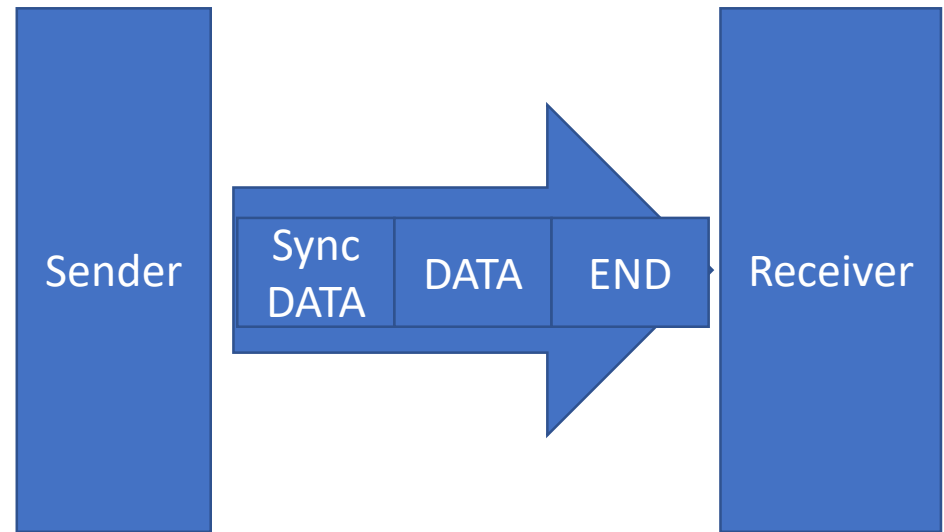
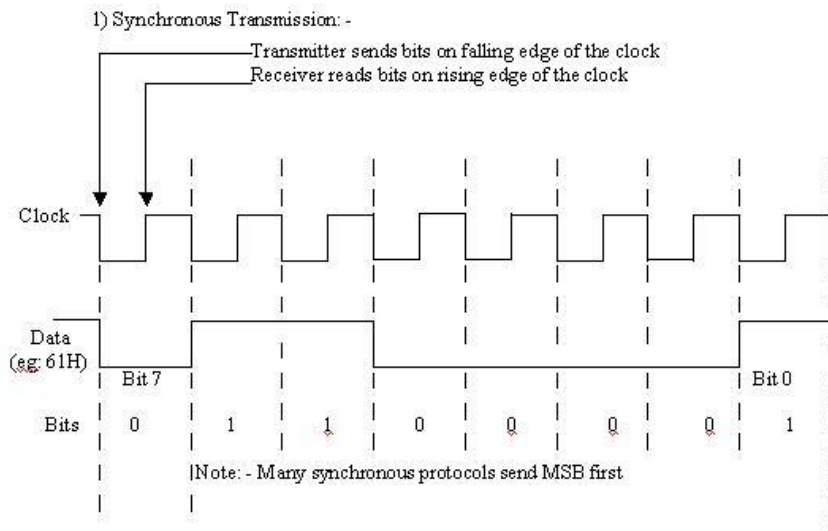
- Serial Interface



# Digital Communication

## Synchronous Transmission

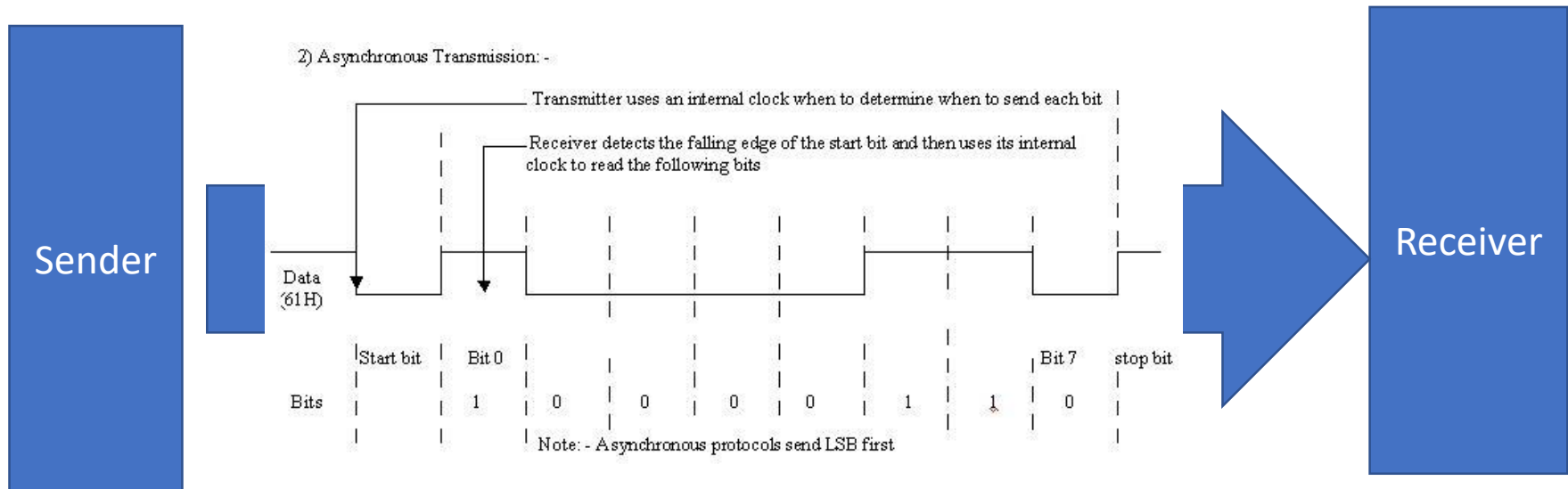
- Block 단위 전송
- 시간이 명시됨
- Overhead가 작음
- 하드웨어가 비쌘



# Digital Communication

## Asynchronous Transmission

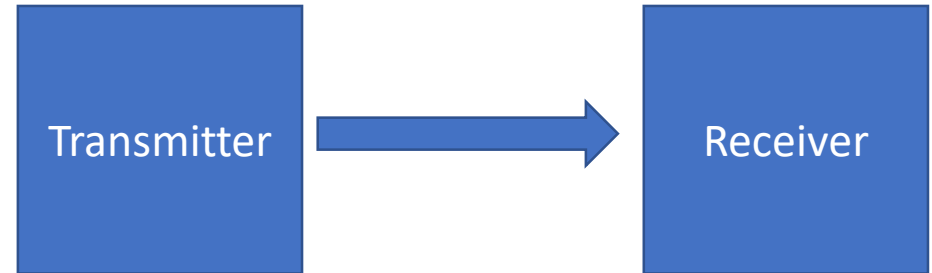
- Byte단위 전송
- 하드웨어가 간단하고 비교적 저렴함
- Clock 조정이 필요
- 추가적인 제어 bit가 필요
- Overhead가 상대적으로 큼



# ■ Digital Communication

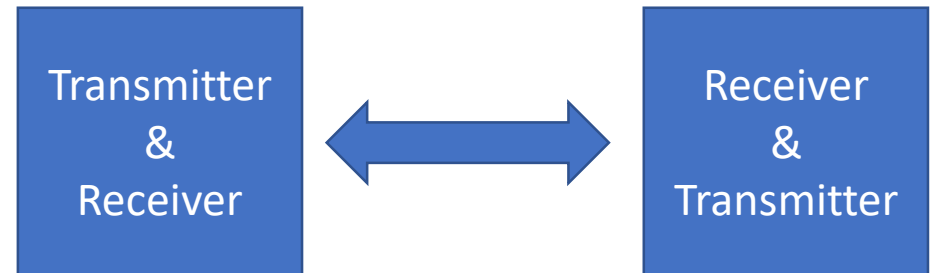
- Simplex transmission

- 한 방향으로만 전송



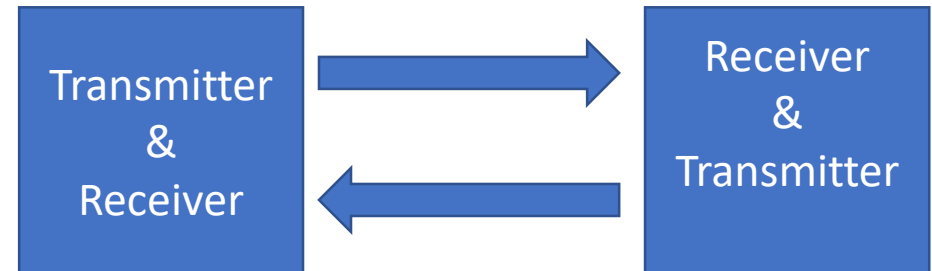
- Half-duplex transmission

- 1개 회선으로 양 방향 전송
- 동시에 양방향 전송 불가



- Full-duplex transmission

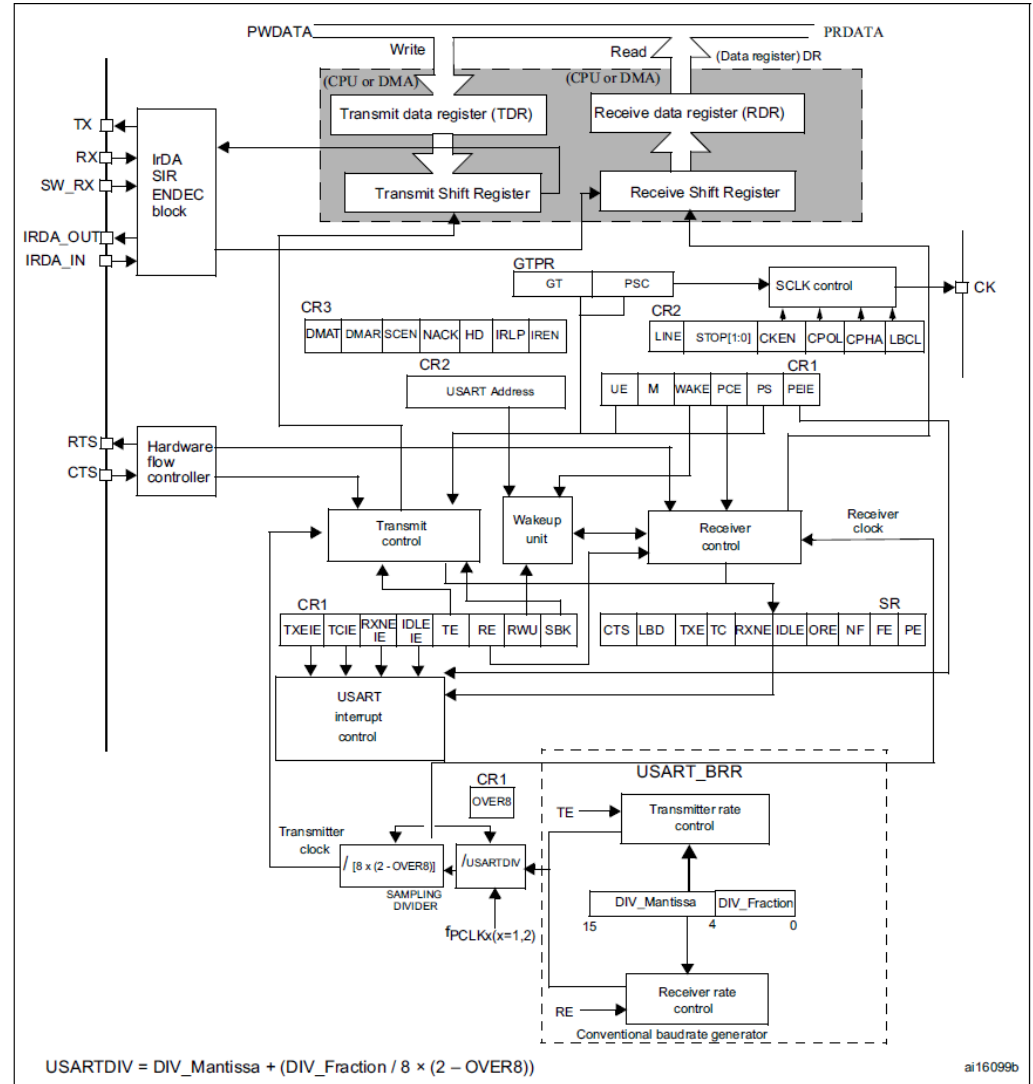
- 2개 회선으로 양방향 전송 가능
- 동시에 양 방향 전송이 가능





- Universal(Serial) Synchronous Asynchronous Receiver Transmitter
  - 직렬 통신을 수행하는 장치
  - Full duplex, asynchronous communications
- RS-232c 직렬 통신
  - 컴퓨터에 내장된 USART의 송수신 신호는 TTL 로직 레벨이기 때문에 원거리 통신을 위해 신호 레벨을 변경하는 규격으로 RS-232c, RS-422, RS-485 등이 있는데, 이 중에서 RS-232c 규격이 가장 널리 사용되는 규격임
  - RS-232c는 전화선을 이용한 데이터 통신을 하기 위해 1962년에 미국의 EIA(Electronic Industries Association)에서 DTE(Data Terminal Equipment)와 모뎀 등과 같은 DCE(Data Communication Equipment) 사이에 데이터 전송용으로 제정한 통신 규격임

- USART block diagram



## USART 기능 소개

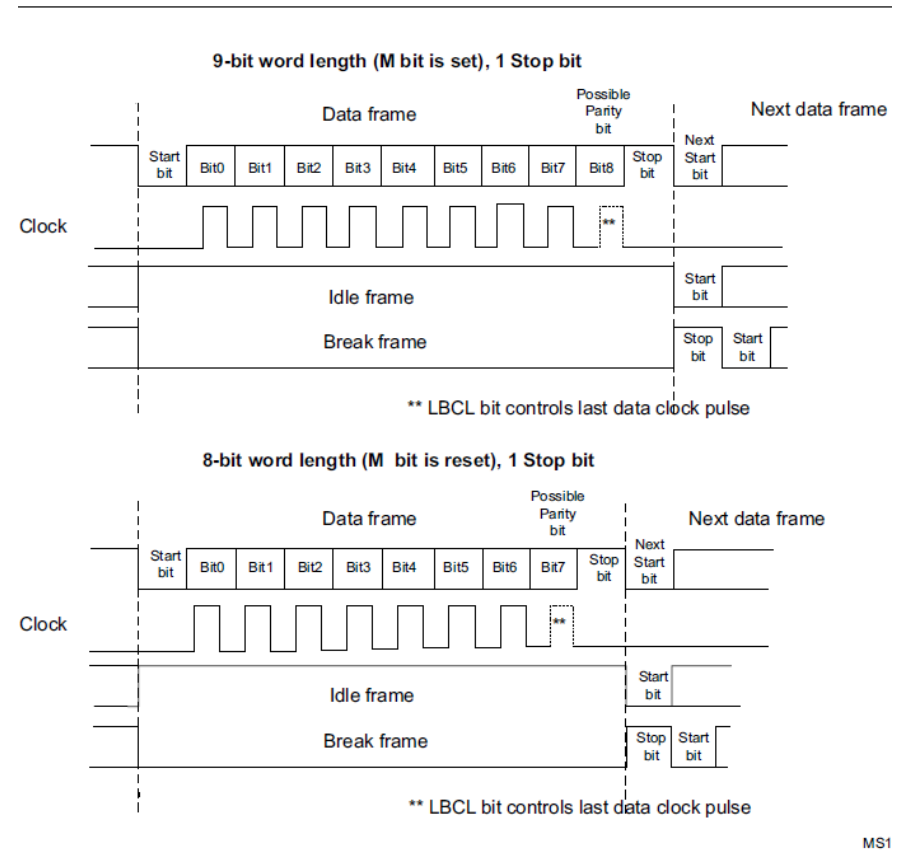
- 통신 속도
  - 직렬 전송 시 송신 측과 수신 측의 전송속도를 일정하게 유지
  - 통신 속도는 시간당 데이터를 전송할 수 있는 양을 나타냄
  - 통신 속도는 Bit Rate(bps : bits per second)를 사용하는데, 이는 1초당 전송되는 비트의 수를 말함
  - RS232 통신의 경우 baud rate라는 용어를 많이 사용하는데, 이는 1초당 전송되는 변조된 신호의 수를 뜻함
- 스톱 비트
  - 전송을 시작할 경우 논리 1을 내보내고 8비트를 전송한 후 스톱 비트를 전송
  - 스타트 비트는 고정되어 있어 사용자가 조정할 필요가 없지만 스톱 비트는 1과 1.5, 2 비트 중 하나를 선택
- 패리티
  - 데이터를 전송할 때 데이터에 패리티 비트를 추가하여 외적인 방해에 의해 오류가 발생했을 경우에 오류를 검출하기 위해 사용
  - 패리티 비트는 데이터 길이가 7인 경우에 8번째 비트를 패리티 비트로 이용

## USART 기능 소개

- 자료 길이
  - 하나의 데이터를 전송하는데 필요한 데이터 길이 즉, 비트 수를 말함  
(2 바이트 한글 전송에서는 데이터 길이를 8로 설정)
- STM32 USART 포트
  - USART 6개 (동기식 4개, 비동기식 2개) 내장
  - STM32의 USART는 송신 완료(TX Complete), 송신 데이터 레지스터 준비완료(TX Data Register Empty), 수신 완료(RX Complete) 등 10가지의 인터럽트를 제공
  - STM32 (Cortex-M4)에서는 세 개의 USART 포트를 USART0, USART1, USART2 라고 일컫음
- STM32 USART 데이터 프레임 포맷
  - (1 비트의 스타트 비트) + (5, 6, 7, 8, 9 비트의 데이터 비트) + (0, 1 비트의 패리티비트) + (0.5, 1, 1.5, 2 비트의 스톱비트) 프레임으로 이루어져 최소 7비트 최대 13비트로 구성가능

## USART 기능 소개

- Word length programming
  - 스타트 비트 : 1비트, 항상 0레벨.  
송신시에 자동적으로 생성
  - 데이터 비트 : 5, 6, 7, 8, 9비트  
가능
  - 패리티 비트 : 패리티를 사용하지 않을 수도 있고 사용하는 경우 홀수 혹은 짝수 패리티 1비트 사용
  - 스톱 비트 : 0.5, 1, 1.5, 2 개의 비트가 가능하며 항상 1레벨. 송신 시에 자동적으로 생성



# I USART

## USART baud rate 설정

- Tx/Rx 모두 같은 baud rate로 설정
- Baud rate

$$\text{Tx/Rx baud} = \frac{f_{\text{CK}}}{8 \times (2 - \text{OVER8}) \times \text{USARTDIV}}$$

- USARTDIV is an unsigned fixed point number that is coded on the USART\_BRR register.
- When OVER8=0, the fractional part is coded on 4 bits and programmed by the DIV\_fraction[3:0] bits in the USART\_BRR register
- When OVER8=1, the fractional part is coded on 3 bits and programmed by the DIV\_fraction[2:0] bits in the USART\_BRR register, and bit DIV\_fraction[3] must be kept cleared.

## USART baud rate 설정

Oversampling by 16 (OVER8=0)							
Baud rate		f <sub>CLK</sub> = 42 MHz			f <sub>CLK</sub> = 84 MHz		
S.No	Desired	Actual	Value programmed in the baud rate register	% Error = (Calculated - Desired)B.Rate / Desired B.Rate	Actual	Value programmed in the baud rate register	% Error
1.	1.2 KBps	1.2 KBps	2187.5	0	1.2 KBps	NA	0
2.	2.4 KBps	2.4 KBps	1093.75	0	2.4 KBps	2187.5	0
3.	9.6 KBps	9.6 KBps	273.4375	0	9.6 KBps	546.875	0
4.	19.2 KBps	19.195 KBps	136.75	0.02	19.2 KBps	273.4375	0
5.	38.4 KBps	38.391 KBps	68.375	0.02	38.391 KBps	136.75	0.02
6.	57.6 KBps	57.613 KBps	45.5625	0.02	57.613 KBps	91.125	0.02
7.	115.2 KBps	115.068 KBps	22.8125	0.11	115.226 KBps	45.5625	0.02
8.	230.4 KBps	230.769 KBps	11.375	0.16	230.137 KBps	22.8125	0.11
9.	460.8 KBps	461.538 KBps	5.6875	0.16	461.538 KBps	11.375	0.16
10.	921.6 KBps	913.043 KBps	2.875	0.93	923.076 KBps	5.6875	0.93
11.	1.792 MBps	1.826 MBps	1.4375	1.9	1.787 MBps	2.9375	0.27
12.	1.8432 MBps	1.826 MBps	1.4375	0.93	1.826 MBps	2.875	0.93
13.	3.584 MBps	N.A	N.A	N.A	3.652 MBps	1.4375	1.9
14.	3.6864 MBps	N.A	N.A	N.A	3.652 MBps	1.4375	0.93

## USART baud rate 설정

- 전송할 데이터에 양에 따라 Baud rate를 잘 설정할 필요가 있음



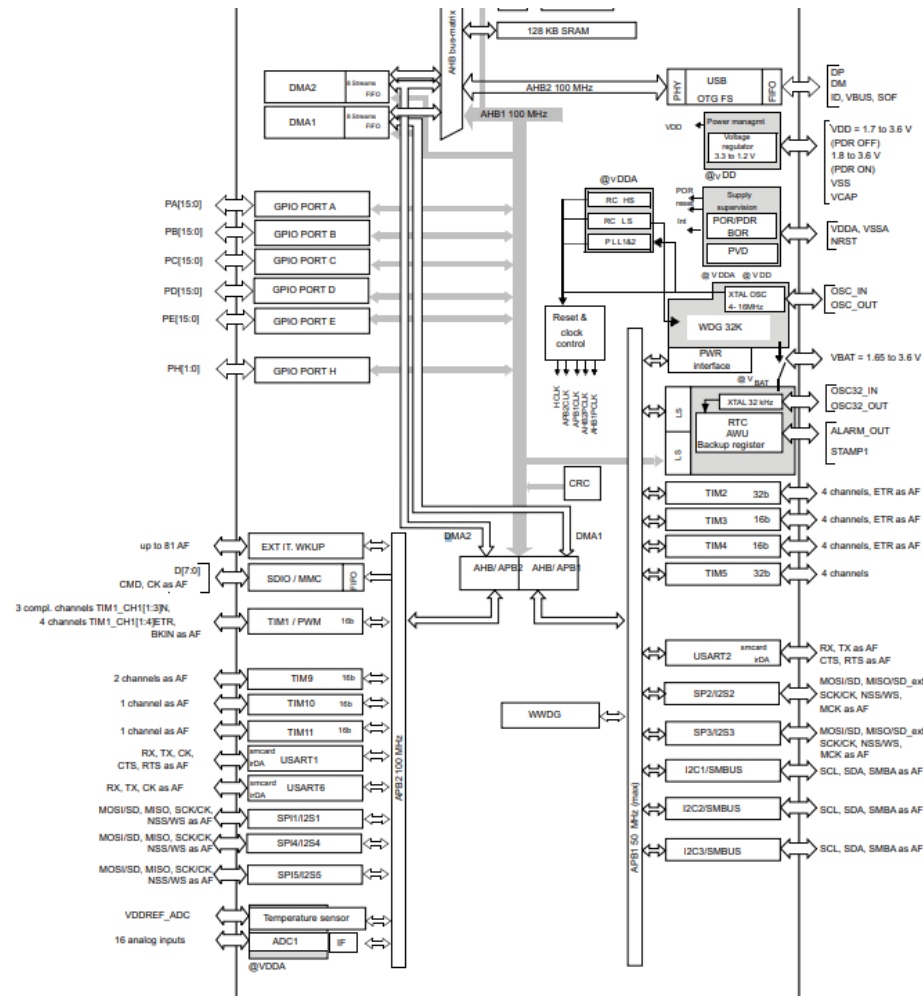


Table 9 Alternate function mapping

Port	AF00	AF01	AF02	AF03	AF04	AF05	AF06	AF07	AF08	AF09	AF10	AF11	AF12	AF13	AF14	AF15
	SYS_AF	TIM1/TIM2	TIM3/ TIM4/ TIM5	TIM9/ TIM10/ TIM11	I2C1/I2C2/ I2C3	SPI1/I2S1S PI2/ I2S2/SPI3/ I2S3	SPI2/I2S2/ SPI3/ I2S3/SPI4/ I2S4/SPI5/ I2S5	SPI3/I2S3/ USART1/ USART2	USART6	I2C2/ I2C3	OTG1_FS		SDIO			
PA0	-	TIM2_CH1/ TIM2_ETR	TIM5_CH1	-	-	-	-	USART2_ CTS	-	-	-	-	-	-	-	EVENT OUT
PA1	-	TIM2_CH2	TIM5_CH2	-	-	SPI4_MOSI/ I2S4_SD	-	USART2_ RTS	-	-	-	-	-	-	-	EVENT OUT
PA2	-	TIM2_CH3	TIM5_CH3	TIM9_CH1	-	I2S2_CKIN	-	USART2_ TX	-	-	-	-	-	-	-	EVENT OUT
PA3	-	TIM2_CH4	TIM5_CH4	TIM9_CH2	-	I2S2_MCK	-	USART2_ RX	-	-	-	-	-	-	-	EVENT OUT
PA4	-	-	-	-	-	SPI1_NSS/I 2S1_WS	SPI3_NSS/I2 S3_WS	USART2_ CK	-	-	-	-	-	-	-	EVENT OUT

# 실습 준비

- 실습 노트북에 virtual comport 드라이버 설치
- <https://www.st.com/en/development-tools/stsw-stm32102.html>

STSW-STM32102 ACTIVE Save to MyST

## STM32 Virtual COM Port Driver

[Get Software](#) [Download databrief](#)

[Overview](#) [Documentation](#)

### Product overview

Description All features Get Software Recommended for you

#### Description

Compatible with the x86 and x64 platforms

The STSW-STM32102 software package contains four installation files based on the various versions of the Microsoft® operating system.

OS versions prior to Windows® 7 are compatible with the Windows® 7 installations included in the package.

Starting from Windows® 10, the STSW-STM32102 driver is no more adequate and the usage of the native inbox driver is recommended.

#### All features

- Virtual COM port driver installation package for Windows® operating systems: 98SE, 2000, XP, Vista®, 7, and 8.x

#### Get Software

Part Number	General Description	Latest version	Supplier	ECCN (EU)	ECCN (US)	Download
STSW-STM32102	STM32 Virtual COM Port Driver	1.5.0	ST	NEC	3D991	<a href="#">Get latest</a>

이메일 인증 후 다운로드 됨  
설치

# 실습

## 준비

- MPU - PC간 통신을 확인하기 위한 프로그램 준비

- Comportmaster

<http://withrobot.com/data/?mod=document&uid=12>

- PuTTY

<https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html>

- 각자 장단점이 있으므로 둘 다 준비

### Alternative binary files

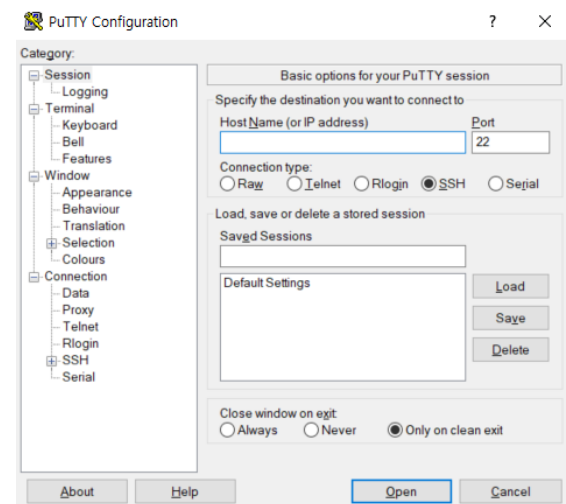
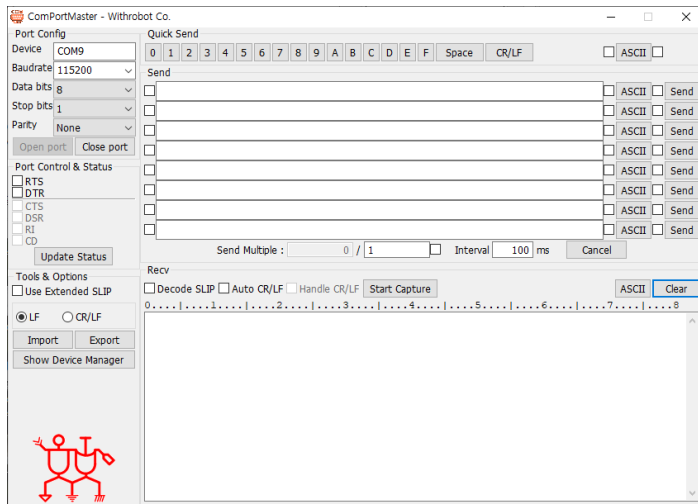
The installer packages above will provide versions of all of these (except PuTTYtel and  
(Not sure whether you want the 32-bit or the 64-bit version? Read the [FAQ entry](#).)

putty.exe (the SSH and Telnet client itself)

64-bit x86: [putty.exe](#) [\(signature\)](#)

64-bit Arm: [putty.exe](#) [\(signature\)](#)

32-bit x86: [putty.exe](#) [\(signature\)](#)



# I 실습

## 실습 1: MCU에서 PC로 문자 전송하기

- MCU에서 PC로 'a' 문자를 전송함
- HW 구성: USB 연결
- FW 구성: 프로젝트 파일 추가
  - 폴더: STM32F4xx\_DSP\_StdPeriph\_Lib\_V1.8.0\Libraries\STM32F4xx\_StdPeriph\_Driver\src
  - 파일: stm32f4xx\_usart.c, misc.c, stm32f4xx\_tim.c, stm32f4xx\_syscfg.c

## 실습 1: MCU에서 PC로 문자 전송하기

```
#include "stm32f4xx.h"
```

```
void LED_init(void)
```

```
{
```

```
    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOA, ENABLE);
```

```
    GPIO_InitTypeDef GPIO_InitStructure;
```

```
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_OUT;
```

```
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_5;
```

```
    GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
```

```
    GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL;
```

```
    GPIO_Init(GPIOA, &GPIO_InitStructure);
```

```
}
```

## 실습 1: MCU에서 PC로 문자 전송하기

```
void USART2_Configuration(void)
{
    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOA, ENABLE);

    GPIO_InitTypeDef GPIO_InitStructure;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF;
    GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
    GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_UP;
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_2 | GPIO_Pin_3;
    GPIO_Init(GPIOA, &GPIO_InitStructure);

    GPIO_PinAFConfig(GPIOA, GPIO_PinSource2, GPIO_AF_USART2);
    GPIO_PinAFConfig(GPIOA, GPIO_PinSource3, GPIO_AF_USART2);

    RCC_APB1PeriphClockCmd(RCC_APB1Periph_USART2, ENABLE);

    USART_InitTypeDef USART_InitStructure;
    USART_InitStructure.USART_BaudRate = 115200;
    USART_InitStructure.USART_WordLength = USART_WordLength_8b;
    USART_InitStructure.USART_StopBits = USART_StopBits_1;
    USART_InitStructure.USART_Parity = USART_Parity_No;
    USART_InitStructure.USART_Mode = USART_Mode_Rx | USART_Mode_Tx;
    USART_Init(USART2, &USART_InitStructure);
    USART_Cmd(USART2, ENABLE);
}
```

	AF00	AF01	AF02	AF03	AF04	AF05	AF06	AF07
Port	SYS_AF	TIM1/TIM2	TIM3/ TIM4/ TIM5	TIM9/ TIM10/ TIM11	I2C1/I2C2/ I2C3	SP11/I2S1S PI2/ I2S2/SP13/ I2S3	SP12/I2S2/ SP13/ I2S3/SP14/ I2S4/SP15/ I2S5	SP13/I2S3/ USART1/ USART2
PA0	-	TIM2_CH1/ TIM2_ETR	TIM5_CH1	-	-	-	-	USART2_ CTS
PA1	-	TIM2_CH2	TIM5_CH2	-	-	SP14_MOSI I2S4_SD	-	USART2_ RTS
PA2	-	TIM2_CH3	TIM5_CH3	TIM9_CH1	-	I2S2_CKIN	-	USART2_ TX
PA3	-	TIM2_CH4	TIM5_CH4	TIM9_CH2	-	I2S2_MCK	-	USART2_ RX
PA4	-	-	-	-	-	SP11_NSS/I 2S1_WS	SP13_NSS/I2 S3_WS	USART2_ CK

## 실습 1: MCU에서 PC로 문자 전송하기

```
void TIM2_Configuration(int intervals)
{
    RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM2, ENABLE);

    TIM_TimeBaseInitTypeDef TIM_TimeBaseStructure;
    NVIC_InitTypeDef NVIC_InitStructure;

    TIM_TimeBaseStructure.TIM_Prescaler = 26880 - 1;
    TIM_TimeBaseStructure.TIM_Period = intervals - 1;

    TIM_TimeBaseStructure.TIM_CounterMode = TIM_CounterMode_Up;
    TIM_TimeBaseInit(TIM2, &TIM_TimeBaseStructure);

    TIM_ClearITPendingBit(TIM2, TIM_IT_Update);
    TIM_ITConfig(TIM2, TIM_IT_Update, ENABLE);
    TIM_Cmd(TIM2, ENABLE);

    NVIC_PriorityGroupConfig(NVIC_PriorityGroup_1);
    NVIC_InitStructure.NVIC_IRQChannel = TIM2_IRQn;
    NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 0;
    NVIC_InitStructure.NVIC_IRQChannelSubPriority = 1;
    NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
    NVIC_Init(&NVIC_InitStructure);
}
```

현재 딱히 하는 일은 없음  
(다음 실습을 위해 구현해 둔 것)

```
void TIM2_IRQHandler(void)
{
    if(TIM_GetITStatus(TIM2, TIM_IT_Update) != RESET)
    {
        GPIO_ToggleBits(GPIOA, GPIO_Pin_5);

        // clear interrupt flag
        TIM_ClearITPendingBit(TIM2, TIM_IT_Update);
    }
}
```



# I 실습

## 실습 1: MCU에서 PC로 문자 전송하기

```
void Serial_Send(unsigned char t)
{
    while(USART_GetFlagStatus(USART2, USART_FLAG_TXE) == RESET);
    USART_SendData(USART2, t);
}
```

데이터 전송(Tx)이 완료되면, 인터럽트 플래그가 변할 것  
그렇지 않다면, 데이터를 전송하고 있다는 뜻  
기존에 전송하는 것이 있다면, 기다려라

USART2 Tx로 t변수 값을 전송하라

# I 실습

## 실습 1: MCU에서 PC로 문자 전송하기

```
int main()
{
    LED_init();
    TIM2_Configuration(1000);
    USART2_Configuration();

    while(1)
    {
        Serial_Send('a');
    }
    return 0;
}
```

# 실습

## 실습 1: MCU에서 PC로 문자 전송하기

- [제어판] - [장치관리자]에서 어떤 Port에 연결되었는지 확인  
(PC마다, 상황에 따라 연결되는 포트가 다르므로 항상 먼저 확인!)
- 이번 실습에서는 ComPortMaster 이용

The image shows a Windows Device Manager window on the left and the ComPortMaster software on the right. In the Device Manager, under 'Ports', 'STMMicroelectronics STLink Virtual COM Port(COM9)' is highlighted with a red box and labeled '1.'. A red arrow points from this box to the 'COM9' selection in the ComPortMaster 'Port Config' section, which is labeled '2.'. In the ComPortMaster 'Port Config' section, the 'Open port' button is highlighted with a red box and labeled '4.'. In the ComPortMaster 'Quick Send' section, the 'ASCII' checkbox is highlighted with a red box and labeled '(3.)'. Below the software window, there is a green text box with the text: '이것은 원래 안 눌러도 되는데, 위 코드를 실행시키면 문제가 발생하여 임시로 누르는 것임'.

1. STMMicroelectronics STLink Virtual COM Port(COM9)

2. COM9

4. Open port

(3.) ASCII

이것은 원래 안 눌러도 되는데,  
위 코드를 실행시키면  
문제가 발생하여 임시로 누르는 것임

# 실습

## 실습 1: MCU에서 PC로 문자 전송하기

- 'a'를 보냄

### ASCII Code 일람표

(Table of American Standard Code for Information Interchange)

10진	16진	문자	10진	16진	문자	10진	16진	문자	10진	16진	문자
0	0x00	NUL	32	0x20	SP	64	0x40	@	96	0x60	`
1	0x01	SOH	33	0x21	!	65	0x41	A	97	0x61	a
2	0x02	STX	34	0x22	"	66	0x42	B	98	0x62	b
3	0x03	ETX	35	0x23	#	67	0x43	C	99	0x63	c
4	0x04	EOT	36	0x24	\$	68	0x44	D	100	0x64	d
5	0x05	ENQ	37	0x25	%	69	0x45	E	101	0x65	e
6	0x06	ACK	38	0x26	&	70	0x46	F	102	0x66	f
7	0x07	BEL	39	0x27	'	71	0x47	G	103	0x67	g
8	0x08	BS	40	0x28	(	72	0x48	H	104	0x68	h
9	0x09	HT	41	0x29	)	73	0x49	I	105	0x69	i
10	0x0A	LF	42	0x2A	*	74	0x4A	J	106	0x6A	j
11	0x0B	VT	43	0x2B	+	75	0x4B	K	107	0x6B	k
12	0x0C	FF	44	0x2C	,	76	0x4C	L	108	0x6C	l
13	0x0D	CR	45	0x2D	-	77	0x4D	M	109	0x6D	m
14	0x0E	SO	46	0x2E	.	78	0x4E	N	110	0x6E	n
15	0x0F	SI	47	0x2F	/	79	0x4F	O	111	0x6F	o
16	0x10	DLE	48	0x30	0	80	0x50	P	112	0x70	p
17	0x11	DC1	49	0x31	1	81	0x51	Q	113	0x71	q
18	0x12	DC2	50	0x32	2	82	0x52	R	114	0x72	r
19	0x13	DC3	51	0x33	3	83	0x53	S	115	0x73	s
20	0x14	DC4	52	0x34	4	84	0x54	T	116	0x74	t
21	0x15	NAK	53	0x35	5	85	0x55	U	117	0x75	u
22	0x16	SYN	54	0x36	6	86	0x56	V	118	0x76	v
23	0x17	ETB	55	0x37	7	87	0x57	W	119	0x77	w
24	0x18	CAN	56	0x38	8	88	0x58	X	120	0x78	x
25	0x19	EM	57	0x39	9	89	0x59	Y	121	0x79	y
26	0x1A	SUB	58	0x3A	:	90	0x5A	Z	122	0x7A	z
27	0x1B	ESC	59	0x3B	;	91	0x5B	[	123	0x7B	{
28	0x1C	FS	60	0x3C	<	92	0x5C	\	124	0x7C	
29	0x1D	GS	61	0x3D	=	93	0x5D	]	125	0x7D	}
30	0x1E	RS	62	0x3E	>	94	0x5E	^	126	0x7E	~
31	0x1F	US	63	0x3F	?	95	0x5F	_	127	0x7F	DEL

범례 : ■ 알파벳 / ■ 숫자 / ■ 구두점 / ■ 제어 문자 / ■ 공백 문자

## 실습 1: MCU에서 PC로 문자 전송하기

- Clock 변경

### system\_stm32f4xx.c 파일 수정

```
void SystemInit(void)
{
    /* FPU settings -----*/
    #if (__FPU_PRESENT == 1) && (__FPU_USED == 1)
        SCB->CPACR |= ((3UL << 10*2)|(3UL << 11*2)); /* set CP10 and CP11 Full Access */
    #endif
    /* Reset the RCC clock configuration to the default reset state -----*/
    /* Set HSION bit */
    RCC->CR |= (uint32_t)0x00000001;

    /* Reset CFGR register */
    RCC->CFGR = 0x00000000;

    /* Reset HSEON, CSSON and PLLON bits */
    RCC->CR &= (uint32_t)0xFEFFFFFF;

    /* Reset PLLCFGR register */
    RCC->PLLCFGR = 0x24003010;

    /* Reset HSEBYP bit */
    RCC->CR &= (uint32_t)0xFFFBFFFF;

    /* Disable all interrupts */
    RCC->CIR = 0x00000000;

    #if defined (DATA_IN_ExtSRAM) || defined (DATA_IN_ExtSDRAM)
        SystemInit_ExtMemCtl();
    #endif /* DATA_IN_ExtSRAM || DATA_IN_ExtSDRAM */

    /* Configure the System clock source, PLL Multiplier and Divider factors,
       AHB/APBx prescalers and Flash settings -----*/
    SetSysClock();

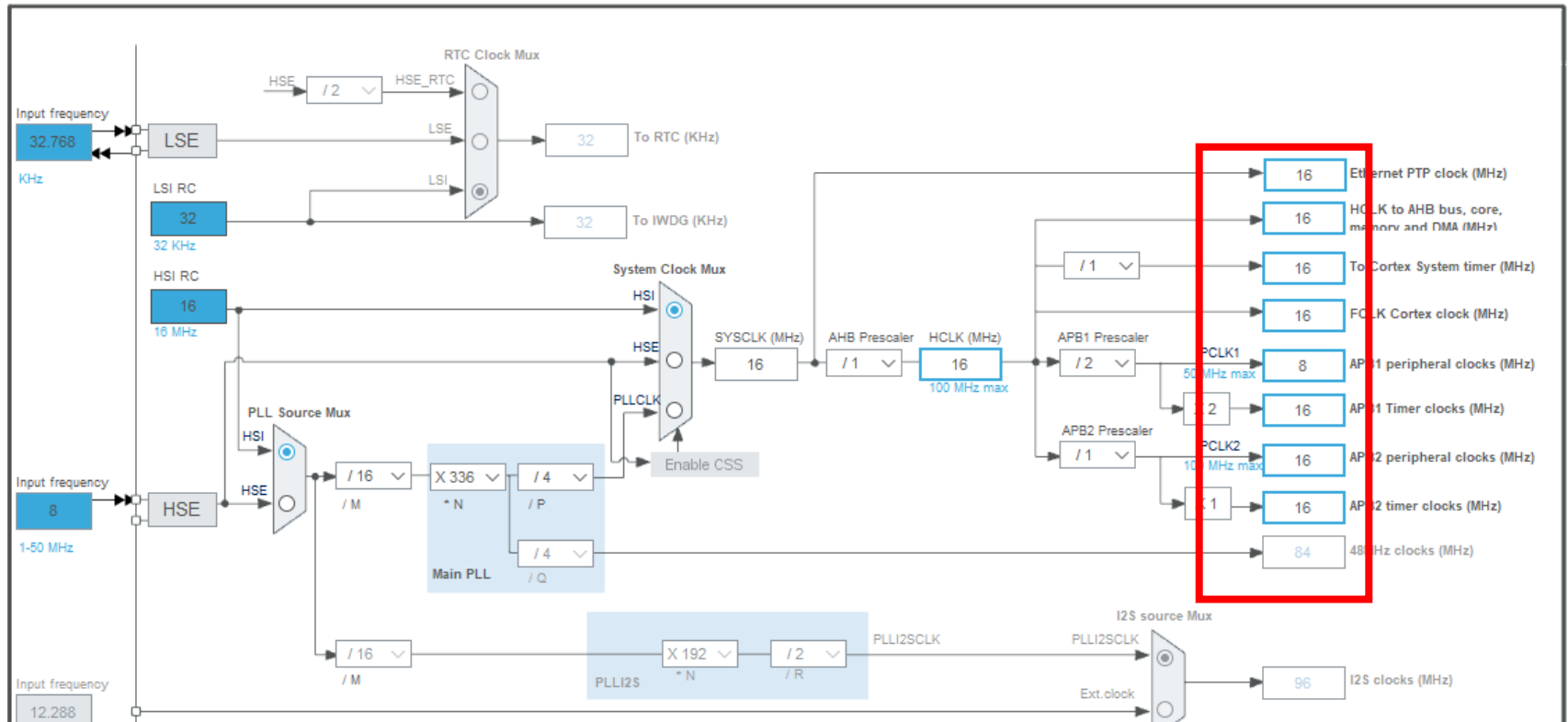
    /* Configure the Vector Table location add offset address -----*/
    #if defined VECT_TAB_SRAM
        SCB->VTOR = SRAM_BASE | VECT_TAB_OFFSET; /* Vector Table Relocation in Internal SRAM */
    #else
        SCB->VTOR = FLASH_BASE | VECT_TAB_OFFSET; /* Vector Table Relocation in Internal FLASH */
    #endif
}
```

주석처리

## 실습 1: MCU에서 PC로 문자 전송하기

- 16MHz로 변경됨

Oversampling by 16 (OVER8=0)							
Baud rate		f <sub>CLK</sub> = 42 MHz			f <sub>CLK</sub> = 84 MHz		
S.No	Desired	Actual	Value programmed in the baud rate register	% Error = (Calculated - Desired)/Desired B.Rate	Actual	Value programmed in the baud rate register	% Error
1.	1.2 KBps	1.2 KBps	2187.5	0	1.2 KBps	NA	0
2.	2.4 KBps	2.4 KBps	1093.75	0	2.4 KBps	2187.5	0
3.	9.6 KBps	9.6 KBps	273.4375	0	9.6 KBps	546.875	0
4.	19.2 KBps	19.195 KBps	136.75	0.02	19.2 KBps	273.4375	0
5.	38.4 KBps	38.391 KBps	68.375	0.02	38.391 KBps	136.75	0.02
6.	57.6 KBps	57.613 KBps	45.5625	0.02	57.613 KBps	91.125	0.02
7.	115.2 KBps	115.068 KBps	22.8125	0.11	115.226 KBps	45.5625	0.02
8.	230.4 KBps	230.769 KBps	11.375	0.16	230.137 KBps	22.8125	0.11
9.	460.8 KBps	461.538 KBps	5.6875	0.16	461.538 KBps	11.375	0.16
10.	921.6 KBps	913.043 KBps	2.875	0.93	923.076 KBps	5.6875	0.93
11.	1.792 MBps	1.826 MBps	1.4375	1.9	1.787 MBps	2.9375	0.27
12.	1.8432 MBps	1.826 MBps	1.4375	0.93	1.826 MBps	2.875	0.93
13.	3.584 MBps	N.A	N.A	N.A	3.652 MBps	1.4375	1.9
14.	3.6864 MBps	N.A	N.A	N.A	3.652 MBps	1.4375	0.93



## 실습 1: MCU에서 PC로 문자 전송하기

```
void TIM2_Configuration(int intervals)
{
    RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM2, ENABLE);

    TIM_TimeBaseInitTypeDef TIM_TimeBaseStructure;
    NVIC_InitTypeDef NVIC_InitStructure;

    TIM_TimeBaseStructure.TIM_Prescaler = 16000 - 1;
    TIM_TimeBaseStructure.TIM_Period = intervals - 1;

    TIM_TimeBaseStructure.TIM_CounterMode = TIM_CounterMode_Up;
    TIM_TimeBaseInit(TIM2, &TIM_TimeBaseStructure);

    TIM_ClearITPendingBit(TIM2, TIM_IT_Update);
    TIM_ITConfig(TIM2, TIM_IT_Update, ENABLE);
    TIM_Cmd(TIM2, ENABLE);

    NVIC_PriorityGroupConfig(NVIC_PriorityGroup_1);
    NVIC_InitStructure.NVIC_IRQChannel = TIM2_IRQn;
    NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 0;
    NVIC_InitStructure.NVIC_IRQChannelSubPriority = 1;
    NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
    NVIC_Init(&NVIC_InitStructure);
}
```

# I 실습

## 실습 1: MCU에서 PC로 문자 전송하기

- 1초에 한번씩 'a'를 전송하도록 코드 수정



# I 실습

## 실습 2: MCU - PC 데이터 송수신 (에코 프로그램. aka 메아리)

- PC 키보드로 누른 값이 프로그램에 출력되도록 구현
  - PC에서 'a' 누름. PC → MCU 'a' 전송 (MCU 관점에서 데이터 수신)
  - MCU는 'a'를 PC로 전송 (MCU 관점에서 데이터 송신)

이 코드는 실습에서 구현하였음

# I 실습

## 실습 2: MPU - PC 데이터 송수신 (에코 프로그램)

- PC 키보드로 누른 값이 프로그램에 출력되도록 구현
  - PC에서 'a' 누름. PC → MCU 'a' 전송 (MPU 관점에서 데이터 수신)
  - MCU는 'a'를 PC로 전송 (MCU 관점에서 데이터 송신)

이 코드는 실습에서 구현하였음

```
unsigned char Serial_Receive(void)
{
    unsigned char data;

    while(USART_GetFlagStatus(USART2, USART_FLAG_RXNE) == RESET);
    data = USART_ReceiveData(USART2);
    return data;
}
```

데이터 수신 중이라면, 대기

데이터 수신이 완료되면,  
data변수에 수신 값 받아 return

# 실습

## 실습 2: MCU - PC 데이터 송수신 (에코 프로그램)

```
int main()
{
    LED_init();
    TIM2_Configuration(1000);
    USART2_Configuration();

    unsigned char getdata;

    while(1)
    {
        getdata = Serial_Receive();
        Serial_Send(getdata);

        if (getdata == 'a')
        {
            GPIO_SetBits(GPIOA, GPIO_Pin_5);
        }
        else
        {
            GPIO_ResetBits(GPIOA, GPIO_Pin_5);
        }
    }
    return 0;
}
```

참고: char는 8비트. 우리는 설정에서 8비트로 정함

데이터 수신 후,  
수신 값을 송신

수신 값이 'a'면 User LED on  
'a'가 아니면 LED off

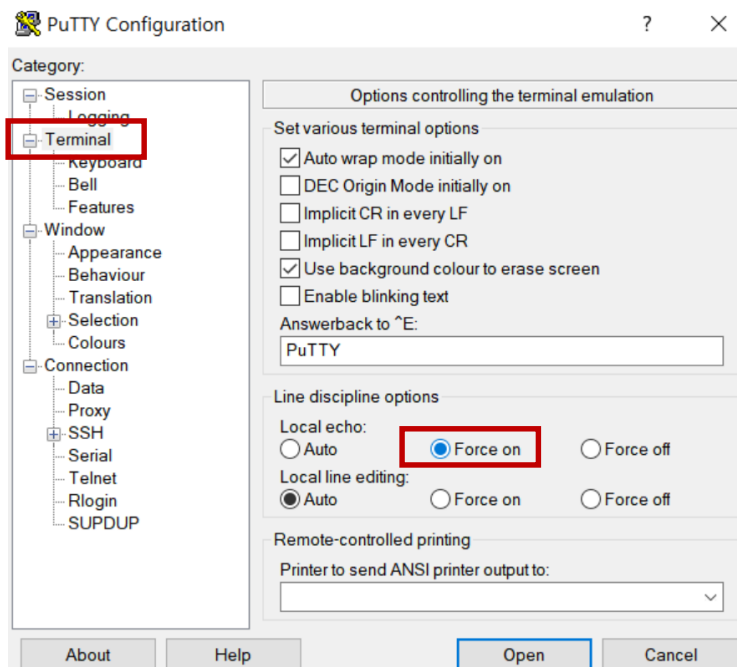
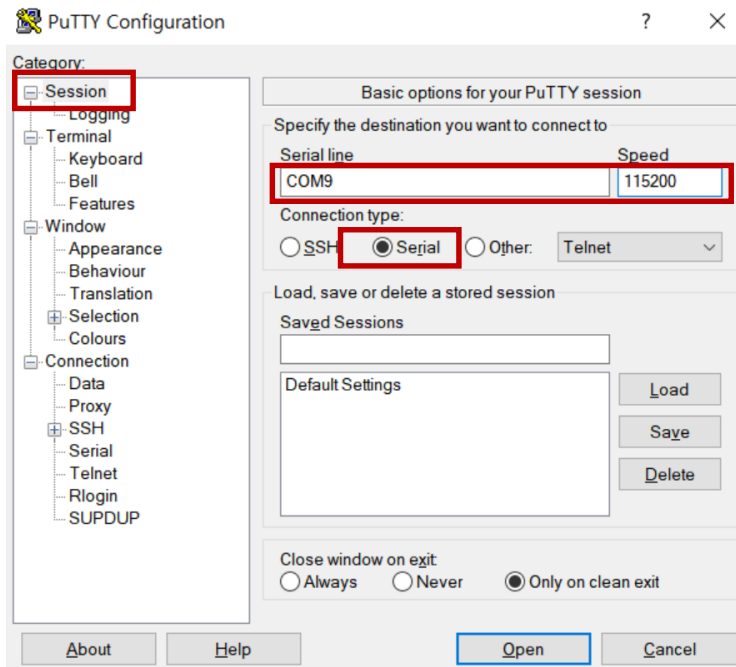
```
void TIM2_IRQHandler(void)
{
    if(TIM_GetITStatus(TIM2, TIM_IT_Update) != RESET)
    {
        //GPIO_ToggleBits(GPIOA, GPIO_Pin_5);

        // clear interrupt flag
        TIM_ClearITPendingBit(TIM2, TIM_IT_Update);
    }
}
```

# 실습

## 실습 2: MCU - PC 데이터 송수신 (에코 프로그램)

- PuTTY로 확인



# 실습

## 기타

- 문자열을 보내고 싶다면?
- For loop이용해도 되고, 포인터를 이용해도 되고
- 끝나는 지점을 알아야 함. 엔터로 합시다 = `\n`

```
char* pch;
```

```
void Serial_Send_Printf(char* pch)
{
    while(*pch != '\0')
    {
        Serial_Send(*pch);
        pch++;
    }
}
```

```
void TIM2_IRQHandler(void)
{
    if(TIM_GetITStatus(TIM2, TIM_IT_Update) != RESET)
    {
        Serial_Send_Printf("Welcome to USRAT\n");

        // clear interrupt flag
        TIM_ClearITPendingBit(TIM2, TIM_IT_Update);
    }
}
```

```
int main()
{
    LED_init();
    TIM2_Configuration(1000);
    USART2_Configuration();

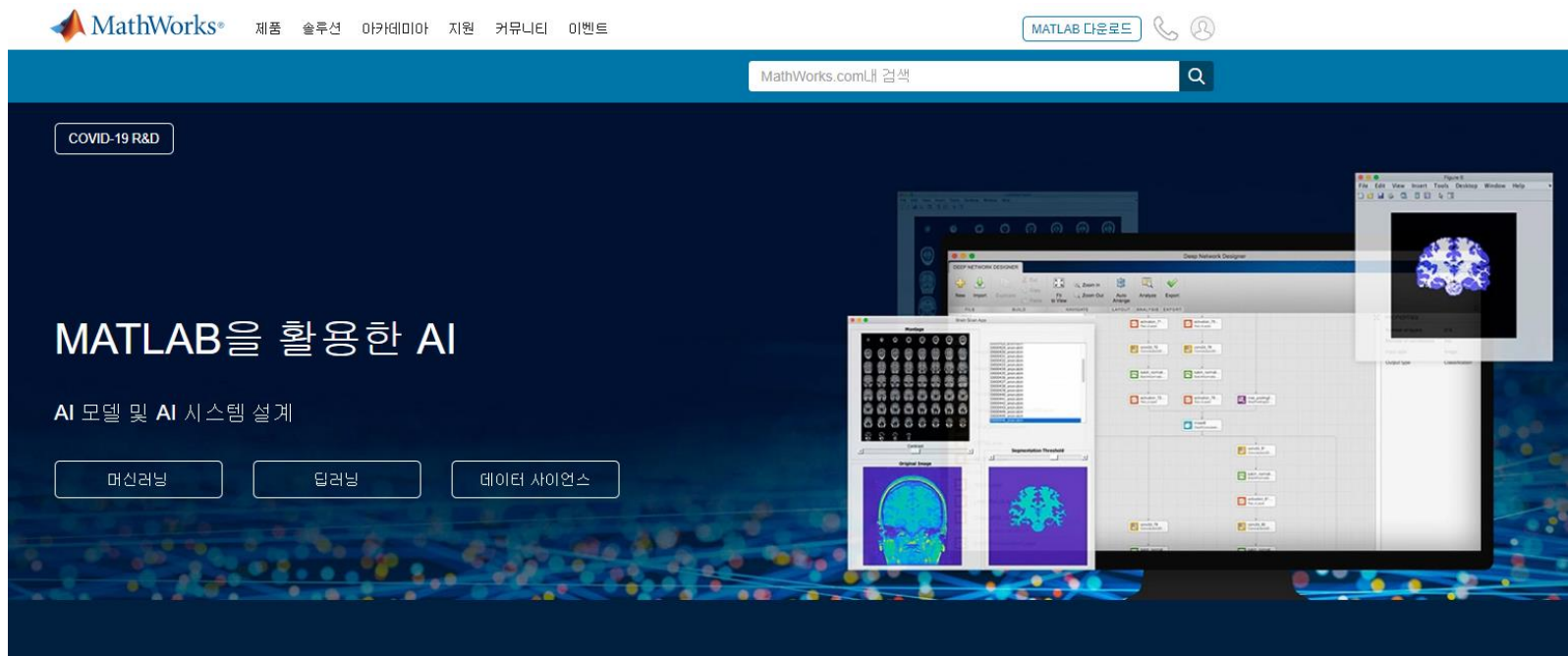
    while(1)
    {
    }

    return 0;
}
```

- USART로 이것저것 해보기
- 새 프로젝트 생성 & ADC, USART configuration 함수 추가해오기
- Matlab 설치

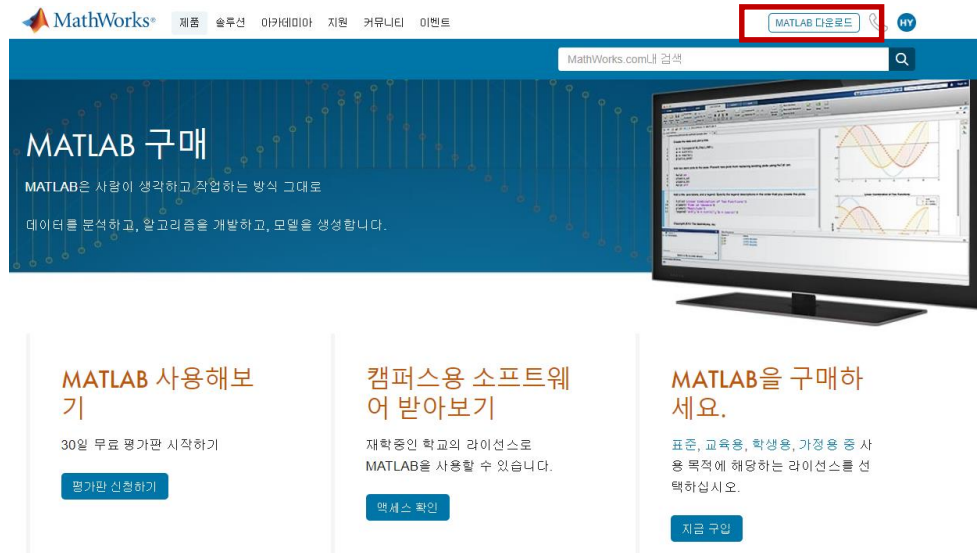
## Matlab

- <https://kr.mathworks.com/>
- 실습 노트북에 설치



## Matlab

- smu.ac.kr 계정으로 가입 후 소프트웨어 설치



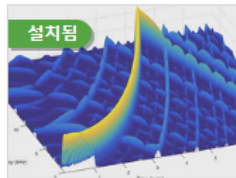
### 내 소프트웨어

라이선스	레이블	옵션	사용 용도			
40618140	MATLAB (Individual)	Total Headcount	Academic	⌵	✕	🛒



## Matlab

- Toolbox 중 <Signal Processing Toolbox>, <DSP System Toolbox>, <Statistics and Machine Learning Toolbox>, <Curve Fitting Toolbox> 반드시 설치







### Signal Processing Toolbox R2020b 작성자: MathWorks

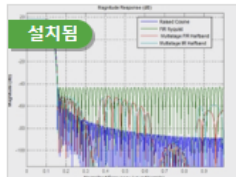


Perform **signal** processing and analysis

Signal Processing Toolbox™ provides functions and apps to analyze, preprocess, and extract features from uniformly and nonuniformly sampled **signals**. The toolbox includes tools for filter design and

-  **Signal Analyzer** - Visualize and compare multiple **signals** and spectra
-  **Signal Labeler** - Label **signal** attributes, regions, and points of interest
-  **Filter Designer** - Design filters starting with algorithm selection
-  **Window Designer** - Design and analyze spectral windows

MathWorks 툴박스



### DSP System Toolbox R2020b 작성자: MathWorks



Design and simulate streaming **signal** processing systems

DSP System Toolbox™ provides algorithms, apps, and scopes for designing, simulating, and analyzing **signal** processing systems in MATLAB® and Simulink®. You can model real-time DSP systems for

MathWorks 툴박스

**Thank you.**

