GPIO&인터럽트

마이크로프로세서

HRI 연구실

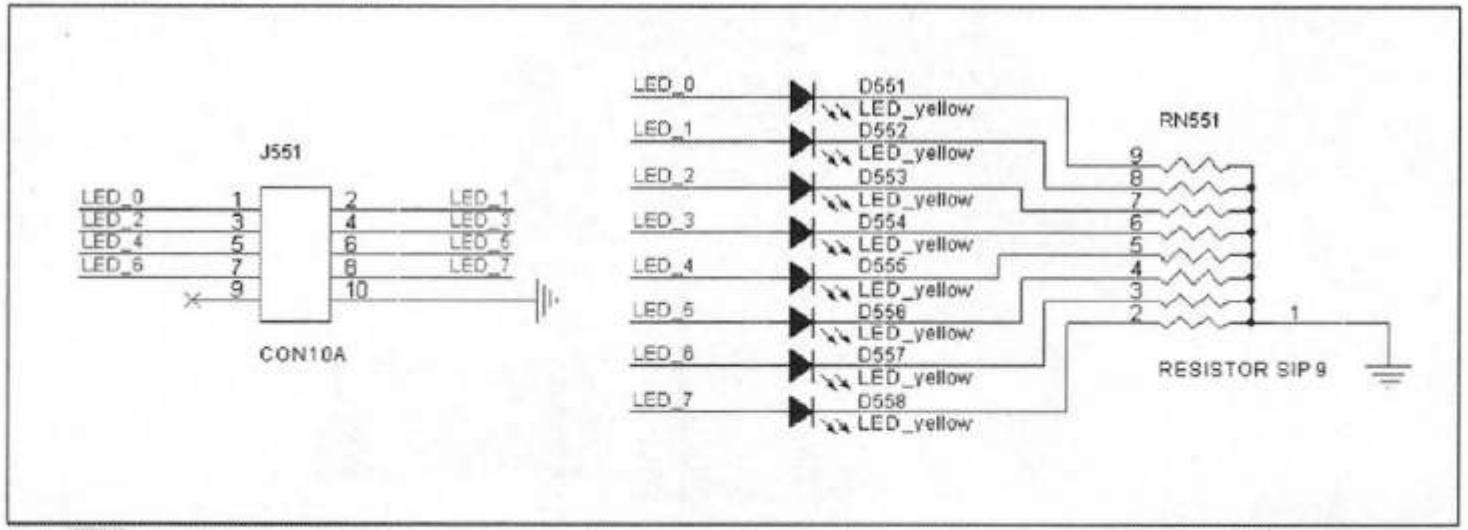
김동한

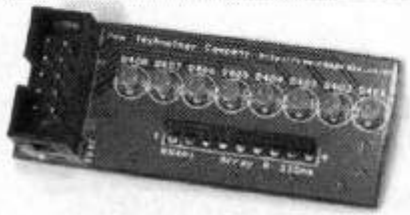
Human-Robot Interaction Laboratory

KYUNG HEE UNIVERSITY

6.1 LED 제어

6.1.1 ATmega128을 이용한 입력(push button)과 출력 LED) 포트





#1	#3	#5	#7	E9
LED 1	LED 3	LED 5	LED 8	전원 5V
#2	**#4**	**#6**	**#8**	**#10**
LED 2	LED 4	LED 6	LED 7	전원 GND

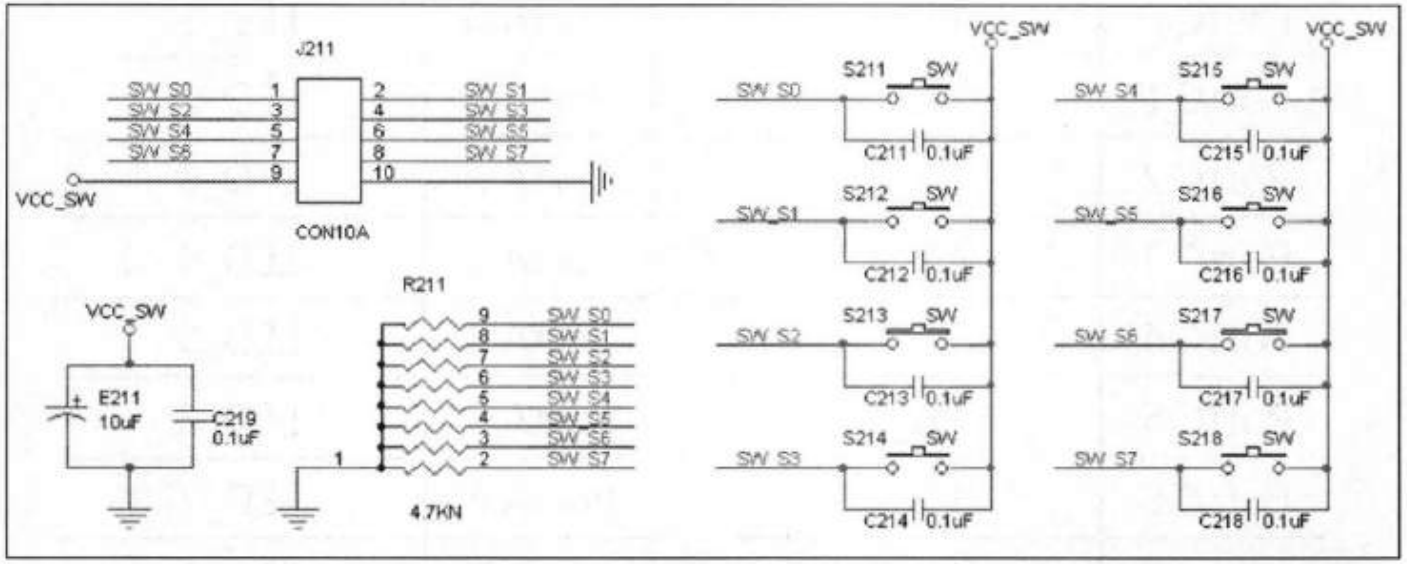
출력회로, LED모듈의 회로.

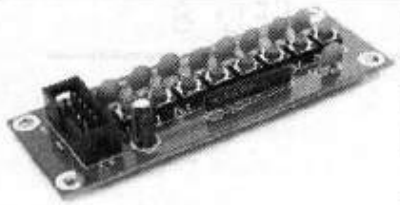
Atmega128에서 High(5V, 1)을 주어야 ON 되는 구조, 공통 GND

10 Pin, 8bit의 제어신호와 Vcc(9번) GND(10번) 으로 구성

6.1 LED 제어

6.1.1 ATmega128을 이용한 입력(push button)과 출력 LED) 포트





#1	#3	#5	#7	E9
S/W 1	S/W 3	S/W 5	S/W 8	전원 5V
#2	**#4**	**#6**	**#8**	**#10**
S/W 2	S/W 4	S/W 6	S/W 7	전원 GND

입력회로, Switch 모듈의 회로.

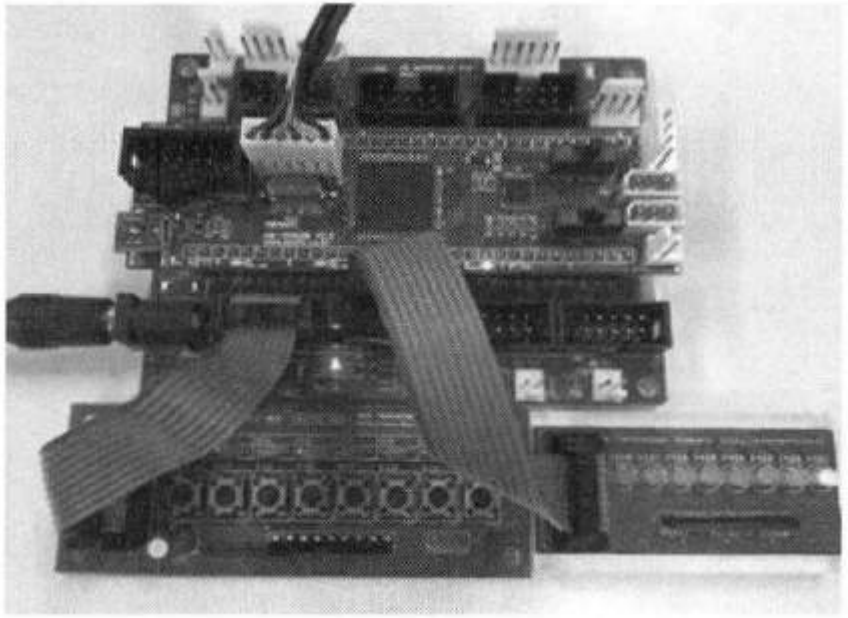
공통 GND, Pull-down 방식으로 회로를 구성

신호 입력은 평상시에 0V(Low,0)을 유지하면서 스위치가 On되면 5V(High,1)이 되는 구조.

KYUNG HEE UNIVERSITY

6.1 LED 제어

6.1.2 ATmega128 포트와 연결된 입력과 출력포트



input = Push Button (PinC)

Pin	Push Button
PortC.0	PB_1
PortC.1	PB_2
PortC.2	PB_3
PortC.3	PB_4
PortC.4	PB_5
PortC.5	PB_6
PortC.6	PB_7
PortC.7	PB_8

output = LED (PortA)

Pin	Push Button
PortA.0	LED_1
PortA.1	LED_2
PortA.2	LED_3
PortA.3	LED_4
PortA.4	LED_5
PortA.5	LED_6
PortA.6	LED_7
PortA.7	LED_8

입력장치는 ATmega128의 PORTC에| 연결하였고, 레지스터는 PINC를 사용한다.
출력장치는 ATmega128의 PORTA에 연결하였고, 레지스터는 PORTA를 사용한다

6.1 LED 제어

6.1.2 ATmega128 포트와 연결된 입력과 출력포트

포트 제어 레지스터의 종류

DDRx : 입력으로 설정할지(0) 출력으로 설정할지(1) 결정. Ex) PORTA를 출력으로 사용하고 싶다. → DDRA = 0xff

PINx : 입력상태를 읽는 레지스터. AVR에서 스위치 입력이나 센서 입력을 판단 할 때 사용.

PORTx : 입출력 설정 레지스터, DDRx에서 입력으로 설정했는지 출력으로 설정 했는지에 따라 설정할 수 있는 것이 다름. 아래의 4가지 경우가 존재함.

입출력설정	PORTx = 비트값 1	PORTx = 비트값 0
DDRx의 비트가 1(출력) 설정되을시	해당 포트에 VCC(5V)를 출력합니다.	해당 포트에 GND(0V)를 출력합니다.
DDRx의 비트가 0(입력) 설정되을시	해당 포트상태를 Pullup(풀업) 입력되기 전상태가 1인 상태로 만듭니다.	해당 포트상태를 Tri-State 입력되기 전상태가 0인 상태로 만듭니다.

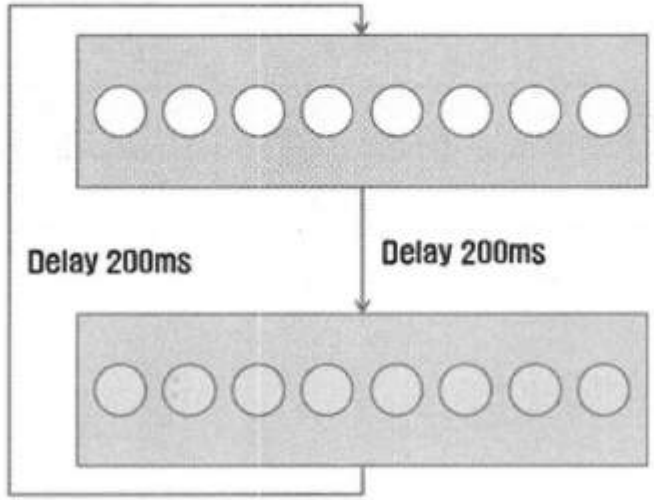
6.1 LED 제어

실습 1. 아래 그림과 같이 LED On 시켜라.

PORTA에 LED를 연결하여 8개의 LED가 0.2초 간격으로 ON, OFF되도록 프로그램을 작성하시오.

여기에서 timer는 delay header file를 이용하여 구성하고 프로그램에 적용하시오.

LED의 연결방법은 GND공통으로 ATmega128에서 High(5V)신호가 발생할 때에 ON된다.



해설

Delay function 이해하기

(1) 의미 : Delay는 프로그램이 작동하는 동안 정해진 시간 동안 지연하는 함수이다.

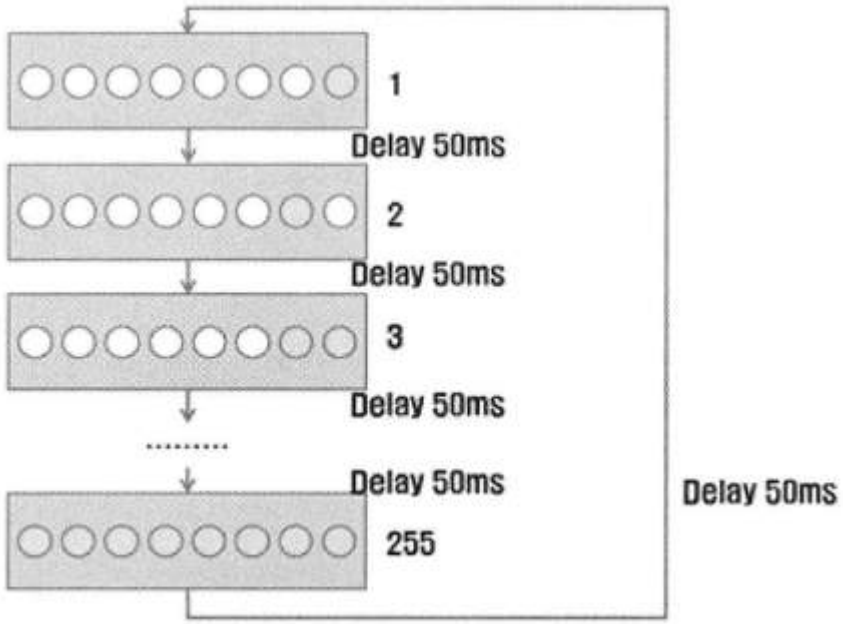
(2) 구조 : #include <delay.h> delay_ms(200); ms는 1/000초로서 200ms이므로 0.2초가 된다.

```
#include <mega128.h> //header file include
#include <delay.h> //use delay function
void main(void)
{
    PORTA = 0x00;
    DDRA = 0xff;
    while (1)
    {
        PORTA = 0x00;
        delay_ms(200);
        PORTA = 0xff;
        delay_ms(200);
    }
}
```

6.1 LED 제어

실습 2. 50ms마다 +1씩을 개수하여 255까지 증가하는 2진 카운터 만들어 LED에 표시하기

(1) LED 8개를 이용하여 숫자 카운터를 할 수 있는 프로그램을 작성한다.
(2) 8 비트는 255까지 카운터를 할 수 있고 실제 2진수의 형태로 지속적으로 LED가 ON되도록 프로그램을 작성하는데 50ms간격으로 구성한다.
(3) 2진수의 형태는 1(0b00000001),2(0b00000010),3(0b00000011), . . . , 255(0b11111111)이 된다.



해설

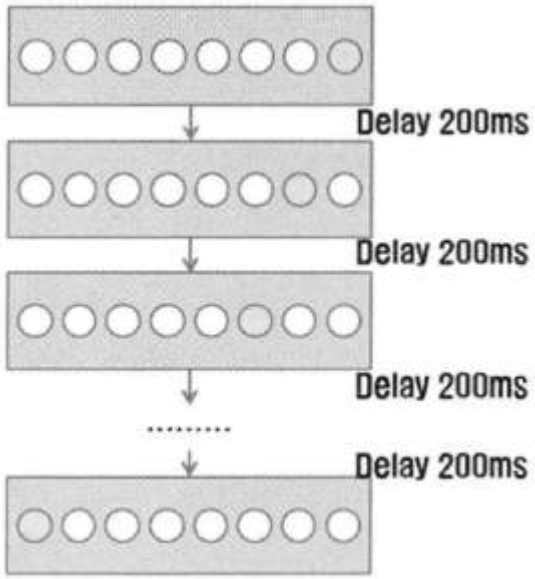
(1) 숫자의 증감을 하기 위해서는 무한루프 안에서 50ms씩 증가함에 따라 숫자가 하나씩 증가해야 한다. 그러므로 i=i+1과 같은 의미의 "i++"이란 증감연산자를 사용한다.
(2) "PORTA = PORTA +1;" 과 "PORTA++;"같은 증감연산자를 사용한다.

```
#include <mega128.h> //header file include
#include <delay.h>

void main(void)
{
        PORTA=0x00;
        DDRA=0xff;

        while (1)
        {
                PORTA++;
                delay_ms(50);
        }
}
```

6.1 LED 제어

실습 3. 20ms 마다 LED가 하나씩 옆으로 이동하는 프로그램을 작성하시오.

PORTA에 연결되어 있는 LED 8개를 200ms간격으로 On-Off되면서 이동하는 프로그램을 작성하시오.
PORTA = 0b00000001; "1"을 하나씩 이동하면서 0b10000000까지 이동하는 프로그램 작성.
"PORTA << 1;" 의미는 이동 연산자 "<<"은 왼쪽으로 1비트씩 이동하는 프로그램 작성.
위 두 가지의 형태로 프로그램을 작성한다.
예1 : 0b00000001 << 2 = 0b00000100, 예2 : 0x40 << 1 = 0x80 , 예3 : 0x01 << 3 = 0x08



해설

```
#include <mega128.h> //header file include
#include <delay.h> //use delay function

void main(void)
{
    PORTA = 0x00;
    DDRA = 0xff;
    PORTA = 0x00;

    while (1)
    {
        PORTA = 0b00000001;
        delay_ms(200);
        .
        .
        .
        PORTA = 0b10000000;
        delay_ms(200);

    }
}
```

```
#include <mega128.h> //header file include
#include <delay.h> //use delay function
void main(void)
{
    PORTA = 0x00;
    DDRA = 0xff;
    PORTA = 0x01; // 초기에 첫번째 자리에 LED On
    while (1)
    {
        PORTA = PORTC << 1;
        delay_ms(200);
    }
}
```

KYUNG HEE UNIVERSITY

6.1 LED 제어

[과제1] 스위치를 누를 때마다 비트별로 일대일 대응에 의한 LED가 On될 수 있도록 프로그램 하시오.

PORTA에 LED, PORTC에 switch를 연결하여 스위치 1번을 누르면 1번 LED가 ON되는 스위치에 따른 순차적인 LED On이 될 수 있도록 프로그램을 작성하시오.

"PORTA = PINC; " 현재의 레지스터는 8 비트를 병렬로 동시에 사용하는 것으로 PINC의 입력신호 변화에 따라 PORTA의 출력신호가 1:1로 변환한다는 뜻이다.

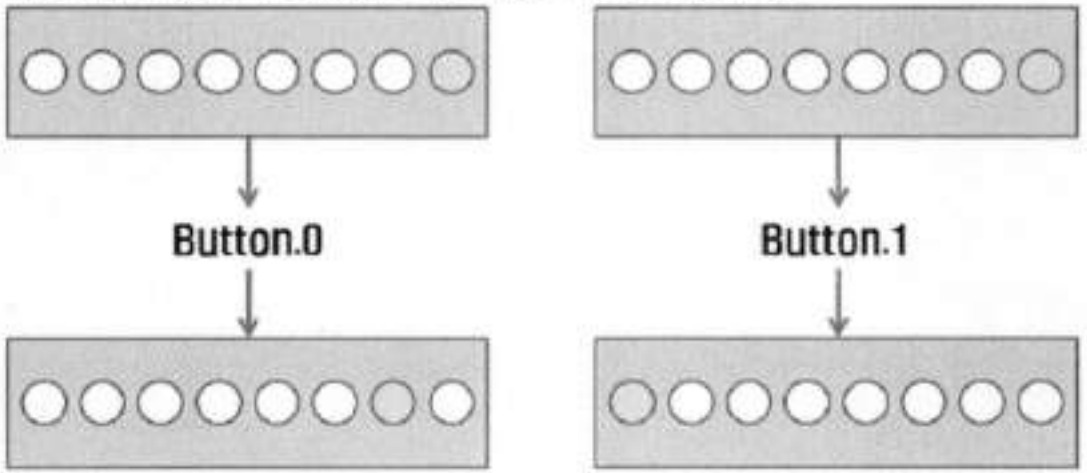
"PORTA.0 = PINC.0;" → 한 비트씩 같다는 뜻으로 만약 현재의 비트를 쓸 경우에는 8개 모두를 사용해야 한다.

포트	7	6	5	4	3	2	1	0
입력	PINC.7	PINC.6	PINC.5	PINC.4	PINC.3	PINC.2	PINC.1	PINC.0
	PINC							
출력	PortA.7	PortA.6	PortA.5	PortA.4	PortA.3	PortA.2	PortA.1	PortA.0
	PORTA							

6.1 LED 제어

[과제2] 스위치를 누를 때마다 다음과 같이 LED가 On될 수 있도록 프로그램 하시오.

Switch 1(PINC.0)번을 누르면 8개의 LED가 좌측으로 이동한다.
Switch 2(PINC.1)번을 누르면 8개의 LED가 우측으로 이동한다.
Delay time은 200ms로 한다.
연산자는 이동연산자와 증감연산자를 사용하여 프로그램 하시오.



6.1 LED 제어

6.1.3 ATmega128을 이용한 LED 제어에서 산술연산자 이해

산술연산자는 사칙연산, 나머지, 이동연산자, 비트논리 연산자로 나누어 볼 수 있다.
사칙연산은 +, - , / ,*로서 bit와 byte끼리 같은 비트 수일 경우는 가능하다.
이동연산자는 1 비트씩 이동한다. << : 왼쪽으로 1비트, >> : 오른쪽으로 1비트
논리연산자는 And, Or, Xor, Not로 나누어 논리연산을 한다.

연산자	설명
+	덧셈
-	뺄셈
*	곱셈
/	나눗셈
%	나머지
<<	왼쪽으로 쉬프트
>>	오른쪽으로 쉬프트
&	and 연산
|	or 연산
^	xor 연산
~	not 연산

1. 사칙 연산자 예

(1) PORTA = 2*5; ➔ 10 0b00001010으로 LED On

(2) PORTA = 13/2 ➔ 6.5 → 6으로 판단 LED On

2. 이동연산자 예

(1) PORTA = 0x32 >>3; → (0x06)

(2) PORTA = 0b10101011 << 5; →(0b01100000)

3. 논리연산자 예

(1) PORTA = 0x32 & 0x54; → (0x86)

(2) PORTA.5 = 1 | 0; → (0)

6.1 LED 제어

실습 6. 진수에 대한 프로그램 (2진수로 변환하여 LED로 나타내기)

- 스위치를 이용하여 각 스위치에 해당하는 문제를 출력

정수형 변수 A=23 일 때, LED에 2 bit 신호로 출력하시오.
정수형 변수 B=18 일 때, LED에 2 bit 신호로 출력하시오.
정수형 변수 C= 0x18 일 때, LED에 2 bit 신호로 출력하시오.
정수형 변수 D= 0x32 일 때, LED에 2 bit 신호로 출력하시오.

10진수와 2진수 변환하기 학습 및 사칙연산에 대한 연산자 익히기

포트	8 bit	7bit	6bit	5bit	4bit	3bit	2bit	1bit
십진값	128	64	32	16	8	4	2	1
비트값	0	0	0	1	0	0	1	0

예) B = 18 2진수 변환 (00010010) → 1*16 + 2*1 = 18
B = 18을 PORTA에 입력시켜 LED에서의 출력을 확인한다.
예) C = 0x18 2진수 변환 (0001 1000) → 1*16 + 1*8 = 24
C = 24을 PORTA에 입력시켜 LED에서의 출력을 확인한다.

```
#include <mega128.h>
#include <delay.h>

void main(void)
{

    /*A = 23;
    B = 18;
    C = 0x18;
    D = 0x32; */

    PORTA = 0x00;
    DDRA = 0xff;
    DDRC =0x00;


while (1)
    {
        if(PINC.0 == 1)
            PORTA = 0b00010111;
        else if(PINC.1 ==1)
            PORTA = 0b00001010;
        else if(PINC.2 == 1)
            PORTA = 0b00011000;
        else if(PINC.3 == 1)
            PORTA = 0b00110010;

        else
         {
            PORTA = 0x00;
         }
    }
}
```

6.1 LED 제어

실습 7. 사칙 연산자에 대한 프로그램을 작성하시오.

정수형 변수 A=34, B=18의 값을 받아서 더하여 PORTA 의 LED에 2 bit 신호로 출력하시오.
정수형 변수 A=34, B=18의 값을 받아서 뺄셈하여 PORTA 의 LED에 2 bit 신호로 출력하시오.
정수형 변수 A=34, B=18의 값을 받아서 곱셈하여 PORTA 의 LED에 2 bit 신호로 출력하시오.
정수형 변수 A=34, B=18의 값을 받아서 나눗셈하여 PORTA 의 LED에 2 bit 신호로 출력하시오.

10진수와 2진수 변환하기 학습 및 사칙연산에 대한 연산자 익히기

연산자	설명
+	덧셈
-	뺄셈
*	곱셈
/	나눗셈

예) A = 34 2진수 변환 (100010)
B = 18 2진수 변환 (10010)
C=A+B D=A-B E=A*B F=A/B

위 네 개의 사칙연산의 값을 PORTA에 입력하여 LED에서의 출력을 확인한다.

6.1 LED 제어

[과제3] 이동 연산자와 비트 논리 연산자에 대한 프로그램을 작성하시오.

A=0xf8을 오른쪽으로 5비트만큼 이동하여 결과 값을 PORTA의 LED에 표현하시오.
B=112를 왼쪽으로 3비트만큼 이동하여 결과 값을 PORTA의 LED에 표현하시오.
A=0xf8과 B=112를 "A&B" AND 논리를 이용하여 결과 값을 PORTA의 LED에 표현하시오.
A=0xf8과 B=112를 "A|B" OR 논리를 이용하여 결과 값을 PORTA의 LED에 표현하시오.
A=0xf8과 B=112를 "A^B" XOR 논리를 이용하여 결과 값을 PORTA의 LED에 표현하시오.
A=0xf8을 "~A" NOT 논리를 이용하여 결과 값을 PORTA의 LED에 표현하시오.

이동연산자(<< ,>>)와 논리연산자에 대하여 이해하자

예1) A=0xf8; 우선 2진수로 변환 (A=0b11111000)
>>5(5비트 오른쪽으로 이동) A=00000111;
예2) B=112 우선 2진수로 변환 (A=0b01100110)
<<3 (3비트 왼쪽으로 이동) A=00110000;

포트	8 bit	7bit	6bit	5bit	4bit	3bit	2bit	1bit
십진값	128	64	32	16	8	4	2	1
비트값	0	1	1	0	0	1	1	0

6.1 LED 제어

6.1.4. 조건 연산자와 연산문을 이용한 LED 제어

1) 조건 연산문 어떤 특정 조건의 만족 여부를 판별할 때 사용하며, 판별 후에 특정 행위를 결정할 때 사용한다.
   ① 만약 조건이 참이면, 참에 해당하는 특정 행위를 지음 (if 문)
      만약 조건이 거짓이면, 거짓에 해당하는 특정 행위를 지음
   ② 조건 값에 해당하는 특정 행위를 지음 (switch case 문)
      종류:
      if 문 switch 문 case :
      if ~ else 문
      if ~ else if ~ else 문
2) 조건 연산자
   : 비트를 처리할 경우에 판단이나 처리를 해야할 경우에 사용한다.
   ① 만약 입력신호를 판단할 경우에
      if(PINC.0 ==1) PINC.0의 입력신호가 1(5V, High)이면 조건문이 실행된다.
   ② 입력 데이터의 크기를 한정할 경우에
      if(A >= 100) → A가 100보다 큰 경우에 조건문이 실행된다.

연산자	설명
==	같다
!=	같지않다
<, <=	작다
>, >=	크다
&&	and 비교
||	or 비교
!	not 비교

6.1 LED 제어

실습 9. A=5, B=6일 때, 관계 연산자를 이용하여 참, 거짓을 판단 후 LED에 출력 하시오.

관련이론

A=5, B=6일 때 관계 연산자를 이용하여 참, 거짓을 판단한다.
(참 = 1, High / 거짓 = 0, Low)

연산자	설명	상태	설명
==	같다	LED_1 (==)	같다
!=	같지않다	LED_2 (!=)	같지 않다
<, <=	작다	LED_3 (<)	~보다 작다
		LED_4 (<=)	~보다 작거나 같다
>, >=	크다	LED_5 (>)	~보다 크다
		LED_6 (>=)	~보다 크거나 같다

실험 결과

LED_1 : 거짓, 0, Low, 꺼짐
LED_2 : 참, 1, High, 켜짐
LED_3 : 참, 1, High, 켜짐
LED_4 : 참, 1, High, 켜짐
LED_5 : 거짓, 0, Low, 꺼짐
LED_6 : 거짓, 0, Low, 꺼짐

프로그램

```
#include <mega128.h>

unsigned int A, B;

void main(void)
{
   PORTA=0x00;
   DDRA=0xff;

   while (1)
   {
      A = 5;
      B = 6;
      PORTA.0 = (A==B);
      PORTA.1 = (A!=B);
      PORTA.2 = (A<B);
      PORTA.3 = (A<=B);
      PORTA.4 = (A>B);
      PORTA.5 = (A>=B);
   }
}
```

6.1 LED 제어

실습 10. IF 제어문을 이용하여 스위치를 LED가 On될 수 있도록 프로그램 하시오.

Button.0 -> LED.0
Button.1 -> LED.1
Button.2 -> LED.2
Button.3 -> LED.3
Button.4 -> LED.4
Button.5 -> LED.5
Button.6 -> LED.6
Button.7 -> LED.7

If(PINC.0 == 1) PORTA.0 = 1;

의미 : PINC에 1번째 자리에 맞는 스위치 신호가 high 이면(if), PORTA의 1번째 자리에 맞는 LED를 발광한다는 의미이다.

```
void main(void)
{
   DDRA=0xFF;
   PORTA = 0xFF;
   DDRC = 0x00;
   PORTC = 0x00;
   while (1)
   {
      PORTA = 0;
      if( PINC.0 == 1 ) PORTA.0 = 1;
      if( PINC.1 == 1 ) PORTA.1 = 1;
      if( PINC.2 == 1 ) PORTA.2 = 1;
     if( PINC.3 == 1 ) PORTA.3 = 1;
   }
}
```

6.1 LED 제어

[과제4] Switch 문을 이용하여 스위치를 누를 때 마다 다음과 같이 LED가 On될 수 있도록 프로그램 하시오.

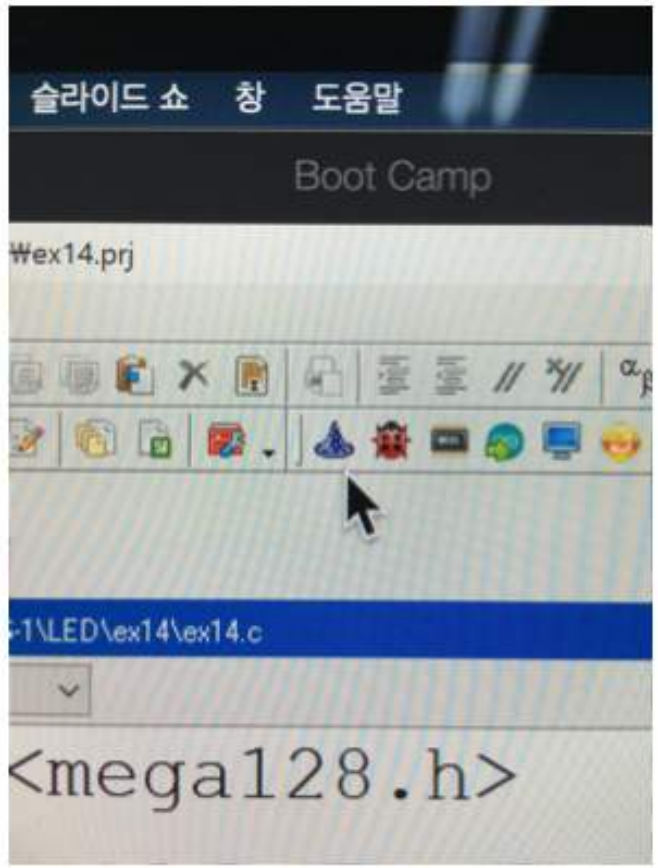
Button.0 -> LED.0
Button.1 -> LED.1
Button.2 -> LED.2
Button.3 -> LED.3
Button.4 -> LED.4
Button.5 -> LED.5
Button.6 -> LED.6
Button.7 -> LED.7

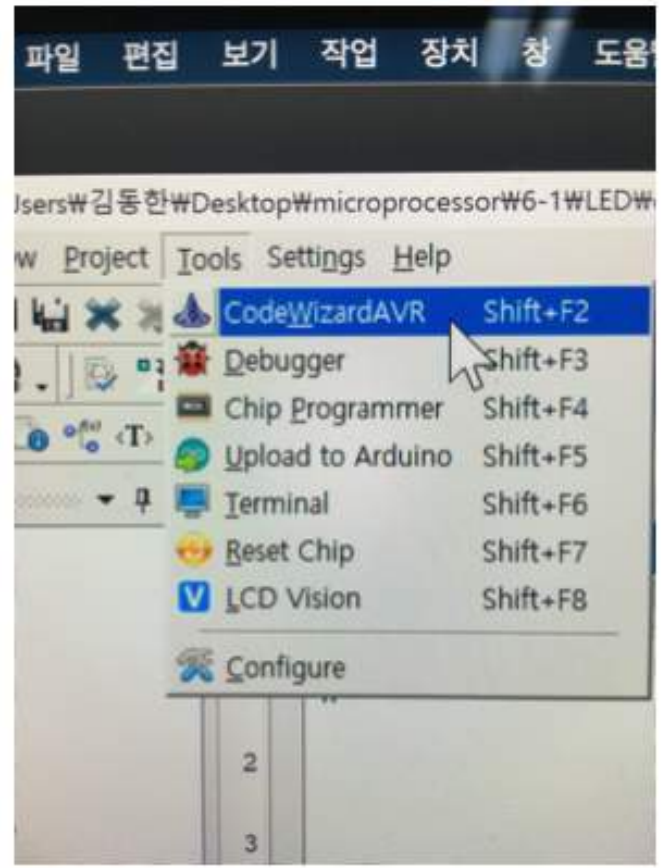
Switch case

의미: 정수를 연산문의 값에 대입하여 사용하는 복수 분기 선택문이다.
Switch(idx) idx값은 PINC에 값으로 대입, switch 문에 값 PINC의 스위치 선택값을 보고 case 값으로 jump

중요. Codewizard 사용법

실행하는 방법

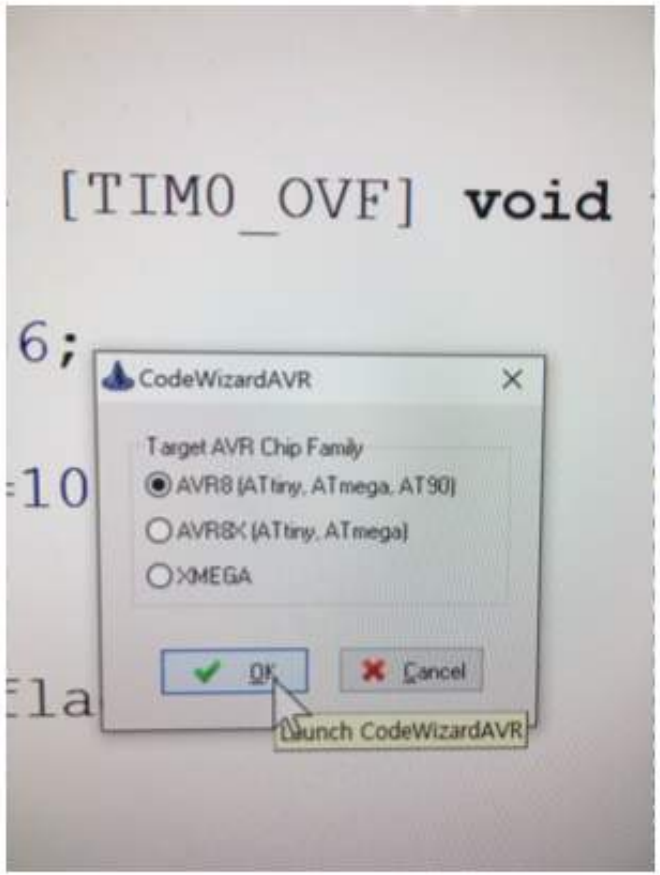


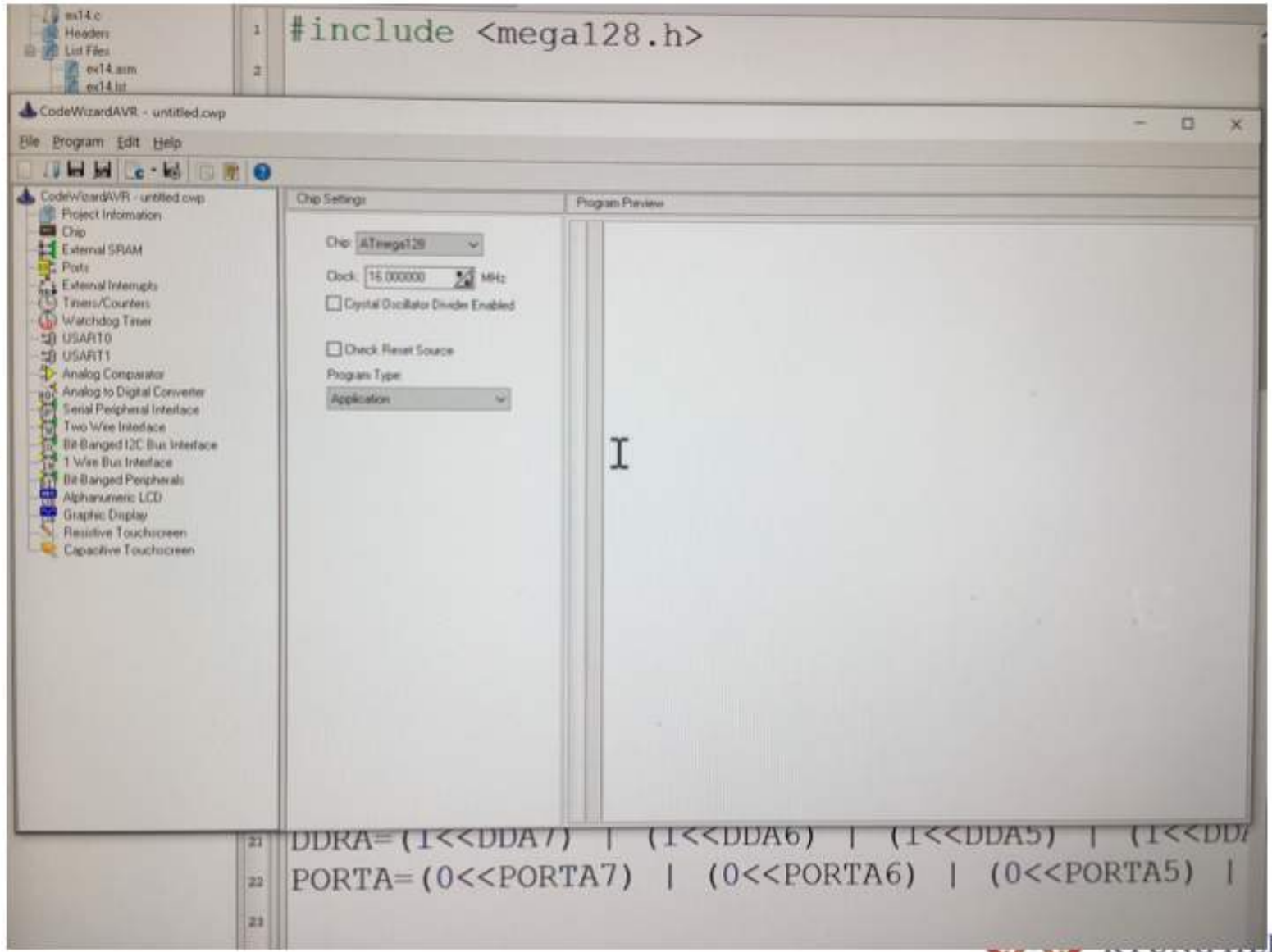


KYUNG HEE UNIVERSITY

중요. Codewizard 사용법

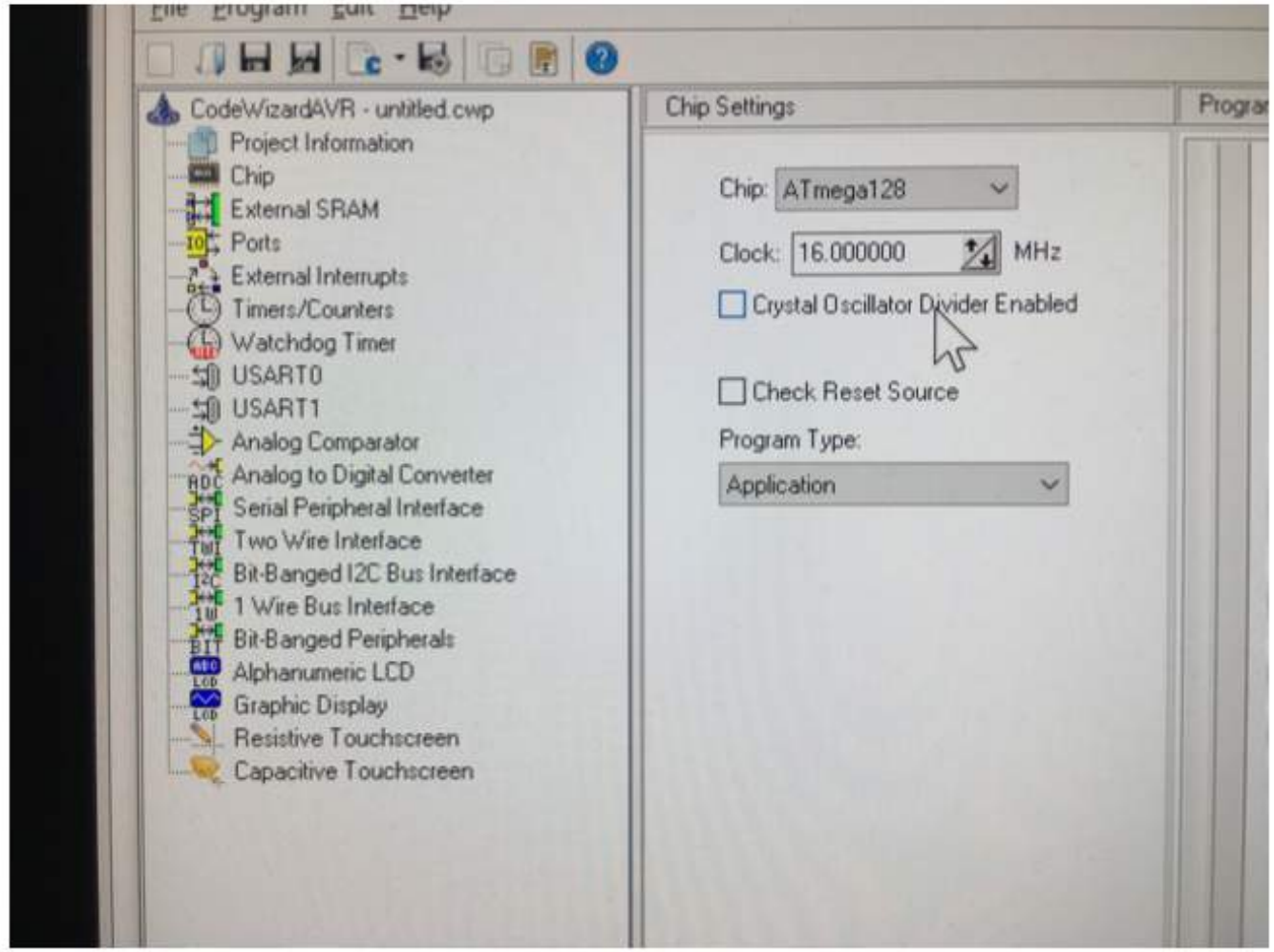
Chip 선택





중요. Codewizard 사용법

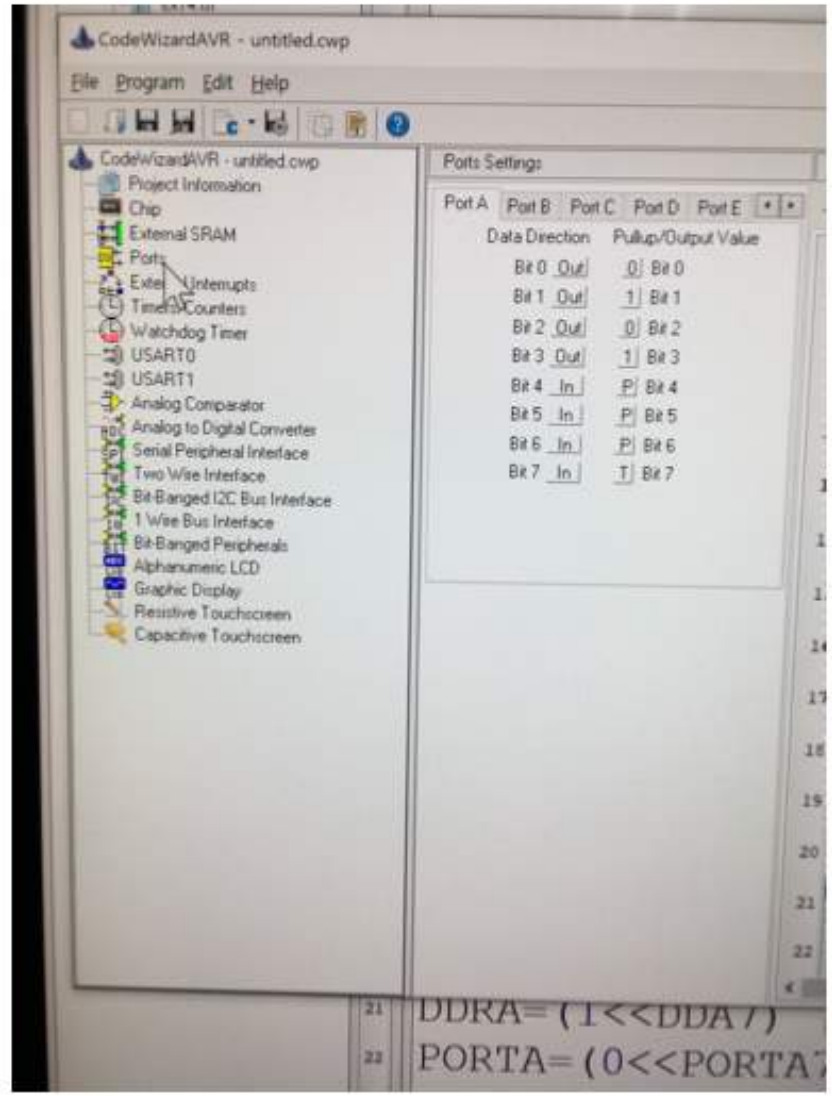
Chip 세팅



KYUNG HEE UNIVERSITY

중요. Codewizard 사용법

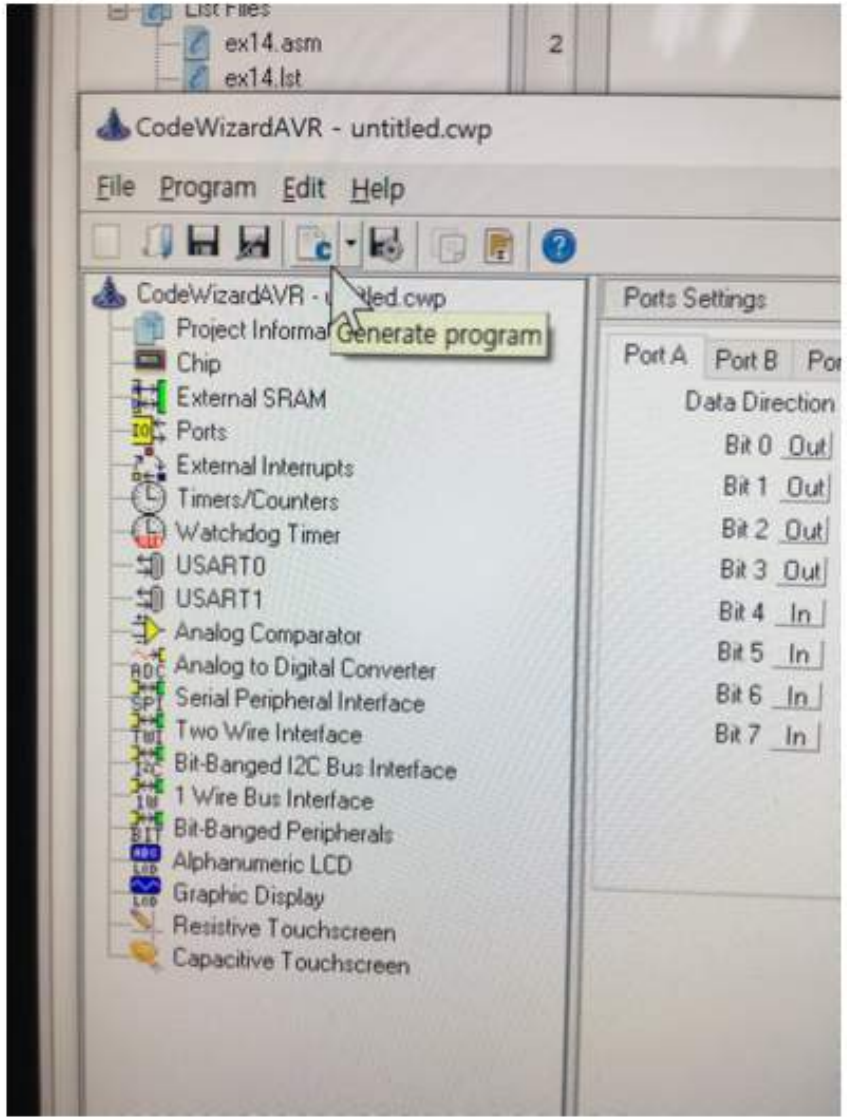
PORT 세팅 예제

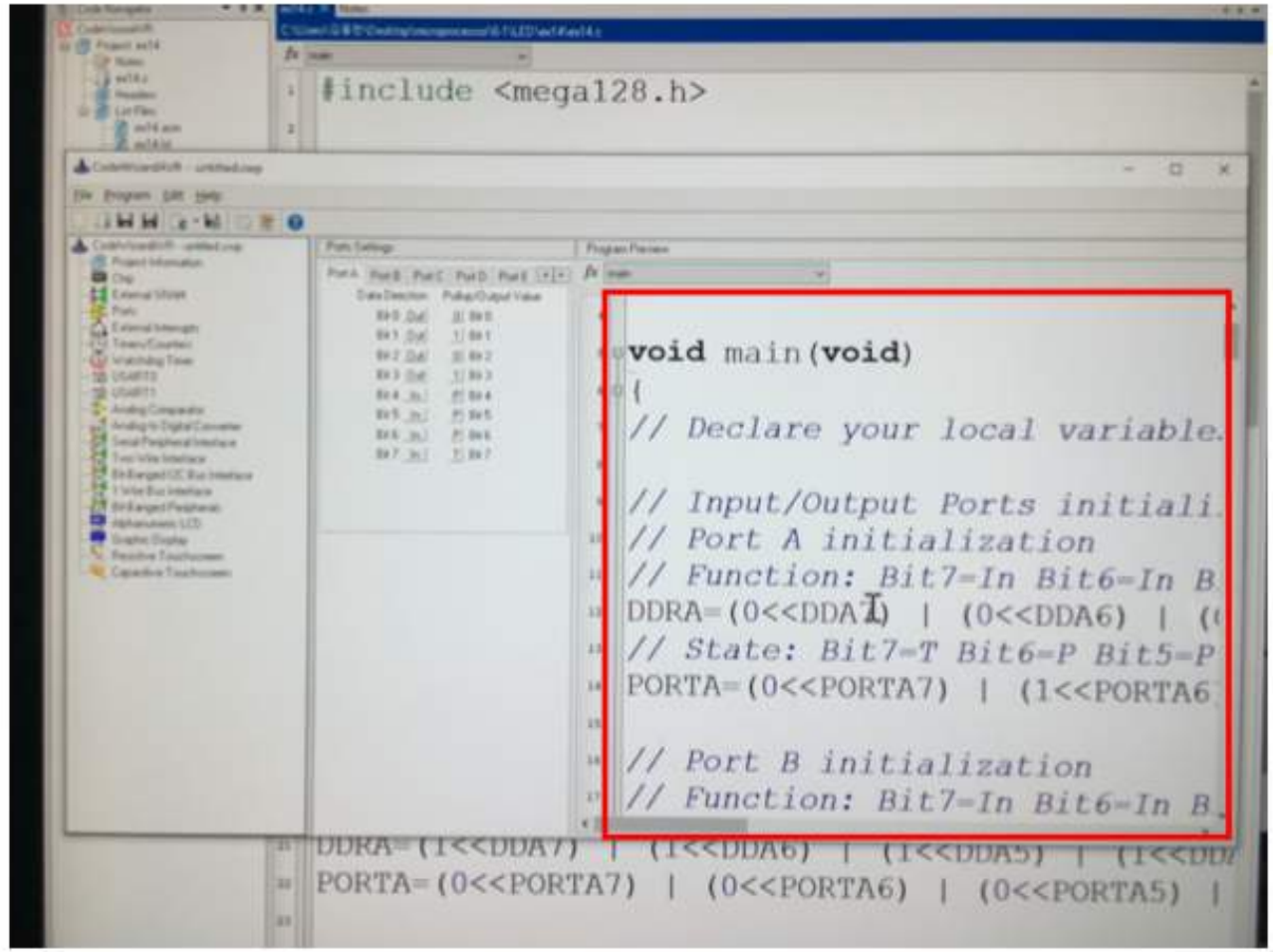


KYUNG HEE UNIVERSITY

중요. Codewizard 사용법

Code generation

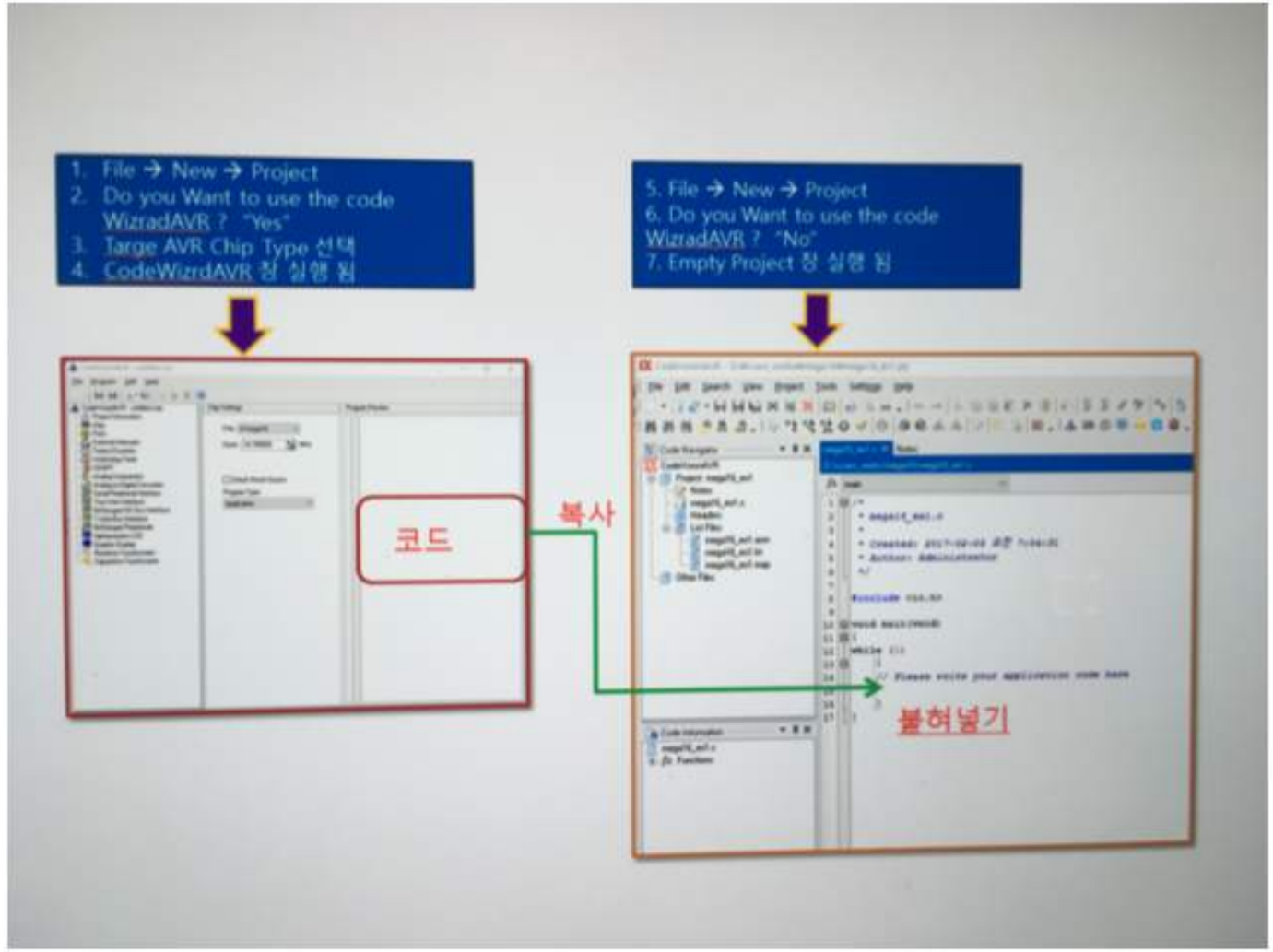




KYUNG HEE UNIVERSITY

중요. Codewizard 사용법

Code generation 후 활용법



KYUNG HEE UNIVERSITY