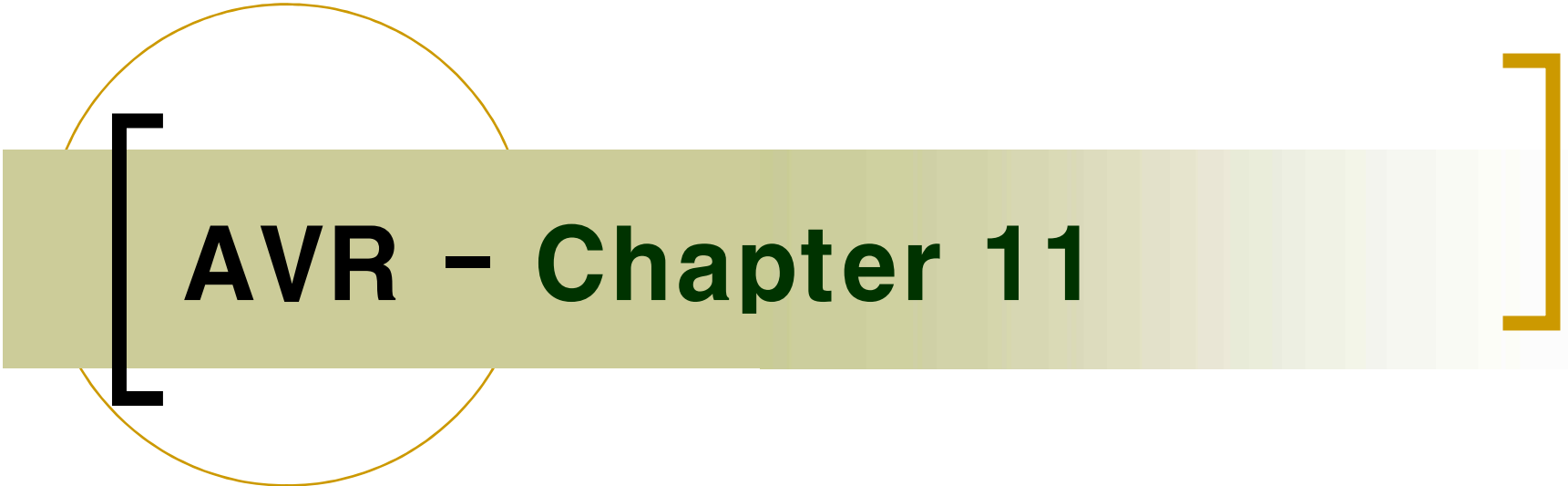




AVR - Chapter 11

류 대 우

davidryu@newtc.co.kr

[SERVO 의 종류



- **Analog servo**

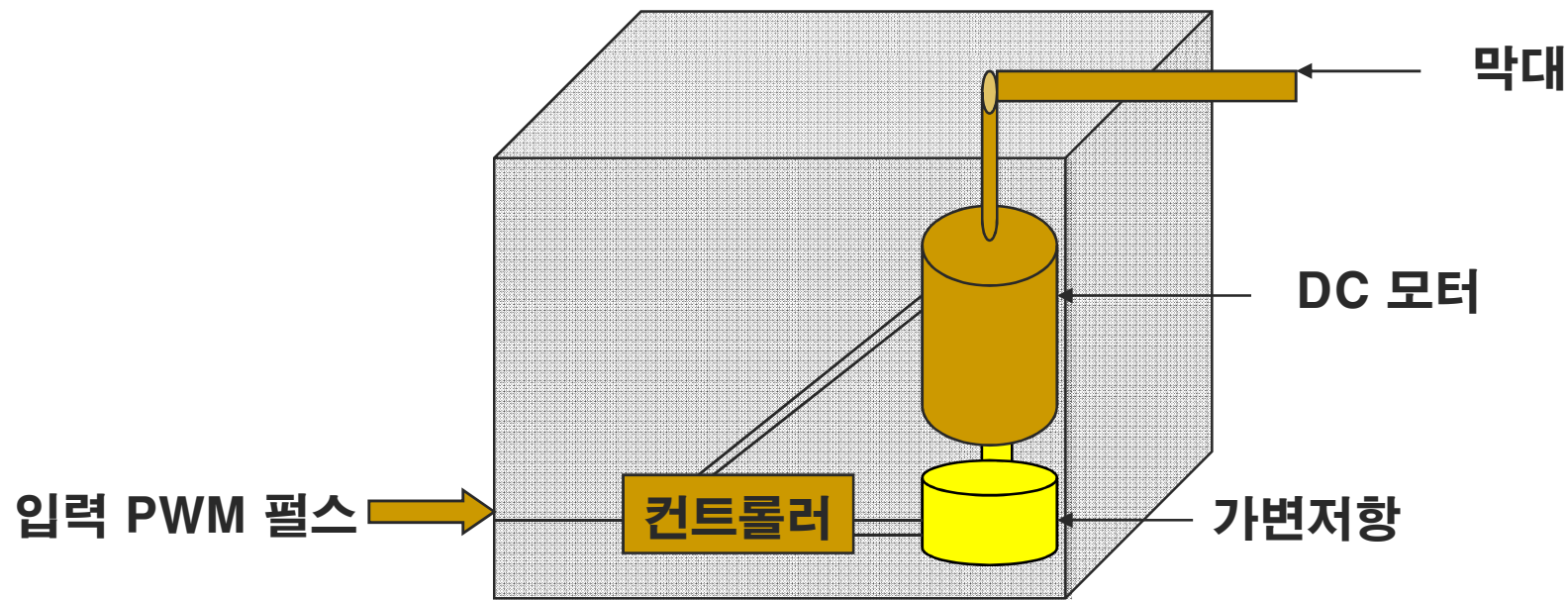
- 원하는 각도에서 servo를 멈춘 채로 고정하려면, PWM 신호를 연속해서 주기적으로 계속 주어야 한다.

- **Digital servo**

- 한 번 원하는 각도의 PWM 신호를 주면 다시 주지 않아도, 다음 신호가 있을 때까지 그 각도를 유지하고 있다.
- 최근 로봇전용으로 RS-232C, RS-422 등을 지원하는 servo도 등장하고 있는 추세다.

[아날로그 SERVO의 제작 원리]

- PWM 펄스를 컨트롤러의 A/D 핀으로 입력하면, 컨트롤러는 전압을 재고, 기 전압 만큼 가변저항에 연결된 전압을 재서 비교하여



[아날로그 SERVO의 제작 원리]

- PWM 펄스 입력 전압만큼 가변저항을 바꾸는 쪽으로 DC 모터를 돌리게 되고, 이것은 PWM 펄스와 가변저항 값이 같아질 때까지 돌리게 된다.
- 이 때, 미세한 각도를 알기 위하여 기어비 조절을 잘 하여야 하며, 기어비가 SERVO의 미세 각도 및 속도도 조절하게 된다.
- SERVO용 전용 컨트롤러를 이용하면, 보다 쉽게 SERVO를 제작할 수 있게 된다.

[디지털 SERVO의 제작 원리]

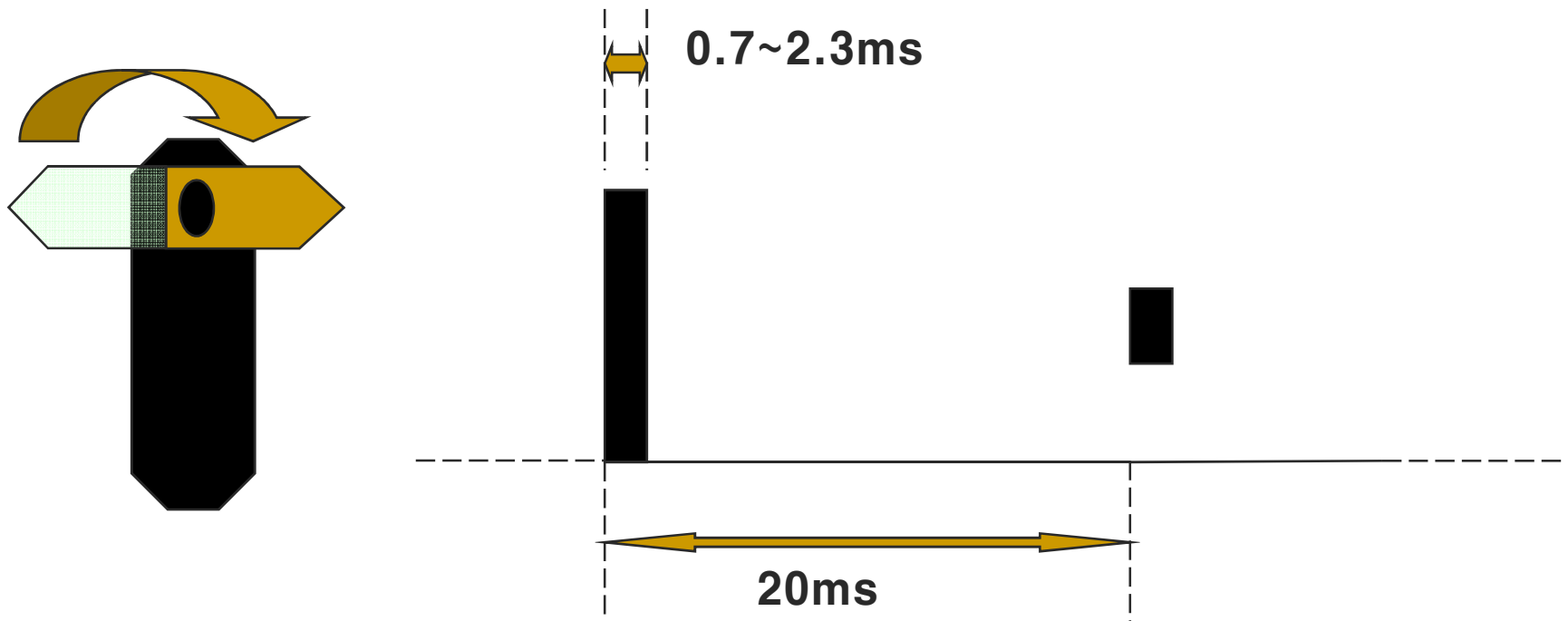
- 아날로그 SERVO와 비슷하지만,
- A/D 포트에서 받아들이는 PWM 신호에서 SERVO가 받아들일 수 있는 범위 내의 PWM 신호가 들어오는지를 판단한 후,
- 들어올 경우:
 - 가변 저항을 그 PWM 신호의 값과 일치 시까지 DC 모터를 돌린다.
- 안 들어올 경우
 - 바로 전에 들어왔던 PWM 신호의 값을 현재 값으로 인식하고, DC 모터를 돌린 상태로 모터를 유지한다.

[디지털 SERVO의 제작 원리]

- RS-232C, RS-422 등으로 모터를 제어할 수 있도록, 명령어 프로토콜을 지정하여 넣고, 명령에 따라서 지정된 각도로 이동될 수 있도록 프로토콜을 제작한다.
- 각 모터마다 고유한 번호를 부여한 후, 모터제조 시에 EEPROM에 저장하고, 라벨에 기록한다.
- Example :
 - C021,1800 : 21번 모터를 180.0도로 이동하라.
- 모터마다 약간씩의 보정을 위하여, 모터 보정용 장치를 만들고, 모터를 물린 후에 보정하여 돌린 후, 모터에 있는 EEPROM에 값을 저장하여 전원 연결시의 초기값을 모터에 전원이 들어가면 읽어 들여 그 위치로 움직인다.

[서버모터의 제어 방법]

- 서보는 20ms 주기로 0.7~2.3ms만 ON을 주면 ON 주는 시간에 따라 -90도~+90도까지 위치가 바뀐 채로, 멈추어 있게 된다.

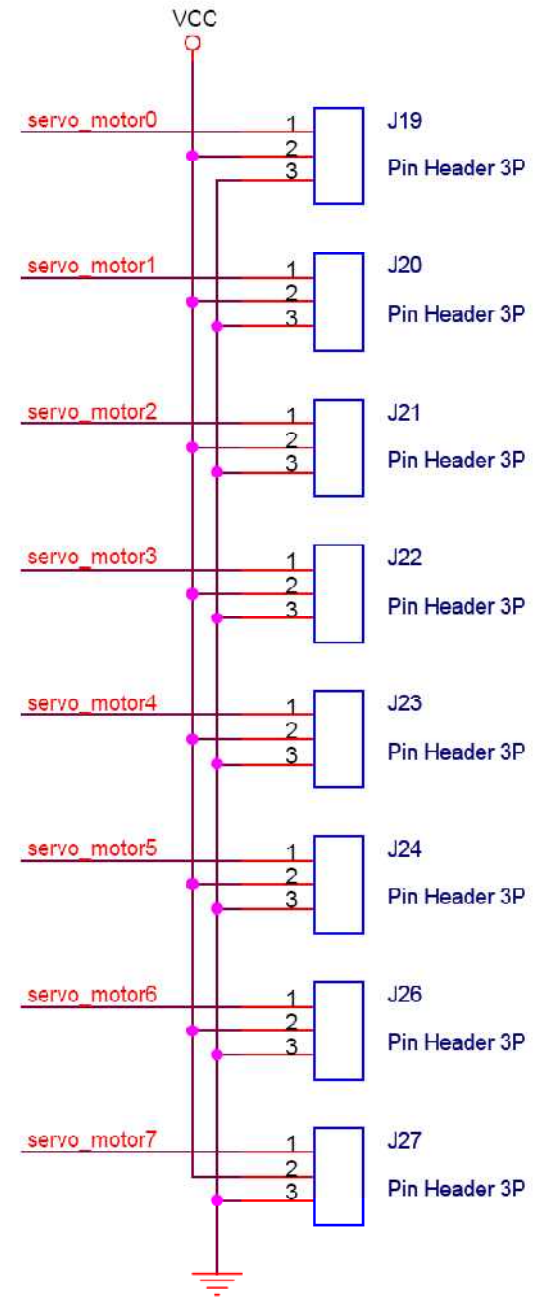


[1개의 서보 제어

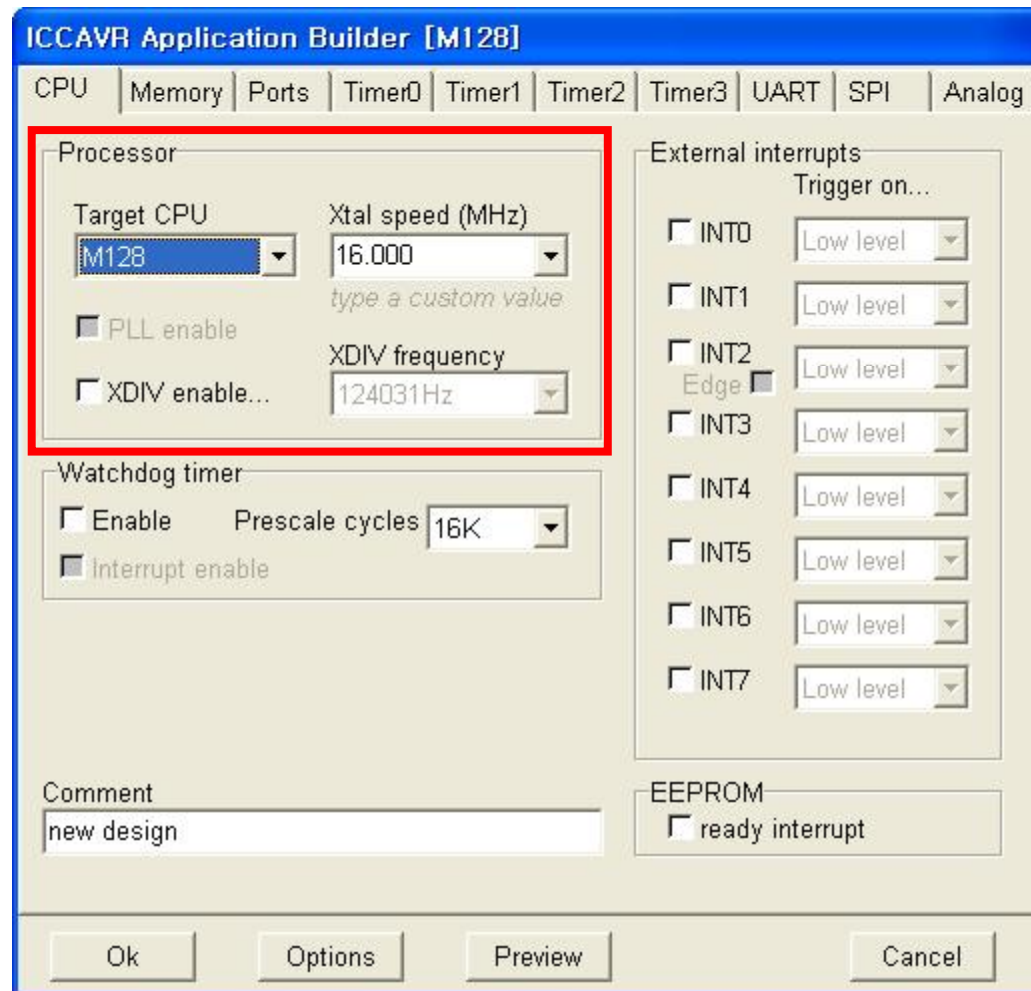
]

- 앞서 밝힌 것처럼,
- 타이머를 발생시켜 0.7~2.3ms 중 원하는 각도의 시간만큼만 1로 인식하도록 프로그래밍 한다.
- 1개의 서보를 제어 하려면,
- ICCAVR의 1us 타이머를 사용하였을 경우,
 - 3700번을 반복하였을 때를 20ms 주기로 하고
 - 47~245를 -90~+90도로 하면 된다.

[Servo Motor



[Servo Motor 실습]



[Servo Motor 실습]

ICCAVR Application Builder [M128]

CPU | Memory | Ports | Timer0 | Timer1 | Timer2 | Timer3 | UART | SPI | Analog

External memory

☒ Enable external memory

External memory page configuration

NA / 0x1100-0xFFFF

☐ Wait states

Lower wait states: None

Upper wait states: None

Absolute address definitions

Address	Name	Type	Count	
0x0000		signed char	1	Add

Modify

Remove

Ok Options Preview Cancel

[Servo Motor 실습]

ICCAVR Application Builder [M128]

CPU | Memory | Ports | Timer0 | Timer1 | Timer2 | Timer3 | UART | SPI | Analog

Port A

	7	6	5	4	3	2	1	0
Direction	☐	☐	☐	☐	☐	☐	☐	☐
Value	☐	☐	☐	☐	☐	☐	☐	☐
Change	☐	☐	☐	☐	☐	☐	☐	☐

Port B

	7	6	5	4	3	2	1	0
Direction	☐	☐	☐	☐	☐	☐	☐	☐
Value	☐	☐	☐	☐	☐	☐	☐	☐
Change	☐	☐	☐	☐	☐	☐	☐	☐

Port C

	7	6	5	4	3	2	1	0
Direction	☐	☐	☐	☐	☐	☐	☐	☐
Value	☐	☐	☐	☐	☐	☐	☐	☐
Change	☐	☐	☐	☐	☐	☐	☐	☐

Port D

	7	6	5	4	3	2	1	0
Direction	☐	☐	☐	☐	☐	☐	☐	☐
Value	☐	☐	☐	☐	☐	☐	☐	☐
Change	☐	☐	☐	☐	☐	☐	☐	☐

Port E

	7	6	5	4	3	2	1	0
Direction	☐	☐	☐	☐	☐	☐	☐	☐
Value	☐	☐	☐	☐	☐	☐	☐	☐

Port F

	7	6	5	4	3	2	1	0
Direction	☐	☐	☐	☐	☐	☐	☐	☐
Value	☐	☐	☐	☐	☐	☐	☐	☐

Port G

	7	6	5	4	3	2	1	0
Direction	☐	☐	☐	☐	☐	☐	☐	☐
Value	☐	☐	☐	☐	☐	☐	☐	☐

right click on a "value" bit to define signal name

Ok Options Preview Cancel

[Servo Motor 실습]

ICCAVR Application Builder [M128]

CPU | Memory | Ports | **Timer0** | Timer1 | Timer2 | Timer3 | UART | SPI | Analog

Initialisation

☒ Use Timer0 Power off ☐ Overflow interrupt ☒

Desired value Units Actual value (error%)

100 uSec 100.000uSec (0.0%)

Prescale select ... TCNT0

64 0xE7

Compare

OCR0

0x19

OCO output mode

Disconnected

☐ Clear on compare

☐ Compare interrupt

Waveform mode

Normal

Asynchronous mode

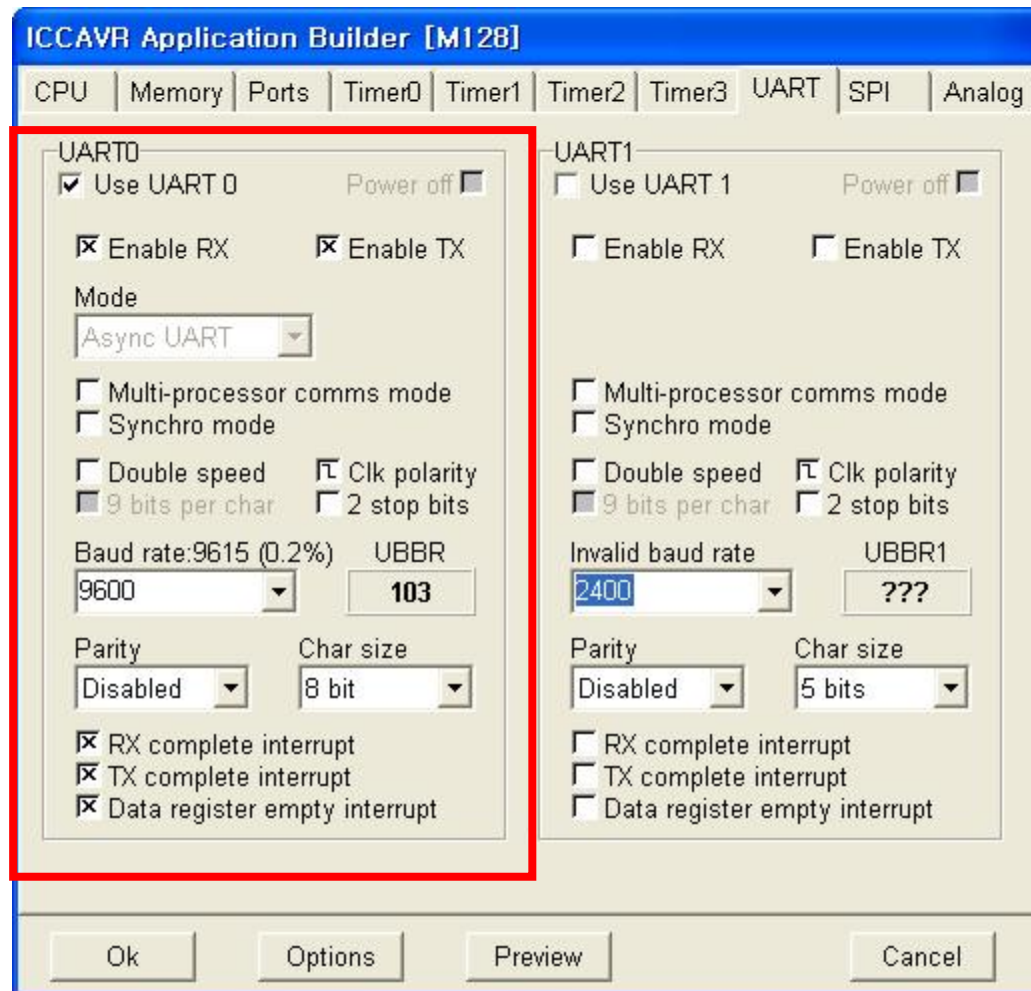
☐ Enable async mode

External crystal (Hz)

32768

Ok Options Preview Cancel

[Servo Motor 실습]



[Servo Motor 실습]

- **#define SERVOMIN 0**
- **#define SERVOMAX 200**

- **#define VALUEMIN 6**
- **#define VALUEMAX 24**

- **volatile unsigned int defValue = VALUEMAX;**
- **volatile unsigned int servoTime = 0;**
- **volatile unsigned int turnTime = 0;**

[Servo Motor 실습



```
#pragma interrupt_handler timer0_ovf_isr:17
void timer0_ovf_isr(void)
{
    TCNT0 = 0xE7; //reload counter value

    if(servoTime >= SERVOMAX){
        servoTime = SERVOMIN;
        turnTime++;
    }

    if(turnTime > SERVOMAX){
        if(defValue == VALUEMIN){
            defValue = VALUEMAX;
        }
        else{
            defValue = VALUEMIN;
        }
        turnTime = 0;
    }
}
```

[Servo Motor 실습]

```
servoTime++;  
  
if(defValue >= VALUEMAX){  
    defValue = VALUEMAX;  
}  
if(defValue <= VALUEMIN){  
    defValue = VALUEMIN;  
}  
  
if(defValue > servoTime){  
    EX_SERVO = 0x01;  
}  
else{  
    EX_SERVO = 0x00;  
}  
  
}
```

[Servo Motor 실습 문제]

- 시리얼을 이용하여, 100단계 이상으로 각 값을 가지고 서보 모터 제어할 수 있도록 작성하시오. 단, 시리얼로 현재 각도를 보내서 모니터링 하세요. (모터 1개 기준)
 - 'A'키를 누르면 각도가 감소(현재 숫자에서 감소)
 - 0이하로는 떨어지지 않게 만들며, 현재 값은 시리얼로 보여진다.
 - 'S'키를 누르면 각도가 증가(현재 숫자에서 증가)
 - 300이상으로는 증가하지 않게 만들며, 현재 값은 시리얼로 보여진다.

[다중 서보모터 제어]

- 3개 이상의 서보 모터를 동시에 원하는 각도로 제어할 수 있도록 프로그램 하세요.
- 8개, 16개 까지 동시에 제어할 수 있도록 프로그램 하시오.
- 한 개의 서보모터를 할 수 있는 범위 내에서 가장 정밀하게 단계를 많이 만들어서 제어할 수 있도록 프로그램 하세요.

[Servo Motor 프로토콜 넣기]

■ 프로토콜로 동작하는 서보 모터 구현

- 서보 모터를 동작시키는 루틴을 작성 후,
- 오실로 스코프 등으로 정확한 타이밍을 재고, 값을 결정한 후,
- RS-232C 나 RS-422 등을 통해 제어할 수 있도록 프로토콜을 집어 넣는다.
- 이 때, 서보의 주기가 너무 길게 되면, 하모닉에 의해 덜덜 떠는 증상이 나타나게 되며, 또한 과도한 피드백으로 인하여 설정된 값으로 수렴하지 못하게 된다. 네거티브 피드백으로 동작하는 서보의 원리를 잘 이해하여, 가능한 정확한 펄스를 넣어줄 수록 떨지 않게 된다.