

프로젝트 실습 I

2007. March. 5

서보모터의 제어

학습 목표

■ 학습 목표

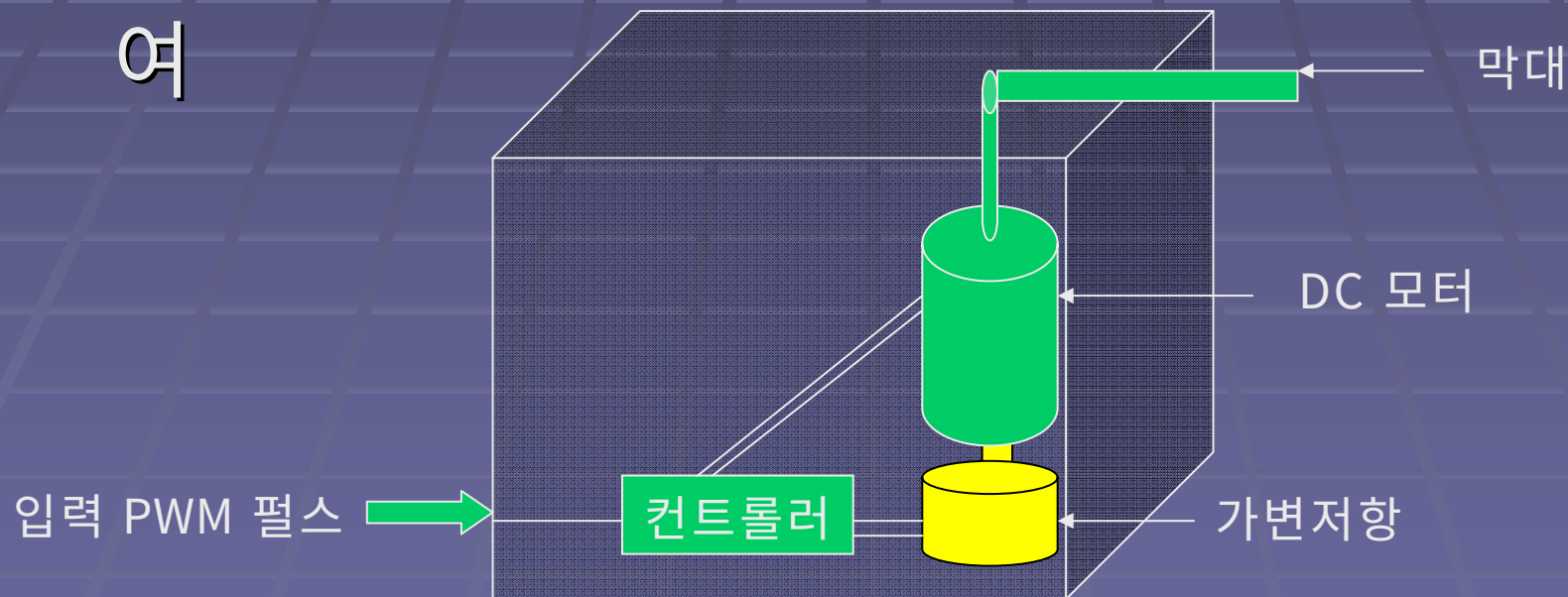
- 서보 모터의 종류와 모터 제작원리를 이해한다.
- 서보 모터의 제어 타이밍과 원리를 이해한다.
- AVR에서 타이머가 무엇인지 이해하고, 오실로스코프 등을 통하여 측정하고 수정하여, 원하는 타이밍의 타이머를 만들 수 있다.
- 1개의 서보 모터를 타이머를 이용하여 제어할 수 있다.
- 여러 개의 서보 모터를 타이머를 이용하여 제어할 수 있다.

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-232, 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

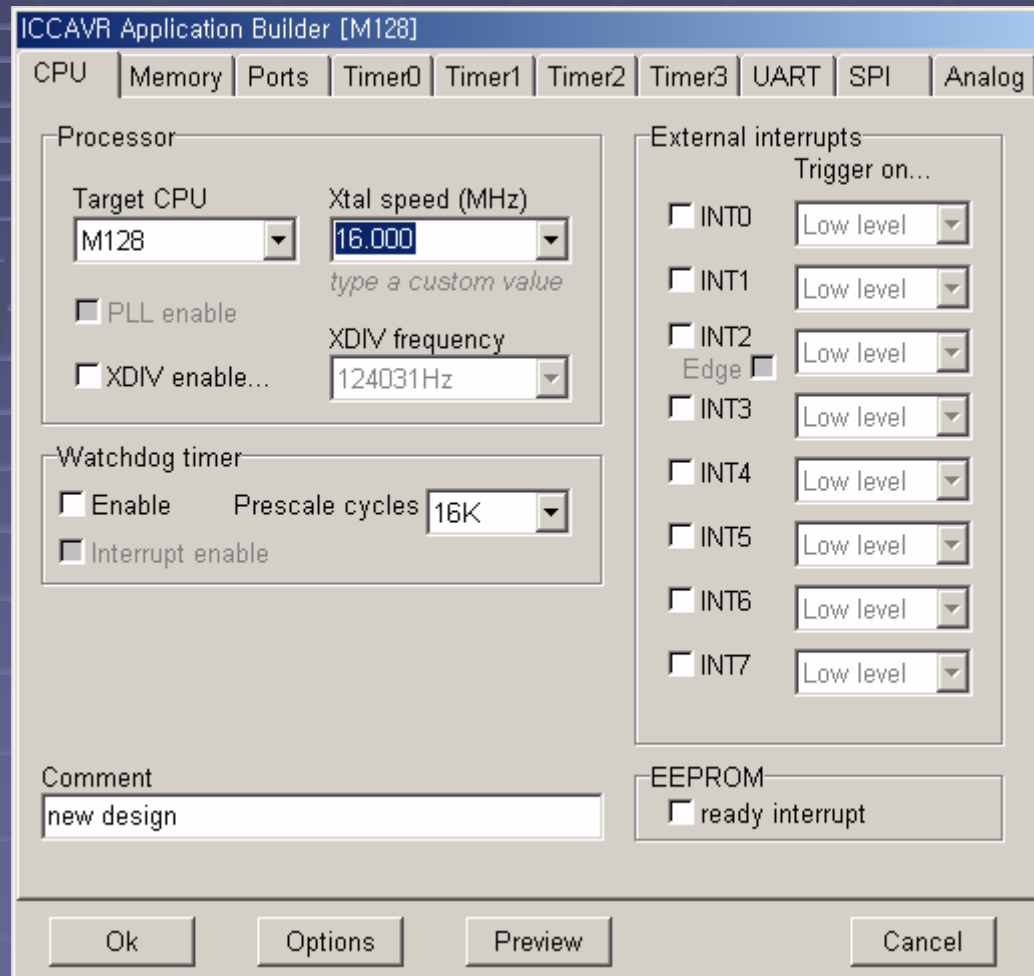
- RS-232C, RS-422 등으로 모터를 제어할 수 있도록, 명령어 프로토콜을 지정하여 넣고, 명령에 따라서 지정된 각도로 이동될 수 있도록 프로토콜을 제작한다.
- 각 모터마다 고유한 번호를 부여한 후, 모터제조 시에 EEPROM에 저장하고, 라벨에 기록한다.
- Example :
 - C021,1800 : 21번 모터를 180.0도로 이동하라.
- 모터마다 약간씩의 보정을 위하여, 모터 보정용 장치를 만들고, 모터를 물린 후에 보정하여 돌린 후, 모터에 있는 EEPROM에 값을 저장하여 전원 연결시의 초기값을 모터에 전원이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

AVR의 타이머란?

- AVR에서 타이머는 지정한 일정 시간 간격마다 불려지게 된다.
- 이 때, 동작된 타이머의 오버 플로우 인터럽트 서비스 루틴 안에서 프로그래머는 원하는 동작을 할 수 있다.
- 인터럽트 서비스 루틴은 ISR이라고도 한다

마법사를 이용한 타이머 루틴 생성

MCU와 X-tal speed 지정



The image shows the 'ICCAVR Application Builder [M128]' dialog box. It has a tabbed interface with 'CPU' selected. The 'Processor' section includes 'Target CPU' (M128), 'Xtal speed (MHz)' (16.000), 'PLL enable' (unchecked), and 'XDIV enable...' (unchecked). The 'Watchdog timer' section has 'Enable' (unchecked) and 'Interrupt enable' (unchecked). The 'External interrupts' section lists INT0 through INT7, each with a 'Trigger on...' dropdown set to 'Low level'. The 'EEPROM' section has 'ready interrupt' (unchecked). The 'Comment' field contains 'new design'. Buttons at the bottom are 'Ok', 'Options', 'Preview', and 'Cancel'.

ICCAVR Application Builder [M128]

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

Processor

Target CPU: M128

Xtal speed (MHz): 16.000
type a custom value

☐ PLL enable

☐ XDIV enable...

XDIV frequency: 124031Hz

Watchdog timer

☐ Enable Prescale cycles: 16K

☐ Interrupt enable

External interrupts

Trigger on...

☐ INT0 Low level

☐ INT1 Low level

☐ INT2 Low level

Edge ☐

☐ INT3 Low level

☐ INT4 Low level

☐ INT5 Low level

☐ INT6 Low level

☐ INT7 Low level

EEPROM

☐ ready interrupt

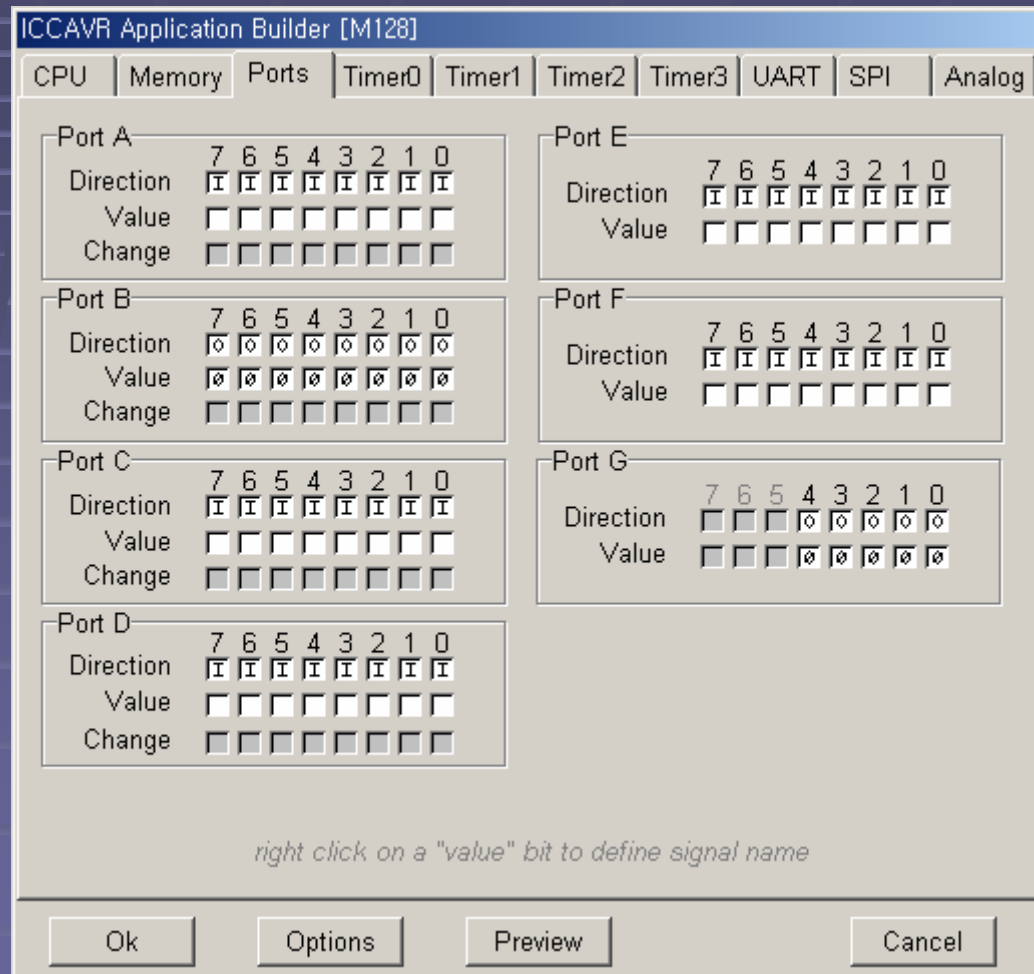
Comment: new design

Ok Options Preview Cancel

마법사를 이용한 타이머 루틴 생성

사용 포트 방향 지정

- 포트 맵 작성 후 계획에 의하여 체크한다.
- 출력은 0, 입력은 1이다.
- 출력 지정 시, 비트 별 초기값 지정이 가능하다.



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	☐	☐	☐	☐	☐	☐	☐	☐
Change	☐	☐	☐	☐	☐	☐	☐	☐

Port F

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

Port G

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

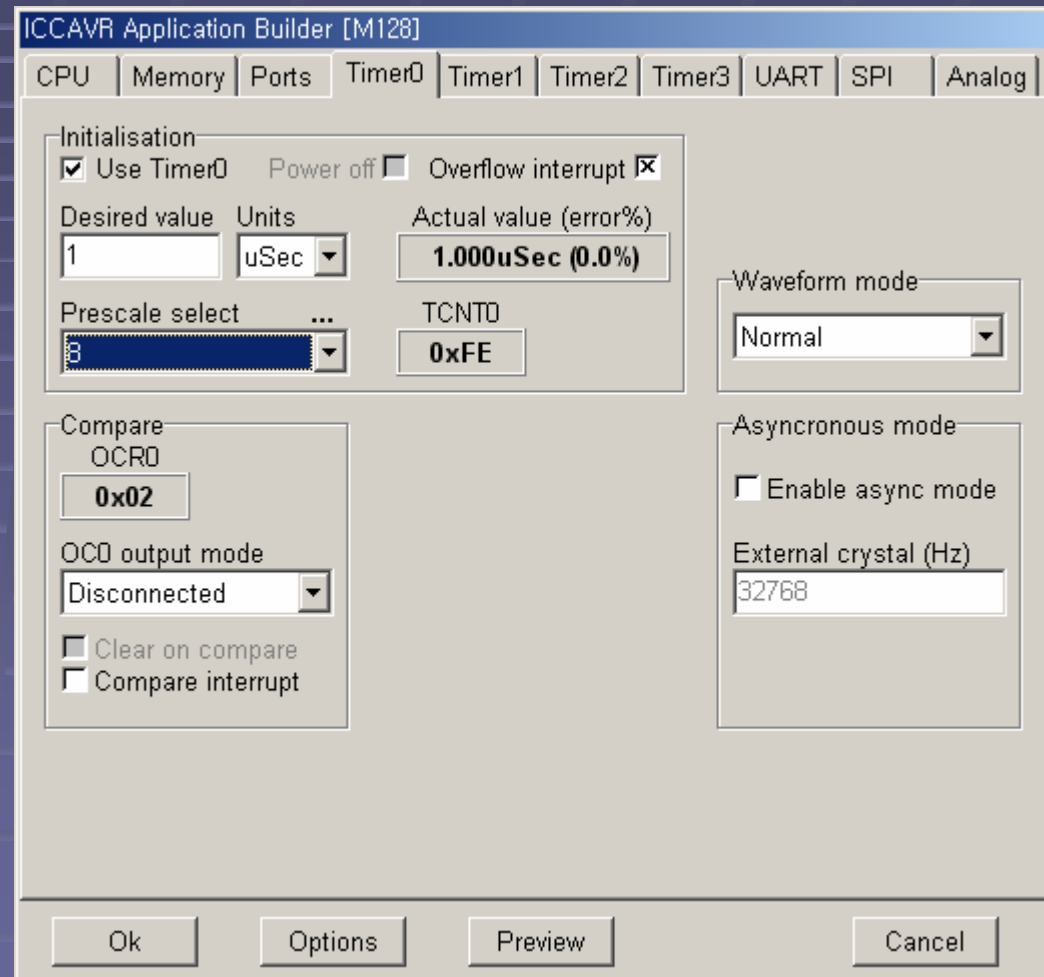
right click on a "value" bit to define signal name

Ok Options Preview Cancel

마법사를 이용한 타이머 루틴 생성

타이머 지정(ATmega128의 경우)

- 0~3까지 8비트 타이머 2개,
- 16비트 타이머 2개 지정이 가능
- 원하는 시간 값과 단위를 설정
- Prescale을 선택하고,
- Overflow interrupt를 체크한다.

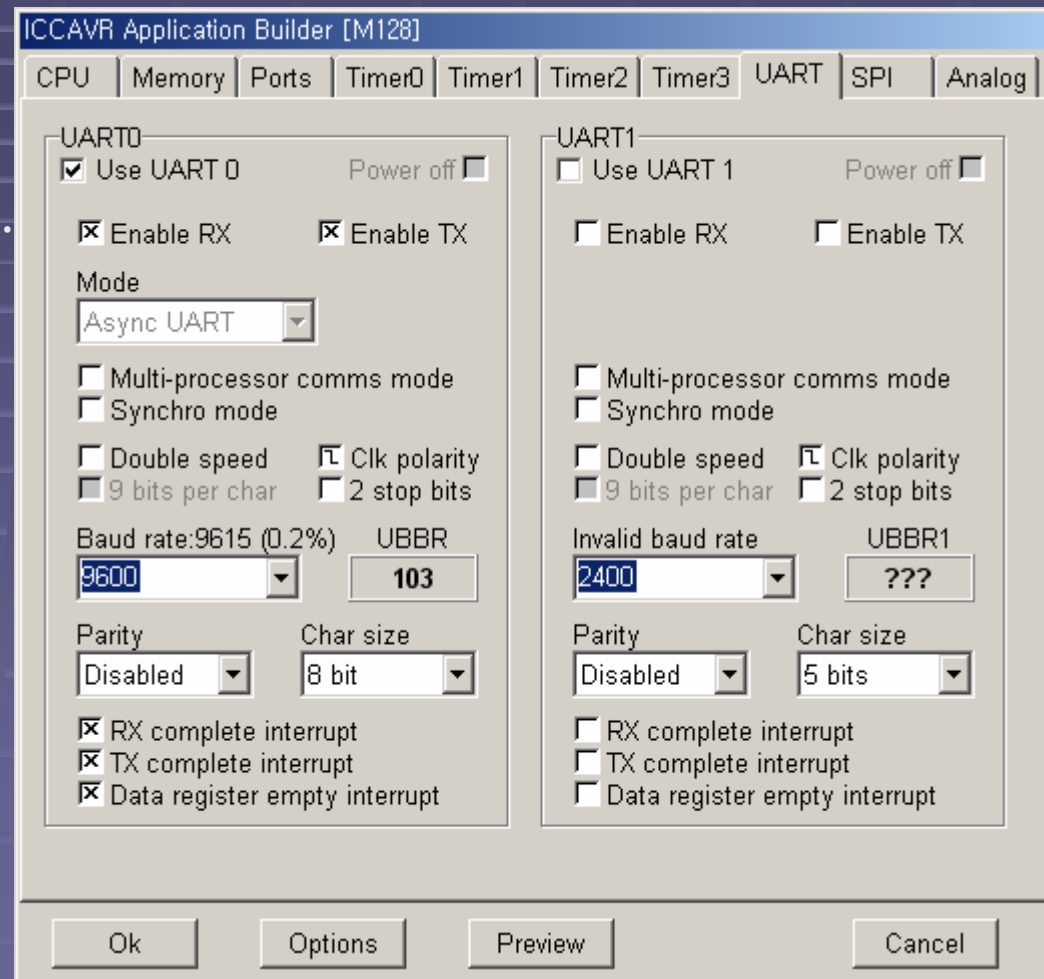


The image shows the 'ICCAVR Application Builder [M128]' window with the 'Timer0' tab selected. The 'Initialisation' section has 'Use Timer0' checked, 'Power off' unchecked, and 'Overflow interrupt' checked. The 'Desired value' is 1, 'Units' is uSec, and 'Actual value (error%)' is 1.000uSec (0.0%). The 'Prescale select' is 8, and 'TCNT0' is 0xFE. The 'Waveform mode' is Normal. The 'Compare' section shows 'OCR0' as 0x02, 'OCD output mode' as Disconnected, and 'Clear on compare' and 'Compare interrupt' are unchecked. The 'Asynchronous mode' section has 'Enable async mode' unchecked and 'External crystal (Hz)' as 32768. At the bottom are 'Ok', 'Options', 'Preview', and 'Cancel' buttons.

마법사를 이용한 타이머 루틴 생성

시리얼 통신(ATmega128의 경우)

- 0~1까지 두 개를 지정 가능하다.
- Enable RX, TX,
- Baudrate, parity, data size
- 9,600bps, none, 8 bit
- 인터럽트 사용 시,
인터럽트 사용부분 체크



The image shows the 'ICCAVR Application Builder [M128]' window with the 'UART' tab selected. It contains two panels for configuring UART0 and UART1.

UART0 Configuration:

- ☒ Use UART 0 (Power off ☐)
- ☒ Enable RX, ☒ Enable TX
- Mode: Async UART (dropdown)
- ☐ Multi-processor comms mode, ☐ Synchro mode
- ☐ Double speed, ☒ Clk polarity, ☐ 9 bits per char, ☐ 2 stop bits
- Baud rate: 9615 (0.2%) (dropdown), UBRR: 103 (text box)
- Parity: Disabled (dropdown), Char size: 8 bit (dropdown)
- ☒ RX complete interrupt, ☒ TX complete interrupt, ☒ Data register empty interrupt

UART1 Configuration:

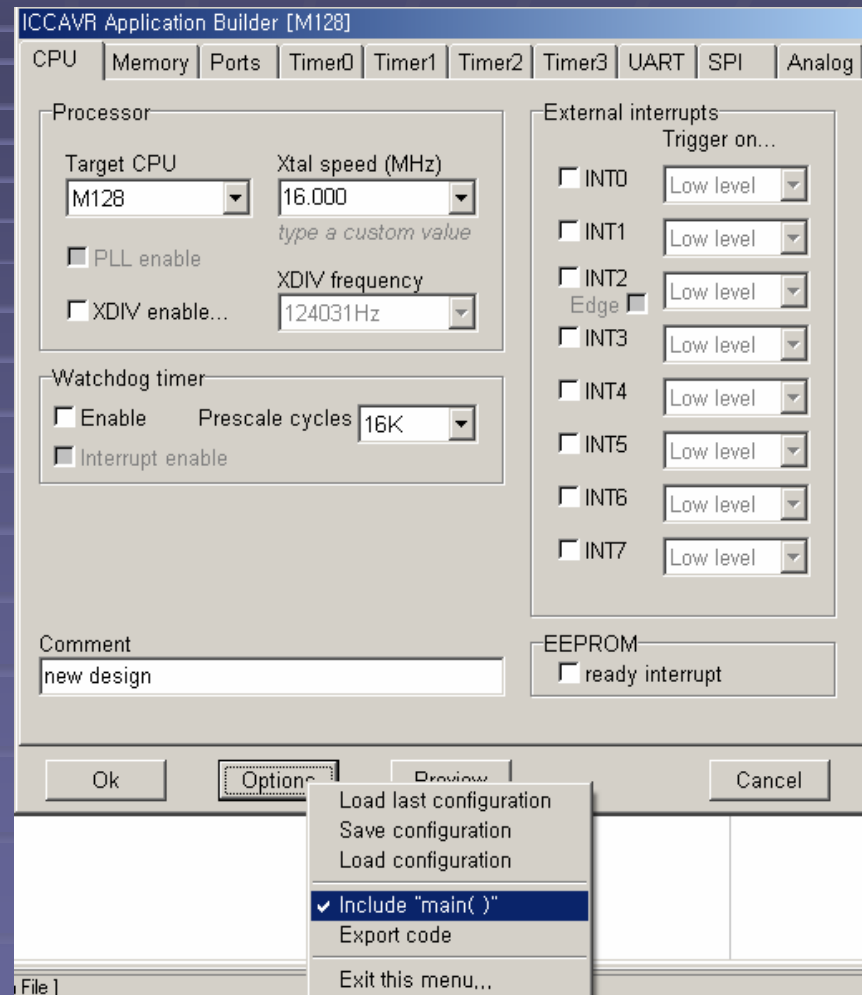
- ☐ Use UART 1 (Power off ☐)
- ☐ Enable RX, ☐ Enable TX
- ☐ Multi-processor comms mode, ☐ Synchro mode
- ☐ Double speed, ☒ Clk polarity, ☐ 9 bits per char, ☐ 2 stop bits
- Invalid baud rate (dropdown), UBRR1: ??? (text box)
- Parity: Disabled (dropdown), Char size: 5 bits (dropdown)
- ☐ RX complete interrupt, ☐ TX complete interrupt, ☐ Data register empty interrupt

Buttons at the bottom: Ok, Options, Preview, Cancel.

마법사를 이용한 타이머 루틴 생성

Option에서 Include를 하고,
OK를 눌러서 소스코드를 생성한다.

- 프로젝트를 생성한 후에,
제작된 파일을 .c로 저장한 후
프로젝트에 Add 한다.
- Project -> Options에서 device
configuration을 ATmega128로
바꾸어 컴파일러에게
사용 MCU를 알려준다.



마법사를 이용한 타이머 루틴 생성

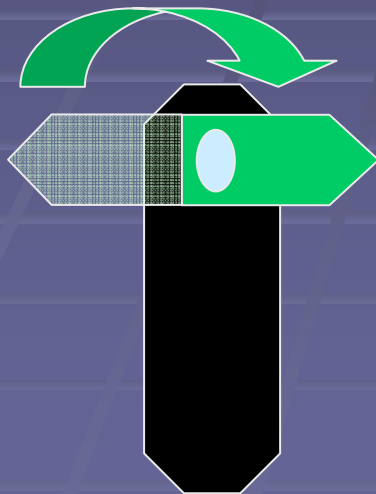
- 타이머 0 Overflow 인터럽트 서비스 루틴의 예
- ```
#pragma interrupt_handler timer0_ovf_isr:17
void timer0_ovf_isr(void)
{
 TCNT0 = 0xFE; //reload counter value
 // 여기에 타이머로 반복하고자 하는 문장 넣음
 // 인터럽트 루틴이므로 길지 않고 가능한 짧게
}
```

# 마법사를 이용한 타이머 루틴 생성

- 타이머 0 Overflow 인터럽트 서비스 루틴의 예
- ```
#pragma interrupt_handler timer0_ovf_isr:17  
void timer0_ovf_isr(void)  
{  
    TCNT0 = 0xFE; //reload counter value  
    // 여기에 타이머로 반복하고자 하는 문장 넣음  
    // 인터럽트 루틴이므로 길지 않고 가능한 짧게  
}
```

서버모터의 제어 방법

- 서보는 20ms 주기로 0.7~2.3ms만 ON을 주면 ON 주는 시간에 따라 -90도~+90도 까지 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.

여러 개의 서보 제어

- 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 짚어 넣는다.