Lab 10: Pulse Width Modulation (PWM)

EE222: Microprocessor Systems

1 Administrivia

Learning Outcomes

By the end of this lab you will be able to;

1. Use Timers of ATmega16 to generate PWM Signals.
2. Adjust the duty cycle of the generated PWM Signal.

Deliverable

You are required to submit

- Appropriately Commented Code
- Explicit Calculations for Timer Values
- Issues in Developing the Solution and your Response

in the beginning of next lab

Hardware Resources

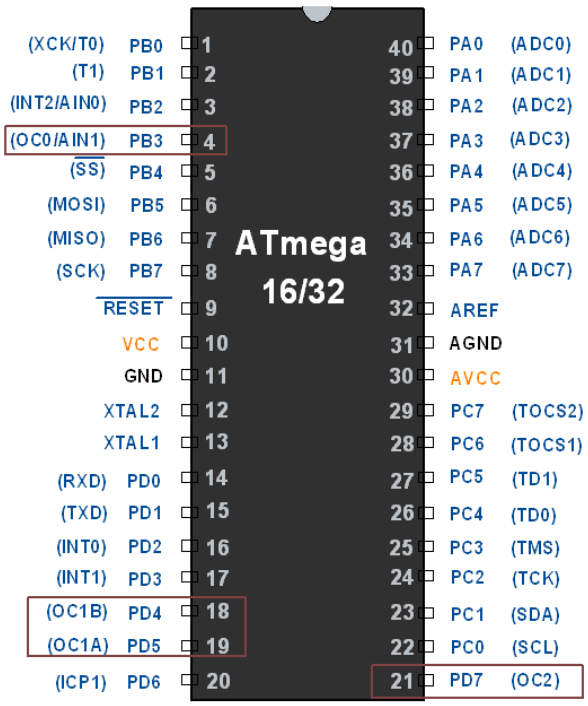
- ATmega16A Microcontroller Unit
- Universal Programmer
- Potentiometer
- LEDs
- Resistance 47Ω
- Oscilloscope

AVR ATmega PWM

ATmega has an inbuilt PWM unit. As we know, ATmega has 3 Timers T0, T1, and T2 which can be used for PWM generation. Mainly there are two modes in PWM.

1. Fast PWM
2. Phase correct PWM

We need to configure the Timer Register for generating PWM. PWM output will be generated on the corresponding Timer's output compare pin (OCx).



Configuring TimerO for PWM

It is simple to configure PWM mode in Timer. We just need to set some bits in the TCCR0 register.

TCCR0: Timer Counter Control Register 0

7	6	5	4	3	2	1	0
FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00

The bits can be modified as follow to achieve different modes:

WGM00	WGM01	Timer0 mode selection bit
0	0	Normal
0	1	CTC (Clear timer on Compare Match)
1	**0**	**PWM, Phase correct**
1	**1**	**Fast PWM**

Go through the following link to learn about setting up of bits for different modes.

www.electronicwings.com/avr-atmega/atmega1632-pwm

FAST PWM

To set Fast PWM mode, we have to set WGM00: 01= 11. To generate a PWM waveform on the OC0 pin, we need to set COM01:00= 10 or 11.

The following code controls the brightness of an LED using Fast PWM.

```
#define F_CPU 8000000UL
#include "avr/io.h"
#include <util/delay.h>
void PWM_init()
{
        /*set fast PWM mode with non-inverted output*/
        TCCR0 = (1<<WGM00) | (1<<WGM01) | (1<<COM01) | (1<<CS00);
        DDRB|=(1<<PB3);  /*set OC0 pin as output*/
}

int main ()
{
        unsigned char duty;
        PWM_init();
        while (1)
        {
                for(duty=0; duty<255; duty++)
                {
                        OCR0=duty;  /*increase the LED light intensity*/
                        _delay_ms(8);
                }
```

```
                for(duty=255; duty>1; duty--)
                {
                        OCR0=duty;  /*decrease the LED light intensity*/
                        _delay_ms(8);
                }
        }
}
```

Lab Tasks

Task A

Implement the code in example above, just to get familiar with ATmega16 PWM.

Task B

Use a potentiometer to control the brightness of an LED using PWM in ATmega16. Simulate the circuit on Proteus and patch the hardware. Use Oscilloscope to observe the duty cycle of the generated PWM signal as the knob of potentiometer changes.

As the value of duty cycle changes, print out the duty cycle in percentage on 7 Segment display as well as on the Serial port of PC using UART.