LAB 4
Servo Controller

NAME: POSSIBLE POINTS: 10

STUDENT ID:

COURSE DATE & TIME:

OBJECTIVE:

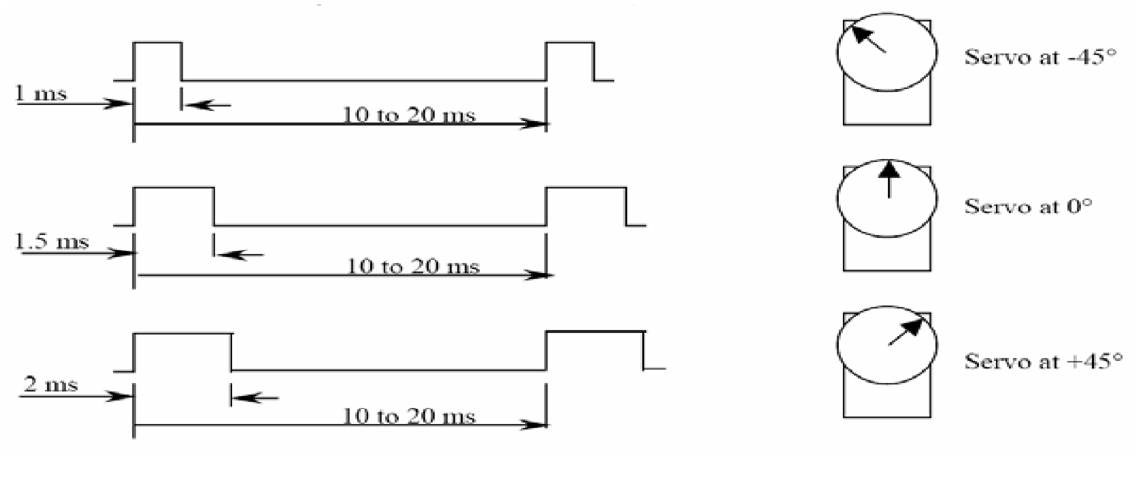
- To interface and drive a “Hobby” type 3wire Servo from the 8051.
- To program the 8051 to use one of the peripheral timers, which will generate an Interrupt that creates a positive timed pulse on one of the I/O Port Pins. The duration of time that the positive pulse stays high will determine the angle that the servo moves to.

OPERATION:

We will be interfacing to a “Hobby” type 3-wire Servo that can be controlled to go from -45 degrees to +45 degrees depending on the duration of the pulse we send to the Servo. Two Pushbutton/Switches will be used to cycle up or down through the desired angle that the servo will be set at in increments of 4.5 degrees, i.e. -45, -40.5, -36, -31.5, …, 0, 4.5, 9, 13.5, …, 40.5, 45. This non-round number was selected to make the math of your program considerably simpler

THEORY:

Driving the Servo to the desired angle will require generating a positive pulse on one of the I/O pins. This can be considered generating a PWM (Pulse Width Modulated) signal. The following diagram and respective timing should make this clear.



Although there are several ways to program a microcontroller to generate this pulse, the required method for this lab is as follows. An 8051 timer will be configured to generate an interrupt every 20ms. Therefore, the ISR - Interrupt Service Handler will execute once everytime the positive pulse is be generated. The ISR will generate this positive pulse by setting the P0.0 Pin High, a hard-coded busy-wait delay will create the timing between 1ms and 2ms (determined by the desired angle). Once this busy-wat 1ms to 2ms delay has finished, P0.0 will be cleared to Low.

A separate function "void PushButton_Handler(void)" will be called inside the Super-Loop of your main function and will handle the polling of the pushbuttons. The desired angle will be tracked by a variable and the state of the pushbuttons will cause this desired angle variable to be incremented or decremented each time the button is pushed. Make sure to catch the situations when the desired angle variable is being decremented/incremented past the extremes of -45 and 45 and to not allow the value to be modified outside this range.

TIPS & TRICKS:

Busy-Wait Delay inside the ISR: The easiest way to generate the required 1-2ms delay is to break up the delay into 2 separate delays. The first would be a constant 1ms delay followed immediately by a variable 0-1ms Delay. Getting precise timing in C-code can be tricky and normally for this type of application, you would use a more complicated implementation of the Timer to achieve this crucial timing. However this more advanced approach is outside the scope of our class and a simpler hard-coded busy wait delay will suffice for us and lower the complexity of our code and system. The following information can be used to derive all timing for this lab.

Given a system clock frequency of 3.0625Mhz

```
unsigned char x;
For(x = 0; x <= 100; x++){      //This for loop delay will busy-wait for 1.03ms (close enough
}                               //    for government work and keeps the math easy)
```

Polling the Pushbuttons/Switches: When making this lab you will encounter an issue with the PushButtons. A human pressing the PushButton is much slower than the software polling the Port Pin. Example:

```
if(PushButton_Is_Pressed){
    Increment Desired Angle }
```

This is the right functionality we want, however the problem is that this piece of code will poll the PushButton many times (10s to 100s) for every one press. Therefore for every one press of the push button the Desired Angle would be incremented or decremented to the max or min angle. One simple solution to this problem is to wait until the button has been released.

```
if(PushButton_Is_Pressed){
    Increment Desired Angle
    Wait Until Pushbutton Is Released }
```

LAB 4
Servo Controller

CONNECTIONS:

A Servo is a relatively easy device to interface with a microcontroller. There are 3 wires, typically:

Black = Ground, Red = 5v, White/Yellow is the signal wire that will be driven by your Port Pin.
Alternatively:
Brown = Ground, Red = 5v, Orange = Signal

LAB WRITE-UP AND DELIVERABLES:

The lab write-up will include this page as the cover sheet with any questions answered and the source code. Also a schematic of your specific implementation will be included. You must draw your schematic using proper Schematic Capture software like CircuitMaker. Also include a picture of your demo in the written submitted report.

DEMO AND GRADING:

When your project is ready, you will demonstrate the functionality to the instructor and hand in the write-up. Your demo will be graded on its ability to correctly resolve the desired angle. Also I will be looking for good coding techniques. Code structure and flow of data in your programs become crucial with the use of interrupts.