USC Viterbi School of Engineering

Ming Hsieh Department of Electrical and Computer Engineering

EE 459Lx - Embedded Systems Design Laboratory

EE459 Lab Assignment 3
Test Operation of the I/O Pins

1 Introduction

This assignment has two purposes: to develop a test program that can be used to confirm the operation of all the general purpose I/O pins of the microcontroller, and to become familiar with the use of the triggering controls on the oscilloscopes. The I/O pins will be exercised in a known way and observed using one of the oscilloscopes. The software should be saved for possible use later in the semester to test the microcontroller in case there are problems with the I/O lines.

2 Write a Test Program

The ATmega328P has several pins (see Table 1 and Figure 1) that can be configured as either inputs or outputs and used for whatever the program's author wishes. The table also shows the number of pins available on three other Atmel microcontrollers that are available for use in EE 459Lx projects.

Microcontroller	I/O pins	Pins available
ATmega1284P	32	PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD7
ATmega328P	21	PB0-PB5, PB7, PC0-PC5, PD0-PD7
ATtiny4313	16	PA1, PB0-PB7, PD0-PD6
ATtiny85	5	PB0-PB4

Table 1: General purpose I/O pins available

In the test program each of the ATmega328P I/O pins will be made to change from the low state to the high state and back to the low state in order to confirm that they are working properly. The oscilloscope will be used to observe the actions of the pins. To clearly see what is happening on the scope it is necessary to have a way to tell the scope when to start acquiring the waveforms of the signals being observed. This is known as setting up a triggering condition and depending on the type of scope in use and types of signals being observed triggering conditions can range from very simple to very complex. For observing the outputs in this assignment, we will designate one of the I/O pins to be the triggering signal to the scope. The scope will be configured to start acquiring the waveforms after the triggering signal has been detected. By knowing that the trigger is being generated at a specific place in the program, the timing relationship between the other pins can be analyzed.

Select one of the available I/O pins to be the trigger source. Write a program that first makes the trigger pin go from the zero state to the one state and then back to the zero state. After that make all the other pins do the same 0-1-0 transition one at a time. Once all the pins have been exercised, start the loop over again. Something like the steps shown in Fig. 2.

2.1 Changing Bits in I/O Ports

Each I/O port of the Atmel ATmega328P consists of up to eight signal lines that can be changed individually by the program by setting or clearing the corresponding bits in the associated PORT register. For more

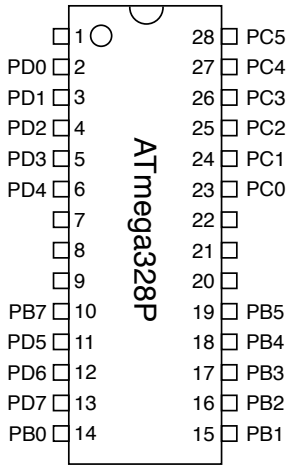

Figure 1: Port pins of the ATmega328P

```
Set all I/O pins to zero
Set all I/O pins to be outputs
Start of loop
    Trigger pin = 1
    Trigger pin = 0
    I/O pin #2 = 1
    I/O pin #2 = 0
    I/O pin #3 = 1
    I/O pin #3 = 0
        .
        .
        .
    I/O pin #n = 1
    I/O pin #n = 0
End of loop
```

Figure 2: Basic idea of how the program should work

detailed information on the I/O registers, see the class handouts on the ATmega328P and on the AVR-gcc programming environment.

All of the bits in the register can be changed at once by simply assigning a value to the register.

```
PORTD = 0x2C;              // Set PORTD to 00101100
```

The bits in the registers are numbered from zero (the LSB) to seven (the MSB). The above instruction sets bits 5, 3 and 2 in port D to ones, the others are zeros. Individual register bits can be changed without affecting the other bits by using the C language bitwise logical OR and AND operators. For example, the following changes port D, bit 2 (PD2).

```
PORTD = PORTD | 0x04;     // Set bit 2 in PORTD
PORTD = PORTD & 0xFB;     // Clear bit 2 in PORTD
```

While the above works, the code can be made simpler by using symbolic names for the bits and some additional C operators.

```
PORTD |= (1 << PD2);      // Set bit 2 in PORTD
PORTD &= ~(1 << PD2);     // Clear bit 2 in PORTD
```

Your program can consist of as many of these bit changing statements as necessary to exercise all the I/O port bits. It's not important what order the pins are tested as long as you run through all the pins in your loop. Once you have the program written, program it into your microcontroller .

3 Set Up Oscilloscope Triggering

The first step in observing the output is to configure the scope to acquire the signal for displaying at the right time. This is done by setting up the triggering conditions and is probably the most important aspect of making the scope into a useful diagnostic tool. The scope can be set to trigger when the input signal meets a set of conditions and it will then display the signals on all the channels that occurred before and after that time.

To configure the proper trigger settings, first press the "Menu" button in the Trigger section at the right side of the scope controls (Fig. 3). This will bring up the trigger settings along the bottom of the screen as shown in Fig. 4.

For this assignment, set the trigger settings to the following by pressing the keys below the labels and then selecting the desired setting.

Figure 3: Trigger controls

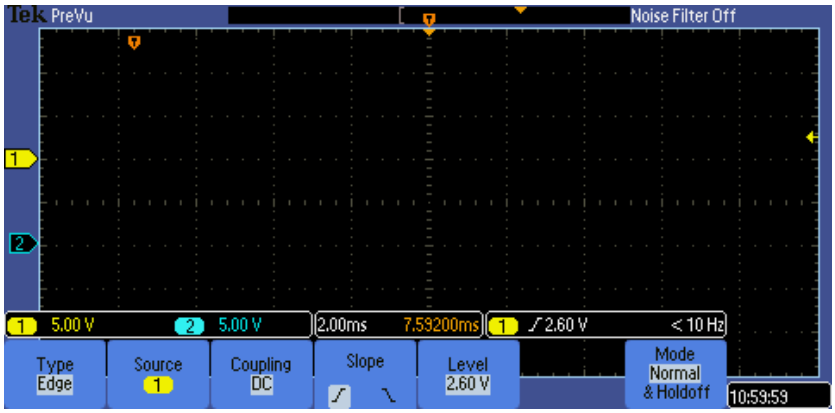

Figure 4: Trigger settings at bottom of screen

Type - Set to "`Edge`". This will cause the scope to trigger on a rising or falling edge of the signal.

Source - Pick one of the channels of the oscilloscope to be the trigger source and use the buttons to set the trigger channel to the channel number you picked.

Coupling - Set to "`DC`".

Slope - Set this to the rising edge although it won't make much difference for this exercise since the duration of the pulse being used for triggering is relatively short.

Level - Set to something about midway between 0 and 5V. The scope will trigger when the input signal passes this level. The triggering level can also be adjusted later using the "Level" knob in the Trigger controls.

Mode - Set to "`Normal`". If the label also says "`& Holdoff`" that's OK.

The scope is now set to trigger when a rising edge of the trigger source channel passes the selected voltage. Use the "Position" knob in the Horizontal section of the controls to move the little orange "`T`" marker at the top of the screen over near the left edge of the display. This marker shows where the triggering occurred and for this assignment we are mostly interested in observing things that happen after the triggering.

4 Observe Microcontroller Output

Connect the probe for the trigger channel to the pin you selected as the trigger source, and connect the probe's ground clip to a ground pin on your board. Connect one or more of the other channels to the next outputs that will be toggled after the trigger. You should see an output similar to that in Fig. 5. Channel 1 at the top is connected to the trigger source and the marker at the top of the screen is showing that the scope triggered on the rising edge of the signal on channel 1. Channel 2 below it is connected to the next bit to be toggled so its pulse is the same shape as the trigger pulse but delayed. Channel 3 is connected to the third bit in the sequence and appears after the one on channel 2. Channel 4 is connected to the last bit in the toggling sequence just before the program loops back to start of the test.

Move one of the scope probes along the microcontroller's pins in the order in which the bits are being toggled and confirm that all of them are working and that the pulses are appearing in the order in which you would expect them to based on the position of the code toggling that bit in the program. Make sure all of the output pins are producing a signal of about the same height when in the one state.

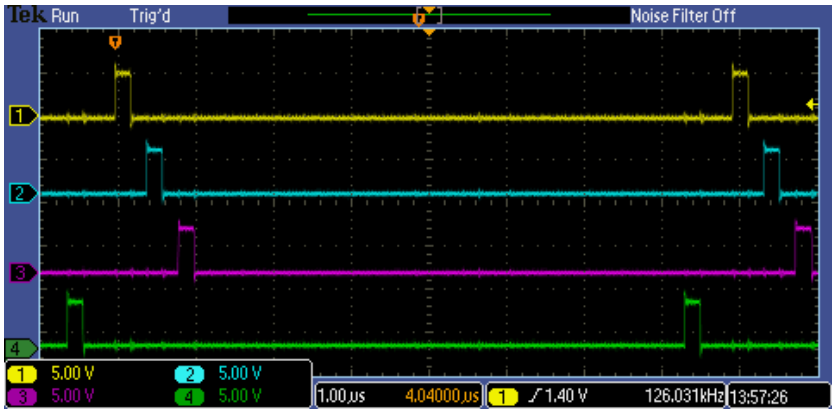

Figure 5: Multiple outputs from micro controller

5 Normal vs. Auto Triggering

For the assignment above we set the scope for "Normal" triggering. In Normal triggering mode, the scope will only trigger and show a display when the triggering conditions are met, such as reaching a certain voltage on a rising edge. In the upper right corner of Fig. 5 it says "`Trig'd`". This indicates that the scope has found a triggering condition and the display is showing the resulting waveforms. Now remove the probe that is on the pin you selected for triggering. The indicator in the upper left should change to "`Trig?`" to show that no trigger condition is currently being found. On these Tektronix scopes the display may continue to show that data acquired from the last time a trigger condition was found, and the `Trig?` indicator is telling you that the display is actually showing something from the past and not what may currently be coming from the circuit. Put the probe back on the trigger pin and the display will go back to the "`Trig'd`" indication and show current output waveforms.

Press the "Menu" button in the Trigger controls to bring up the labels above the button at the bottom of the screen. Press the "Mode" button and select "Auto (Untriggered roll)" from the buttons along the right side of the screen. The waveform display probably won't change and it will still indicate "`Trig'd`". Once again remove the probe from the trigger pin. The indicator at the top left will change to "`Auto`" and the display will start shifting horizontally with the pulses appearing to scroll across the display in a way that makes analysis of them nearly impossible.

This illustrates the difference between Normal and Auto triggering modes. When a trigger condition is present, Auto and Normal mode produce the same waveform display. The difference is what happens when the trigger disappears. While Normal mode shows nothing, or maybe waveforms from a triggered display from the past, in Auto mode the scope automatically triggers, acquires a waveform, draws it on the screen and then continues to do this over and over. Without a defined condition to trigger on, the waveforms are just drawn on top of each other as they come into the scope, resulting in things appearing to fly back an forth across the screen. However, this type of display can still be very useful in some situations. If you are analyzing a circuit where you are not sure what signals are present, it may be difficult to set up a triggering condition that acquires the signals. Using Auto mode lets you get a glimpse of the waveform, and from that you might be able to select a triggering condition to make the display more readable. In Normal mode, you'll probably end up just seeing a blank screen with no idea what the circuit is doing.

For example, suppose you are looking at one of the microcontroller's output pins in Normal mode with the triggering level set to 2.4V which should be correct to trigger when the output pin goes from low to high state. But nothing ever shows up on the screen. Switching to Auto mode shows waveforms moving rapidly across the screen but it's still evident that nothing is going above 2.4V, perhaps because some other part of the circuit is holding the output down and not letting it rise to the full logical one voltage. Learning this information can be crucial in debugging the circuit even though the scope was never able to actually trigger on the signal in this condition.