Lab 0: Introduction to the laboratory equipment

Goals:

The main goal of this lab is to become familiar with the main equipment that will be used in subsequent labs.

- A microcomputer with a serial (RS-232) port
- The STK-500 programmer
- Multimeter
- Oscilloscope
- Breadboard

The STK-500

This is the primary tool for programming the Atmel processors we will be using in the subsequent labs. It also contains several other components which may allow us to test parts of our programs. Generally, we will use the STK-500 as a simple development platform.

The STK-500 is powered from a 9 – 15V DC supply capable of providing at least 0.5 amps of current. (We will use 12V DC power supplies capable of supplying about 1 amp.) There is a power (on/off) switch near the power supply connector.

There are two 9-pin RS-232 connectors on the board. the one closest to the power connector, labeled `RS232 CTRL`, is the connection used to communicate with the STK-500.

Turn on the power switch, and connect a led input to one of the VTG pins. What happens?

Connect a led input to one of the GND pins. What happens?

Power down the STK-500, then connect the SWITCH bank to the LED bank using one of the 10-wire cable connectors. Then turn the power back on. (We normally power down the STK-500 whenever we make connections to it.) What happens when a switch is pressed?

Power down the STK-500 and connect the RS-232 cable between the processor and the STK-500 (to `RS232 CTRL`).

In Linux, the program `avrdude` can be used to actually program the processor in the STK-500, and also determine the state of the programmer. The latter is what we do next.

Execute the command:

```
avrdude -P /dev/ttyS0 -t -p m1284p -c stk500v2
```

Then type the following commands:

- help
- part
- sig
- dump lfuse
- dump hfuse
- dump efuse
- parms
- quit

Ensure that you understand the outputs. In particular, what do the fuse settings mean? Refer to the data sheet for the ATmega1284P to determine these.

What is the target voltage for the device? Measure VTG using the multimeter to confirm this value. (Set the multimeter to the 20V range. Then touch the + lead (red) to VTG, and the − lead (black) to GND.)

In Windows, the AVR Studio 4 environment provides a menu for displaying the fuse and other configurational settings.

Next we will compile and run a simple assembler program. the program simply copies the input at port D to output at port C.

Programming a chip

Now we will actually write a program into the memory of the ATMega1284P processor on the STK-500 board.

The program is the same as in the class notes, and can be downloaded from the lab home page.

After the program is compiled, under linux it can be loaded by the avrdude program. Instructions for compiling the program are included as comments near the top of the program file. The STK can be programmed using

```
avrdude -P /dev/ttyS0  -p m1284p -c stk500v2 -U flash:w:app.srec
```

(here, `app` is the name of the program).

In Windows, it can be programmed through the AVR Studio 4 environment.

After programming the device, turn off the power and connect one of the 10 pin cables between the switch and port D, and the the other between the LEDs and port C. (Show this to the instructor before turning the power back on.)

Then power up the device, and observe what happens when you press one or more buttons.

Another program

This is the second program from class, and it uses a single pin from port B (Pin B4) as input, and outputs a sequence of 0 1 0 1 ... with a period determined by the program code to Port B5 as long as the button is pressed.

This program can also be downloaded from the home page.

It can be downloaded to the STK-500 similarly.

For this example, use the wire connectors to connect a switch to pin B4, and connect a small speaker between pin B5 and ground. This can conveniently be done using Expansion Connector 1 on the STK-500. (Insert the speaker pins in socket positions 21 and 25 - see the example board in the lab).

Connect an oscilloscope probe to the output pin (pin B5) to see the shape of the waveform generated.

Modify the frequency of the code (changing the loop variables, for example) and observe the different frequencies.

This same method can be used to vary the intensity of and LED, for example, or even to play music. (We will find more efficient ways to do this later.)

Optional:

Modify the assembly language program so the sound frequency varies continuously while the button is pressed.