Mechatronic Project 234

Laboratory B: Digital and analogue inputs and the ADC

ADC stands for Analogue to Digital Converter. It is an interface which converts an analogue voltage (e.g 0-5V) into a digital signal (e.g an 8-bit or 10-bit number). An ADC is essential if you wish to be able to process analogue signals with your Microcontroller.

The ATmega can read up to 16 different analogue voltages (using the Port F & Port K pins).

In addition, the ADC requires two reference voltages. AREF is the maximum measured voltage (that is, the voltage which will return the largest digital output), while AGND is the minimum measured voltage (often 0V, connected to the GND pin on the ATmega1280). AVCC is the ADC power supply, and will usually be simply the same 5V power supply as the VCC pin on the ATmega1280/2560.

You will need the following command to initialise the ADC on the ATmega:

```
adc_init(); //initialse ADC
_delay_ms(20); //it's a good idea to wait a bit after your init section
```

Then to read the value from the ADC (the voltage of the pin)

```
adc_read(3); //read the voltage at ADC3
```

where 3 is the number of the ADC pin you want to read from (remember to check the pin mapping 3 is for ADC3, 0 is for ADC0 etc.).

You will also need to revise reading digital inputs using the `PINx` registers (e.g `PINA` for Port A etc).

Lab tasks:

1. An analog joystick is provided, which consists of two potentiometers and a push button. When the joystick is connected to power and ground, the two analog output pins will deliver voltages proportional to the stick position in the horizontal and vertical directions. The push-button is configured active-low (meaning the output on the SEL line is low when pressed and open otherwise). Connect the joystick to your Arduino so that you are able to read both the analog inputs and the button state. You will need to either configure an internal pull-up resistor for the select line, or else connect an external pull-up resistor. Sketch a circuit diagram which shows how the potentiometers and buttons are configured, including the pull-up resistor, and showing all Arduino/microcontroller input/output pins. **(B1)**

2. Write a program which reads the input voltage from one channel and displays the result using up to eight LEDs (you may reeuse a circuit from Lab A). **(B2)**

3. Modify your program to use the button press to switch the display between the input channels **(B3)**. Include a flowchart of this program in your workbook. **(B4)**

4. Collect a thermistor. Use your multimeter to estimate the range of resistances from room temperature to body temperature (heating the thermistor with your fingers). Find a way to represent the temperature of the thermistor using the LEDs. You may need to adjust AREF to maximize the range of input. **(B5*)**

All your work must be comprehensively recorded in your workbook. A separate formal writeup is not required, but your workbook must contain all the information that would be contained in a formal writeup including:

- Equipment used
- Procedure followed
- Calculations
- Circuit diagrams
- Formal description of program logic (preferably a flowchart)
- Results
- Reflection on what was learned