ECE3411 – Fall 2016

Lab 5b.

ADC: Analog to Digital Conversion

Marten van Dijk, Chenglu Jin
Department of Electrical & Computer Engineering
University of Connecticut
Email: {marten.van_dijk, chenglu.jin}@uconn.edu

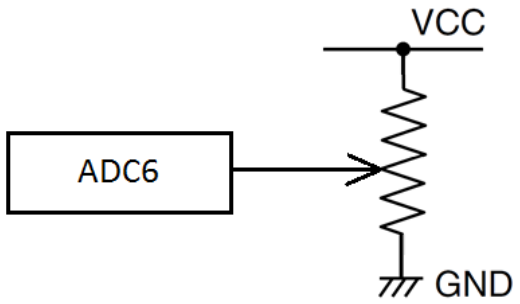
Copied from Lab 5b, ECE3411 – Fall 2015, by
Marten van Dijk and Syed Kamran Haider

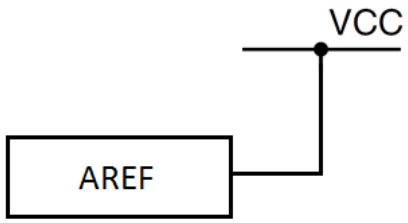
UCONN

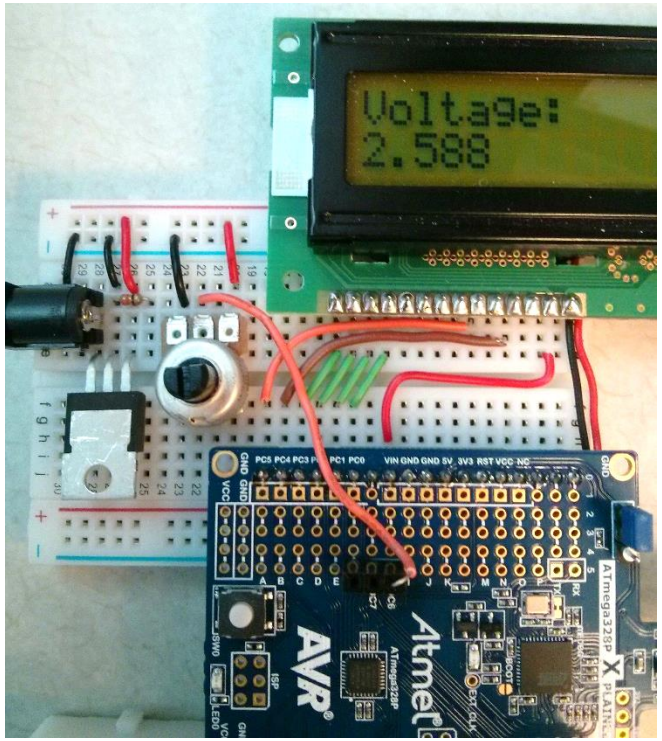
Hardware Changes for Task1

In this lab, we'll be measuring analog voltages using ADC

- Connect the potentiometer to ADC6 pin as shown below.
- Connect AREF pin to VCC → Reference voltage becomes 5V.







Task1: Simple Voltmeter

We are going to design a simple voltmeter that measures voltages between 0-5V with ~4mV resolution.

- Connect a potentiometer to produce variable voltage at ADC6 pin.
- Connect AREF pin to VCC
- Read the analog input voltage using ADC every 100ms
- Convert the ADC reading to voltage measurement and print the voltage on LCD.

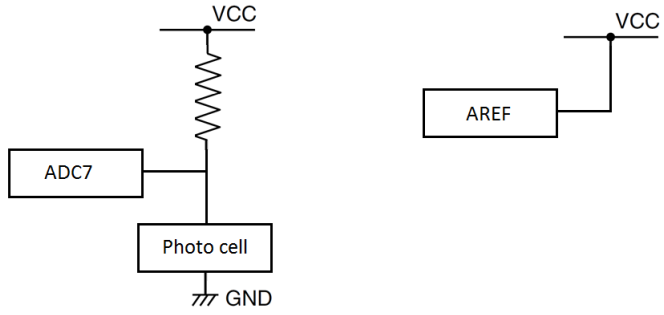
Note: Use the full 10-bit resolution of the ACD

Now you should be able to observe different voltage readings as you twist the potentiometer knob.

Task2: Light Sensor

We are going to control the brightness of LEDs based on the ambient light in the room.

- Replace the potentiometer with a resistor divider network with a photo cell (light dependent resistor) and a 10kOhm resistor to produce variable voltage at ADC7 pin.
- Connect AREF pin to VCC
- Read the analog input voltage using ADC every 100ms
- Change the duty cycle of the PWM signal for LEDs to control their brightness.
- When the Photo cell detects dark, increase the brightness and vice versa.



Task3: Voltage Measurement Statistics

Write a program with the following specifications:

- The program allows two modes for measuring voltages: Normal mode and ADC sleep mode. The user can change mode through UART while the program is executing.
- The program measures with 10 bits accuracy the input voltage (against AREF with VREF=5V) every 1ms.
- The program keeps track of the average and standard deviation of your measurements over the last 50 ms.
- Every 150 ms display on the LCD screen on two separate lines the most recent measured average voltage with its standard deviation (also use words/symbols to make clear what is being displayed).

Open Questions in Task3

Task3 leaves some open questions:

- How often do you measure? And when do you measure? E.g., you may want to wait say 1 ms after each print statement to the terminal and LCD screen such that the print buffers are emptied before the next ADC conversion.
- How do you average if some of your ADC conversions are closer together in time than others?

Pick your solution and explain its accuracy and what you could do in future versions to improve the result.

Further Observations

- Before entering the sleep mode, one needs to check bit RXC0=0 and bit TXC0=1, i.e., everything is received and transmitted.
- One may still see the following artifact:
  - Typing a character while in sleep mode only echoes part of the character or a corrupted character over the UART.
  - E.g. I typed two random strings while in sleep mode and the MCU echoed back this:

    kadjflskadjflk³djflksdjflksdlkfs

    skajsdhksajhdksjahdkjashdj³Øhkdjhkasjd
- Why does that happen?
- The backspace functionality helps the user to correct such an artifact.

Homework Task for Practice

You want to design a digital thermometer using your ATmega328P

- Replace the Potentiometer with a temperature sensor (included in the ordered components, its datasheet is uploaded on Piazza and can be found here)
- Convert the analog voltage coming from the temperature sensor to equivalent temperature value and print it on the LCD.
- Play with different resolutions of the ADC and different internal and external voltage reference values. What observations do you make?