Last Time

- Finite State Machines
- 3-bit Digital to Analog (D2A) Converter

Today

- Finish D2A
- Analog to Digital (A2D) conversion

Back today:

- HW4, Group quiz

Due today: project 4

Next Time

- Discussion of HW 4 and group quiz
- Exam preparation

- Submit suggested exam questions and answers to the D2L discussion group

Digital to Analog and Back

- Analog: encoding information using voltage
  - Many sensors use voltage as an output
  - Motors torque is determined by current passing through the motor
- Digital: encoding information with bits

How to move between these?

Digital to Analog Conversion

On the group quiz: 3-bit D2A converter

- Process specifies a digital output
- Within a short period of time (~ 1 ns), the voltage settles to the value that we computed

Analog to Digital Conversion

For a given voltage, what is the digital representation of the voltage?

- How would we implement this?

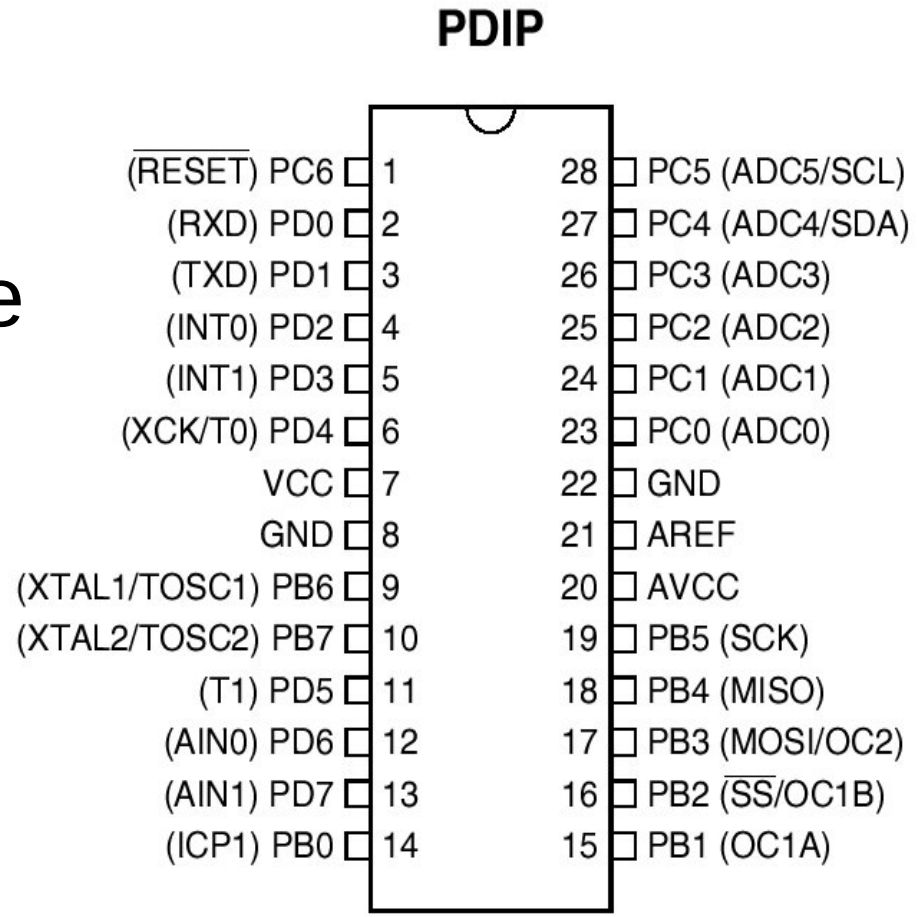
Analog to Digital Conversion

- For a given voltage, what is the digital representation of the voltage?

- Common approach: successive approximation
  - Use a D2A converter to produce a voltage V
  - Compare this with the input voltage Vi
  - If different, then increase/decrease V
  - Repeat (stopping when V is close to Vi)

A2D in the Mega8

- The mega8 contains hardware that implements successive approximation
- 6 mega8 pins can be configured as analog input pins



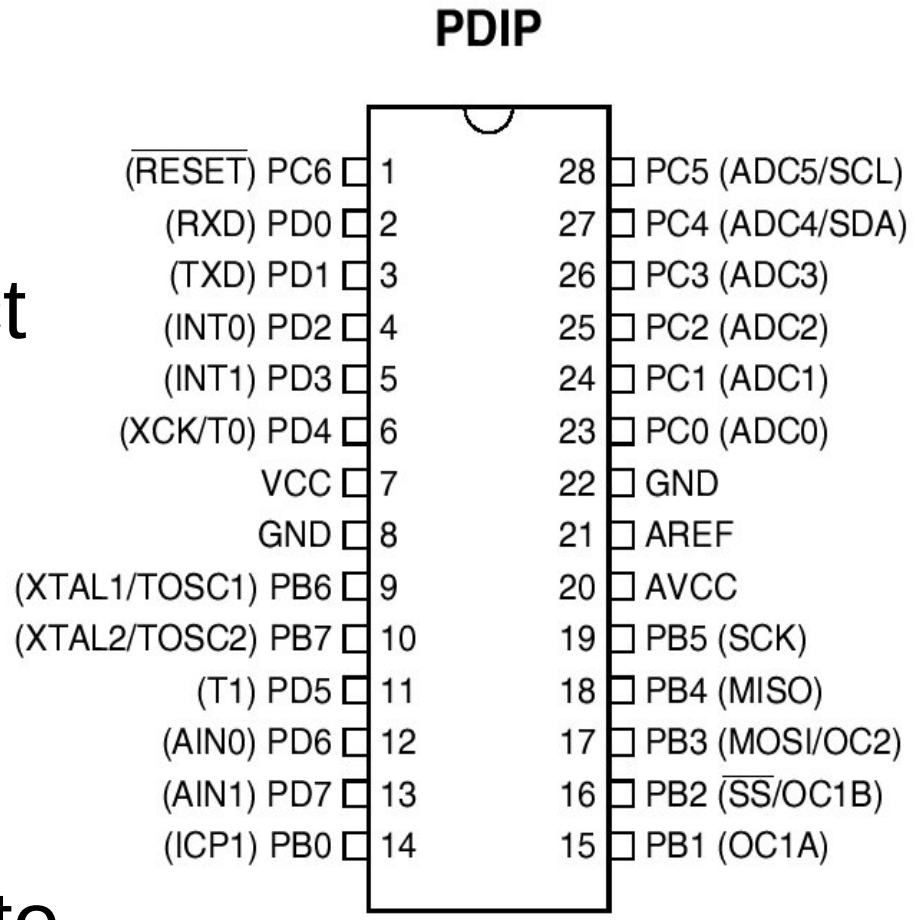
A2D in the Mega8

AVCC: connect to +5V

AREF: (optional) connect to +5V

- Measuring voltages between 0 and +5V

Connect input analog signal to the appropriate ADC pin



A Code Example

```
// Initialize adc
adc_set_reference(ADC_REF_AREF);       // Use the AREF reference pin
adc_set_adlar(0);                      // For our purposes, always use 0
adc_set_prescalar(ADC_PRESCALAR_128);  // Necessary with 16MHz clock
                                       //  and 10 bit resolution
// Turn on ADC Converter
adc_set_enable(ADC_ENABLE);

        :
        :
long val;

// Can do the following an arbitrary number of times

adc_set_channel(ADC_CHANNEL_0);        // ADC0
// Actually start a conversion
adc_start_conversion();

<Could go off and do something else for a while>

val = adc_read();  // Read analog value
```

Analog Conversion Notes

- All functions are provided in oulib
- See oulib.h for the definition of constants

- Can get to the example code from the Atmel HowTo

  www.cs.ou.edu/~fagg/classes/general/atmel

Analog Conversion Notes

- Setting the maximum voltage:

```
adc_set_reference(ADC_REF_AREF);       // Use the AREF reference pin
```

- Can also used a fixed voltage (+2.56V):

```
adc_set_reference(ADC_REF_2p56V);
```

Analog Conversion Notes

- Determining how fast the conversion requires:

```
adc_set_prescalar(ADC_PRESCALAR_128);  // Necessary with 16MHz clock
                                       //  and 10 bit resolution
```

- Conversion requires:

  128 * 15 / 16000000 seconds

  – Can convert faster, but may not get the full 10-bit resolution

Analog Conversion Notes

- Reading out the value:

```
val = adc_read();       // Read the analog value
```

- Blocks until conversion is complete
- Will return a value between 0 and 0x3FF (1023)

Analog Conversion Notes

- Can configure the mega8 to interrupt on conversion completion

Other Devices

- External devices are available that will perform D2A and A2D
- Often interface to the microprocessor via I^2C or SPI
  - (these are high-speed serial protocols)
- Many options
  - Resolution
  - Conversion speed
  - Number of channels