Hardware Interfacing (Input)

• **Switches—Bouncing**
  • A switch is a mechanical device that rarely opens/closes cleanly.

Any problems here?
Analog to Digital Conversion

- **Specifications—Resolution**
  - Is a function of the range and the number of bits (N)

\[
\text{Resolution} = Q = \frac{V_{\text{max}} - V_{\text{min}}}{2^N} = \frac{\text{Range}}{2^N}
\]

- 12-bit resolution for 0-10V range:
  \[
  \frac{10}{2^{12}} = \frac{10}{4096} = 2.44\text{mV}
  \]

- 16-bit resolution for -10-10V range:
  \[
  \frac{20}{2^{16}} = 0.305\text{mV}
  \]

- **Quantization error**
  - Due to finite resolution:
    \[
    U_Q = \frac{1}{2} \left( \frac{\text{Range}}{2^N} \right)
    \]
Analog to Digital Converters

- **Successive Approximation**
  - Relatively fast and inexpensive
  - Uses a D/A (DAC) in a feedback loop
  - Successively tests each bit
  - requires N steps to complete (typically 15-20 $\mu$s)

- input signal is < $\frac{1}{2}$ FS (MSB = 0)
- input signal is > $\frac{1}{4}$ FS ($2^{nd}$ bit = 1)
- input signal is > ($\frac{1}{4} + \frac{1}{8}$) FS ($3^{rd}$ bit = 1)
- input signal is < ($\frac{1}{4} + \frac{1}{8} + \frac{1}{16}$)FS (LSB = 0)
Multiplexers

CH1
CH2
CH3

MSP430 Analog to Digital Conversion

© 2013 Stephen R. Platt
MSP430 ADC10

This table is found in the device datasheet.
### ADC10 Configuration

<table>
<thead>
<tr>
<th>Register</th>
<th>Short Form</th>
<th>Register Type</th>
<th>Address</th>
<th>Initial State</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADC10 input enable register 0</td>
<td>ADC10AE0</td>
<td>Read/write</td>
<td>04Ah</td>
<td>Reset with POR</td>
</tr>
<tr>
<td>ADC10 input enable register 1</td>
<td>ADC10AE1</td>
<td>Read/write</td>
<td>04Bh</td>
<td>Reset with POR</td>
</tr>
<tr>
<td>ADC10 control register 0</td>
<td>ADC10CTL0</td>
<td>Read/write</td>
<td>01B0h</td>
<td>Reset with POR</td>
</tr>
<tr>
<td>ADC10 control register 1</td>
<td>ADC10CTL1</td>
<td>Read/write</td>
<td>01B2h</td>
<td>Reset with POR</td>
</tr>
<tr>
<td>ADC10 memory</td>
<td>ADC10MEM</td>
<td>Read</td>
<td>01B4h</td>
<td>Unchanged</td>
</tr>
<tr>
<td>ADC10 data transfer control register 0</td>
<td>ADC10DTC0</td>
<td>Read/write</td>
<td>048h</td>
<td>Reset with POR</td>
</tr>
<tr>
<td>ADC10 data transfer control register 1</td>
<td>ADC10DTC1</td>
<td>Read/write</td>
<td>049h</td>
<td>Reset with POR</td>
</tr>
<tr>
<td>ADC10 data transfer start address</td>
<td>ADC10SA</td>
<td>Read/write</td>
<td>01BCh</td>
<td>0200h with POR</td>
</tr>
</tbody>
</table>

- **ADC10CTL0 & ADC10CTL1**—ADC10 Control Registers
  - Used to configure how the ADC runs
- **ADC10AE0 & ADC10AE1**—Analog Input Enable Registers
  - Specifies which pins should be used for analog input
- **ADC10MEM**—Conversion Memory Register
  - This is where the conversion result is stored
- **ADC10DTC0 & ADC10DTC1**—Data Transfer Control Registers
  - Used to control direct memory transfer of conversion results
- **ADC10SA**—Start Address Register for Data Transfer
  - Specifies the starting address in memory where results are transferred
We must tell ADC10 what voltage reference source to use. For example, if we want to use the supply voltage and ground as the references, set bits 15-13 = %000 = SREF_0 (defined in header file).

Set these bits to ensure accurate conversion voltage.
Sample and Hold

- Section 22.2.5.1 in MSP430x2xx Users Guide

\[ t_{\text{sample}} > (R_S + R_I) \times \ln(2^{11}) \times C_I \]
ADC10—ADC10CTL0 Control Register 0

<table>
<thead>
<tr>
<th>Bit 15</th>
<th>Bit 14</th>
<th>Bit 13</th>
<th>Bit 12</th>
<th>Bit 11</th>
<th>Bit 10</th>
<th>Bit 9</th>
<th>Bit 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>SREFx</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
</tr>
<tr>
<td>ADC10SHTx</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
</tr>
<tr>
<td>ADC10R</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
</tr>
<tr>
<td>REFOUT</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
</tr>
<tr>
<td>REFBURST</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
</tr>
</tbody>
</table>

**REF2_5V** Bit 6
Reference-generator voltage. REFON must also be set.
- 0: 1.5 V
- 1: 2.5 V

**REFON** Bit 5
Reference generator on
- 0: Reference off
- 1: Reference on

**ADC10ON** Bit 4
ADC10 on
- 0: ADC10 off
- 1: ADC10 on

**ADC10IE** Bit 3
ADC10 interrupt enable
- 0: Interrupt disabled
- 1: Interrupt enabled

**ADC10IFG** Bit 2
ADC10 interrupt flag. This bit is set if ADC10MEM is loaded with a conversion result. It is automatically reset when the interrupt request is accepted, or it may be reset by software. When using the DTC this flag is set when a block of transfers is completed.
- 0: No interrupt pending
- 1: Interrupt pending

Don’t forget to turn the ADC on!

You do want interrupts, don’t you?

Internal voltage reference control.
ADC10—ADC10CTL1 Control Register 1

INCHx: Input channel select. These bits select the channel for a single-conversion or the highest channel for a sequence of conversions.

- 0000: A0
- 0001: A1
- 0010: A2
- 0011: A3
- 0100: A4
- 0101: A5
- 0110: A6
- 0111: A7
- 1000: V_{REF+}
- 1001: V_{REF}/V_{REF-}
- 1010: Temperature sensor
- 1011: (V_{CC} - V_{SS}) / 2
- 1100: (V_{CC} - V_{SS}) / 2, A12 on MSP430x22xx devices
- 1101: (V_{CC} - V_{SS}) / 2, A13 on MSP430x22xx devices
- 1110: (V_{CC} - V_{SS}) / 2, A14 on MSP430x22xx devices
- 1111: (V_{CC} - V_{SS}) / 2, A15 on MSP430x22xx devices

Choose your channels.
Choose your clock rate.

Choose your clock.

Specify your conversion sequence.
ADC10—ADC10AE0 A I Enable Control Reg 0

ADC10AE0x Bits

ADC10 analog enable. These bits enable the corresponding pin for analog input. BIT0 corresponds to A0, BIT1 corresponds to A1, etc.

- 7-0
  - 0: Analog input disabled
  - 1: Analog input enabled

ADC10—ADC10MEM Conversion Memory Reg

Conversion Results

The 10-bit conversion results are right justified, straight-binary format. Bit 9 is the MSB. Bits 15-10 are always 0.
MSP430 Analog to Digital Conversion

MSP430F2272 Project Creator 3.0

ME 461 - S. R. Platt
Fall 2010
Lab #3 Example Code

Based on code written by: Steve Kersey
College of Engineering Control Systems Lab
University of Illinois at Urbana-Champaign

A single sample is made on A1 with reference to AVcc.
Software sets ADC10SC to start sample and conversion -- ADC10SC
automatically cleared at EOC. ADC10 internal oscillator times sample
and conversion. Print voltage corresponding to ADC value in MAIN().

#include "msp430x22x2.h"
#include "UART.h"

unsigned v_adc = 0

void main(void) {
    DCOCTL = CALDCO_16MHZ;
    BCSCTL1 = CALBC1_16MHZ;

    // Timer A Config
    TACCTL0 = CCIE;
    TACCMOD = 16000;
    TACTL = TASSEL_2 + MC_1;

    // ADC10 Config
    ADC10CTL0 |= SREF_0 + ADC10SHT_2 + ADC10ON + ADC10IE;
    ADC10CTL1 = INCH_1 + CONSEQ_0 + ADC10SSELE_0 + SHS_0;
    ADC10AEO = 0x3;
    ADC10CTL0 |= ENC;

    // Example Code:
    // Configure ADC
    ADC10CTL0 = SREF_0 + ADC10SHT_2 + ADC10ON + ADC10IE;
    ADC10CTL1 = INCH_1 + CONSEQ_0 + ADC10SSELE_0 + SHS_0;

    // Start ADC conversion
    ADC10CTL0 |= ENC;

    // Read ADC value
    v_adc = ADC10C0H;

    // Print ADC value
    printf("ADC value: %d\n", v_adc);
}

SREF_0 = 0 * 0x2000 = 0x0000 (i.e., Vcc & Vss)
ADC10SHT_2 = 2 * 0x800 = 0x1000
ADC10ON = 0x0010
ADC10IE = 0x0008
= 0x1018 = %0001 0000 0001 1000

SREFx
ADC10ON
ADC10SHTx
ADC10IE
Based on code written by: Steve Kerse
College of Engineering Control Systems Lab
University of Illinois at Urbana-Champaign

A single sample is made on A1 with reference to AVcc.
Software sets ADC10SC to start sample and conversion -- ADC10SC
automatically cleared at EOC. ADC10 internal oscillator times sample
and conversion. Print voltage corresponding to ADC value in MAIN().

```
#include "msp430x22x2.h"
#include "UART.h"

unsigned v_adc = 0;

void main(void) {
    DCOCTL = CALDCO_16MHZ;
    BCSC7L1 = CALBC1_16MHZ;

    // Timer A Config
    TACCTL0 = CCIE;
    TACCR0 = 16000;
    TACTL = TASSEL_2 + MC_A;

    // ADC10 Config
    ADC10CTL0 = SRC0 + ADC10SHT_2 + ADC10CN + ADC10IE;
    ADC10CTL1 = INCH_1 + CONSEQ_0 + ADC1OSSEL_0 + SHS_0;
    ADC10AEQ = 0x3;
    ADC10CTL0 |= ENC;
```
## ADC10—ADC10CTL1 Control Register 1

<table>
<thead>
<tr>
<th>Bit 15-12</th>
<th>Bit 11-8</th>
<th>Bit 7-4</th>
<th>Bit 3-0</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCHx</td>
<td>SHSx</td>
<td>ADC10DF</td>
<td>ISSH</td>
</tr>
</tbody>
</table>

**INCHx**

- Bits 15-12: Input channel select. These bits select the channel for a single-conversion or the highest channel for a sequence of conversions.
  - 0000: A0
  - 0001: A1
  - 0010: A2
  - 0011: A3
  - 0100: A4
  - 0101: A5
  - 0110: A6
  - 0111: A7
  - 1000: VREF+
  - 1001: VREF-
  - 1010: Temperature sensor
  - 1011: (VCC – VSS) / 2
  - 1100: (VCC – VSS) / 2, A12 on MSP430x22xx devices
  - 1101: (VCC – VSS) / 2, A13 on MSP430x22xx devices
  - 1110: (VCC – VSS) / 2, A14 on MSP430x22xx devices
  - 1111: (VCC – VSS) / 2, A15 on MSP430x22xx devices

Choose your channels.
Integrated Temperature Sensor

- Set INCHx = 1010
- Set $t_{\text{sample}} \geq 30 \ \mu\text{S}$

Typical Temperature Sensor Transfer Function

In general, all sensors must be calibrated.

Do not trust the datasheets!