# ECE/CS 5780/6780: Embedded System Design

Chris J. Myers

Lecture 19: Analog-to-Digital Conversion



## ADC Parameters

- *Precision* is number of distinguishable ADC inputs.
- *Range* is maximum and minimum ADC inputs.
- *Resolution* is change in input that causes digital output to change by 1.

$$\text{Range (volts)} = \text{Precision (alternatives)} \cdot \text{Resolution (volts)}$$

- *Accuracy* usually given for entire instrument (including transducer, analog circuit, ADC, and software).
- ADC is *monotonic* if it has no missing codes.
- ADC is *linear* if resolution is constant through the range.
- ADC *speed* is time to convert.



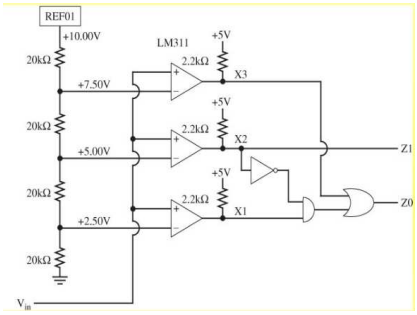
## Common Encoding Schemes

| Unipolar codes | Straight binary | Complementary binary |
|---|---|---|
| +5.00 | 1111,1111 | 0000,0000 |
| +2.50 | 1000,0000 | 0111,1111 |
| +0.02 | 0000,0001 | 1111,1110 |
| +0.00 | 0000,0000 | 1111,1111 |

| Bipolar codes | Offset binary | 2s Complement binary |
|---|---|---|
| +5.00 | 1111,1111 | 0111,1111 |
| +2.50 | 1100,0000 | 0100,0000 |
| +0.04 | 1000,0000 | 0000,0001 |
| +0.00 | 1000,0000 | 0000,0000 |
| -2.50 | 0100,0000 | 1100,0000 |
| -5.00 | 0000,0000 | 1000,0000 |



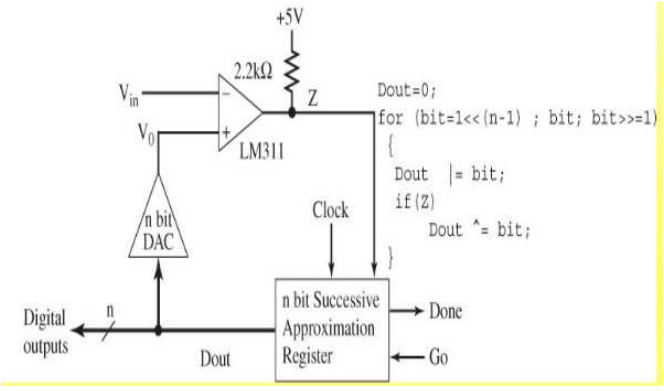
## Two-Bit Flash ADC




| $V_{in}$ | X3 | X2 | X1 | Z1 | Z0 |
|---|---|---|---|---|---|
| $2.5 > V_{in}$ | 0 | 0 | 0 | 0 | 0 |
| $5.0 > V_{in} \geq 2.5$ | 0 | 0 | 1 | 0 | 1 |
| $7.5 > V_{in} \geq 5.0$ | 0 | 1 | 1 | 1 | 0 |
| $V_{in} \geq 7.5$ | 1 | 1 | 1 | 1 | 1 |

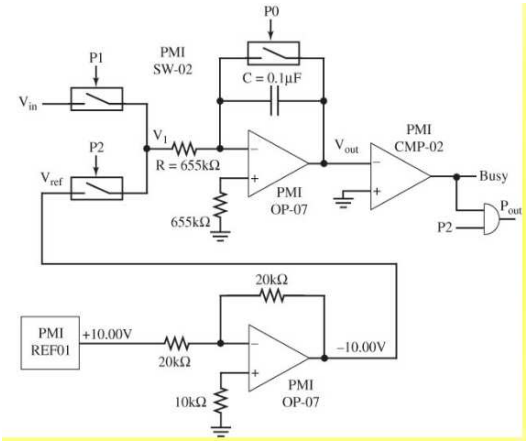


## Successive Approximation ADC

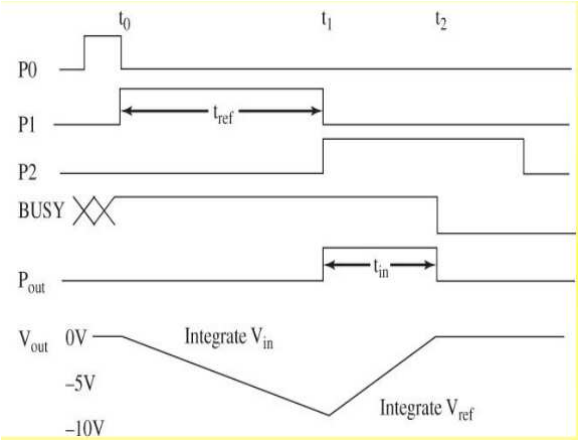





## Sixteen-Bit Dual Slope ADC



## Waveforms During Dual Slope ADC Conversion



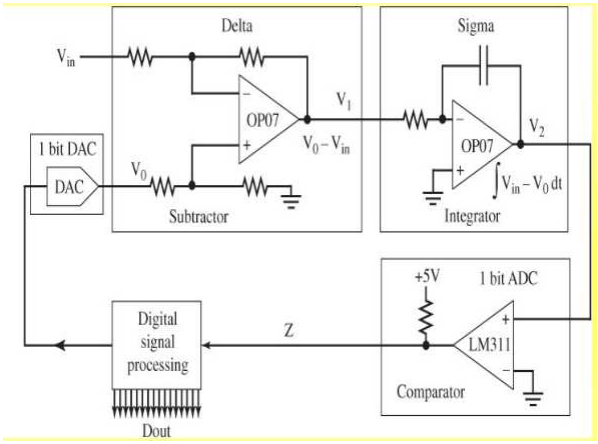



## Dual Slope ADC Conversion: Theory

$$
\begin{aligned}
V_{out}(t_0) &= 0 \\
t_{ref} &= t_1 - t_0 = 65,535\mu s \\
V_{out}(t_1) &= V_{out}(t_0) - \frac{1}{RC}\int_{t_0}^{t_1} V_{in}(s)ds = -\frac{1}{RC}V_{in}t_{ref} \\
V_{out}(t_2) &= 0 \\
t_{in} &= t_2 - t_1 \\
V_{out}(t_2) &= V_{out}(t_1) - \frac{1}{RC}\int_{t_1}^{t_2} V_{ref}(s)ds = V_{out}(t_1) - \frac{1}{RC}V_{ref}t_{in} = 0 \\
0 &= -\frac{1}{RC}V_{in}t_{ref} - \frac{1}{RC}V_{ref}t_{in} \\
0 &= V_{in}t_{ref} + V_{ref}t_{in} \\
V_{in} &= -V_{ref}\frac{t_{in}}{t_{ref}} = 10V\frac{t_{in}}{65,535\mu s}
\end{aligned}
$$



## Sigma Delta ADC



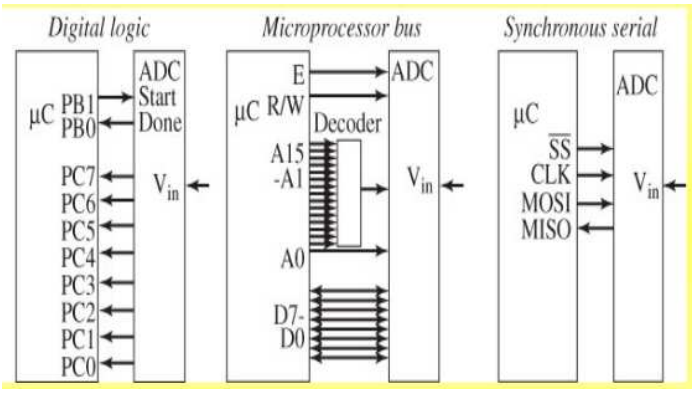



## Software Implementation of Sigma Delta ADC

```
unsigned char DOUT; // 8-bit sample
unsigned char SUM;  // number of times Z=0 and V0=1
unsigned char CNT;  // 8-bit counter
void interrupt 13 TOC5handler(void){
  TFLG1 = OC5;        // ack C5F
  TC5 = TC5+rate;     // interrupt 256 times faster
  if(Z())             // check input
    DACout(0);        // too high, set D/A output, V0=0
  else {
    DACout(1);        // too low, set D/A output, V0=+5v
    SUM++;
  }
  if(++CNT==0){       // end of 256 loops?
    DOUT = SUM;       // new sample
    SUM = 0;          // get ready for the next
  }}
```

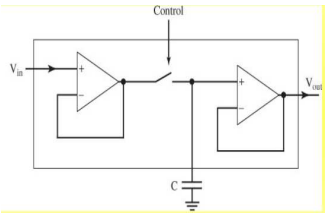


## ADC Interface






## Sample and Hold

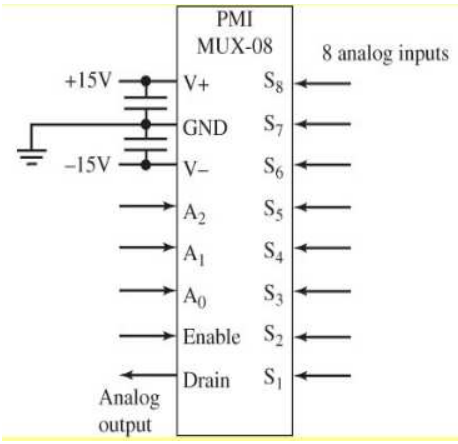



1. Should use polystyrene capacitor because of its high insulation resistance and low dielectric absorption.
2. A larger value of C decreases (improves) droop rate. If droop current is $I_{DR}$, then droop rate is:

$$\frac{dV_{out}}{dt} = \frac{I_{DR}}{C}$$

3. A smaller C decreases (improves) acquisition time.

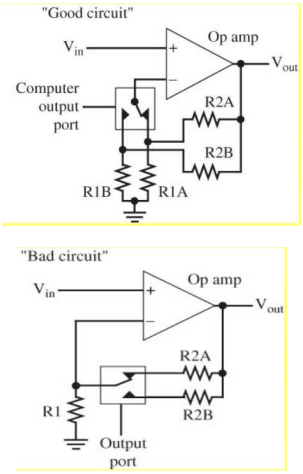
## BiFET Analog Multiplexer




| Enable | $A_2$ | $A_1$ | $A_0$ | Drain |
|---|---|---|---|---|
| 0 | X | X | X | None |
| 1 | 1 | 1 | 1 | $S_8$ |
| 1 | 1 | 1 | 0 | $S_7$ |
| 1 | 1 | 0 | 1 | $S_6$ |
| 1 | 1 | 0 | 0 | $S_5$ |
| 1 | 0 | 1 | 1 | $S_4$ |
| 1 | 0 | 1 | 0 | $S_3$ |
| 1 | 0 | 0 | 1 | $S_2$ |
| 1 | 0 | 0 | 0 | $S_1$ |

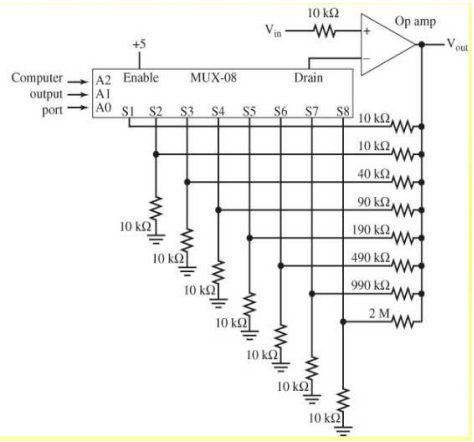


## Bilateral Switch



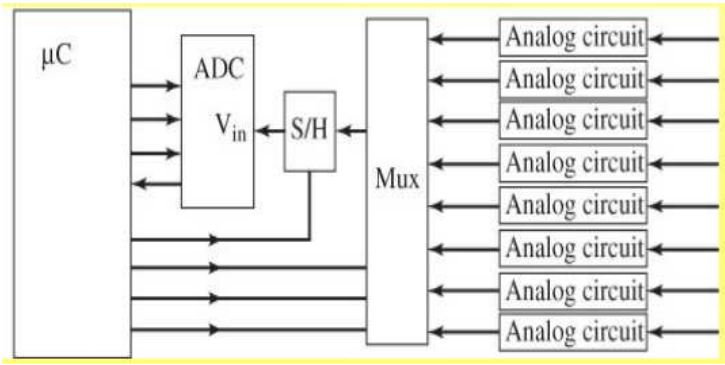



## Variable-Gain Amplifier



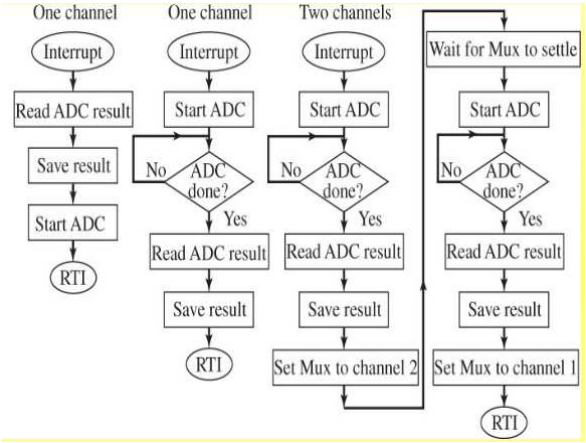



## ADC Block Diagram



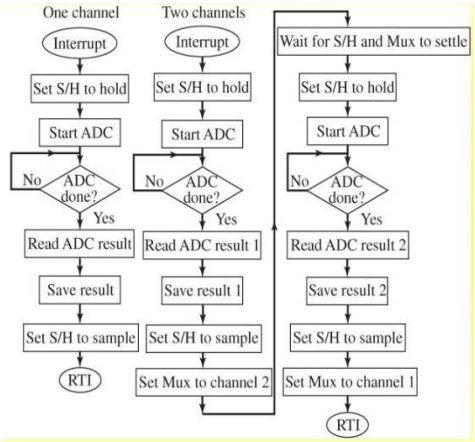



## ADC Interrupt Software Without S/H

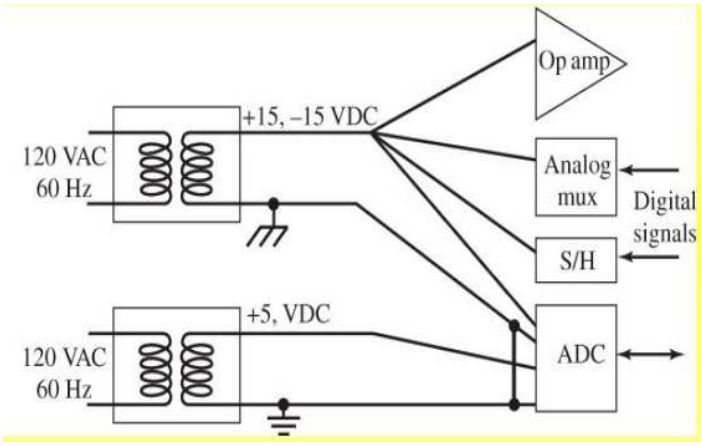





## ADC Interrupt Software With S/H

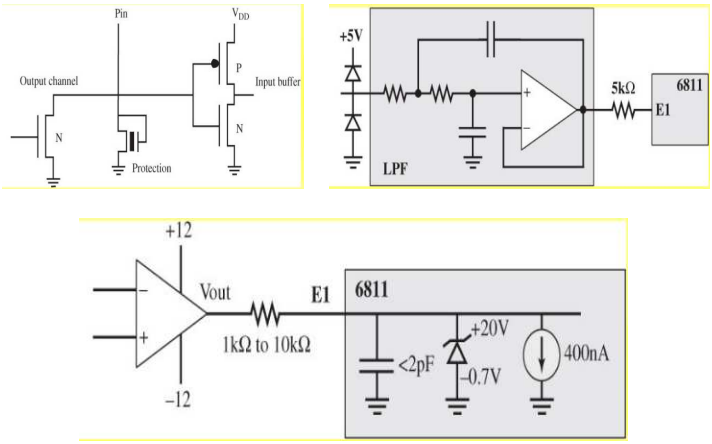


## Power and Grounding for the ADC System






## Input Protection for CMOS Analog Inputs






## Internal ADCs

- 6812 has built-in ADCs with following features:
  - Eight-channel operation.
  - 8-bit or 10-bit resolution.
  - Successive approximation conversion technique.
  - Clock and charge pump to create higher voltages.
  - Two operation modes: single sequence of conversions then stop, and continuous conversion.
  - Supports multiple conversion of single channel, and one conversion each for group of channels.
  - External $V_{RH}$, $V_{RL}$ analog high/low references.



## 6812 ADC System: Setup

- 8 pins of Port AD can be individually configured as analog or digital inputs using the ATDDIEN register (1 for digital, 0 for analog).
- If pin is digital, DDRAD register is used to set pins direction.
- Can use 8-bit or 10-bit resolution which is selected by setting the SRES8 bit (1=8-bit) in the ADTCTL4 register
- ATDCTL2 register:
  - ADC system is enabled by setting ADPU to 1.
  - Interrupts are enabled by setting ASCIE to 1.
  - ASCIF flag is set 1 when conversion sequence is complete, if ASCIE is 1.



## 6812 ADC System: Conversions

- When ADC is triggered, it performs 1 to 8 conversions.
- Number of conversions is selected by the value written into the S8C, S4C, S2C, and S1C bits of ATDCTL3 (values of 0, 8-15 are all 8).
- The channel used is selected by the CC, CB, CA bits of ATDCTL5.
- All conversions can be on one channel or on multiple channels if MULT in ATDCTL5 is set (channel sequence determined by S8C, S4C, S2C, S1C).
- ATDSTAT0 register:
  - SCF flag in ATDSTAT0 is set to 1 when conversion is complete.
  - CC2, CC1, and CC0 bits are a counter to show conversion progress.
- ATDSTAT1 register contains CCFn flag bits for each conversion.



## 6812 ADC System: Triggers

- Conversion can be triggered in three ways:
  - Writing to ATDCTL5 and when done SCF bit in ATDSTAT0 is set.
  - Trigger continuously if SCAN in ATDCTL5 is set.
  - Using an external trigger connected to PAD7.
- External trigger enabled when ETRIGE bit in ATDCTL2 is set.

| ETRIGLE | ETRIGP | External trigger mode |
|---|---|---|
| 0 | 0 | Falling edge of PAD7 |
| 0 | 1 | Rising edge of PAD7 |
| 1 | 0 | Convert while PAD7 is low |
| 1 | 1 | Convert while PAD7 is high |

## 6812 ADC System: Sample Period

- Determined by ATDCTL4 register and E clock.
- Sample done in two phases:
  - Phase one transfers sample to ADC's storage node.
  - Phase two attaches external analog signal to the storage node.

| SMP1 | SMP0 | First sample | Second sample | Total |
|---|---|---|---|---|
| 0 | 0 | 2 ADC clocks | 2 ADC clocks | 4 ADC clocks |
| 0 | 1 | 2 ADC clocks | 4 ADC clocks | 6 ADC clocks |
| 1 | 0 | 2 ADC clocks | 8 ADC clocks | 10 ADC clocks |
| 1 | 1 | 2 ADC clocks | 16 ADC clocks | 18 ADC clocks |

- If $m$ is 5-bit number formed by PRS4-0 and $f_E$ is E clock frequency:

$$\text{ATD clock frequency} = \frac{1}{2}\frac{f_E}{(m+1)}$$



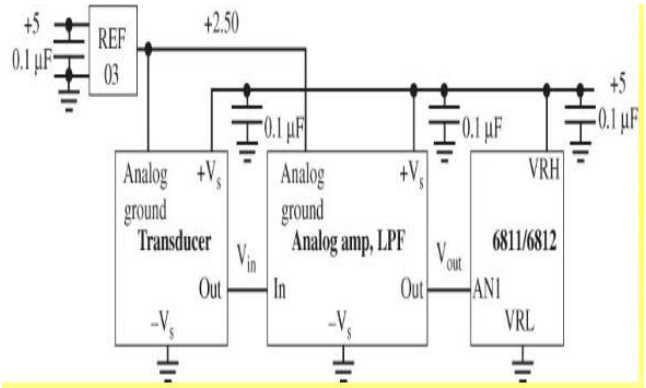
## 6812 ADC System: Binary Formats

- Results returned in the 16-bit ATDDR0 to ATDDR7 registers.
- 10-bit results can unsigned or signed (DSGN=1 in ATDCTL5).
- Results can be left or right justified (DJM=1 in ATDCTL5).

| Input (V) | 8-bit(u) | 10-bit(ur) | 10-bit (ul) | 10-bit (sr) | 10-bit (sl) |
|---|---|---|---|---|---|
| 0.000 | $00 | $0000 | $0000 | $FE00 | $8000 |
| 0.005 | $00 | $0001 | $0040 | $FE01 | $8040 |
| 0.020 | $01 | $0004 | $0100 | $FE04 | $8100 |
| 2.500 | $80 | $0200 | $8000 | $0000 | $0000 |
| 3.750 | $C0 | $0300 | $C000 | $0100 | $4000 |
| 5.000 | $FF | $03FF | $FFC0 | $01FF | $7FC0 |



## ADC Example



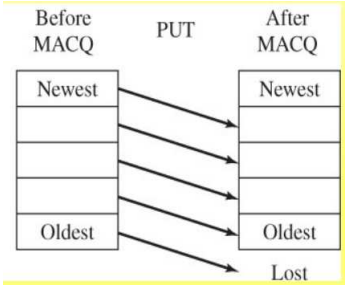


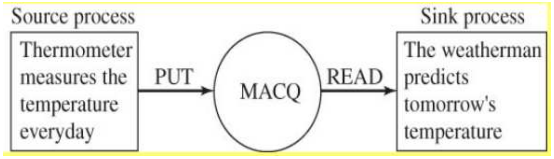

## ADC Software

```
void ADC_Init(void){
  ATDCTL2 = 0x80; // enable ADC
  ATDCTL3 = 0x08;
  ATDCTL4 = 0x05; // 10-bit, divide by 12
}
unsigned short ADC_In(unsigned short chan){
  ATDCTL5 = (unsigned char)chan;  // start sequence
  while((ATDSTAT1&0x01)==0){};    // wait for CCF0
  return ATDDR0;
}
```



## Multiple-Access Circular Queue






## Computing a Derivative

- Simple approach:

$$d(n) = \frac{x(n)-x(n-1)}{\Delta t}$$

- Approach that is more robust to noise:

$$d(n) = \frac{x(n)-3x(n-1)-3x(n-2)-x(n-3)}{\Delta t}$$

## Software for First Derivative Using a MACQ

```
#define RATE 2000
#define OC5  0x20
unsigned short x[4];  // MACQ (mV)
unsigned short d;     // derivative (V/s)
void interrupt 13 TOC5handler(void){
  TC5 = TC5+RATE;      // Executed every 1 ms
  TFLG1 = 0x20;        // ack OC5F
  x[3] = x[2];         // shift MACQ data
  x[2] = x[1];         // units of mV
  x[1] = x[0];
  x[0] = ADC_In(0x85); // current data, from Channel 5
  d = x[0]+3*x[1]-3*x[2]-x[3]; // mV/ms
}
```