

ADC – Analogue to Digital Conversion

- 13 pins support ADC operations
- ANn
- AN2 and AN3 may be used to provide reference voltages V_{REF-} and V_{REF+} for ADC. Can't be then used for ADC.

DIO – 1 bit (2^1 values) voltmeter

1 – 5 V

0 – 0 V

ADC – 10 bit (2^{10} values) voltmeter

1023 – 5 V

1022

•
•
•

1

0 – 0 V

- What will 3.125 V be?
- What voltage is 632?
- ADC is digital or grainy
- Voltages are continuous or smooth
- Conversions will always be imprecise
- Digital values are bins. If voltage falls in a bin, assign that number.

- How big is each bin?

$$\frac{5V - 0V}{2^{10}} = 0.00488V$$

- Which bin would have 2.272 V?

$$\frac{2.272V}{5V - 0V} \times 2^{10} = 465$$

- What voltages lie in bin 762?

$$0 = [0 \leftrightarrow 0.00488V]$$

$$1 = [0.00488V \leftrightarrow 0.00976V]$$

$$\vdots = \vdots$$

$$b = [b \times 0.00488V \leftrightarrow (b + 1) \times 0.00488V]$$

$$\therefore 762 = [3.7201V \leftrightarrow 3.7256V]$$

- We ordinarily don't know V_{input} , so we should assign voltage to middle of bin and then precision is half the bin size

$$V = \left(b + \frac{1}{2} \right) \times \frac{(5V - 0V)}{2^n} \pm \frac{1}{2} \frac{(5V - 0V)}{2^n}$$

$$\therefore .762 \leftrightarrow V = 3.7231 \pm 0.0024V$$

- **Warning** Don't try to find b from known V using this formula, we find b by simply dividing V by bin size

$$b = V \div \frac{(5V - 0V)}{2^n}$$

$$b = \frac{2.272V}{5V - 0V} \times 2^{10} = 465.31 = 465$$

Check:

$$\begin{aligned} V &= \left(465 + \frac{1}{2} \right) \times \frac{(5V - 0V)}{2^{10}} \pm \frac{1}{2} \frac{(5V - 0V)}{2^{10}} \\ &= 2.2729 \pm 0.0024 \end{aligned}$$

Bin	Voltage (V)
178	
0b10 1010 1111	
0x1FA	
	1.7152
	4.6850
	-1.0037
	5.7214

We have used 0V to 5 V as the range of the ADC conversions. We actually have some control over the range.

Suppose we use 1.2 V to 4.0 V instead. What is the bin size? Convert the following.

1000	
22	
418	
	1.5121 V
	3.1780 V
	2.2222 V

Suppose we use 8-bit ADC instead on a range of 0 V to 5.0 V. What is the bin size? Convert the following.

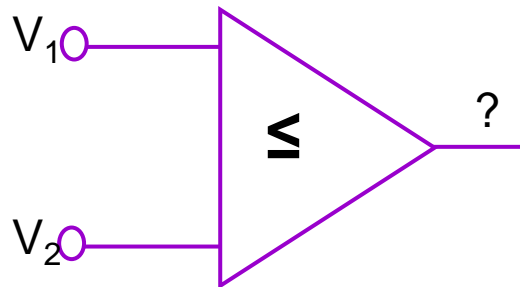
52	
112	
41	
	2.773 V
	4.245 V
	1.222 V

How ADC Works (Simplified)

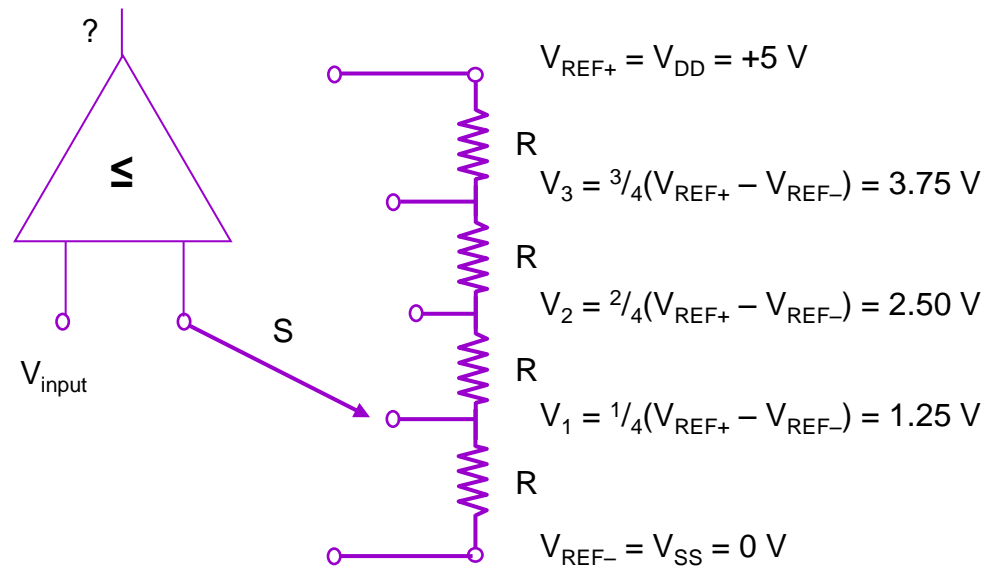
- Two elements
 - Comparator
 - Voltages to compare to (use a voltage divider)
- Algorithm (simple example is Method of Successive Approximation or the High-Low Game Strategy)

Comparator

- V_1 is input and V_2 is reference.
- Output is 0 ($V_1 < V_2$) or 1 ($V_1 \geq V_2$)

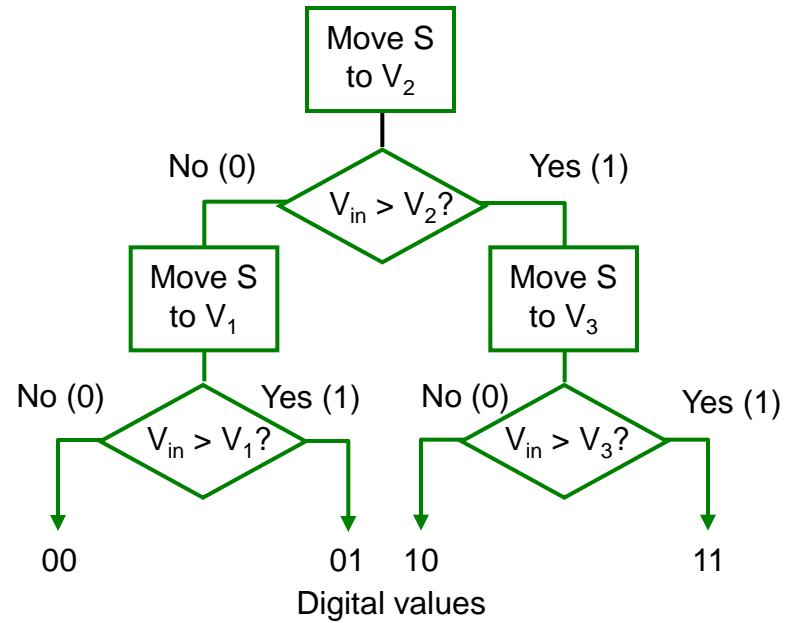


Voltage Divider to Compare to

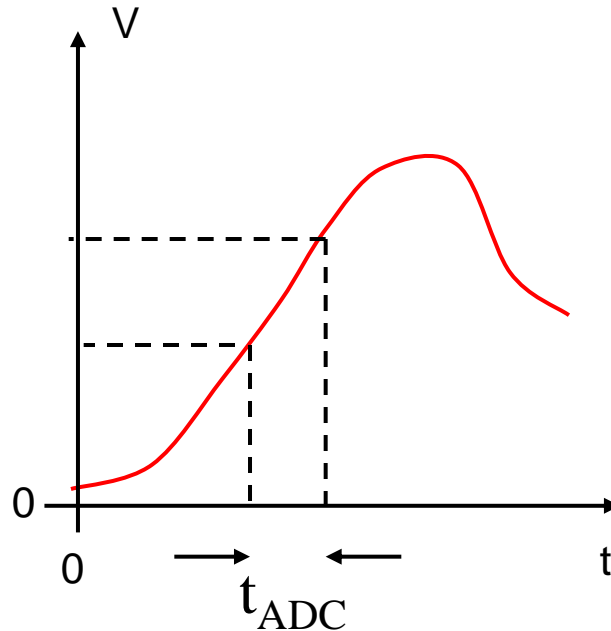


- 2 bit – 2^2 resistors; 2^n bit – 2^n resistors

Algorithm



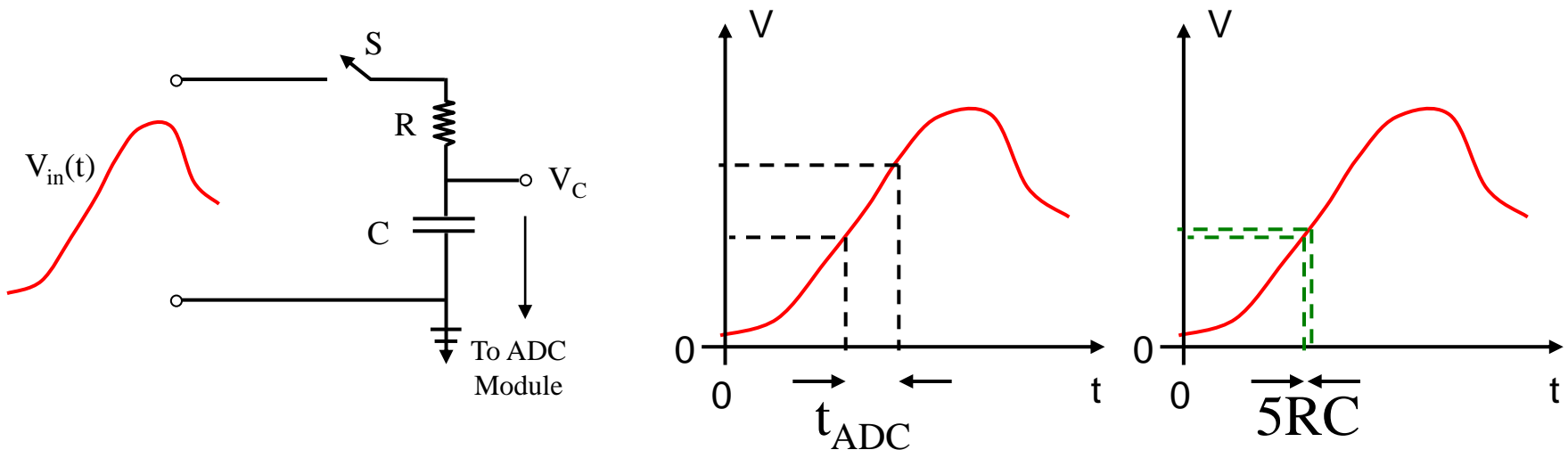
Acquisition time



- Varying voltage signals
- Cannot accurately do comparison

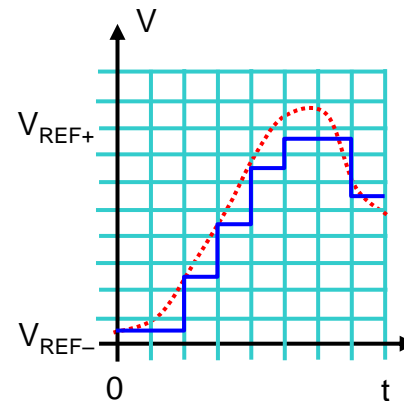
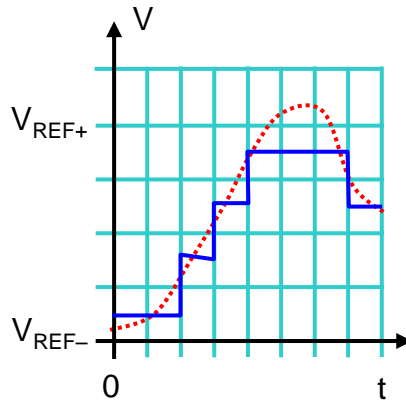
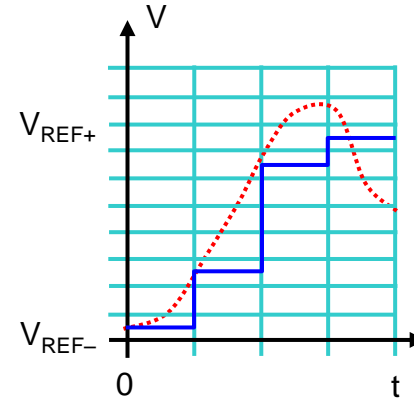
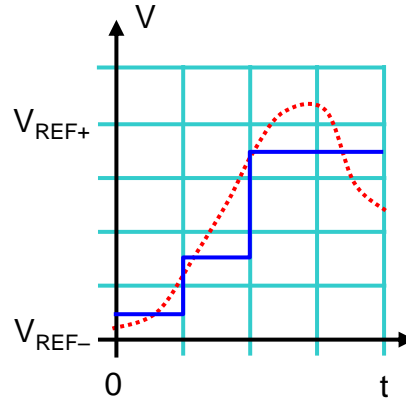
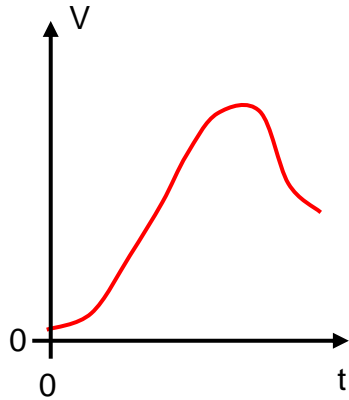
Sample and Hold Circuit

- Have an RC circuit with small value of RC (time constant).
- Will fully charge C in 5 to 10 RC.
- Want $RC \ll t_{ADC}$.
- Also want RC to be small that ΔV is small



According to the datasheet on p. 227, $C = 25$ pF and R (actually the series sum of the R , the input resistance, and the switch resistance) is about $4 \text{ k}\Omega$. What is τ ? How long does it take to fully charge the capacitor (i.e. to more than 99%)?

- Higher precision (more bits) takes more comparisons and more time. Tradoff.



Conversion Time

- Measured in units called T_{AD}
- Need $1 T_{AD} + 1 T_{AD}/\text{bit} = 11 T_{AD}$ per conversion.
- T_{AD} is fixed $\sim 0.7 \mu\text{s}$
- Capacitor charge time usually measured in T_{AD} as well.

Software

```
#include <adc.h>
```

```
OpenADC(unsigned char config, unsigned char config2,  
        unsigned char portconfig )
```

```
SetChanADC(unsigned char channel) // which input pin  
                                     // ANn to use
```

```
ConvertADC() // start conversion
```

```
BusyADC() // = 1 if working, = 0 if finished
```

```
ReadADC() // returns 10-bit ADC result
```

```
CloseADC() // finish
```

channel in SetChanADC()

- Parameter can have value `ADC_CHn` where n is 0 to 12 and refers to pin ANn.
- Note that while many pins can be ADC inputs, you can only read voltage from one pin at a time.
- Cannot be run simultaneously.

portconfig in OpenADC()

- Value 0 to 15
- Configures sets of pins as Ann
- Only certain sets available

port config	ANn												
	12	11	10	9	8	7	6	5	4	3	2	1	0
0	A	A	A	A	A	A	A	A	A	A	A	A	A
1	A	A	A	A	A	A	A	A	A	A	A	A	A
2	A	A	A	A	A	A	A	A	A	A	A	A	A
3	D	A	A	A	A	A	A	A	A	A	A	A	A
4	D	D	A	A	A	A	A	A	A	A	A	A	A
5	D	D	D	A	A	A	A	A	A	A	A	A	A
6	D	D	D	D	A	A	A	A	A	A	A	A	A
7	D	D	D	D	D	A	A	A	A	A	A	A	A
8	D	D	D	D	D	D	A	A	A	A	A	A	A
9	D	D	D	D	D	D	D	A	A	A	A	A	A
10	D	D	D	D	D	D	D	D	A	A	A	A	A
11	D	D	D	D	D	D	D	D	D	A	A	A	A
12	D	D	D	D	D	D	D	D	D	D	A	A	A
13	D	D	D	D	D	D	D	D	D	D	D	A	A
14	D	D	D	D	D	D	D	D	D	D	D	D	A
15	D	D	D	D	D	D	D	D	D	D	D	D	D

D – Digital A – Analogue Input (ADC)

config2 in OpenADC()

3 choices

- `ADC_INT_ON` or `ADC_INT_OFF` ✓
- `ADC_CHn`, n = 0 to 12, default ADC channel/pin
- `ADC_VREFPLUS_VDD`, ✓ (use VDD)
`ADC_VREFMINUS_VSS`, ✓ (use VSS)
`ADC_VREFPLUS_EXT`, (+ Ref is Pin 5/AN3)
`ADC_VREFMINUS_EXT` (– Ref is Pin 4/AN2)

config in OpenADC()

3 choices

- **ADC_LEFT_JUST** or **ADC_RIGHT_JUST** ✓
- **ADC_FOSC_n**, n = 2, 4, 8, 16, or 32

allow time for AD conversion to complete

$$T_{AD} = n \times T_{OSC}$$

- **ADC_n_TAD**, n = 0, 2, 4, 6, 12, 16, or 20

allow time for capacitor to fully charge

$$T_{CAP} = n \times T_{AD}$$

Timing Calculations

- At 32 MHz, $T_{OSC} = 0.03125 \mu s$
- $T_{AD_{required}} = 0.7 \mu s$
- $T_{AD_{required}}/T_{OSC} = 22.4$, so use
`ADC_FOSC_32`
- $T_{AD} = 32T_{OSC} = 1 \mu s$
- $T_{CAP} = 1 \mu s$, so use `ADC_2_TAD`

```

#include <p18F4525.h>
#include <adc.h>
#include <delays.h>
#include "osc.h"
void main (void)
{
    int firstADCvalue, secondADCvalue;
    set_osc_32MHz();           // change the internal oscillator frequency
    // Configure pins AN0 and AN1 only for ADC operation using
    // VDD and VSS as the references. The digital value is right
    // justified. The other values are default settings
    OpenADC( ADC_FOSC_32 & ADC_2_TAD & ADC_RIGHT_JUST,
            ADC_CH0 & ADC_VREFPLUS_VDD & ADC_VREFMINUS_VSS, 13);
    Delay10TCYx(5);          // Delay for 50TCY to stabilize
    while(1)
    {
        SetChanADC(ADC_CH0); // Next read from pin AN0
        ConvertADC();        // Start ADC operation
        while( BusyADC() ); // Wait for completion
        firstADCvalue = ReadADC(); // Read result
        SetChanADC(ADC_CH1); // Next read from pin AN1
        ConvertADC();        // Start ADC operation
        while( BusyADC() ); // Wait for completion
        secondADCvalue = ReadADC(); // Read result
    }
}

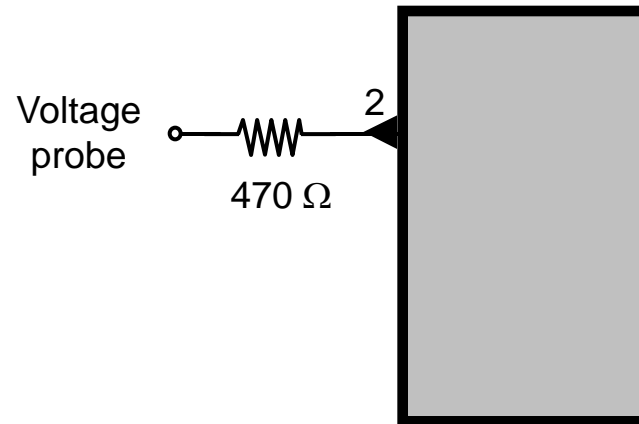
```

How much time does the AD conversion take in μs at 32 MHz?

If the MCU operates at 8MHz, what values of ADC_FOSC_n and ADC_n_TAD should be used? How much time does the conversion take in μs ?

ADC Probe Wiring

- Must buffer input pin so current is not too large.



Transducers

- Transducers are sensors that convert a physical parameter such as Temperature, pH, light intensity, etc into an electronic signal.
- An electronic signal is a change in resistance, current, or voltage.
- Signal to parameter (say V vs T for a temperature probe) may not be linear.
- May need reference points to calibrate signal, e.g. 0°C or 100°C .