

The University of Texas at Arlington

Lecture 9

ADC: Analog-to-Digital Conversion



CSE@UTA

CSE 3442/5442

Embedded Systems 1

Based heavily on slides by Dr. Gergely Záruba and Dr. Roger Walker



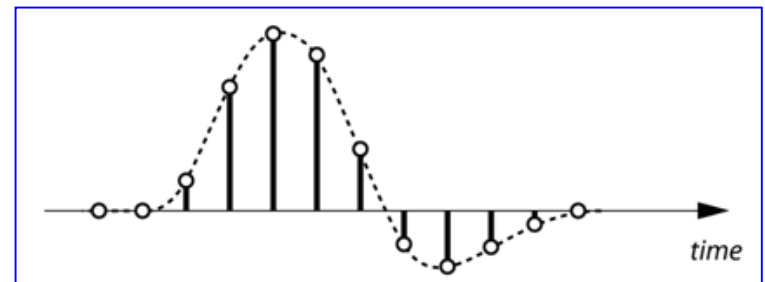
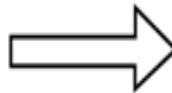
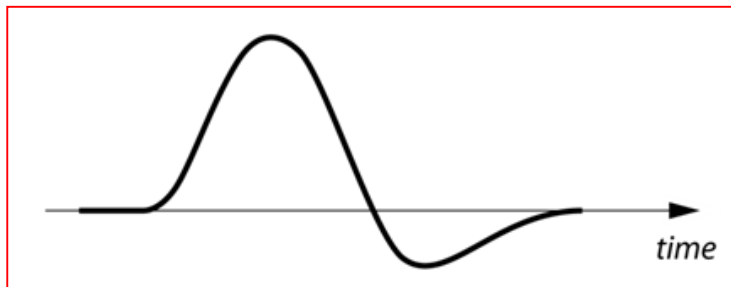
Measuring Physical Quantities

- **Digital** computers use binary **discrete values**
 - 0s and 1s, ON OFF, Logic HIGH LOW, etc.
- In the physical world most everything is **continuous (analog)**
 - voltage, current, temperature, pressure, acceleration, location, etc.
- Need to convert these **physical quantities** into **electrical quantities**

ADC

Analog-to-Digital Conversion

- Converts **analog** data to **digital** data
 - Analog Signal → Digital Value (number)
 - Continuous → Discrete





Sensors and Transducers

- **Transducer** often used interchangeably with **Sensor**
- Transducers convert a physical quantity to an electrical signal (voltage, current)
 - Temperature
 - Velocity
 - Pressure
 - Light



Sensors with Analog Outputs

- Many sensors output **analog** (only) signals where output is proportional to some kind of measured **physical quantity** (temperature, acceleration, etc.)
- **Accuracy** or **precision** of a sensor determines how close the measured (and output) value is to the “real world” value.
 - What is the smallest unit that makes a difference in the measurement
- **In order to use the outputs of analog sensors in calculations, their outputs have to be digitized.**

Example

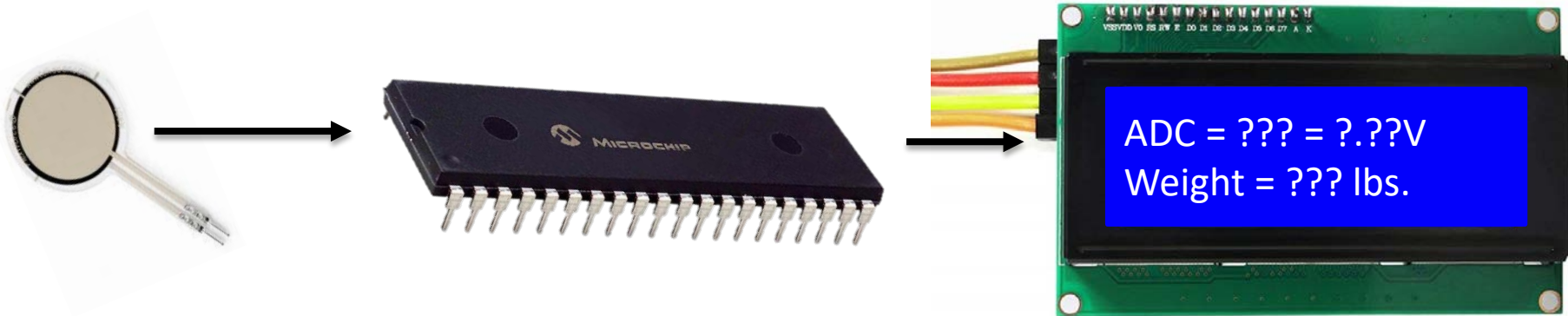


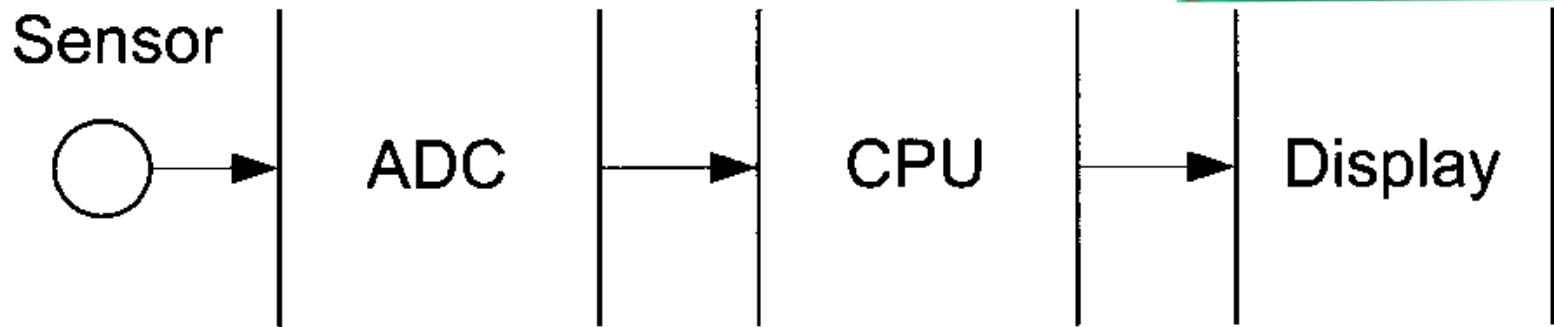
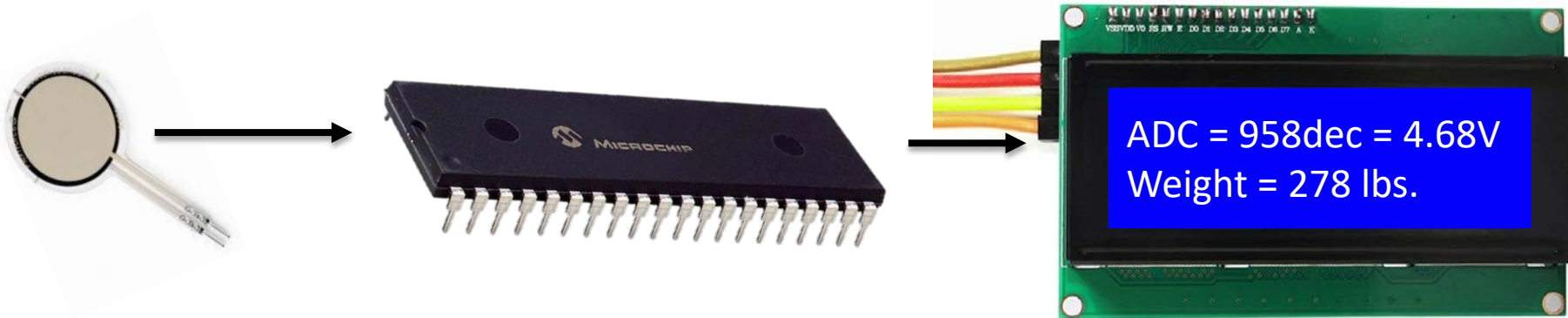
Image Sources:

<http://tutorialsmax.com/pic18f-interfacing-with-i2c-24c512-memory/>

<http://www.microdesignsinc.com/picbook/qwikflash.html>

<https://www.tekscan.com/products-solutions/force-sensors/a401>

Example

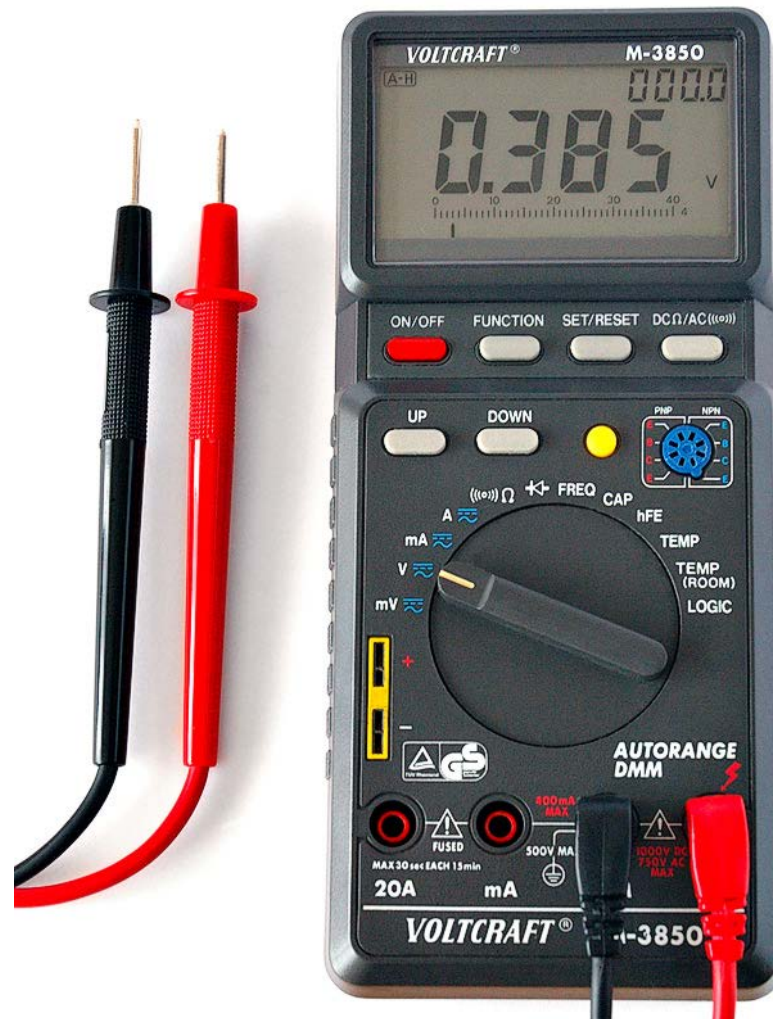


Weight → Resistance (voltage) → Discrete Number (int, float, etc.) → LED "Matrix"





Think of the ADC on your PIC as your personal Voltmeter

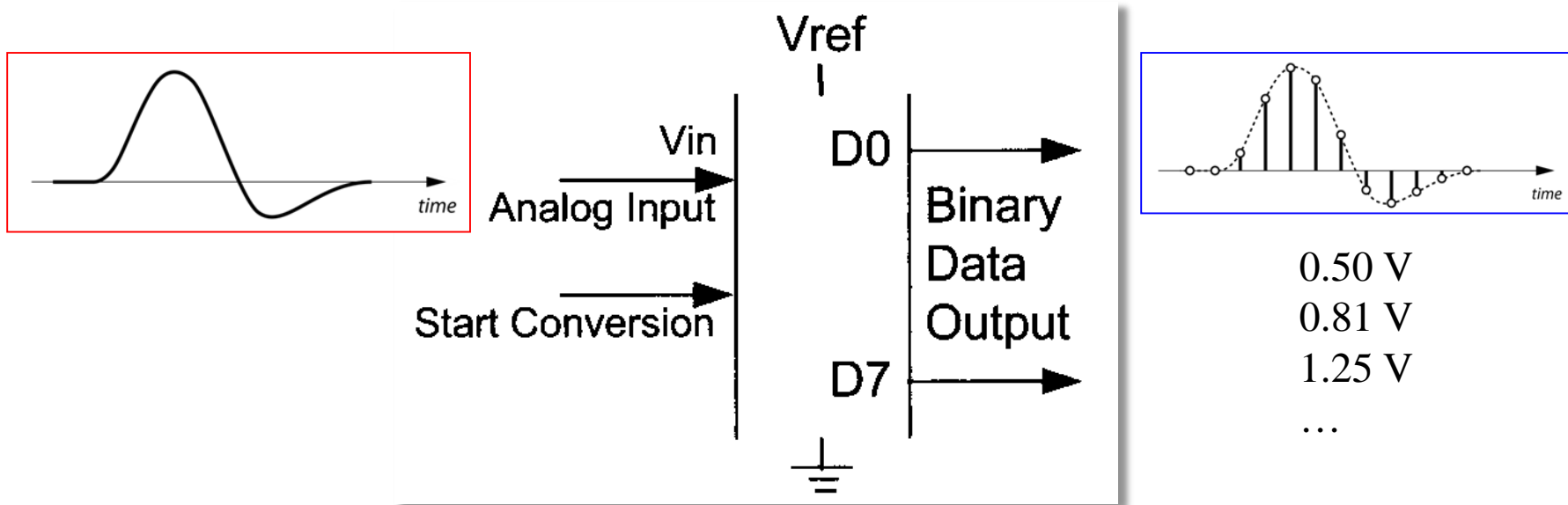




ADC Characteristics

- **Resolution**
 - usually expressed in “n-bits”
- **Conversion time**
 - how long it takes to convert from analog to digital
- **Voltage Reference**
 - what is the voltage range (min and max)
- **Linear vs. non-linear transfer**
 - equal step size vs. changing step size

8-bit ADC Block Diagram



n -bit	Number of steps	Step size (mV)
8	256	$5/256 = 19.53$
10	1,024	$5/1,024 = 4.88$
12	4,096	$5/4,096 = 1.2$
16	65,536	$5/65,536 = 0.076$

Notes: $V_{CC} = 5\text{ V}$

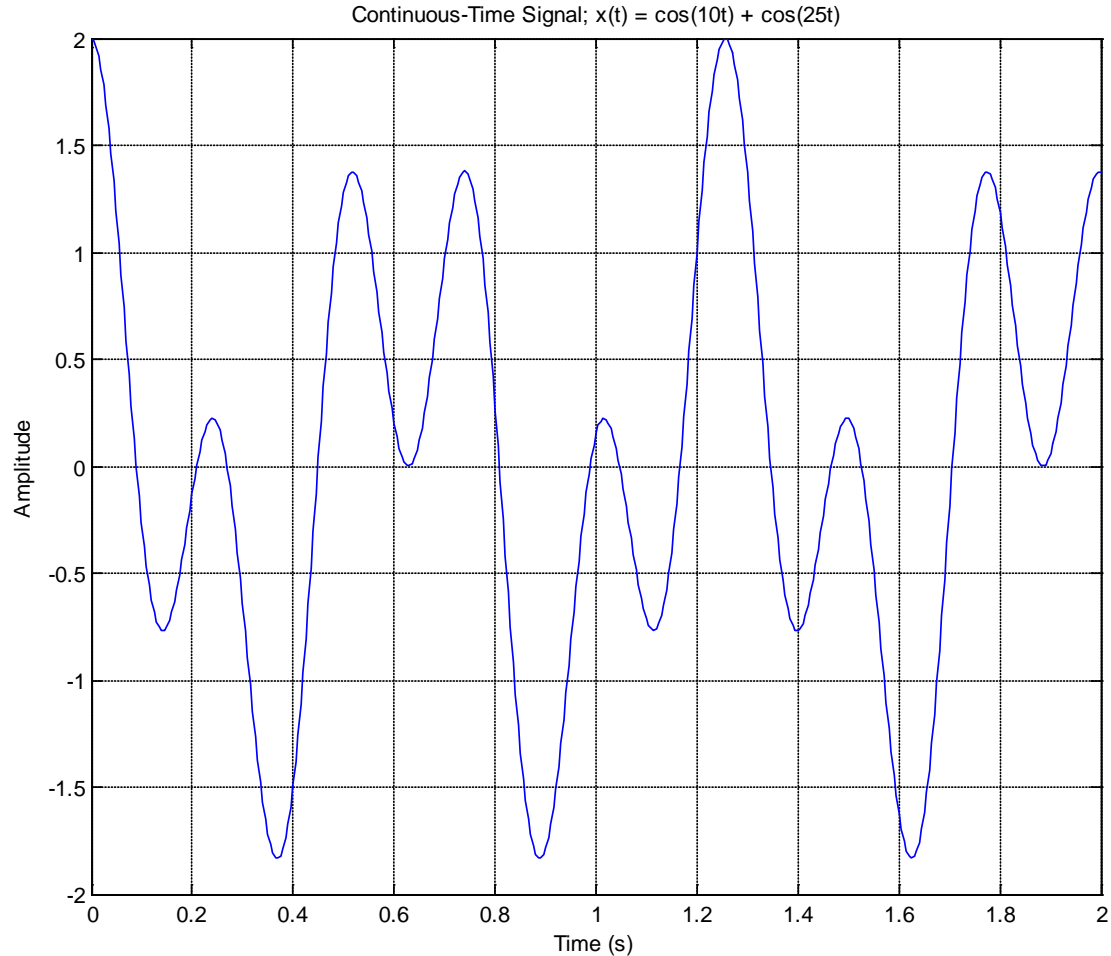
Step size (resolution) is the smallest change that can be discerned by an ADC.



A/D Conversion

- The process of A/D conversion involves
 - Band-limiting the analog signal
 - Setting periodic sampling points at which the continuous time analog signal is going to be digitized
 - Freezing the analog signal at the sampling points so that the signal does not change for the duration of conversion (***sample and hold***)
 - Determining what digital value best represents the analog signal level (***quantizing***)

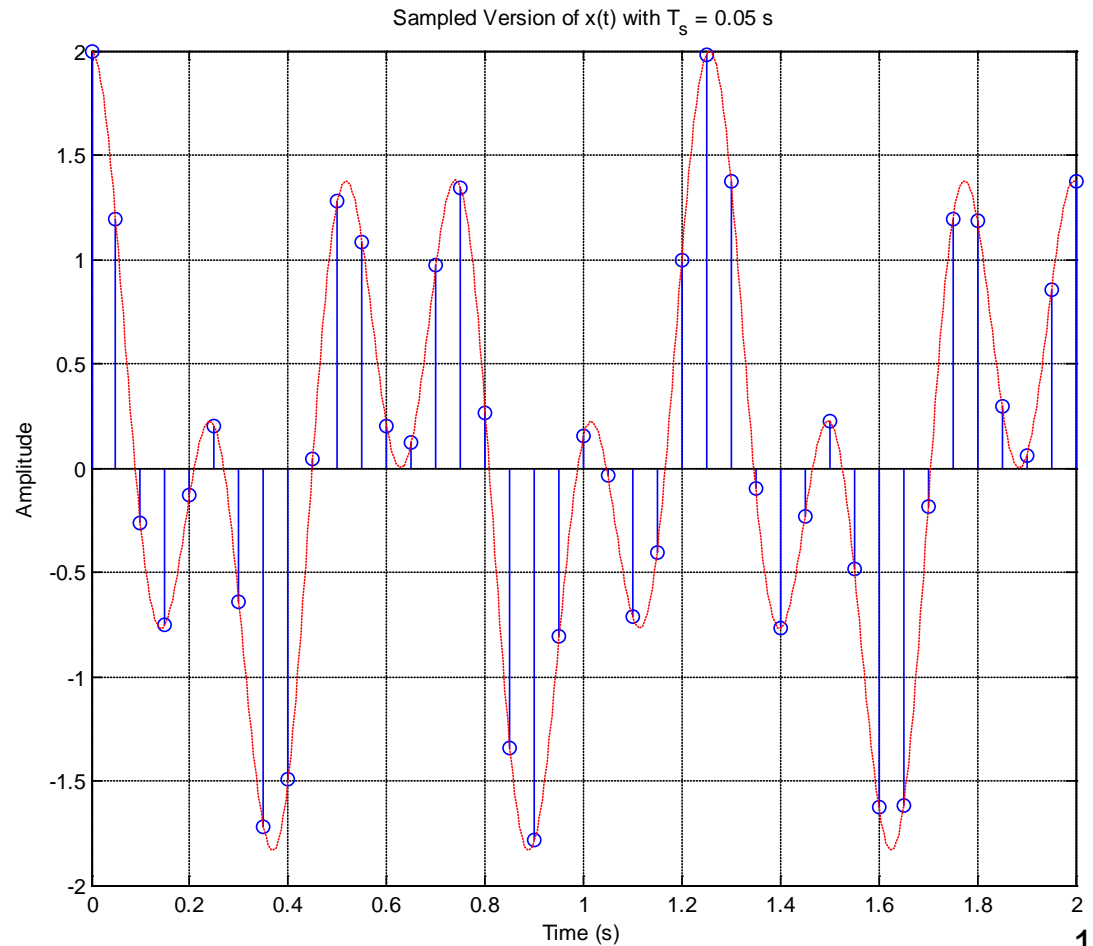
Example: Output of a Sensor



This is the superposition of two sine signals (two frequencies present)

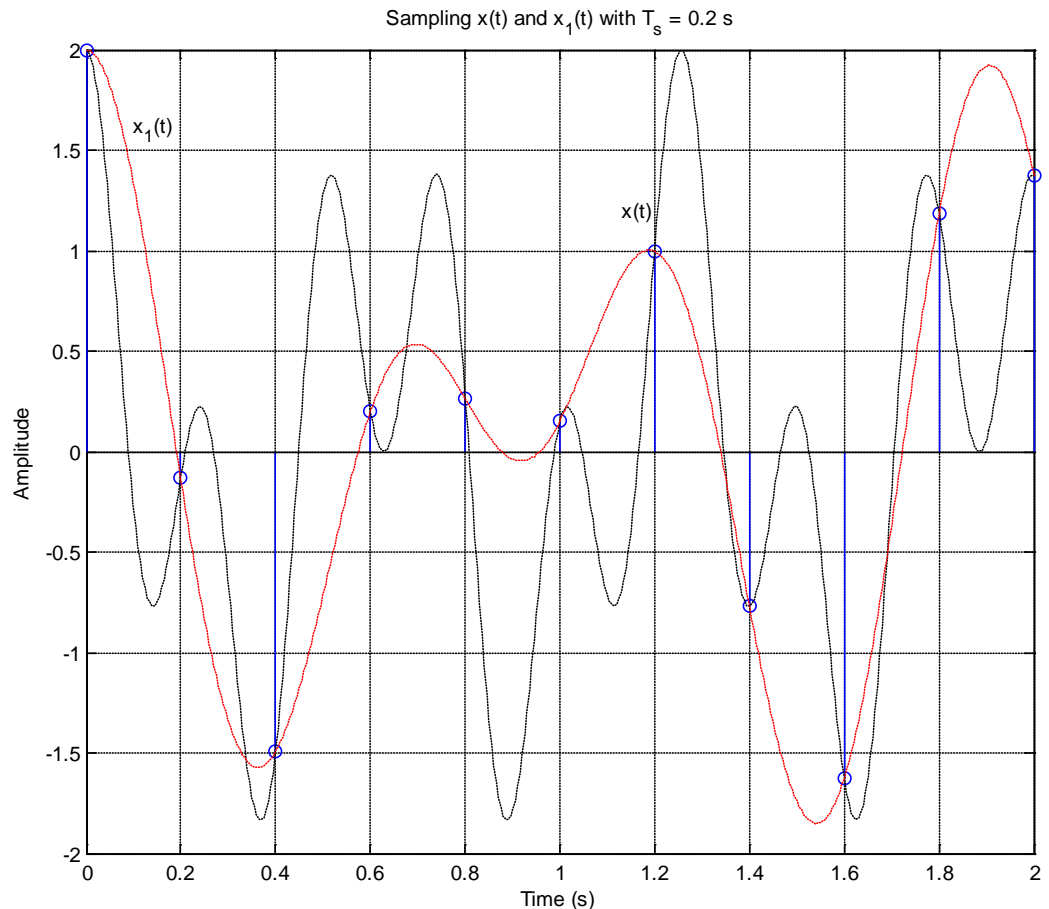
Example: Oversampling

- If original signal was indeed nicely limited, **oversampling** will result in more samples than we need for processing and reconstruction (i.e., burden on our calculations/ processor time)
- Shown here is 20 samples per second (a sampling clock of 20Hz)



Example: Undersampling

- **Undersampling** will result in losing the features of a signal.
- If we wanted to reconstruct the signal the reconstructed signal (red) would not look the same as the original (black) signal
- **Nyquist Rate**
 - Sample at twice the rate of the signal's max frequency



Hold after Sample

- We need circuitry that can “remember” (hold) the analog voltage at the sampling time
- The Ctrl signal in the circuit below should have the periodicity of the sampling rate and a duty cycle corresponding to the holding (charging time)
- Trade-offs of the holding time!

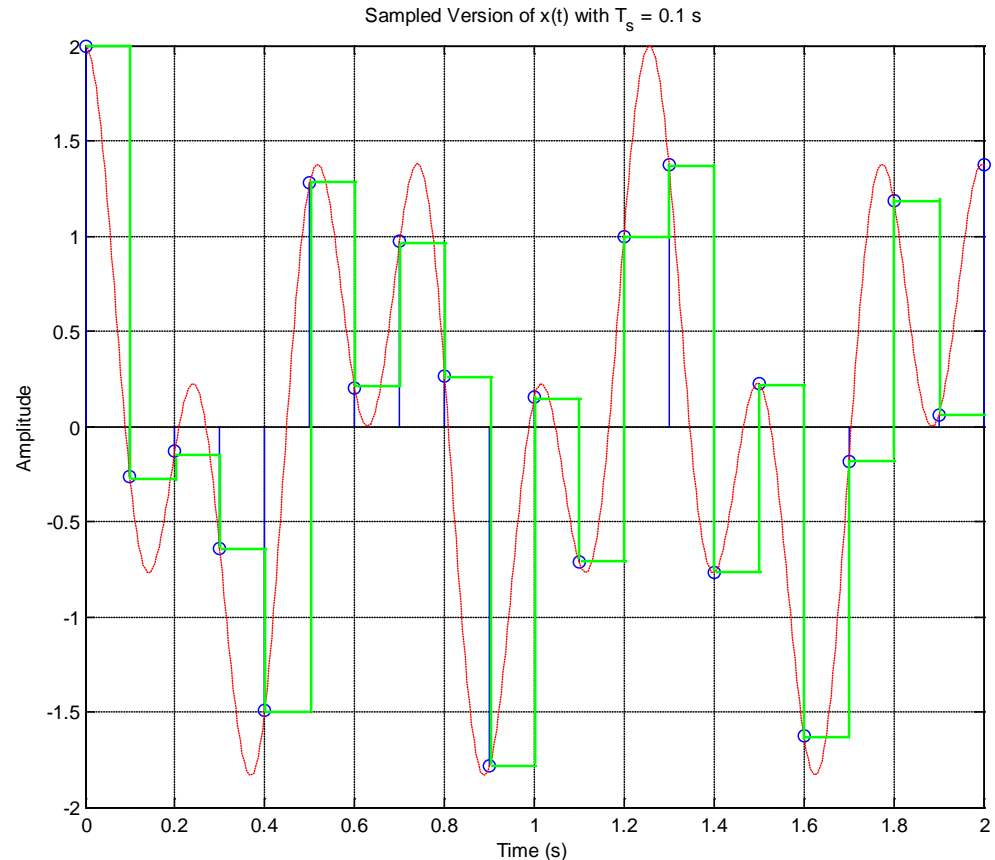
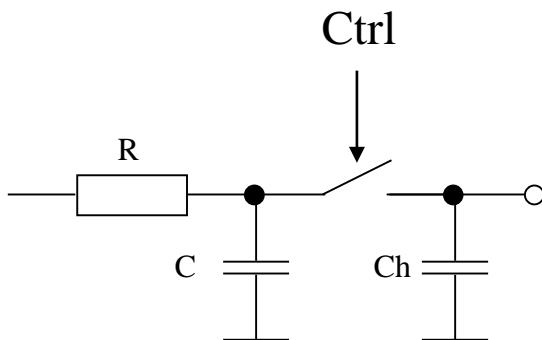
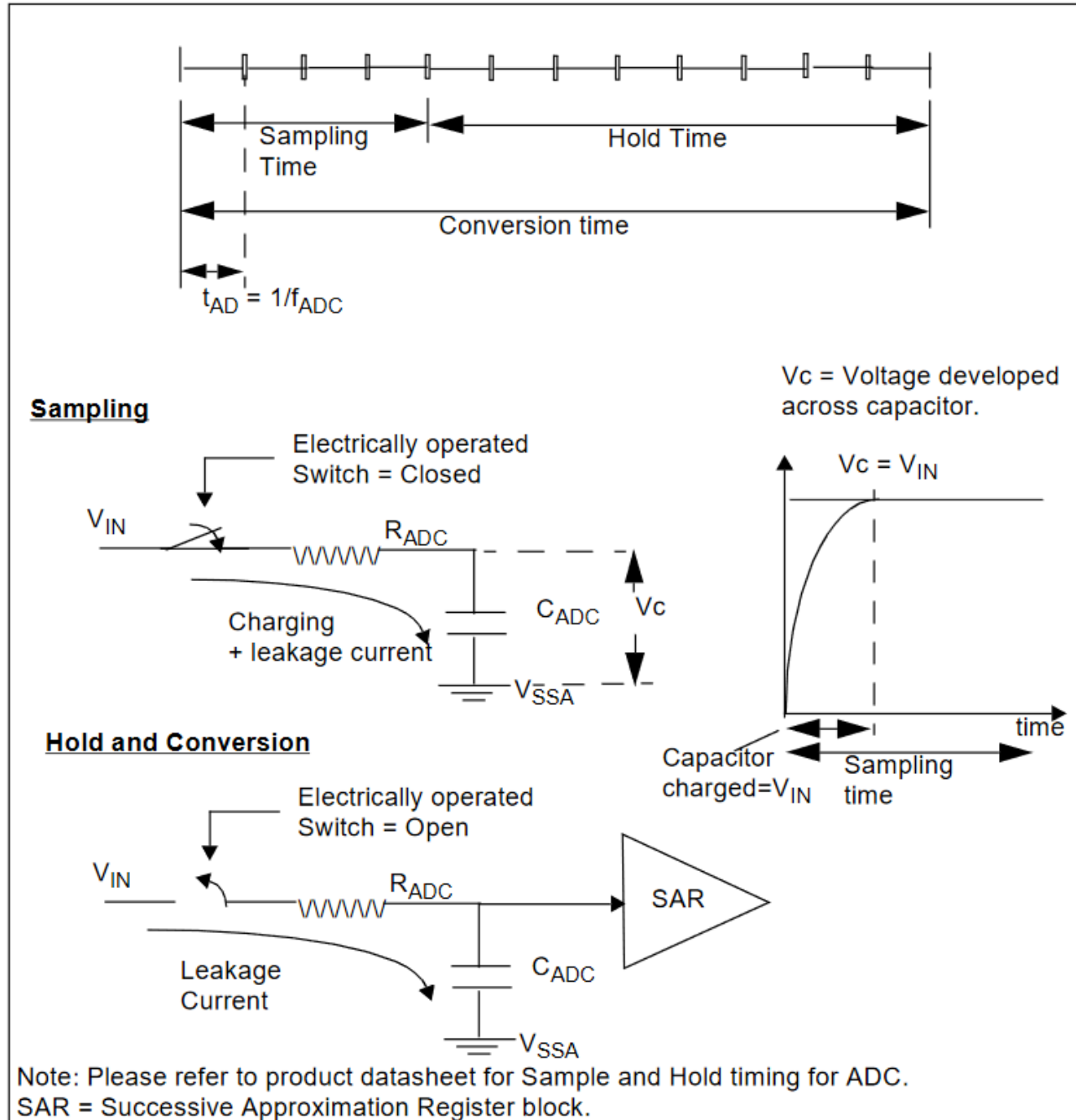


Figure 8. Sample and Hold timing and electrical diagram





Quantization

- Now that the signal is not changing for some time (depending on the sampling period), we need to determine what digital value best represents the analog level.
- This will take some time, which **should not** be more than the sampling period!

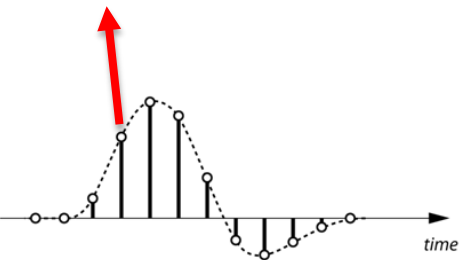
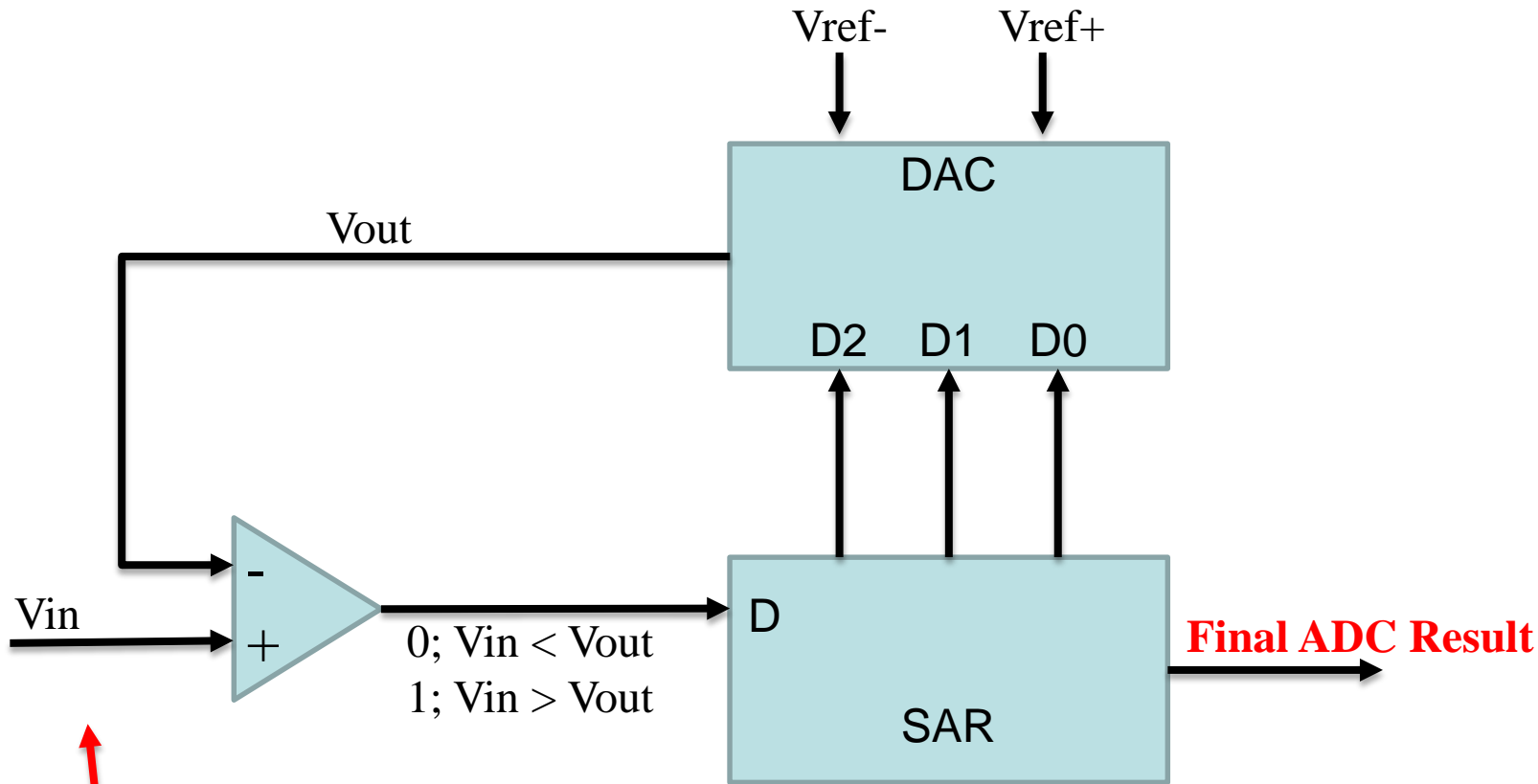


Quantization

- There are many methods to determine what digital value best represents the analog level
 - Direct-Conversion
 - Integrating
 - Delta-Encoded
- Many μ Cs use “**Successive Approximation**” to do this, as our PIC18 does

Successive Approximation ADC

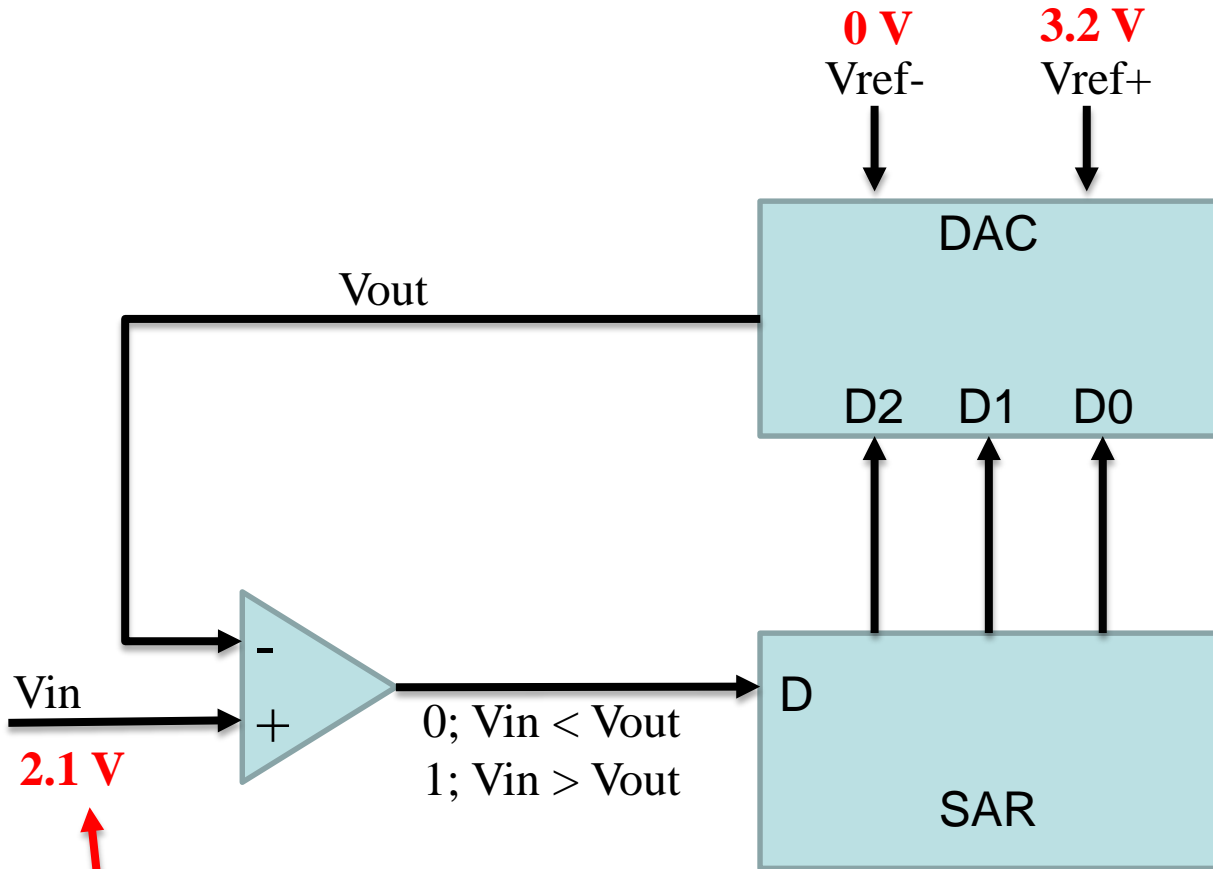
3-bit Example



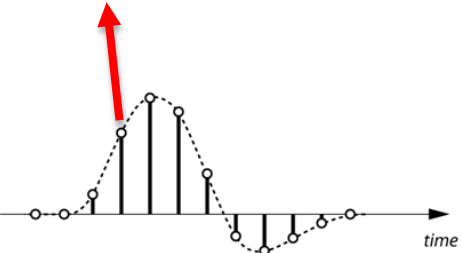


Successive Approximation ADC

3-bit Example



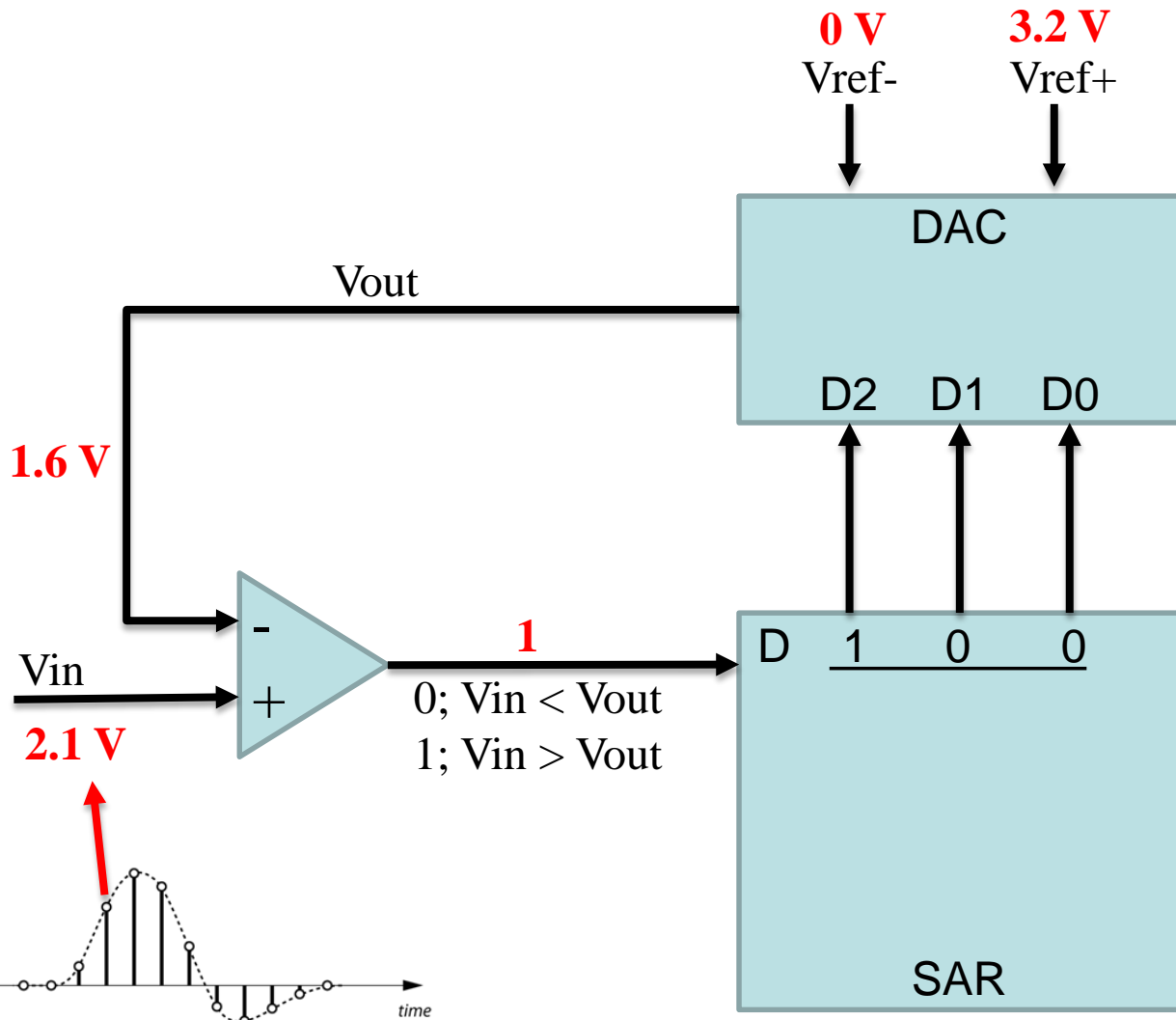
3-bit Result	Voltage Window (Step Size)
111	2.8 – 3.2
110	2.4 – 2.8
101	2.0 – 2.4
100	1.6 – 2.0
011	1.2 – 1.6
010	0.8 – 1.2
001	0.4 – 0.8
000	0.0 – 0.4





Successive Approximation ADC

3-bit Example

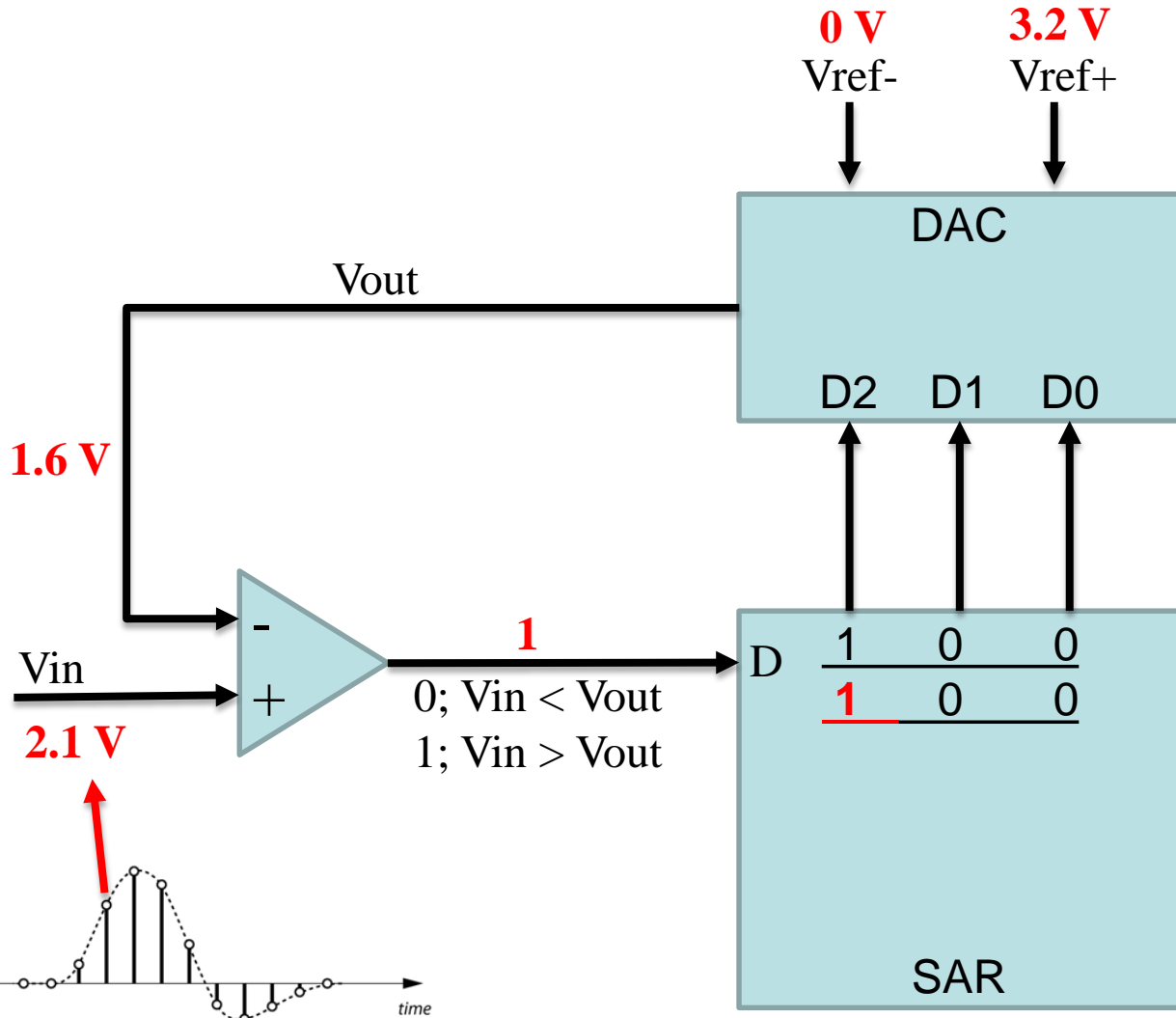


3-bit Result	Voltage Window (Step Size)
111	2.8 – 3.2
110	2.4 – 2.8
101	2.0 – 2.4
100	1.6 – 2.0
011	1.2 – 1.6
010	0.8 – 1.2
001	0.4 – 0.8
000	0.0 – 0.4



Successive Approximation ADC

3-bit Example

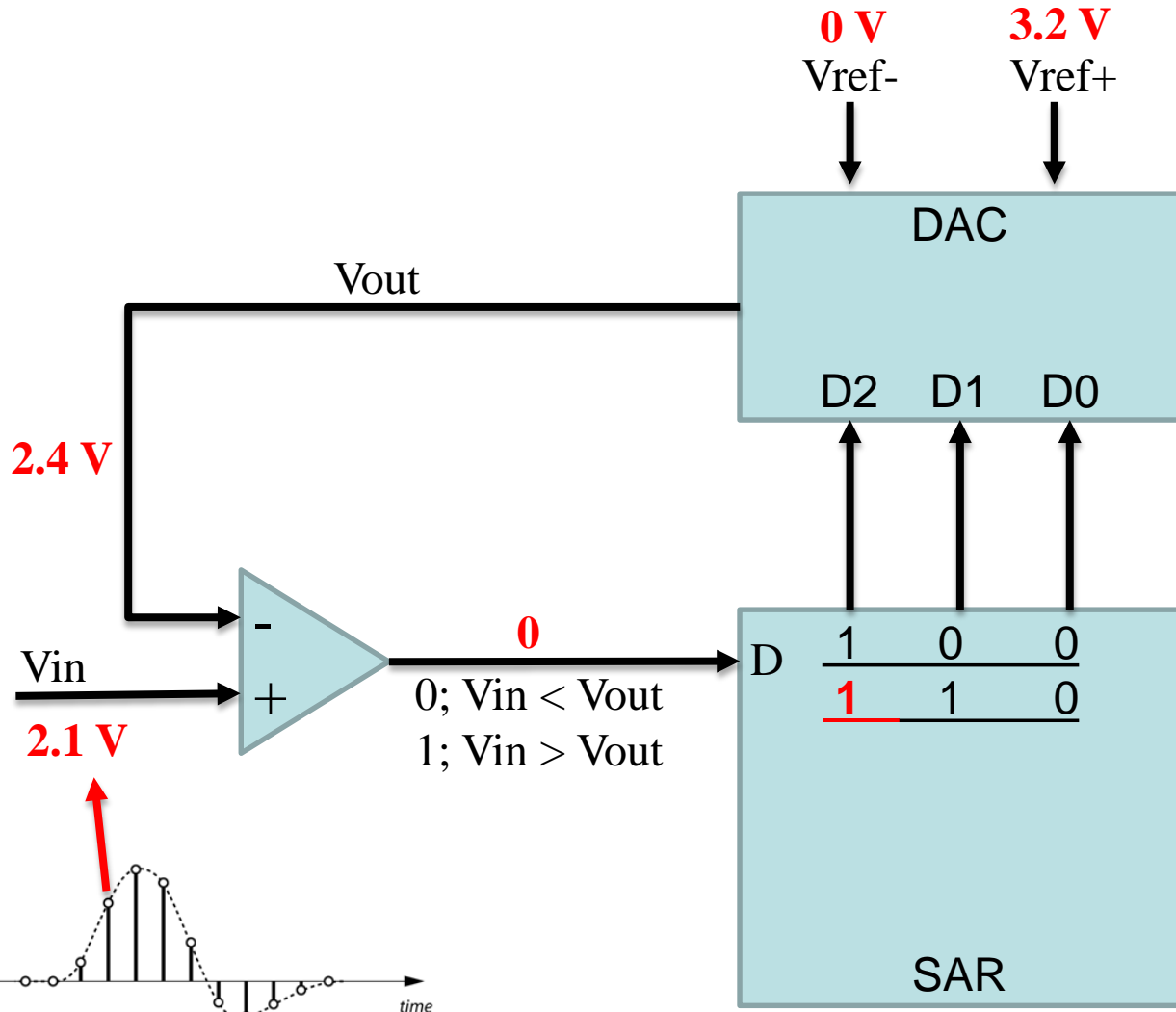


3-bit Result	Voltage Window (Step Size)
111	2.8 – 3.2
110	2.4 – 2.8
101	2.0 – 2.4
100	1.6 – 2.0
011	1.2 – 1.6
010	0.8 – 1.2
001	0.4 – 0.8
000	0.0 – 0.4



Successive Approximation ADC

3-bit Example

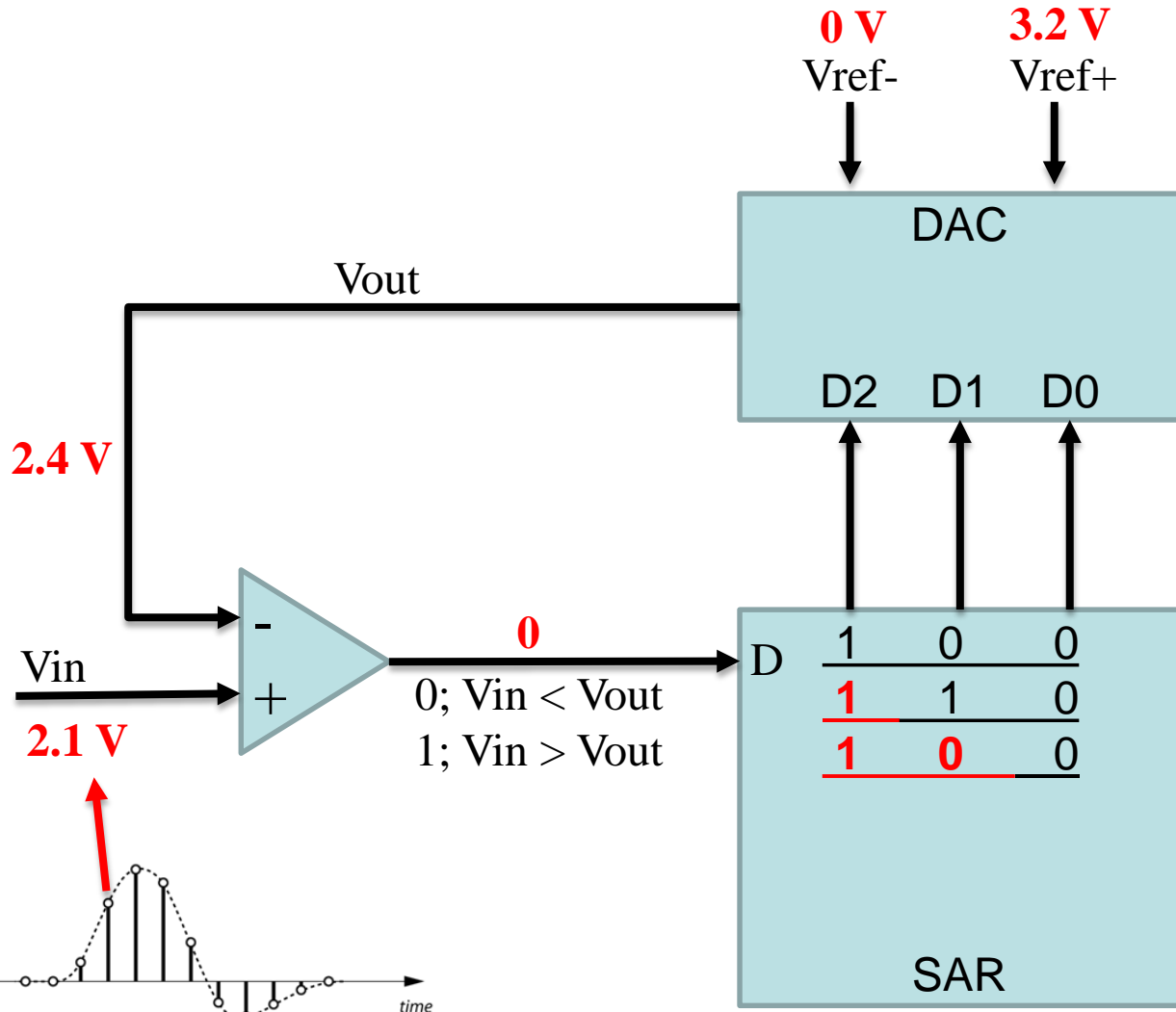


3-bit Result	Voltage Window (Step Size)
111	2.8 – 3.2
110	2.4 – 2.8
101	2.0 – 2.4
100	1.6 – 2.0
011	1.2 – 1.6
010	0.8 – 1.2
001	0.4 – 0.8
000	0.0 – 0.4



Successive Approximation ADC

3-bit Example

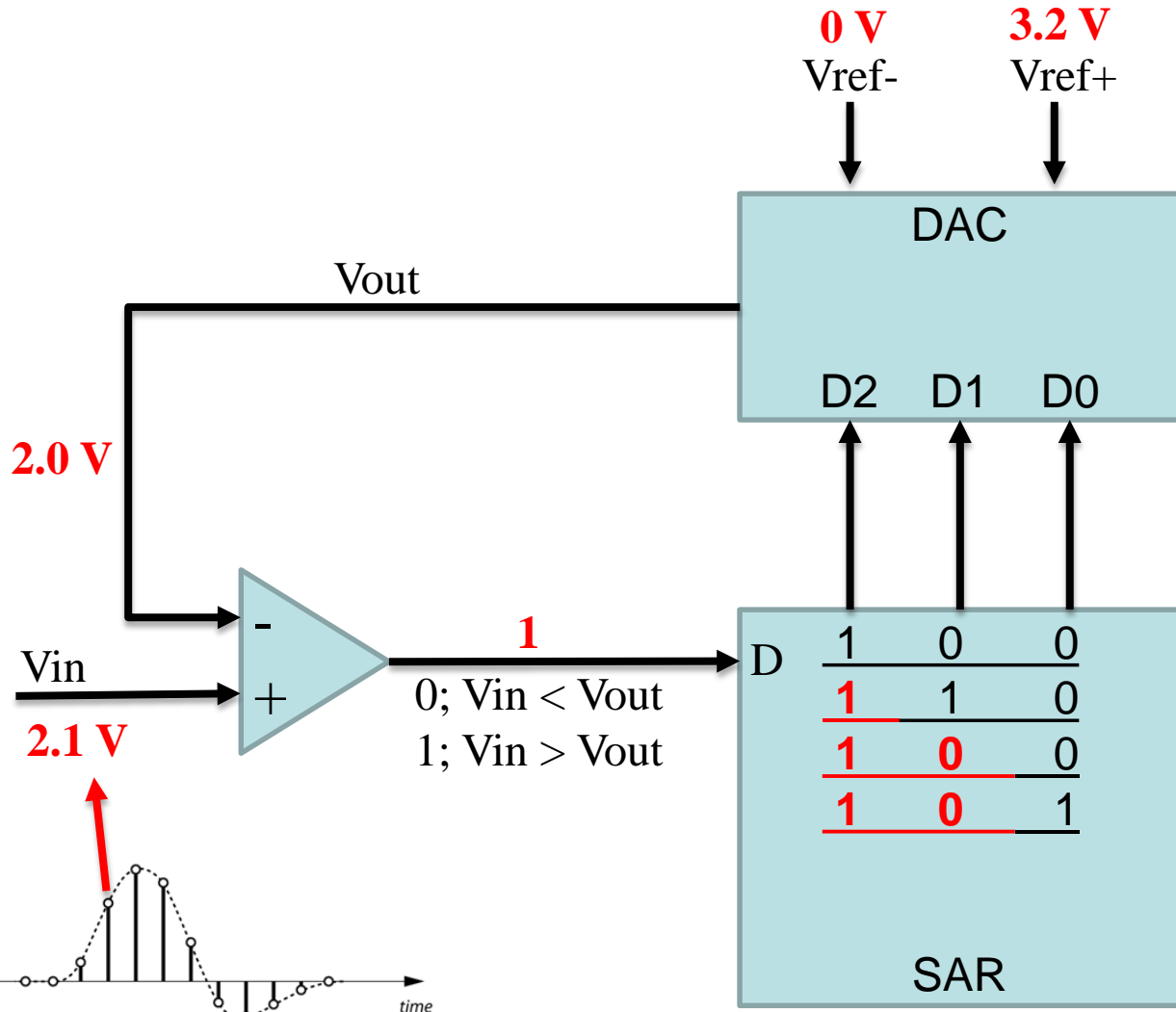


3-bit Result	Voltage Window (Step Size)
111	2.8 – 3.2
110	2.4 – 2.8
101	2.0 – 2.4
100	1.6 – 2.0
011	1.2 – 1.6
010	0.8 – 1.2
001	0.4 – 0.8
000	0.0 – 0.4



Successive Approximation ADC

3-bit Example

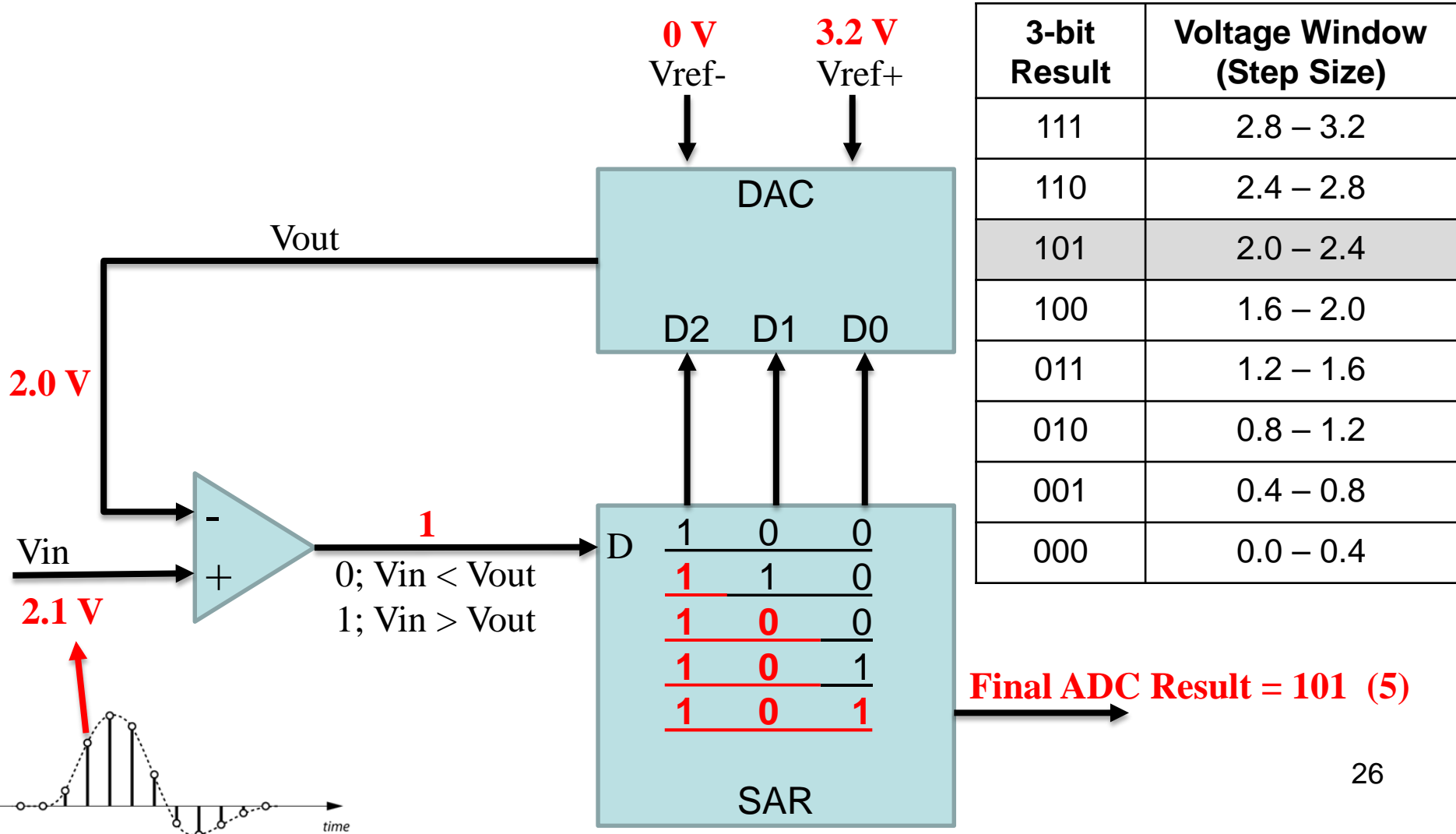


3-bit Result	Voltage Window (Step Size)
111	2.8 – 3.2
110	2.4 – 2.8
101	2.0 – 2.4
100	1.6 – 2.0
011	1.2 – 1.6
010	0.8 – 1.2
001	0.4 – 0.8
000	0.0 – 0.4

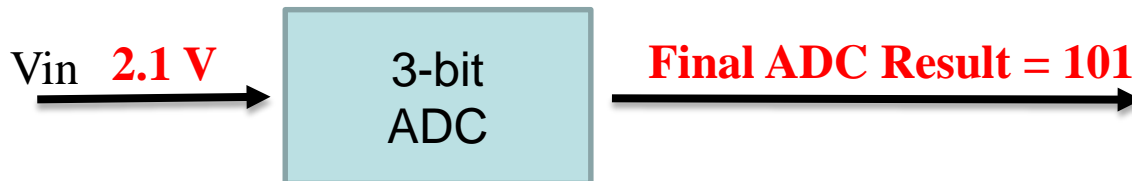


Successive Approximation ADC

3-bit Example



More Bits = More Accuracy



3-bit Result	Voltage Window (Step Size)
111	2.8 – 3.2
110	2.4 – 2.8
101	2.0 – 2.4
100	1.6 – 2.0
011	1.2 – 1.6
010	0.8 – 1.2
001	0.4 – 0.8
000	0.0 – 0.4



ADC Characteristics

- **Resolution**
 - usually expressed in “n-bits”
- **Conversion time**
 - how long it takes to convert from analog to digital
- **Voltage Reference**
 - what is the voltage range (min and max)
- **Linear vs. non-linear transfer**
 - equal step size vs. changing step size



ADC Resolution

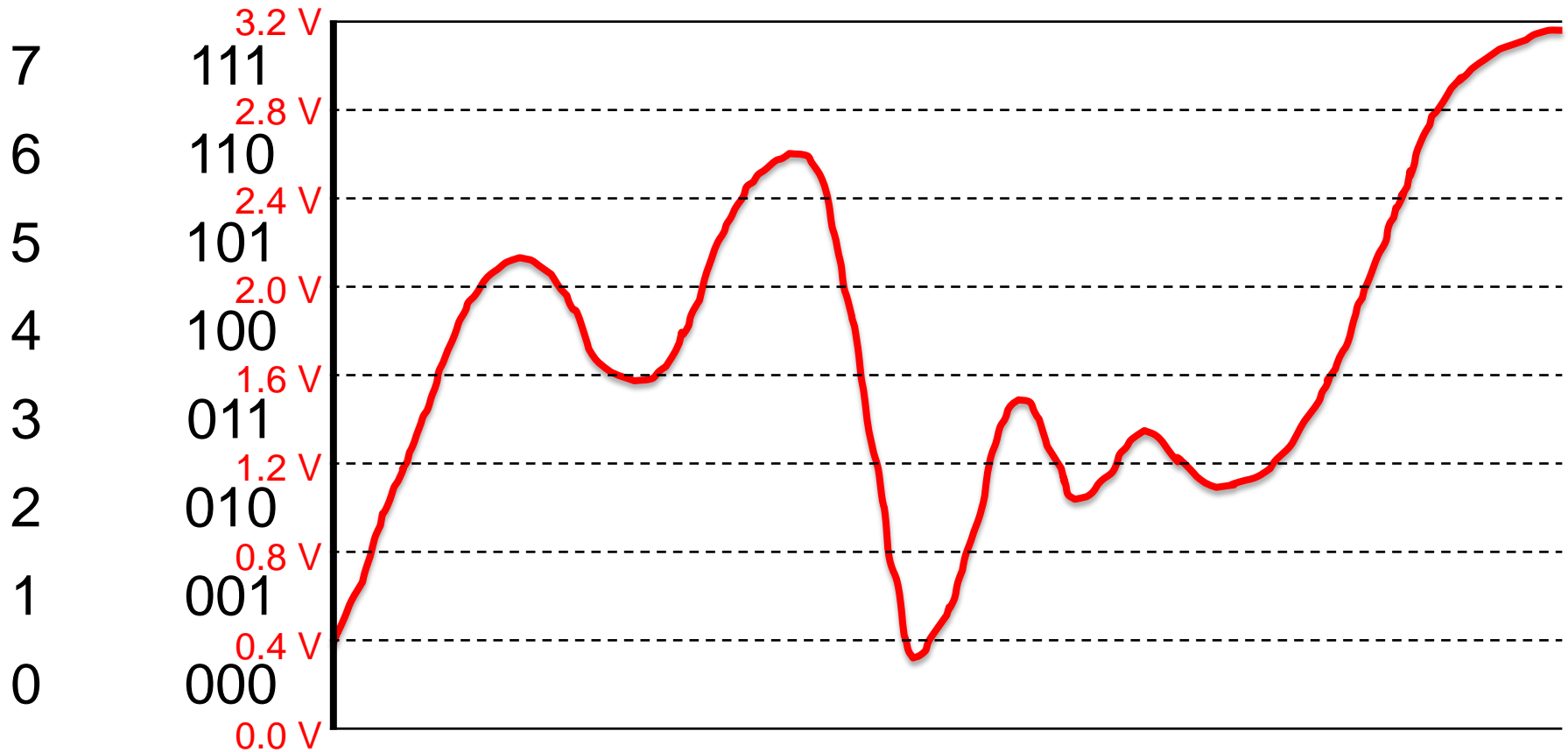
- “n-bits” n can be any number
 - usually 8 – 24
- Resolution determines the quantization steps or how precise/small each voltage window is
- Unchangeable by the programmer (fixed in PIC)
- Step Size = $\frac{V_{ref}^+ - V_{ref}^-}{2^n}$

n-bit	Number of steps	Step size (mV)
8	256	$5/256 = 19.53$
10	1,024	$5/1,024 = 4.88$
12	4,096	$5/4,096 = 1.2$
16	65,536	$5/65,536 = 0.076$

Notes: $V_{CC} = 5\text{ V}$

Step size (resolution) is the smallest change that can be discerned by an ADC.

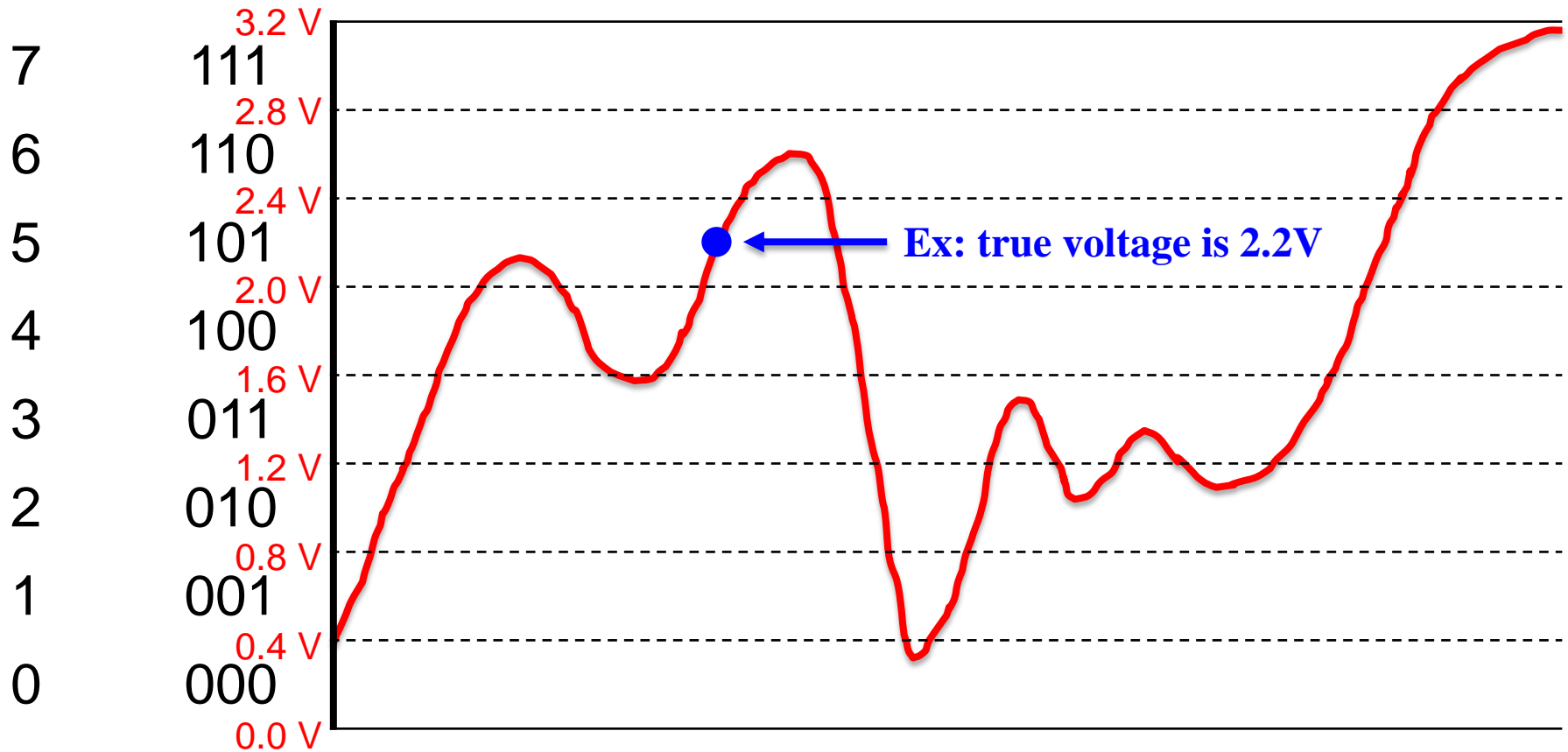
ADC Resolution



$$\text{Step Size} = \frac{V_{ref}^+ - V_{ref}^-}{2^n} = \frac{3.2 - 0.0}{2^3} = 0.4V / \text{step}$$



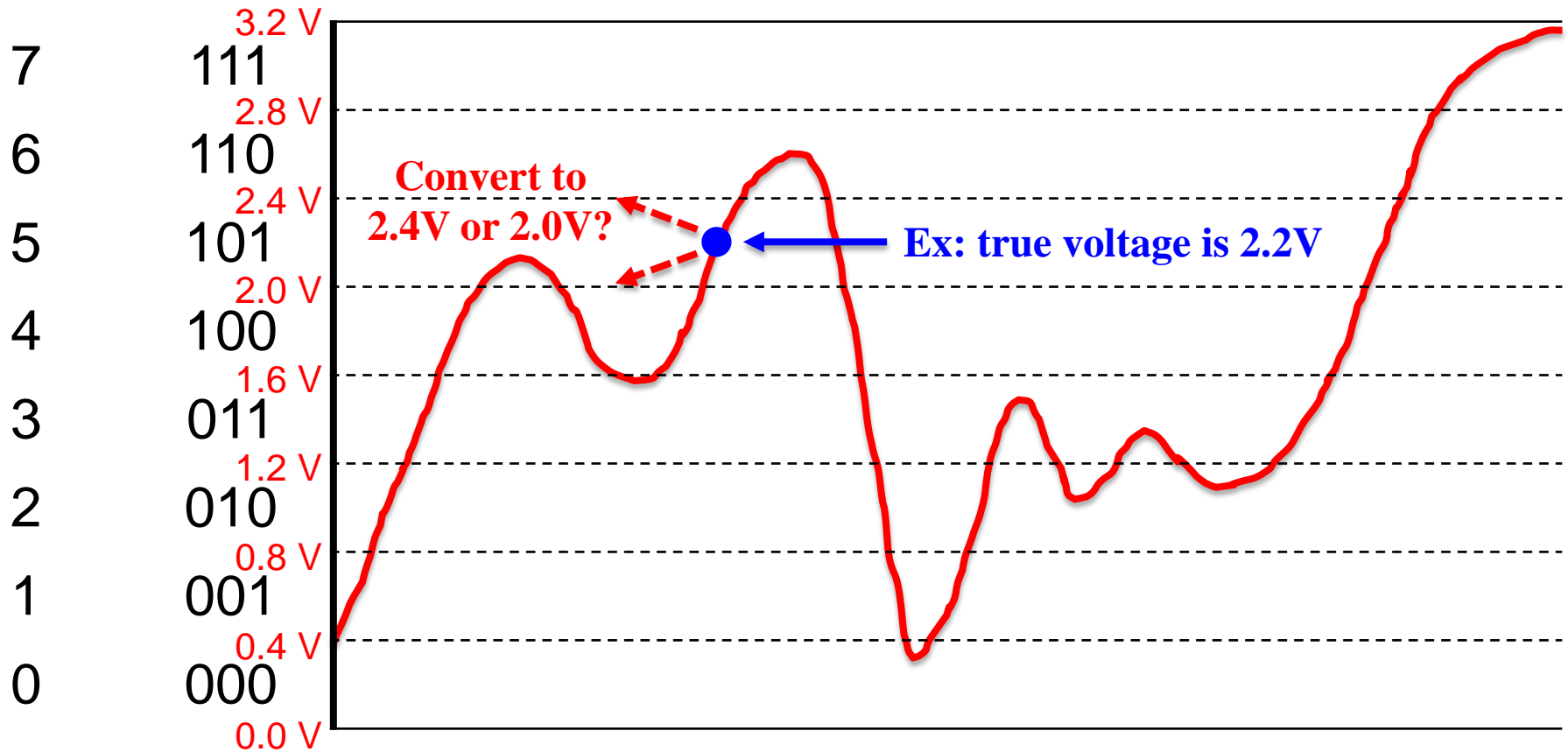
ADC Can Only “Trap” an Input Voltage in a Window



$$\text{Step Size} = \frac{V_{ref}^+ - V_{ref}^-}{2^n} = \frac{3.2 - 0.0}{2^3} = 0.4V / \text{step}$$



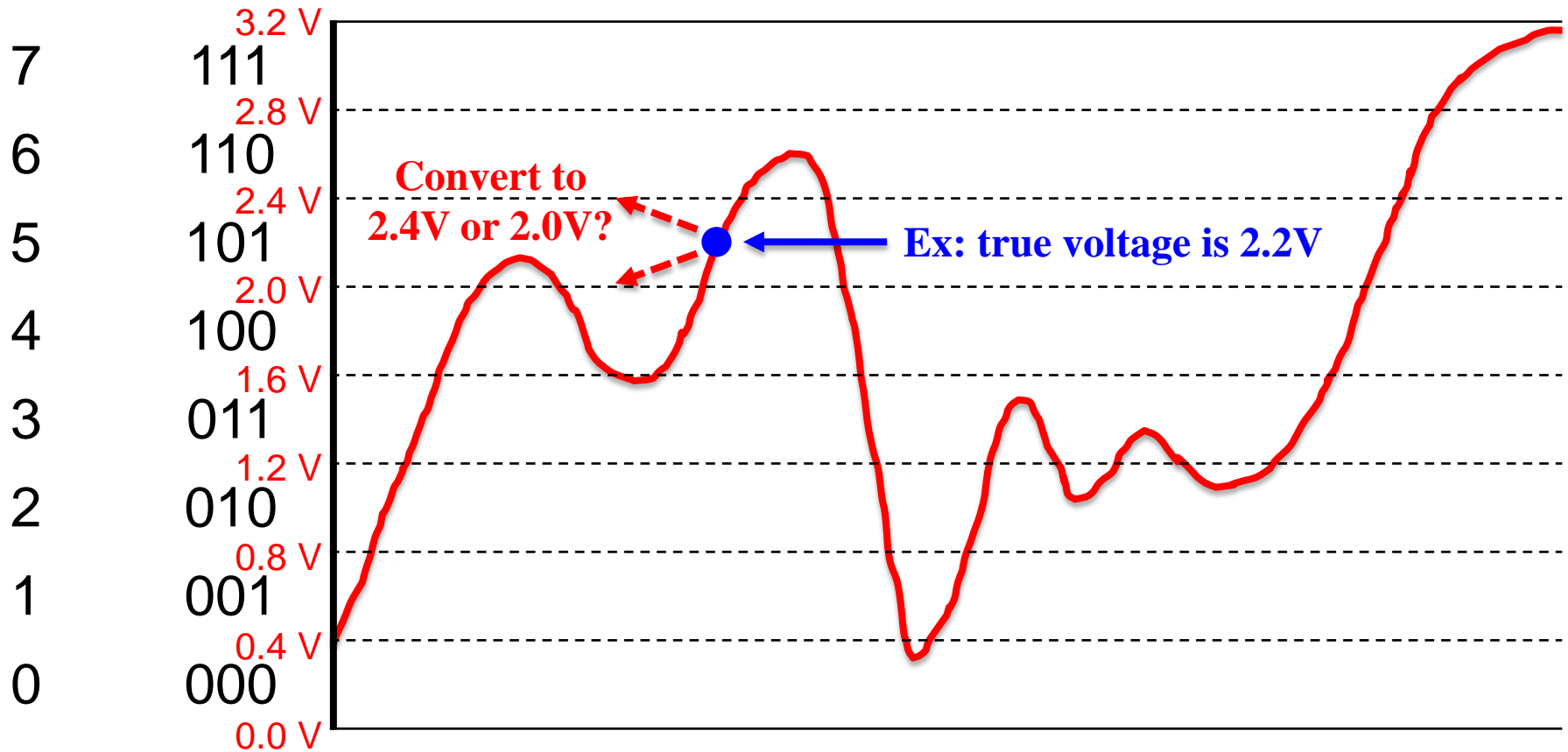
ADC Can Only “Trap” an Input Voltage in a Window



$$\text{Step Size} = \frac{V_{ref}^+ - V_{ref}^-}{2^n} = \frac{3.2 - 0.0}{2^3} = 0.4V / \text{step}$$



All we know is the input voltage is between 2.0V and 2.4V



$$\text{Step Size} = \frac{V_{ref}^+ - V_{ref}^-}{2^n} = \frac{3.2 - 0.0}{2^3} = 0.4V / \text{step}$$



ADC Conversion Time

- A/D conversion takes a **FIXED** amount of time (PIC)
- The analog part of the circuit needs to remember the analog voltage at the exact time of the sampling (usually done by charging a capacitor for a finite amount of time)
- Once that value is remembered, the ADC needs to determine (quantize) what the corresponding digital value to that voltage is. This is usually determined by a quantization clock.
- The conversion time has to be smaller than the sampling time!



V_{ref} – Voltage Reference

- We know that the resolution (n-bits) determines how many quantization steps there are, thus determining the relative scale
- What determines the absolute scale though
 - what voltage value represents 0 and what voltage value represents 0b11111111 (for 8-bit resolution)?
- V_{ref}^- and V_{ref}^+ are used for that
- ADC circuits usually have a default of $V_{\text{ref}}^- = V_{\text{ss}}$ and $V_{\text{ref}}^+ = V_{\text{cc}}$
- However most ADCs let you feed in analog voltages representing V_{ref}^- and V_{ref}^+

V_{ref} (cont'd)

- In general V_{ref} s could be any voltage but many times it is required to be within $[V_{ss}, V_{cc}]$.
- The PICs built-in ADCs do require that.
- So, what's the step size of an 8-bit ADC where $V_{ref}^- = 2V$ and $V_{ref}^+ = 3V$?

$$\text{Step Size} = \frac{V_{ref}^+ - V_{ref}^-}{2^n} = \frac{3 - 2}{2^8} = 3.9mV/step$$

- What would happen if $V_{ref}^- = -2V$ and $V_{ref}^+ = 2V$, how would we want the values represented?
- **If we know all this, we can reconstruct the voltage level in software.**



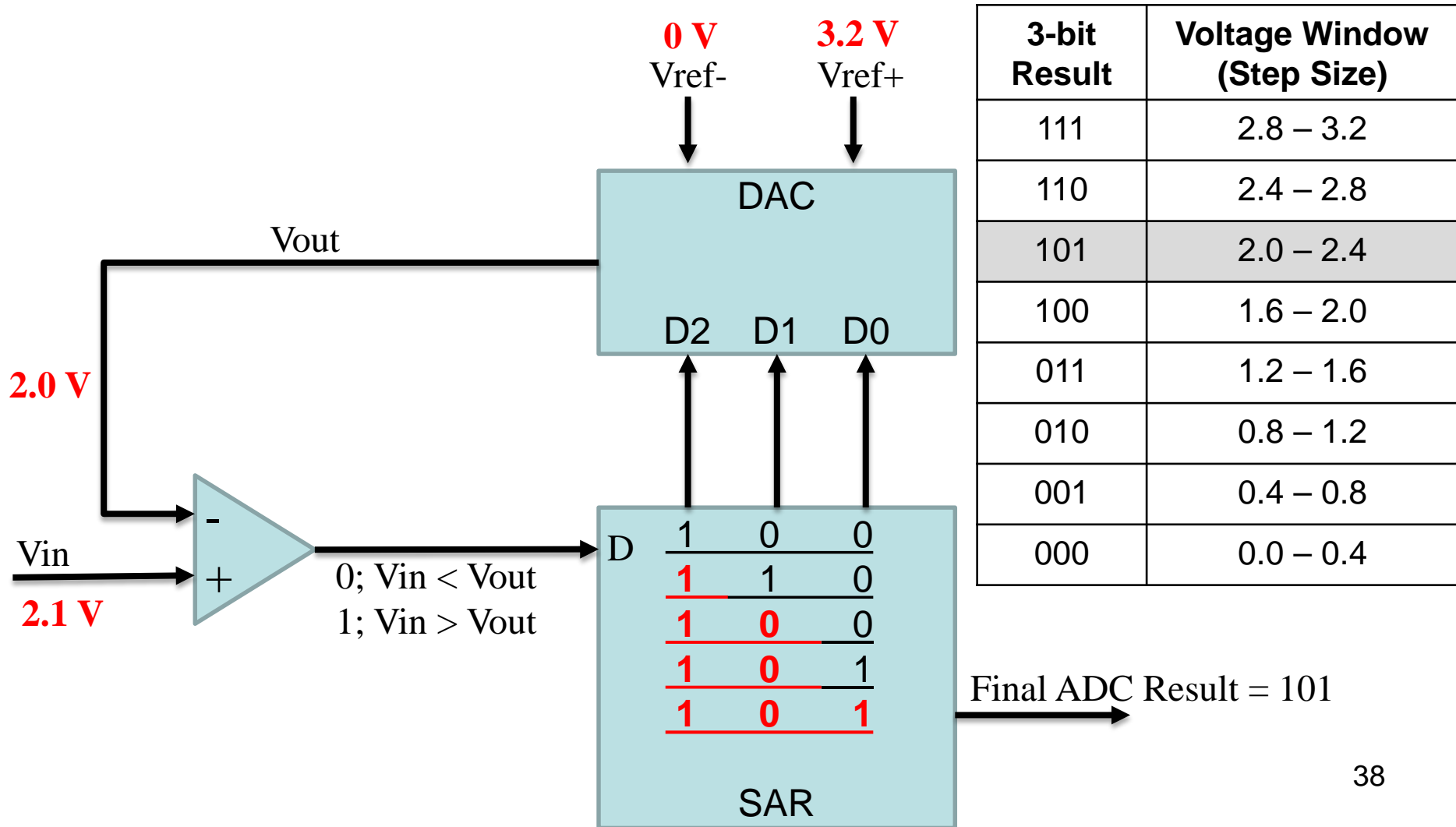
Handling AD Result

- n-bit result ($0 \rightarrow 2^n - 1$)
 - 3-bit: $0 \rightarrow 7$ (000 \rightarrow 111)
 - 8-bit: $0 \rightarrow 255$ (0000 0000 \rightarrow 1111 1111)
 - 10-bit: $0 \rightarrow 1023$ (00 0000 0000 \rightarrow 11 1111 1111)



Successive Approximation ADC

3-bit Example





Handling AD Result

- n-bit result ($0 \rightarrow 2^n - 1$)
 - 3-bit: $0 \rightarrow 7$ (000 \rightarrow 111)
 - 8-bit: $0 \rightarrow 255$ (0000 0000 \rightarrow 1111 1111)
 - 10-bit: $0 \rightarrow 1023$ (00 0000 0000 \rightarrow 11 1111 1111)

- Voltage = $\left[\frac{\text{result}}{2^n - 1} * (V_{ref}^+ - V_{ref}^-) \right] + V_{ref}^-$

↑
“Percentage”
or ratio

↑
Voltage
Range

↗
Voltage Offset
(min value)



Handling AD Result

- 8-bit result ($0 \rightarrow 2^n-1$): $0 \rightarrow 255$ dec. range

- Voltage = $\left[\frac{\text{result}}{2^n-1} * (V_{ref}^+ - V_{ref}^-) \right] + V_{ref}^-$

**“Percentage”
or ratio** **Voltage
Range** **Voltage Offset
(min value)**



Handling AD Result = 198

- 8-bit result ($0 \rightarrow 2^n - 1$): $0 \rightarrow 255$ dec. range

- Voltage = $\left[\frac{198}{255} * (4v - 1v) \right] + 1v$

- Voltage = $\left[\frac{result}{2^n - 1} * (V_{ref}^+ - V_{ref}^-) \right] + V_{ref}^-$

↑
“Percentage”
or ratio

↑
Voltage
Range

↗
Voltage Offset
(min value)



Handling AD Result = 198

- 8-bit result ($0 \rightarrow 2^n - 1$): $0 \rightarrow 255$ dec. range

- Voltage = $\left[\frac{198}{255} * (4v - 1v) \right] + 1v$

- Voltage = $[0.7765 * (3v)] + 1v$

- Voltage = $\left[\frac{result}{2^n - 1} * (V_{ref}^+ - V_{ref}^-) \right] + V_{ref}^-$

↑
“Percentage”
or ratio

↑
Voltage
Range

↗
Voltage Offset
(min value)



Handling AD Result = 198

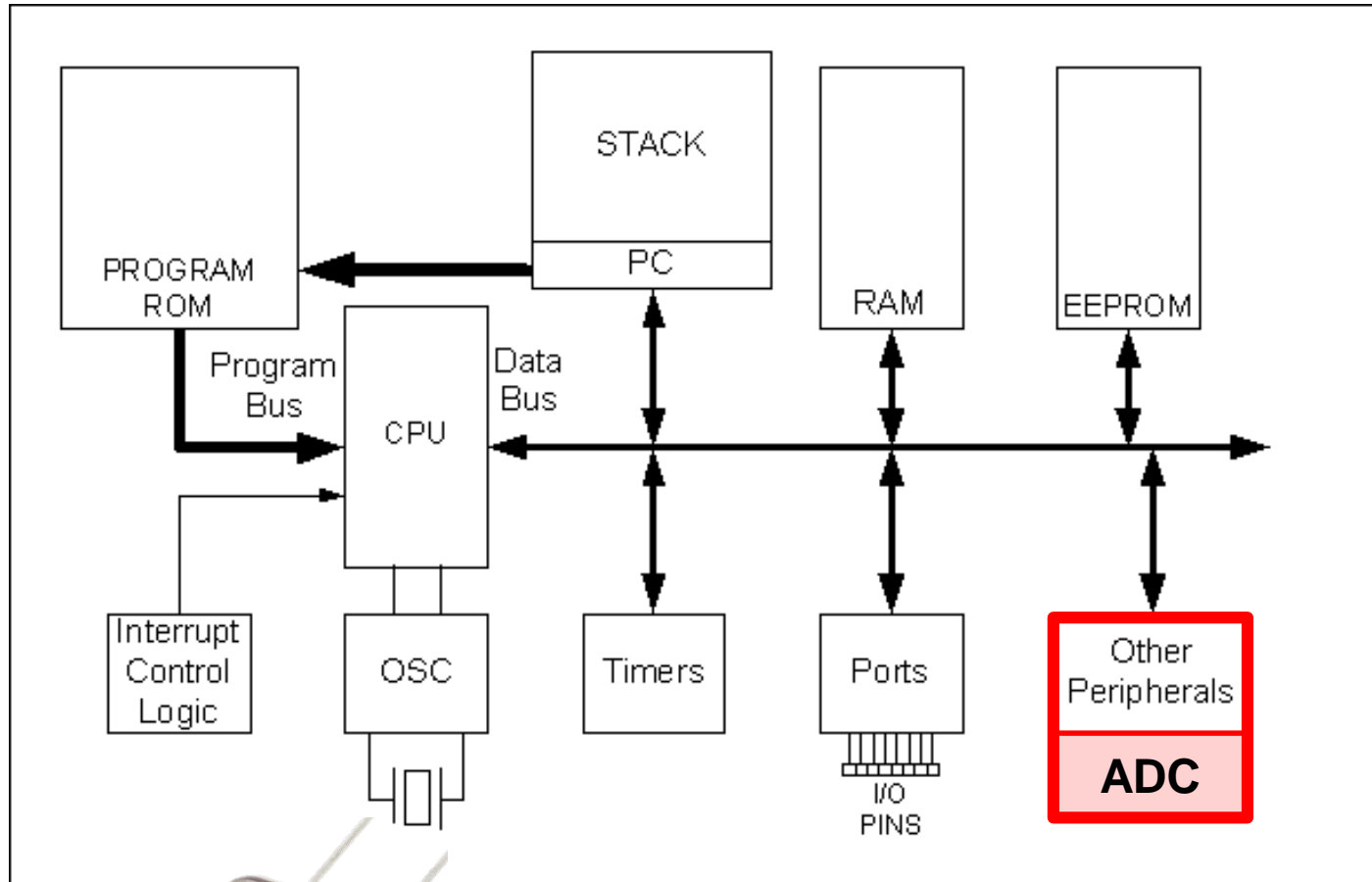
- 8-bit result ($0 \rightarrow 2^n-1$): $0 \rightarrow 255$ dec. range
- Voltage = $\left[\frac{198}{255} * (4v - 1v) \right] + 1v$
- Voltage = $[0.7765 * (3v)] + 1v$
- Voltage = $[2.329v] + 1v = \mathbf{3.329 \text{ Volts}}$
- Voltage = $\left[\frac{\text{result}}{2^n-1} * (V_{ref}^+ - V_{ref}^-) \right] + V_{ref}^-$
 - ↑ “Percentage” or ratio
 - ↑ Voltage Range
 - ↗ Voltage Offset (min value)



Stand-alone ADCs

- **Parallel versus serial output:**
 - An n-bit parallel output ADC has n-pins to talk to a CPU unit in addition to data ready and channel selection bits. Less CPU time, more data pins.
 - A serial output ADC has two pins to talk to the CPU, one for clock, the other for data. More CPU time, few pins.
 - Can be a combination, where (similarly to how we talk to the LCD) x-bits at a time are sent out serially. Can balance CPU time and pins.
- **Number of analog input channels** (i.e., how many sensor can be connected). They can be multiplexed (one ADC) or parallel (many ADCs inside).
- **Start and end of conversion signals.** We need to tell the ADC when to start conversion (indication of sampling) and it has to have feedback on when the conversion is finished.
- Thus there can be **many pins** in an ADC for communication and ADC channels.
- Most PIC microcontrollers have **ADC peripherals built in.** The above will have an effect on these peripherals.

PIC18 ADC Peripheral

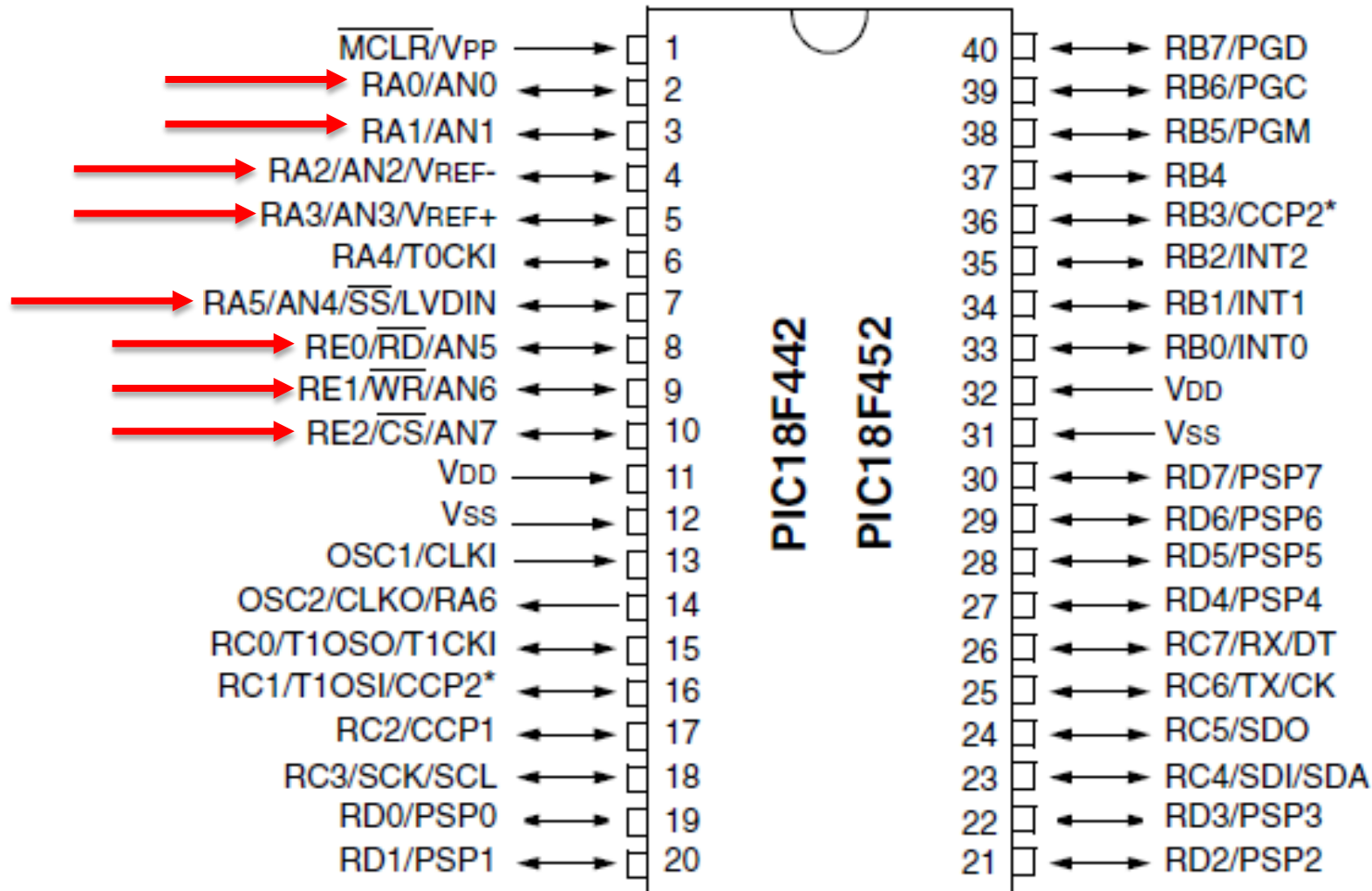




PIC18 ADC

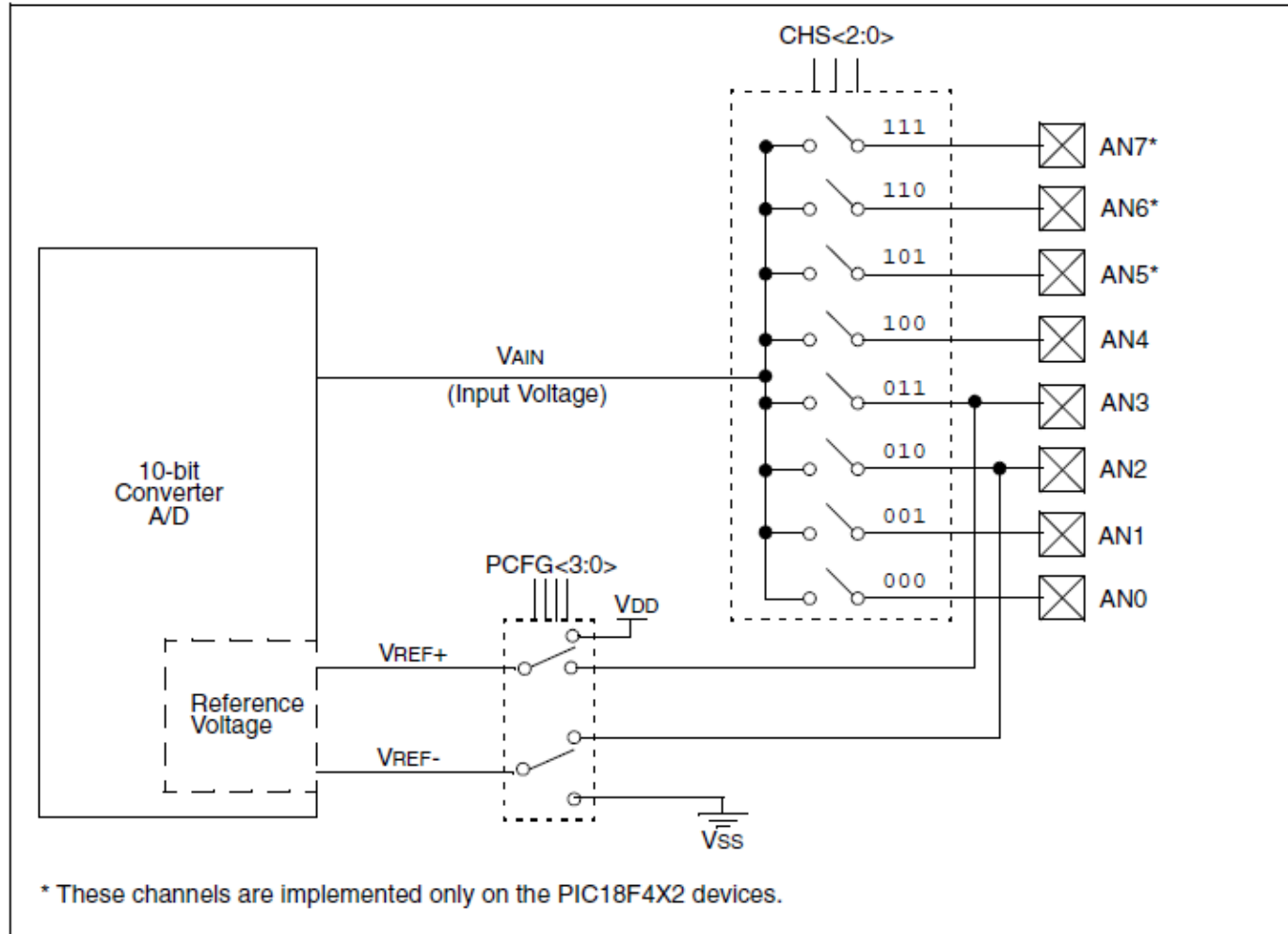
- ADC can be found on most PIC18s as an integrated peripheral. The on-chip ADC peripheral needs to be programmed before use.
- SFR registers are used for all communications between the ADC peripheral and the CPU.
- **PIC18 ADCs are 10-bit successive approximation.** (Other microcontrollers have different precision and architecture on chip ADCs).
- Thus two 8-bit registers are needed to store result
 - **ADRESL** and **ADRESH**
- Up to 16 analog channels (8 for PIC18F452; AN0-AN7) multiplexed to a single ADC.
- Reference voltage can be fed in
- Control registers used to set-up and start conversion.

8 Analog Channels



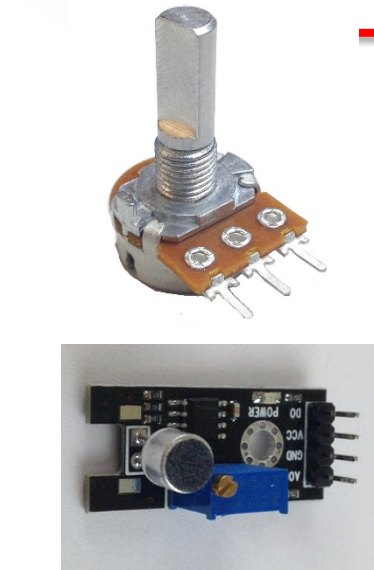
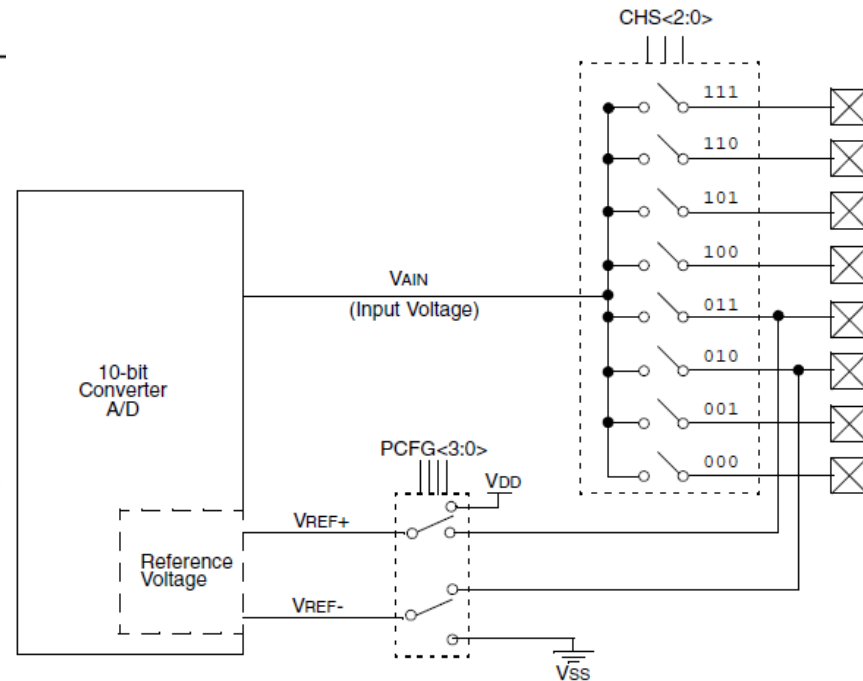
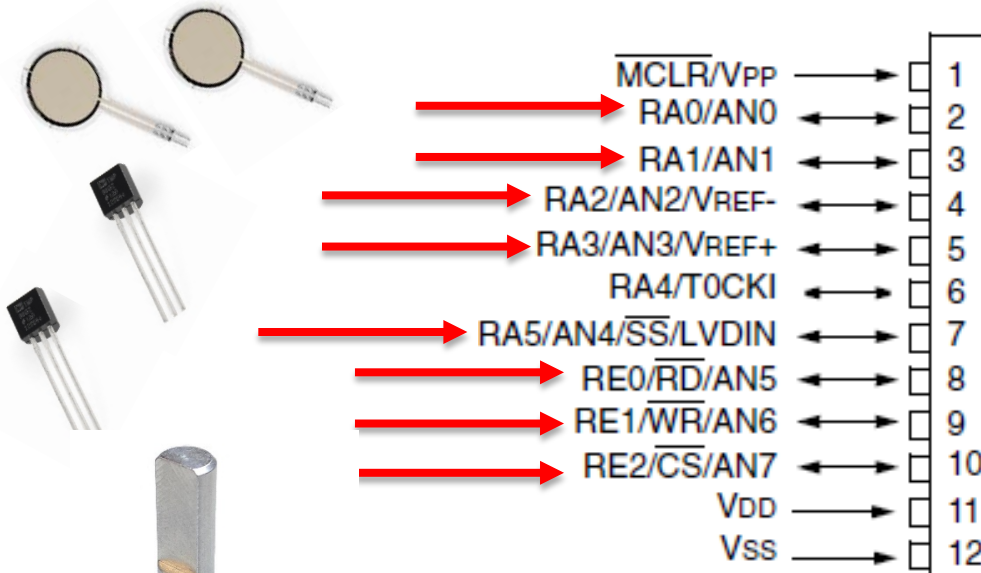
PIC18 ADC Channel Selection

FIGURE 17-1: A/D BLOCK DIAGRAM

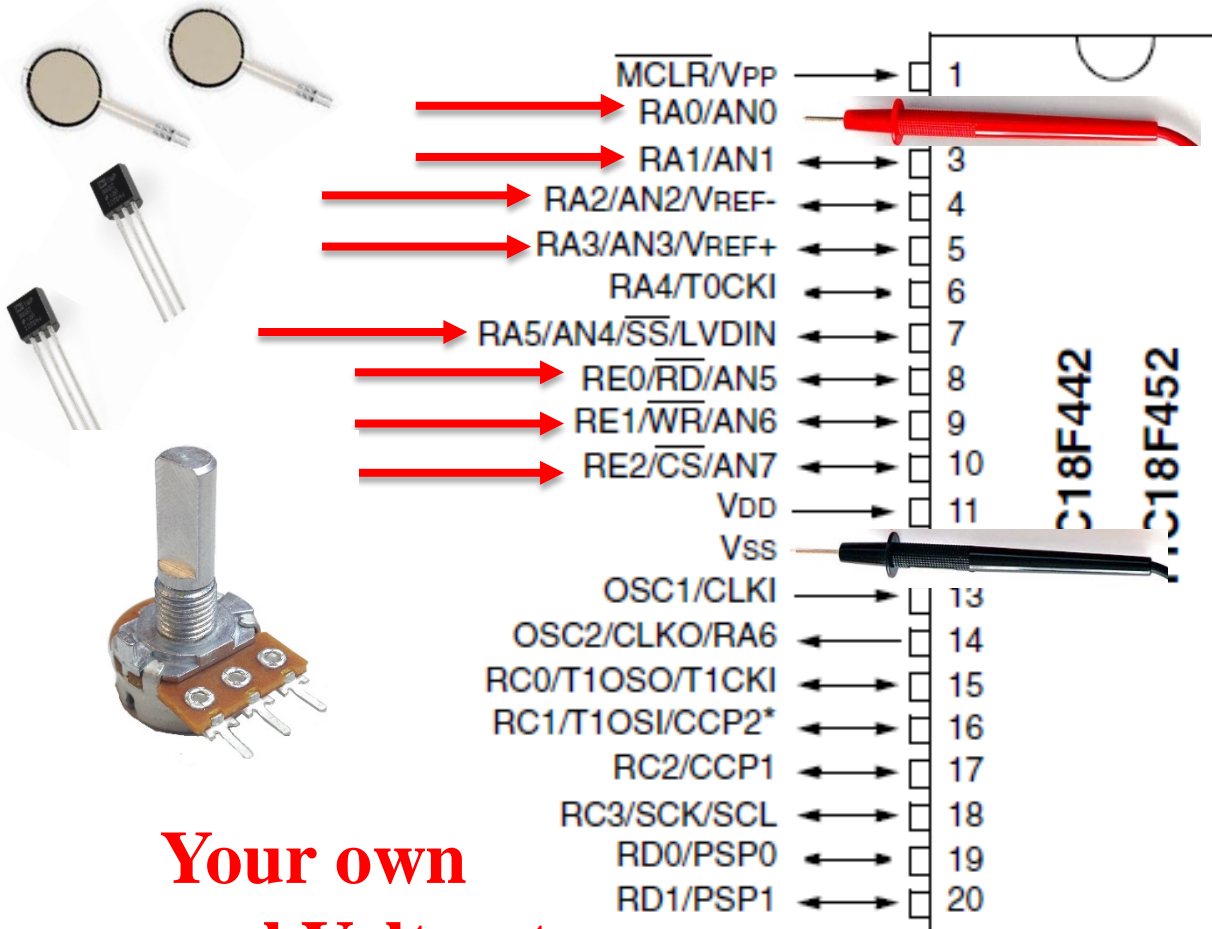




8 Analog Channels (8 Inputs) Only 1 AD Conversion at a time



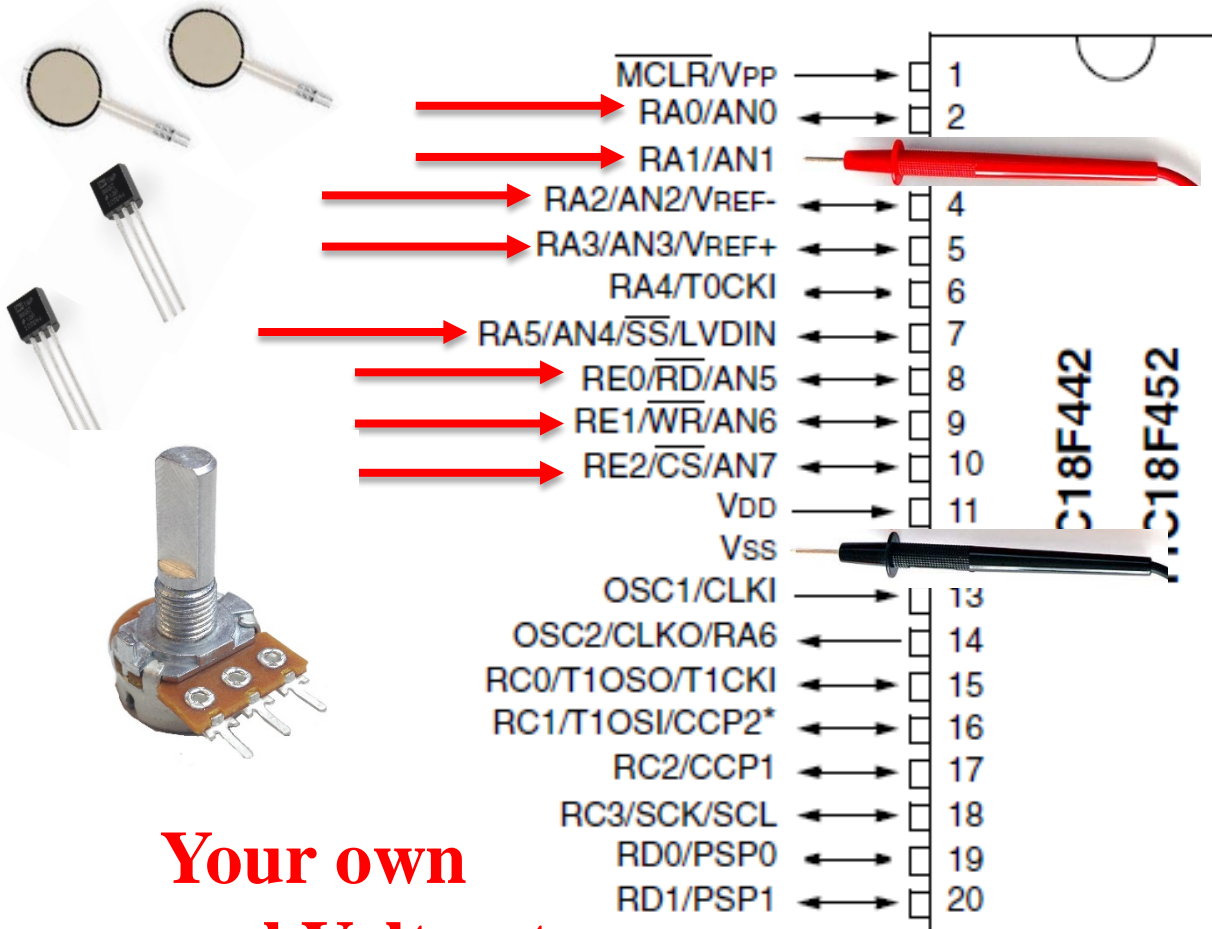
8 Analog Channels (8 Inputs) Only 1 AD Conversion at a time



**Your own
personal Voltmeter**



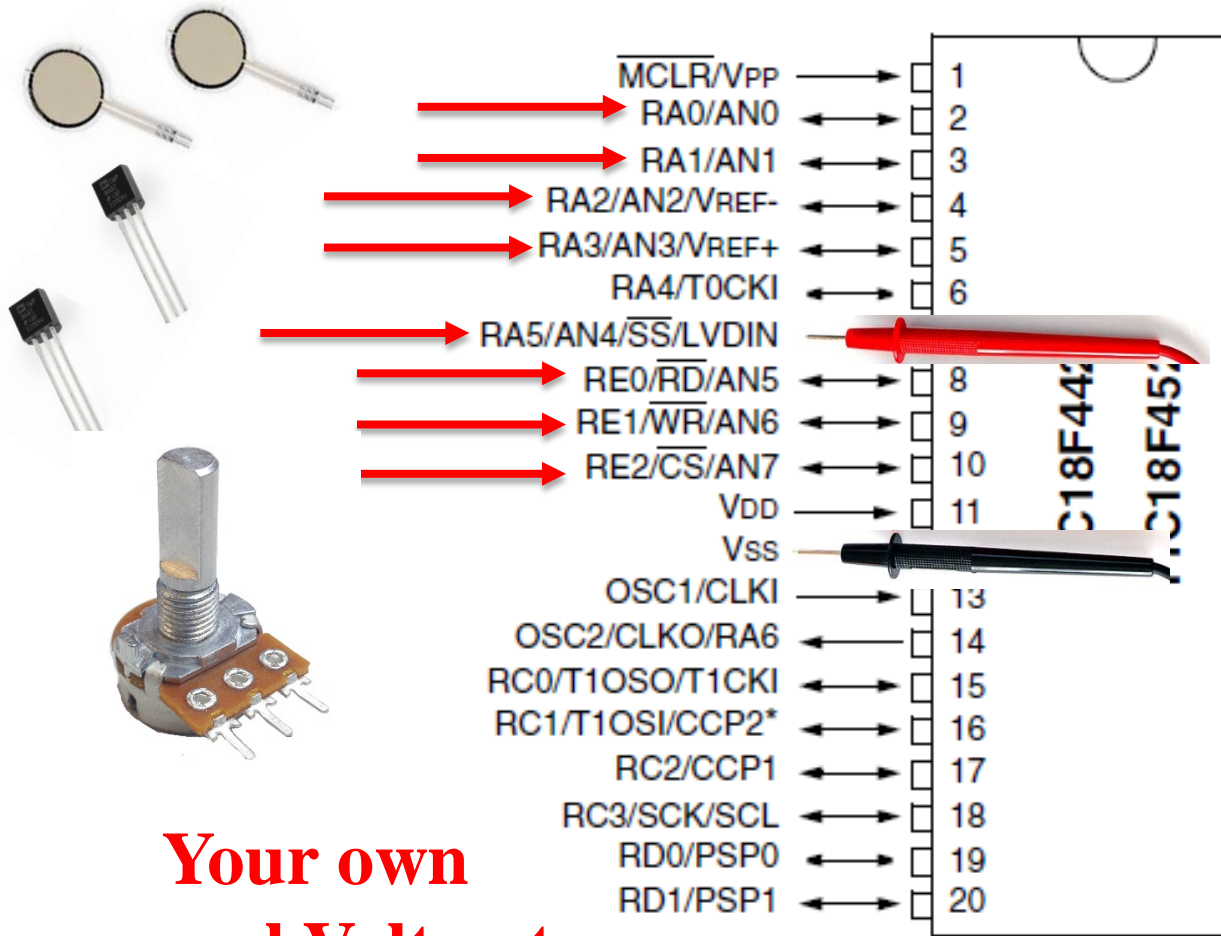
8 Analog Channels (8 Inputs) Only 1 AD Conversion at a time



**Your own
personal Voltmeter**



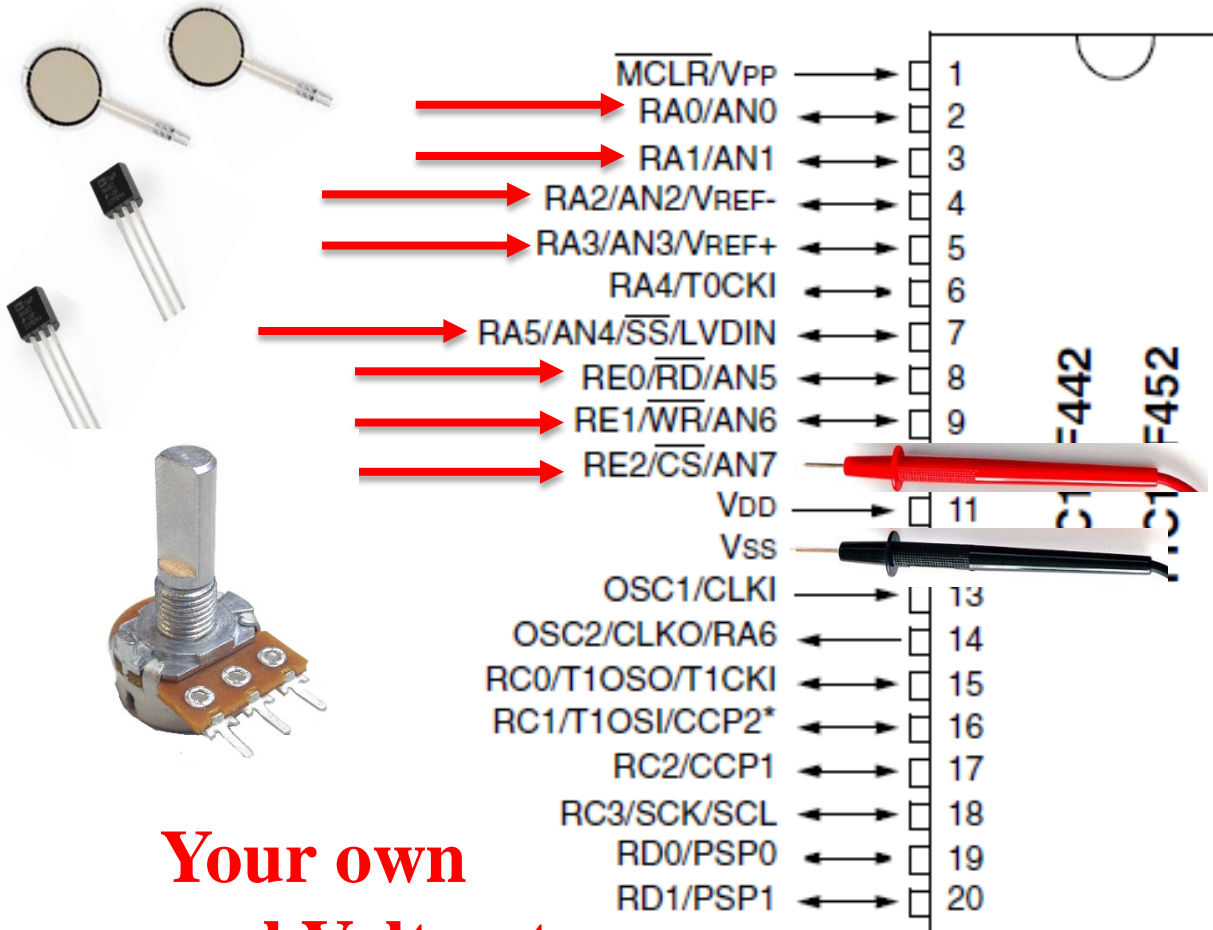
8 Analog Channels (8 Inputs) Only 1 AD Conversion at a time



**Your own
personal Voltmeter**

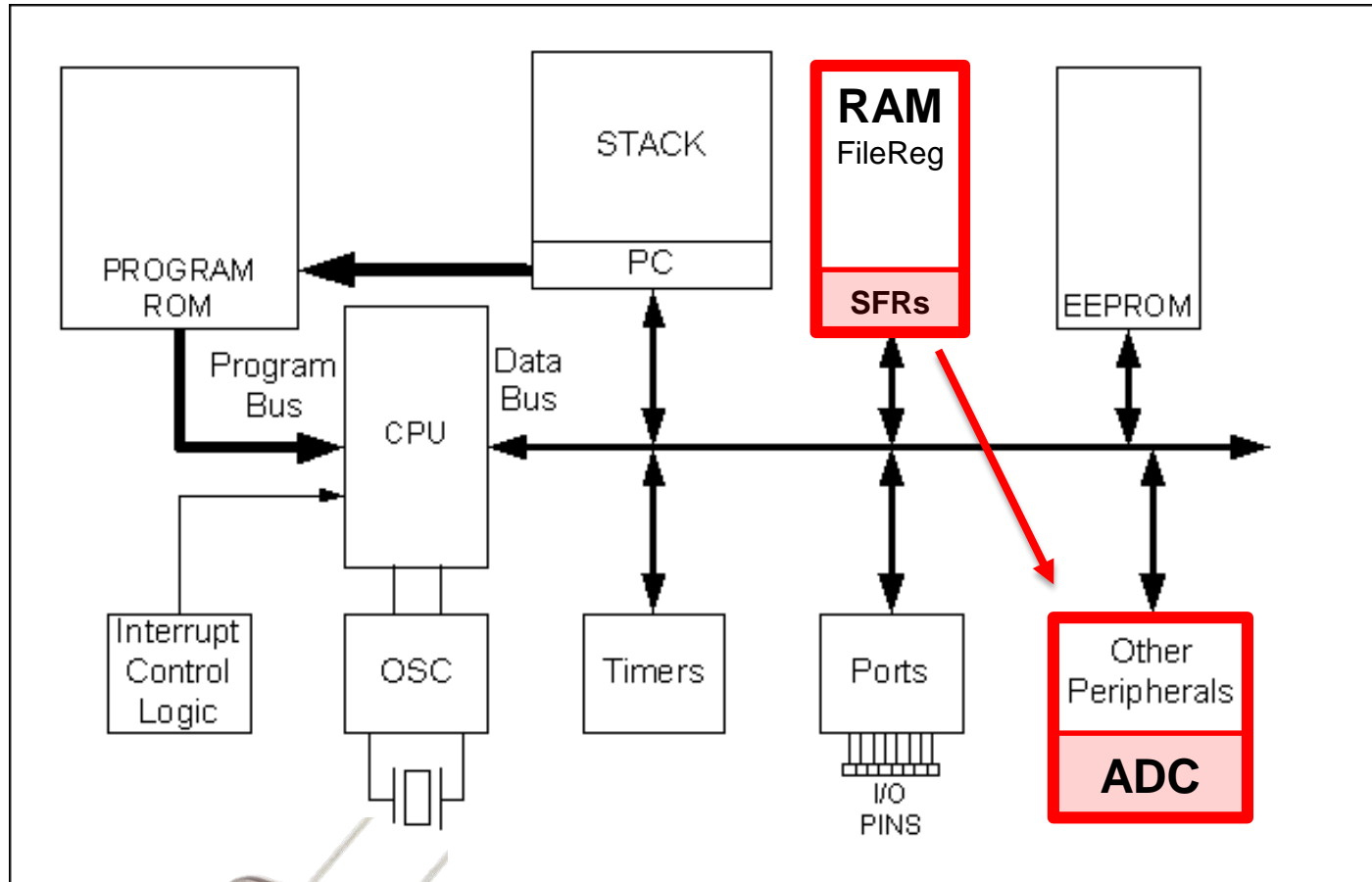


8 Analog Channels (8 Inputs) Only 1 AD Conversion at a time

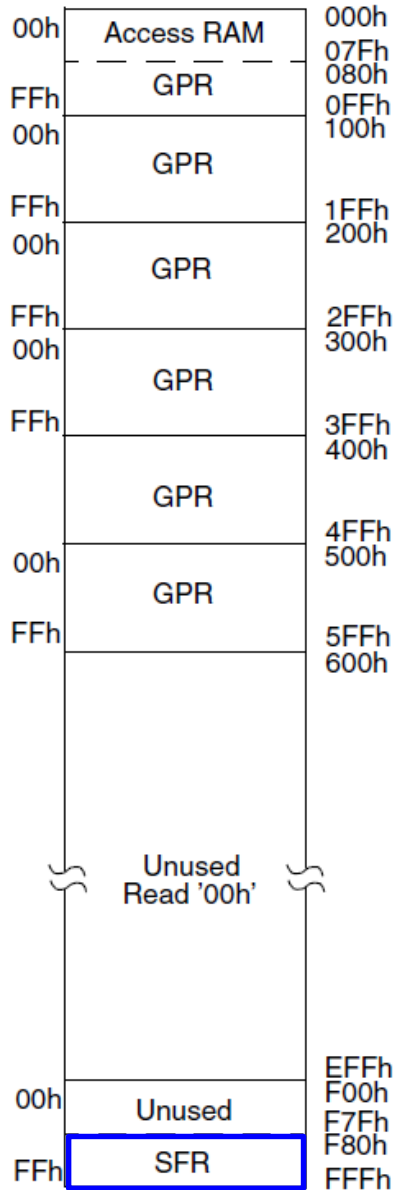


**Your own
personal Voltmeter**

SFRs Used to “Control” the ADC Module



Data Memory Map



PIC18 ADC has 4 SFRs

TABLE 4-1: SPECIAL FUNCTION REGISTER MAP

Address	Name	Address	Name	Address	Name	Address	Name
FFFh	TOSU	FDfH	INDF2 ⁽³⁾	FBFh	CCPR1H	F9Fh	IPR1
FFEh	TOSH	FDEh	POSTINC2 ⁽³⁾	FBEh	CCPR1L	F9Eh	PIR1
FFDh	TOSL	FDDh	POSTDEC2 ⁽³⁾	FBDh	CCP1CON	F9Dh	PIE1
FFCh	STKPTR	FDCh	PREINC2 ⁽³⁾	FBCh	CCPR2H	F9Ch	—
FFBh	PCLATU	FDBh	PLUSW2 ⁽³⁾	FBBh	CCPR2L	F9Bh	—
FFAh	PCLATH	FDAh	FSR2H	FBAh	CCP2CON	F9Ah	—
FF9h	PCL	FD9h	FSR2L	FB9h	—	F99h	—
FF8h	TBLPTRU	FD8h	STATUS	FB8h	—	F98h	—
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	—	F97h	—
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	—	F96h	TRISE ⁽²⁾
FF5h	TABLAT	FD5h	T0CON	FB5h	—	F95h	TRISD ⁽²⁾
FF4h	PRODH	FD4h	—	FB4h	—	F94h	TRISC
FF3h	PRODL	FD3h	OSCCON	FB3h	TMR3H	F93h	TRISB
FF2h	INTCON	FD2h	LVDCON	FB2h	TMR3L	F92h	TRISA
FF1h	INTCON2	FD1h	WDTCON	FB1h	T3CON	F91h	—
FF0h	INTCON3	FD0h	RCON	FB0h	—	F90h	—
FEFh	INDF0 ⁽³⁾	FCFh	TMR1H	FAFh	SPBRG	F8Fh	—
FEEh	POSTINC0 ⁽³⁾	FCEh	TMR1L	FAEh	RCREG	F8Eh	—
FEDh	POSTDEC0 ⁽³⁾	FCDh	T1CON	FADh	TXREG	F8Dh	LATE ⁽²⁾
FECh	PREINC0 ⁽³⁾	FCCh	TMR2	FACH	TXSTA	F8Ch	LATD ⁽²⁾
FEbH	PLUSW0 ⁽³⁾	FCbH	PR2	FABh	RCSTA	F8Bh	LATC
FEAh	FSR0H	FCAh	T2CON	FAAh	—	F8Ah	LATB
FE9h	FSR0L	FC9h	SSPBUF	FA9h	EEADR	F89h	LATA
FE8h	WREG	FC8h	SSPADD	FA8h	EEDATA	F88h	—
FE7h	INDF1 ⁽³⁾	FC7h	SSPSTAT	FA7h	EECON2	F87h	—
FE6h	POSTINC1 ⁽³⁾	FC6h	SSPCON1	FA6h	EECON1	F86h	—
FE5h	POSTDEC1 ⁽³⁾	FC5h	SSPCON2	FA5h	—	F85h	—
FE4h	PREINC1 ⁽³⁾	FC4h	ADRESH	FA4h	—	F84h	PORTE ⁽²⁾
FE3h	PLUSW1 ⁽³⁾	FC3h	ADRESL	FA3h	—	F83h	PORTD ⁽²⁾
FE2h	FSR1H	FC2h	ADCON0	FA2h	IPR2	F82h	PORTC
FE1h	FSR1L	FC1h	ADCON1	FA1h	PIR2	F81h	PORTB
FE0h	BSR	FC0h	—	FA0h	PIE2	F80h	PORTA



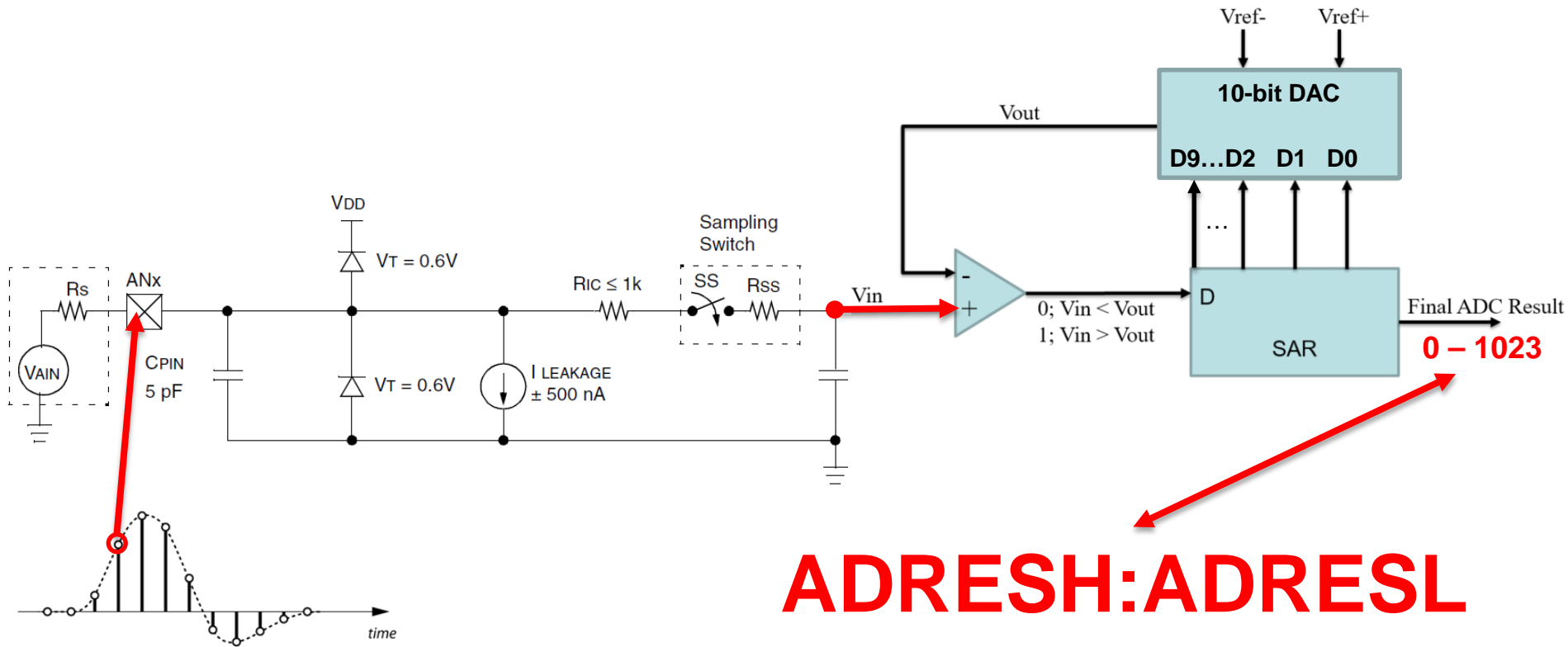
PIC18 ADC has 4 SFRs

The A/D module has four registers. These registers are:

- A/D Result High Register (ADRESH)
- A/D Result Low Register (ADRESL)
- A/D Control Register 0 (ADCON0)
- A/D Control Register 1 (ADCON1)

ADRESH/L

AD Result is 10-bit (0 – 1023)

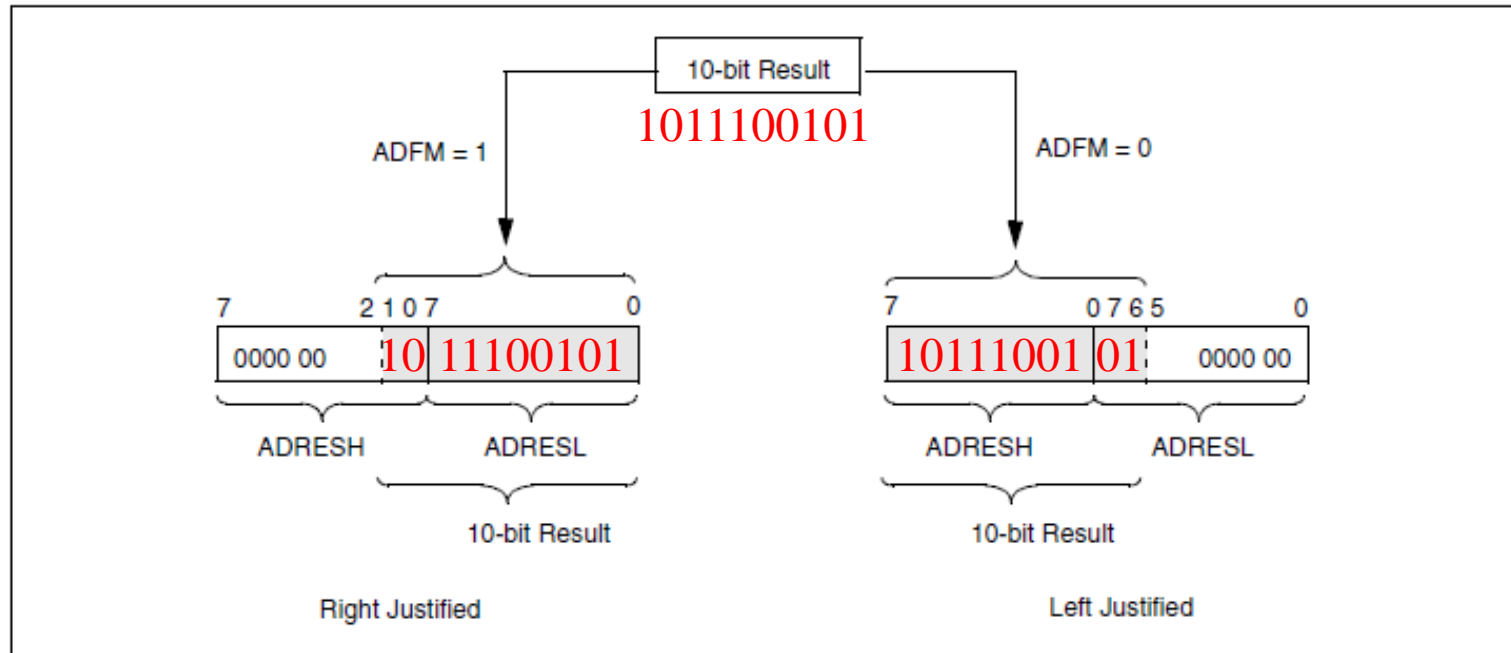


ADRESH/L

AD Result is 10-bit (0 – 1023)

- Ex: AD Result = 741 decimal
= 1011100101 binary
= 10 11100101 binary

FIGURE 17-4: A/D RESULT JUSTIFICATION



ADCON0

17-1: ADCON0 REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/DONE	—	ADON
bit 7							bit 0

bit 7-6 **ADCS1:ADCS0**: A/D Conversion Clock Select bits (ADCON0 bits in bold)

ADCON1 <ADCS2>	ADCON0 <ADCS1:ADCS0>	Clock Conversion
0	00	Fosc/2
0	01	Fosc/8
0	10	Fosc/32
0	11	FRC (clock derived from the internal A/D RC oscillator)
1	00	Fosc/4
1	01	Fosc/16
1	10	Fosc/64
1	11	FRC (clock derived from the internal A/D RC oscillator)

Choose the speed of the AD Conversion



Which analog channel to convert



bit 5-3 **CHS2:CHS0**: Analog Channel Select bits

- 000 = channel 0, (AN0)
- 001 = channel 1, (AN1)
- 010 = channel 2, (AN2)
- 011 = channel 3, (AN3)
- 100 = channel 4, (AN4)
- 101 = channel 5, (AN5)
- 110 = channel 6, (AN6)
- 111 = channel 7, (AN7)

Note: The PIC18F2X2 devices do not implement the full 8 A/D channels; the unimplemented selections are reserved. Do not select any unimplemented channel.

START the actual ADC process (final step)



bit 2 **GO/DONE**: A/D Conversion Status bit

When ADON = 1:

- 1 = A/D conversion in progress (setting this bit starts the A/D conversion which is automatically cleared by hardware when the A/D conversion is complete)
- 0 = A/D conversion not in progress

Unimplemented: Read as '0'

bit 1

bit 0

ADON: A/D On bit

- 1 = A/D converter module is powered up
- 0 = A/D converter module is shut-off and consumes no operating current

Turn on or “Power Up” the ADC module



ADCON1

R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	ADCS2	—	—	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

ADFM: A/D Result Format Select bit

1 = Right justified. Six (6) Most Significant bits of ADRESH are read as '0'.
 0 = Left justified. Six (6) Least Significant bits of ADRESL are read as '0'.

ADCS2: A/D Conversion Clock Select bit (ADCON1 bits in **bold**)

ADCON1 <ADCS2>	ADCON0 <ADCS1:ADCS0>	Clock Conversion
0	00	Fosc/2
0	01	Fosc/8
0	10	Fosc/32
0	11	Frc (clock derived from the internal A/D RC oscillator)
1	00	Fosc/4
1	01	Fosc/16
1	10	Fosc/64
1	11	Frc (clock derived from the internal A/D RC oscillator)

Unimplemented: Read as '0'

PCFG3:PCFG0: A/D Port Configuration Control bits

PCFG <3:0>	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0	VREF+	VREF-	C / R
0000	A	A	A	A	A	A	A	A	VDD	VSS	8 / 0
0001	A	A	A	A	VREF+	A	A	A	AN3	VSS	7 / 1
0010	D	D	D	A	A	A	A	A	VDD	VSS	5 / 0
0011	D	D	D	A	VREF+	A	A	A	AN3	VSS	4 / 1
0100	D	D	D	D	A	D	A	A	VDD	VSS	3 / 0
0101	D	D	D	D	VREF+	D	A	A	AN3	VSS	2 / 1
011x	D	D	D	D	D	D	D	D	—	—	0 / 0
1000	A	A	A	A	VREF+	VREF-	A	A	AN3	AN2	6 / 2
1001	D	D	A	A	A	A	A	A	VDD	VSS	6 / 0
1010	D	D	A	A	VREF+	A	A	A	AN3	VSS	5 / 1
1011	D	D	A	A	VREF+	VREF-	A	A	AN3	AN2	4 / 2
1100	D	D	D	A	VREF+	VREF-	A	A	AN3	AN2	3 / 2
1101	D	D	D	D	VREF+	VREF-	A	A	AN3	AN2	2 / 2
1110	D	D	D	D	D	D	D	A	VDD	VSS	1 / 0
1111	D	D	D	D	VREF+	VREF-	D	A	AN3	AN2	1 / 2

A = Analog input D = Digital I/O

C/R = # of analog input channels / # of A/D voltage references

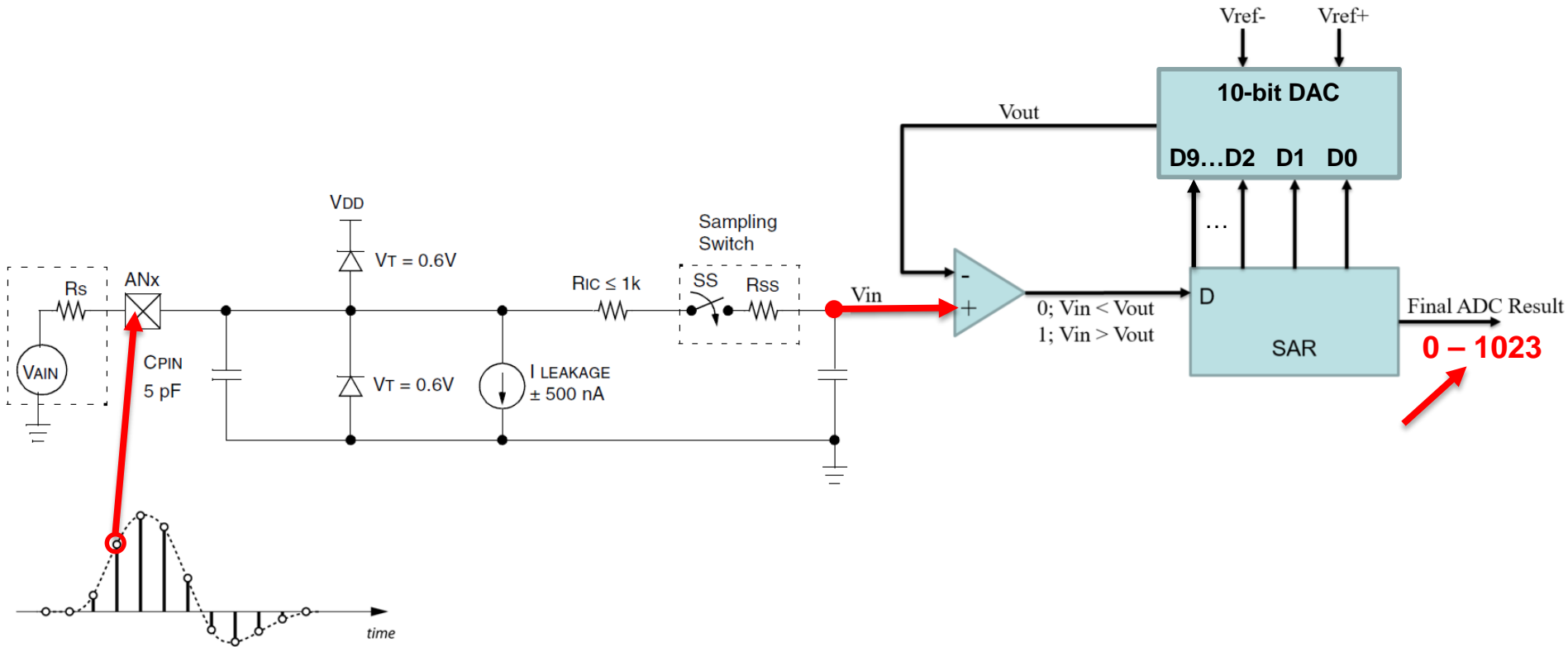
ADC result right or left justified

Choose the speed of the AD Conversion

Pick from this table, which analog pins are AN or DIG and what Vrefs you want

ADC Time

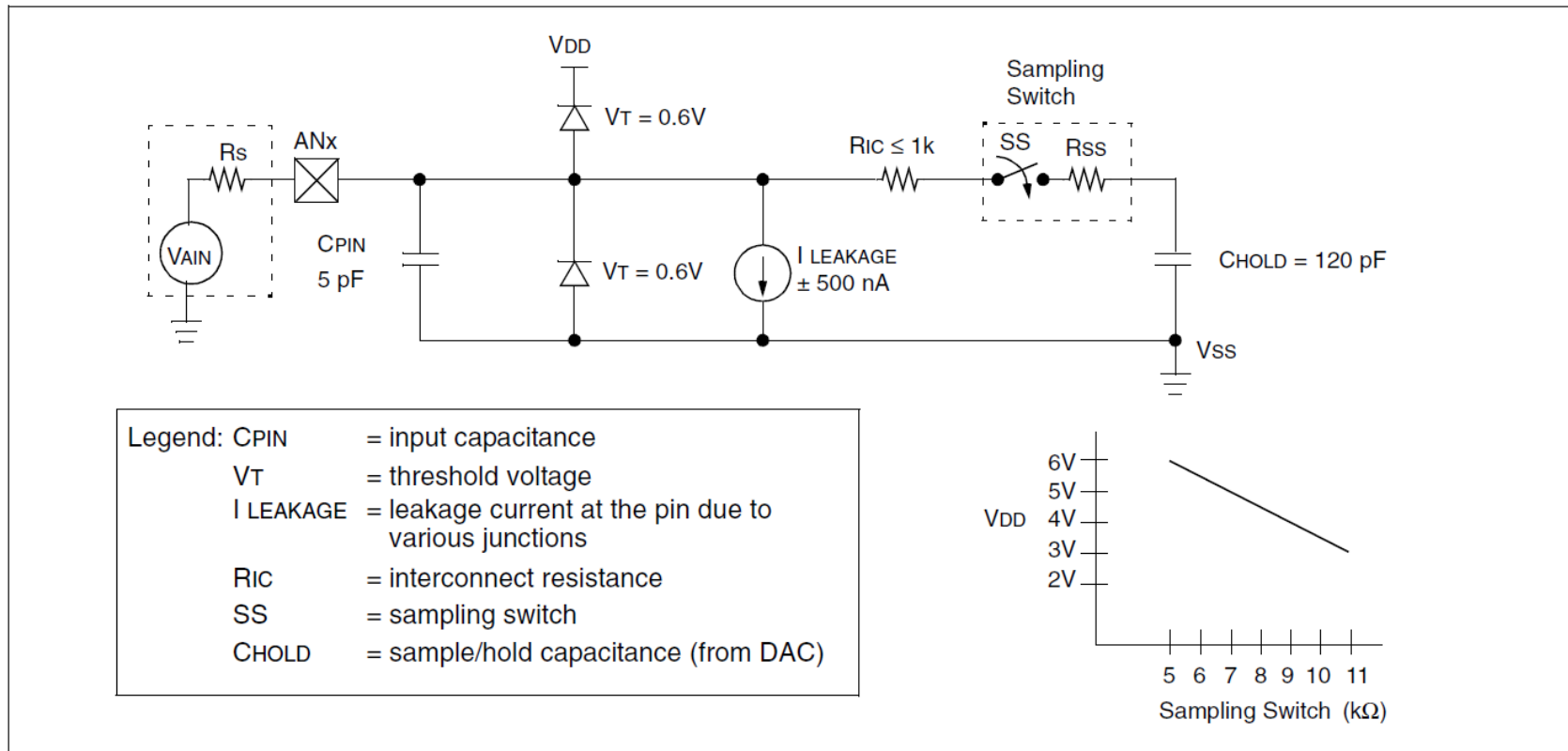
Acquisition Time + Conversion Time



PIC18 A/D Acquisition Time

- The A/D acquisition time is essentially the time required for the **hold capacitor to charge**. Typical value of 5-15 μ s.
- Newer PICs have specific control for **T_{Acq}** (determined in T_{ADs})

FIGURE 17-2: ANALOG INPUT MODEL





PIC18 A/D Acquisition Time

$$\begin{aligned}T_{ACQ} &= \text{Amplifier Settling Time} + \text{Holding Capacitor Charging Time} + \text{Temperature Coefficient} \\ &= T_{AMP} + T_C + T_{COFF}\end{aligned}$$

$$T_{ACQ} = T_{AMP} + T_C + T_{COFF}$$

Temperature coefficient is only required for temperatures $> 25^{\circ}\text{C}$.

$$T_{ACQ} = 2 \mu\text{s} + T_C + [(\text{Temp} - 25^{\circ}\text{C})(0.05 \mu\text{s}/^{\circ}\text{C})]$$

$$\begin{aligned}T_C &= -\text{CHOLD} (R_{IC} + R_{SS} + R_S) \ln(1/2048) \\ &= -120 \text{ pF} (1 \text{ k}\Omega + 7 \text{ k}\Omega + 2.5 \text{ k}\Omega) \ln(0.0004883) \\ &= -120 \text{ pF} (10.5 \text{ k}\Omega) \ln(0.0004883) \\ &= -1.26 \mu\text{s} (-7.6246) \\ &= 9.61 \mu\text{s}\end{aligned}$$

$$\begin{aligned}T_{ACQ} &= 2 \mu\text{s} + 9.61 \mu\text{s} + [(50^{\circ}\text{C} - 25^{\circ}\text{C})(0.05 \mu\text{s}/^{\circ}\text{C})] \\ &= 11.61 \mu\text{s} + 1.25 \mu\text{s} \\ &= 12.86 \mu\text{s}\end{aligned}$$



PIC18 A/D Conversion Time

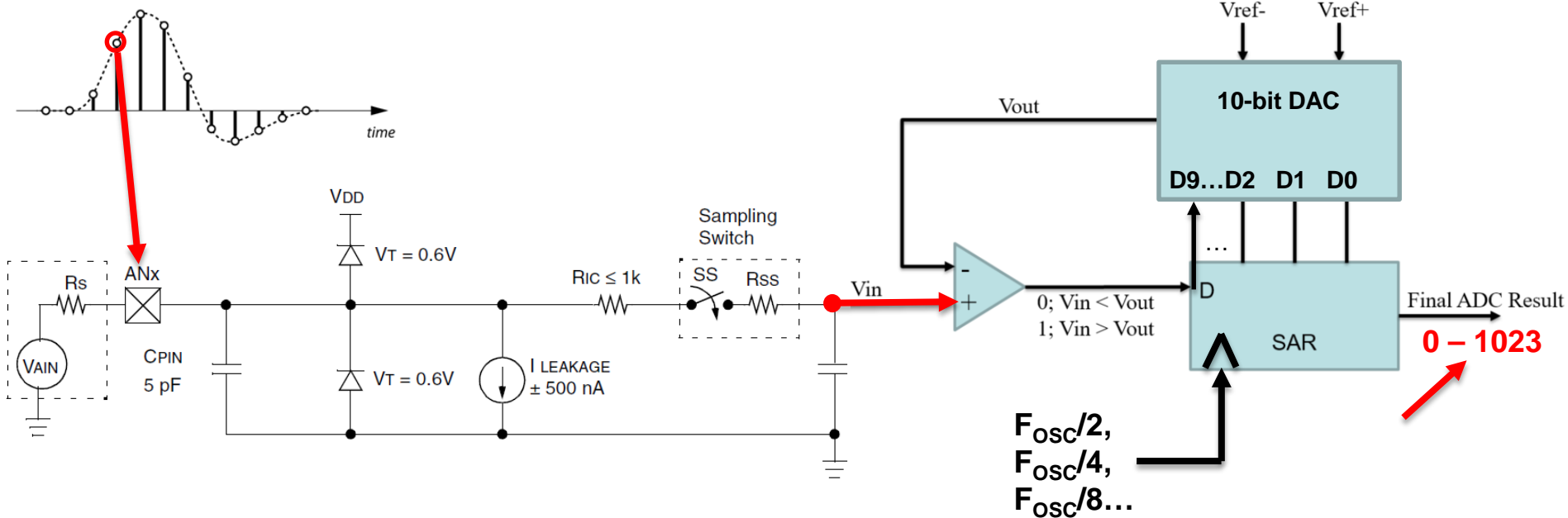
- **T_{AD} is the conversion time per bit**, i.e., the clock at which the successive approximation runs plus overhead
- In addition to the 10 T_{AD} cycles for conversion we need two more T_{AD} cycles. Thus the PIC18 takes a total conversion time of **$12 T_{ADS}$** .
- **However, a single T_{AD} has to be at least $1.6\mu s$!**
- T_{AD} is determined by the conversion clock (from the **F_{osc}**)

bit 7-6 **ADCS1:ADCS0**: A/D Conversion Clock Select bits (ADCON0 bits in bold)

ADCON1 <ADCS2>	ADCON0 <ADCS1:ADCS0>	Clock Conversion
0	00	$F_{osc}/2$
0	01	$F_{osc}/8$
0	10	$F_{osc}/32$
0	11	FRC (clock derived from the internal A/D RC oscillator)
1	00	$F_{osc}/4$
1	01	$F_{osc}/16$
1	10	$F_{osc}/64$
1	11	FRC (clock derived from the internal A/D RC oscillator)

Total ADC Time

Acquisition Time + Conversion Time

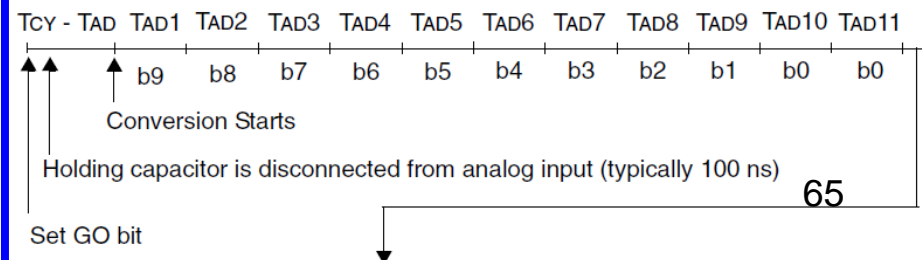


$$T_{ACQ} = 2 \mu s + 9.61 \mu s + [(50^\circ C - 25^\circ C)(0.05 \mu s/^\circ C)]$$

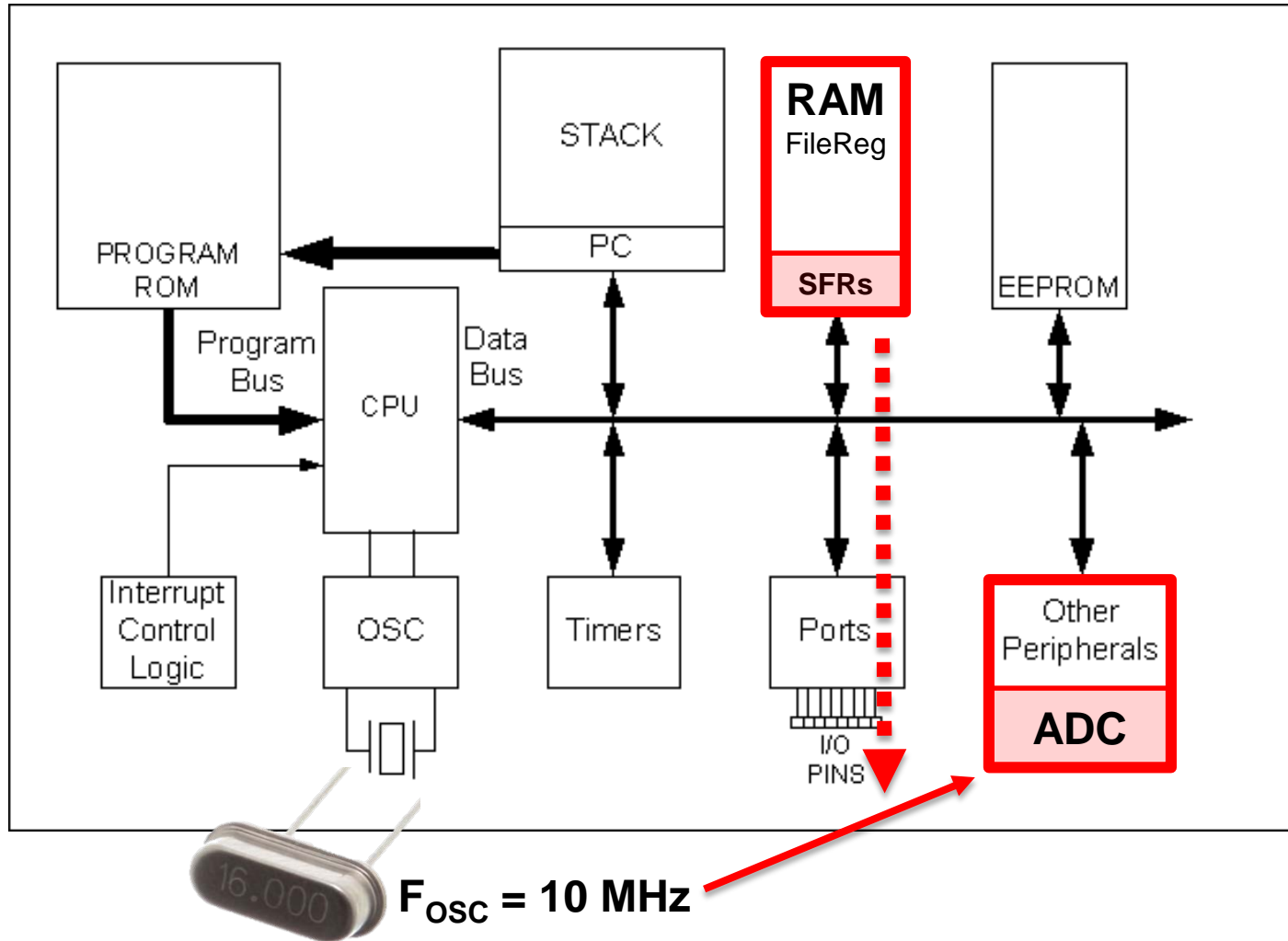
$$11.61 \mu s + 1.25 \mu s$$

$$12.86 \mu s$$

E 17-3: A/D CONVERSION TAD CYCLES



F_{osc} has to be “Cut Down” Slow Enough for the ADC





Setting the ADC's Clock Speed

$$F_{\text{osc}} = \mathbf{10\text{MHz}}$$

$$T_{\text{osc}} = \frac{1}{10 \text{ MHz}} = \frac{1}{10} \mu\text{s} = \mathbf{0.1 \mu\text{s}}$$



Setting the ADC's Clock Speed

$$F_{\text{osc}} = \mathbf{10\text{MHz}}$$

$$T_{\text{osc}} = \frac{1}{10 \text{ MHz}} = \frac{1}{10} \mu\text{s} = \mathbf{0.1 \mu\text{s}}$$

Note: $F_{\text{osc}}/x = x * T_{\text{osc}}$



Setting the ADC's Clock Speed

$$F_{\text{OSC}} = \mathbf{10\text{MHz}} \qquad T_{\text{OSC}} = \frac{1}{10 \text{ MHz}} = \frac{1}{10} \mu\text{s} = \mathbf{0.1 \mu\text{s}}$$

Note: $F_{\text{OSC}}/x = x * T_{\text{OSC}}$

Need: a single T_{AD} to be at least **1.6 μs**



Setting the ADC's Clock Speed

$$F_{\text{OSC}} = \mathbf{10\text{MHz}} \qquad T_{\text{OSC}} = \frac{1}{10 \text{ MHz}} = \frac{1}{10} \mu\text{s} = \mathbf{0.1 \mu\text{s}}$$

Note: $F_{\text{OSC}}/x = x * T_{\text{OSC}}$

Need: a single T_{AD} to be at least **1.6 μs**

$$1 * T_{\text{OSC}} = 0.1 \mu\text{s} < 1.6 \mu\text{s} \quad \mathbf{X}$$



Setting the ADC's Clock Speed

$$F_{\text{OSC}} = \mathbf{10\text{MHz}} \qquad T_{\text{OSC}} = \frac{1}{10 \text{ MHz}} = \frac{1}{10} \mu\text{s} = \mathbf{0.1 \mu\text{s}}$$

Note: $F_{\text{OSC}}/x = x * T_{\text{OSC}}$

Need: a single T_{AD} to be at least **1.6 μs**

$$1 * T_{\text{OSC}} = 0.1 \mu\text{s} < 1.6 \mu\text{s} \quad \mathbf{X}$$

$$2 * T_{\text{OSC}} = 0.2 \mu\text{s} < 1.6 \mu\text{s} \quad \mathbf{X}$$



Setting the ADC's Clock Speed

$$F_{\text{OSC}} = \mathbf{10\text{MHz}} \qquad T_{\text{OSC}} = \frac{1}{10 \text{ MHz}} = \frac{1}{10} \mu\text{s} = \mathbf{0.1 \mu\text{s}}$$

Note: $F_{\text{OSC}}/x = x * T_{\text{OSC}}$

Need: a single T_{AD} to be at least **1.6 μs**

$$1 * T_{\text{OSC}} = 0.1 \mu\text{s} < 1.6 \mu\text{s} \text{ X}$$

$$2 * T_{\text{OSC}} = 0.2 \mu\text{s} < 1.6 \mu\text{s} \text{ X}$$

$$4 * T_{\text{OSC}} = 0.4 \mu\text{s} < 1.6 \mu\text{s} \text{ X}$$



Setting the ADC's Clock Speed

$$F_{\text{OSC}} = \mathbf{10\text{MHz}} \qquad T_{\text{OSC}} = \frac{1}{10 \text{ MHz}} = \frac{1}{10} \mu\text{s} = \mathbf{0.1 \mu\text{s}}$$

Note: $F_{\text{OSC}}/x = x * T_{\text{OSC}}$

Need: a single T_{AD} to be at least **1.6 μs**

$$1 * T_{\text{OSC}} = 0.1 \mu\text{s} < 1.6 \mu\text{s} \text{ X}$$

$$2 * T_{\text{OSC}} = 0.2 \mu\text{s} < 1.6 \mu\text{s} \text{ X}$$

$$4 * T_{\text{OSC}} = 0.4 \mu\text{s} < 1.6 \mu\text{s} \text{ X}$$

$$8 * T_{\text{OSC}} = 0.8 \mu\text{s} < 1.6 \mu\text{s} \text{ X}$$



Setting the ADC's Clock Speed

$$F_{\text{OSC}} = \mathbf{10\text{MHz}} \qquad T_{\text{OSC}} = \frac{1}{10 \text{ MHz}} = \frac{1}{10} \mu\text{s} = \mathbf{0.1 \mu\text{s}}$$

Note: $F_{\text{OSC}}/x = x * T_{\text{OSC}}$

Need: a single T_{AD} to be at least **1.6 μs**

$$1 * T_{\text{OSC}} = 0.1 \mu\text{s} < 1.6 \mu\text{s} \quad \mathbf{X}$$

$$2 * T_{\text{OSC}} = 0.2 \mu\text{s} < 1.6 \mu\text{s} \quad \mathbf{X}$$

$$4 * T_{\text{OSC}} = 0.4 \mu\text{s} < 1.6 \mu\text{s} \quad \mathbf{X}$$

$$8 * T_{\text{OSC}} = 0.8 \mu\text{s} < 1.6 \mu\text{s} \quad \mathbf{X}$$

$$16 * T_{\text{OSC}} = 1.6 \mu\text{s} \geq 1.6 \mu\text{s} \quad \mathbf{\checkmark}$$

T_{AD} 1.6us Minimum

FIGURE 17-3: A/D CONVERSION TAD CYCLES

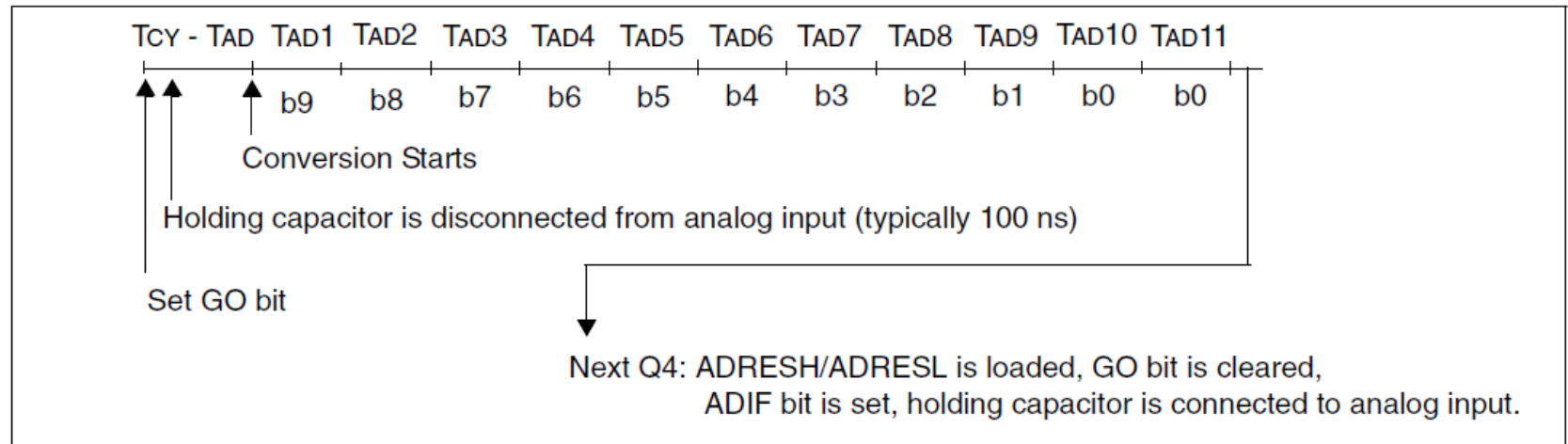
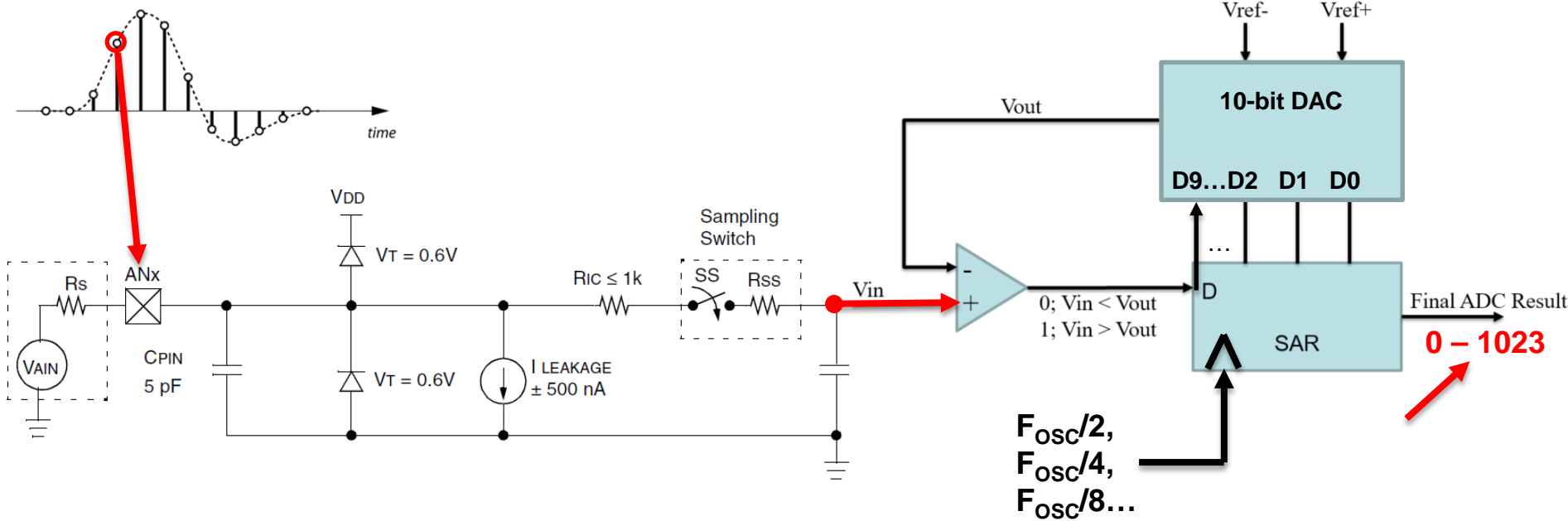


TABLE 17-1: TAD vs. DEVICE OPERATING FREQUENCIES

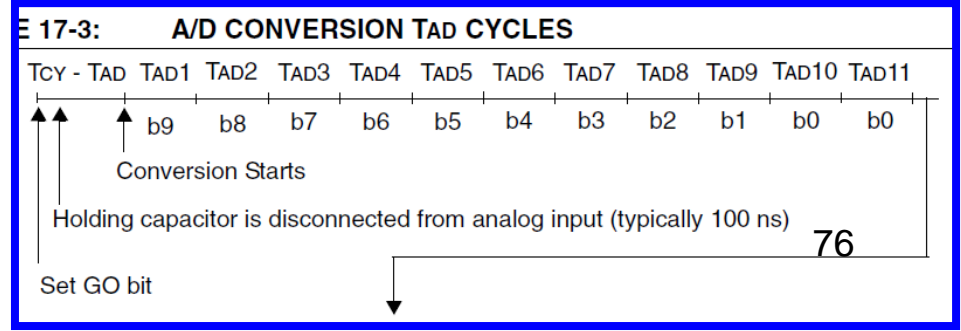
AD Clock Source (TAD)		Maximum Device Frequency	
Operation	ADCS2:ADCS0	PIC18FXX2	PIC18LFX2
2 Tosc	000	1.25 MHz	666 kHz
4 Tosc	100	2.50 MHz	1.33 MHz
8 Tosc	001	5.00 MHz	2.67 MHz
16 Tosc	101	10.00 MHz	5.33 MHz
32 Tosc	010	20.00 MHz	10.67 MHz
64 Tosc	110	40.00 MHz	21.33 MHz
RC	011	—	—

Total ADC Time

Acquisition Time + Conversion Time

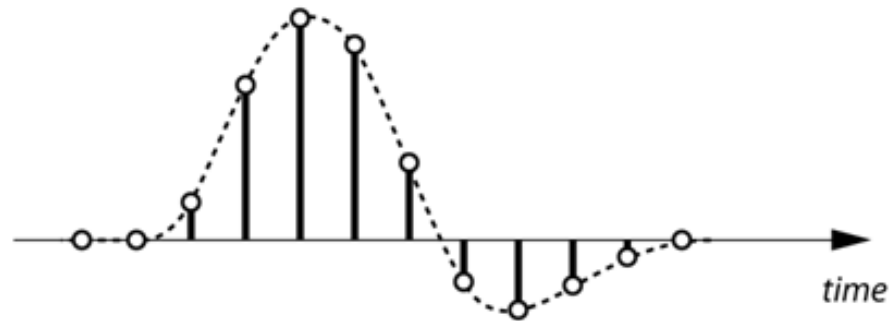


$$\begin{aligned}
 T_{ACQ} &= 2 \mu s + 9.61 \mu s + [(50^{\circ}C - 25^{\circ}C)(0.05 \mu s/^{\circ}C)] \\
 &= 11.61 \mu s + 1.25 \mu s \\
 &= 12.86 \mu s
 \end{aligned}$$



Sampling Frequency

- Note, that we have only described a single conversion
- If periodical conversion needs to be set up, the GO bit has to be turned on periodically and data produced (ADRESH, ADRESL) has to be read periodically



- This can be done by:
 - Calculating timing (i.e., NOP instructions after DONE)
 - Using timer peripherals (not covered yet)
 - Proper interrupt processing



Steps When Using the PIC18 ADC

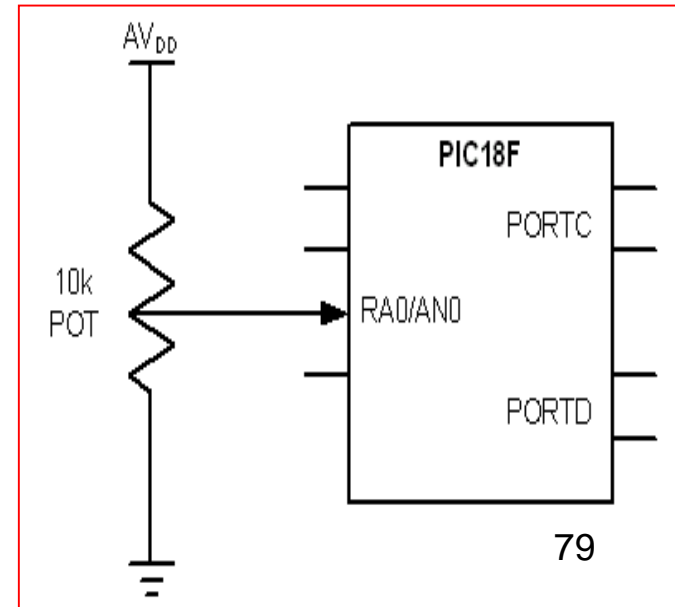
1. Configure the A/D module:
 - Configure analog pins, voltage reference and digital I/O (ADCON1)
 - Select A/D input channel (ADCON0)
 - Select A/D conversion clock (ADCON0)
 - Turn on A/D module (ADCON0)
2. Configure A/D interrupt (if desired):
 - Clear ADIF bit
 - Set ADIE bit
 - Set GIE bit
3. Wait the required acquisition time.
4. Start conversion:
 - Set GO/DONE bit (ADCON0)
5. Wait for A/D conversion to complete, by either:
 - Polling for the GO/DONE bit to be cleared OR
 - Waiting for the A/D interrupt
6. Read A/D Result registers (ADRESH/ADRESL);
 - clear bit ADIF if required.
7. For next conversion, go to step 1 or step 2 as required.
 - The A/D conversion time per bit is defined as TAD. A minimum wait of 2 TAD is required before next acquisition starts.



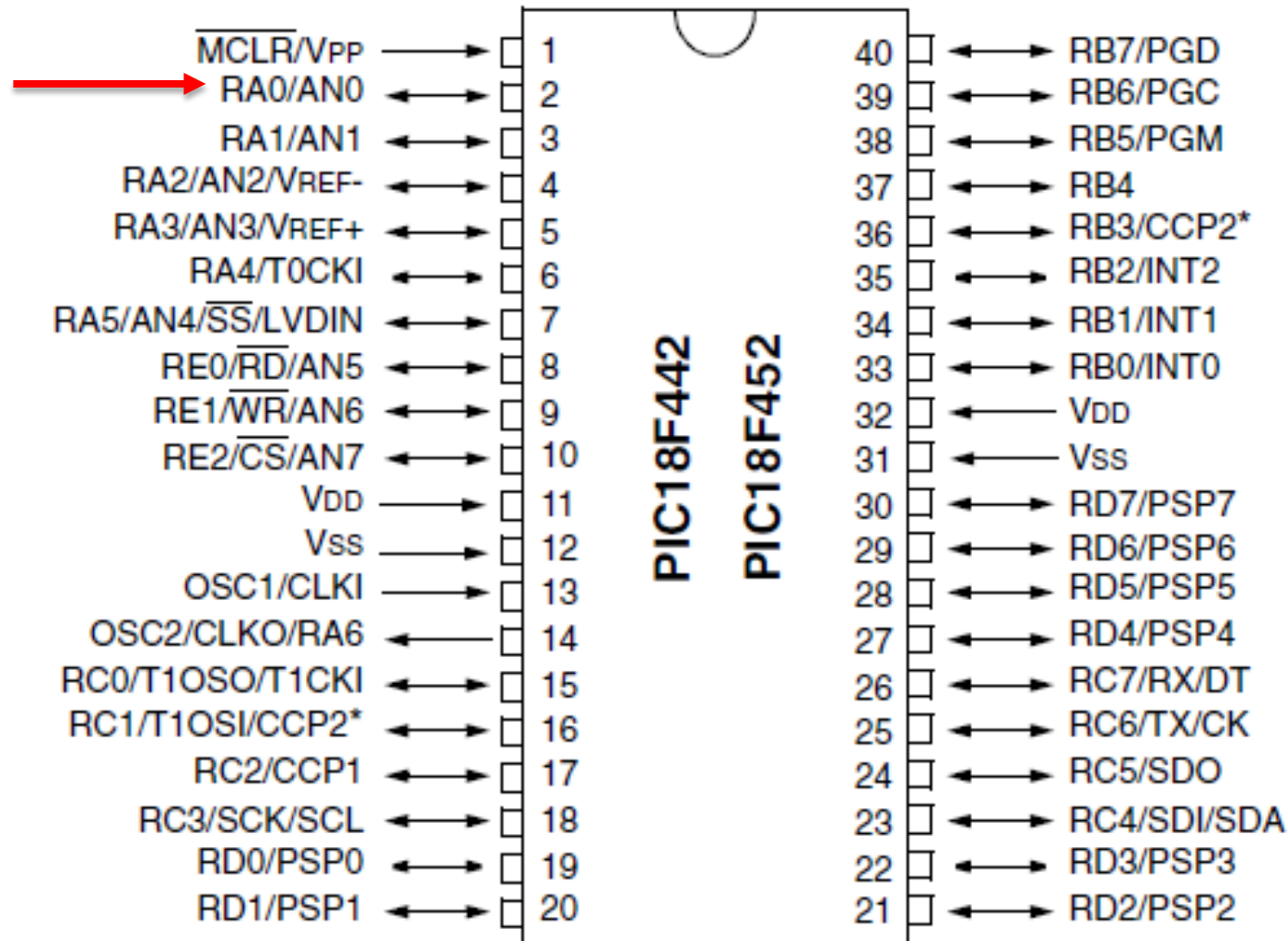
Programming the PIC18 ADC

```
unsigned char adLow = 0;
unsigned char adHigh = 0;
int fullAD = 0;
TRISAbits.TRISA0 = 1; //A0 is an input
ADCON0 = 0b01000001; //FOSC/16, channel 0, A/D module turned ON
ADCON1 = 0b11001110; //right justified (FOSC/16) AN0=analog
```

```
while(1)
{
    //Delay >= TACQ //Datasheet pg. 185
    ADCON0bits.GO = 1;
    while(ADCON0bits.DONE == 1) ;
    adLow = ADRESL;
    adHigh = ADRESH;
    //combine into a single 10-bit number
    DELAY(250); // set your sampling rate (in ms)
}
```



AN0 as Analog Input





17-1: ADCON0 REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/DONE	—	ADON
bit 7							bit 0

bit 7-6 **ADCS1:ADCS0: A/D Conversion Clock Select bits (ADCON0 bits in bold)**

ADCON1 <ADCS2>	ADCON0 <ADCS1:ADCS0>	Clock Conversion
0	00	Fosc/2
0	01	Fosc/8
0	10	Fosc/32
0	11	FRC (clock derived from the internal A/D RC oscillator)
1	00	Fosc/4
1	01	Fosc/16
1	10	Fosc/64
1	11	FRC (clock derived from the internal A/D RC oscillator)



bit 5-3 **CHS2:CHS0: Analog Channel Select bits**



- 000 = channel 0, (AN0)
- 001 = channel 1, (AN1)
- 010 = channel 2, (AN2)
- 011 = channel 3, (AN3)
- 100 = channel 4, (AN4)
- 101 = channel 5, (AN5)
- 110 = channel 6, (AN6)
- 111 = channel 7, (AN7)

Note: The PIC18F2X2 devices do not implement the full 8 A/D channels; the unimplemented selections are reserved. Do not select any unimplemented channel.

bit 2 **GO/DONE:** A/D Conversion Status bit

When ADON = 1:

- 1 = A/D conversion in progress (setting this bit starts the A/D conversion which is automatically cleared by hardware when the A/D conversion is complete)
- 0 = A/D conversion not in progress

bit 1 **Unimplemented:** Read as '0'

bit 0 **ADON:** A/D On bit



- 1 = A/D converter module is powered up
- 0 = A/D converter module is shut-off and consumes no operating current

TER 17-2: ADCON1 REGISTER


R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	ADCS2	—	—	PCFG3	PCFG2	PCFG1	PCFG0
bit 7				bit 0			

- bit 7 **ADFM:** A/D Result Format Select bit
 1 = Right justified. Six (6) Most Significant bits of ADRESH are read as '0'.
 0 = Left justified. Six (6) Least Significant bits of ADRESL are read as '0'.
- bit 6 **ADCS2:** A/D Conversion Clock Select bit (ADCON1 bits in **bold**)

ADCON1 <ADCS2>	ADCON0 <ADCS1:ADCS0>	Clock Conversion
0	00	Fosc/2
0	01	Fosc/8
0	10	Fosc/32
0	11	Frc (clock derived from the internal A/D RC oscillator)
1	00	Fosc/4
1	01	Fosc/16
1	10	Fosc/64
1	11	Frc (clock derived from the internal A/D RC oscillator)

- bit 5-4 **Unimplemented:** Read as '0'
- bit 3-0 **PCFG3:PCFG0:** A/D Port Configuration Control bits

PCFG <3:0>	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0	VREF+	VREF-	C / R
0000	A	A	A	A	A	A	A	A	VDD	Vss	8 / 0
0001	A	A	A	A	VREF+	A	A	A	AN3	Vss	7 / 1
0010	D	D	D	A	A	A	A	A	VDD	Vss	5 / 0
0011	D	D	D	A	VREF+	A	A	A	AN3	Vss	4 / 1
0100	D	D	D	D	A	D	A	A	VDD	Vss	3 / 0
0101	D	D	D	D	VREF+	D	A	A	AN3	Vss	2 / 1
011x	D	D	D	D	D	D	D	D	—	—	0 / 0
1000	A	A	A	A	VREF+	VREF-	A	A	AN3	AN2	6 / 2
1001	D	D	A	A	A	A	A	A	VDD	Vss	6 / 0
1010	D	D	A	A	VREF+	A	A	A	AN3	Vss	5 / 1
1011	D	D	A	A	VREF+	VREF-	A	A	AN3	AN2	4 / 2
1100	D	D	D	A	VREF+	VREF-	A	A	AN3	AN2	3 / 2
1101	D	D	D	D	VREF+	VREF-	A	A	AN3	AN2	2 / 2
1110	D	D	D	D	D	D	D	A	VDD	Vss	1 / 0
1111	D	D	D	D	VREF+	VREF-	D	A	AN3	AN2	1 / 2

A = Analog input D = Digital I/O

C/R = # of analog input channels / # of A/D voltage references

ADCON1

TER 17-2: ADCON1 REGISTER

R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	ADCS2	—	—	PCFG3	PCFG2	PCFG1	PCFG0
bit 7				bit 0			

bit 7 **ADFM:** A/D Result Format Select bit

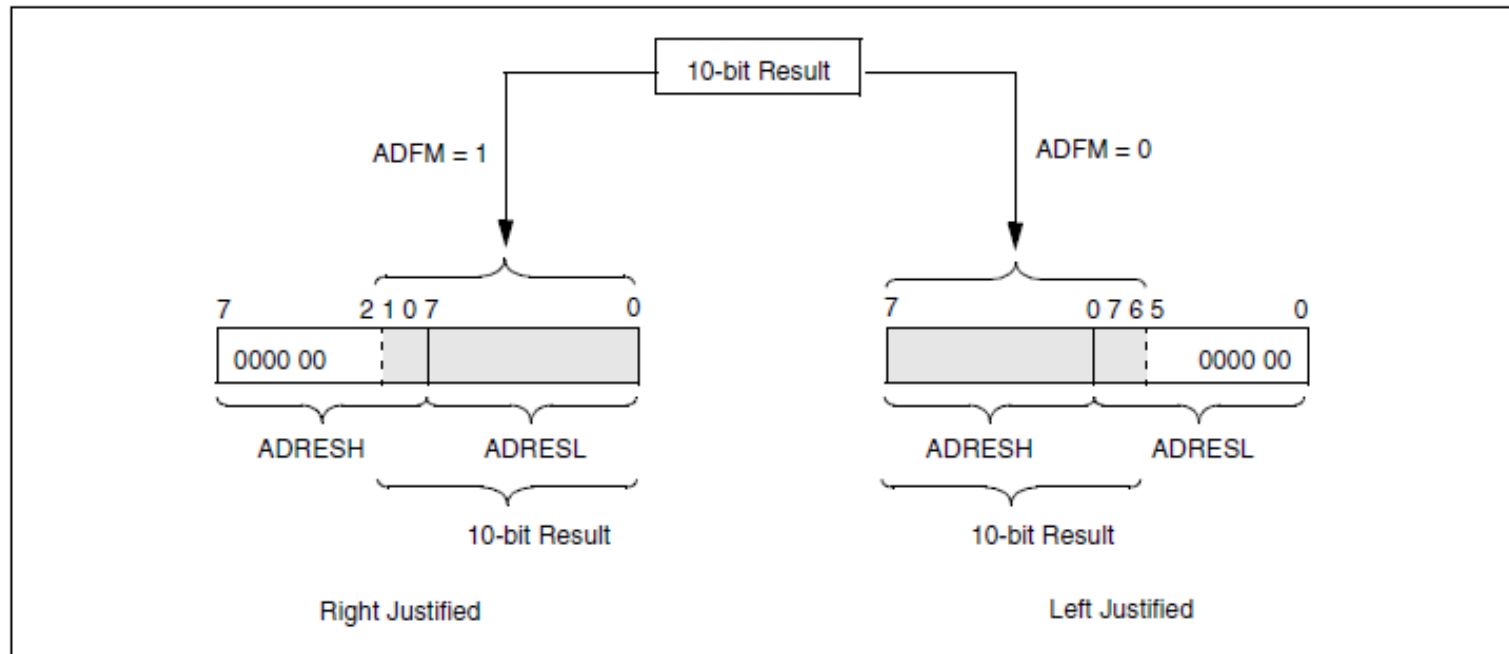
1 = Right justified. Six (6) Most Significant bits of ADRESH are read as '0'.
 0 = Left justified. Six (6) Least Significant bits of ADRESL are read as '0'.

bit 6 **ADCS2:** A/D Conversion Clock Select bit (ADCON1 bits in **bold**)

ADCON1 <ADCS2>	ADCON0 <ADCS1:ADCS0>	Clock Conversion
0	00	$F_{osc}/2$
0	01	$F_{osc}/8$
0	10	$F_{osc}/32$
0	11	FRC (clock derived from the internal A/D RC oscillator)
1	00	$F_{osc}/4$
1	01	$F_{osc}/16$
1	10	$F_{osc}/64$
1	11	FRC (clock derived from the internal A/D RC oscillator)

Format of A/D Result

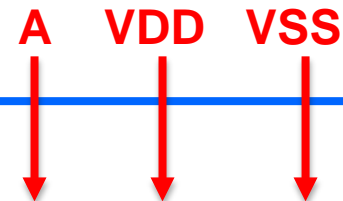
FIGURE 17-4: A/D RESULT JUSTIFICATION





ADCON1

We Want



bit 3-0 PCFG3:PCFG0: A/D Port Configuration Control bits

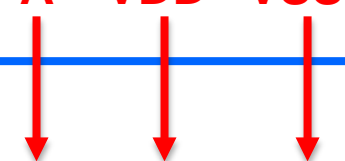
PCFG <3:0>	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0	VREF+	VREF-	C / R
0000	A	A	A	A	A	A	A	A	VDD	VSS	8 / 0
0001	A	A	A	A	VREF+	A	A	A	AN3	VSS	7 / 1
0010	D	D	D	A	A	A	A	A	VDD	VSS	5 / 0
0011	D	D	D	A	VREF+	A	A	A	AN3	VSS	4 / 1
0100	D	D	D	D	A	D	A	A	VDD	VSS	3 / 0
0101	D	D	D	D	VREF+	D	A	A	AN3	VSS	2 / 1
011x	D	D	D	D	D	D	D	D	—	—	0 / 0
1000	A	A	A	A	VREF+	VREF-	A	A	AN3	AN2	6 / 2
1001	D	D	A	A	A	A	A	A	VDD	VSS	6 / 0
1010	D	D	A	A	VREF+	A	A	A	AN3	VSS	5 / 1
1011	D	D	A	A	VREF+	VREF-	A	A	AN3	AN2	4 / 2
1100	D	D	D	A	VREF+	VREF-	A	A	AN3	AN2	3 / 2
1101	D	D	D	D	VREF+	VREF-	A	A	AN3	AN2	2 / 2
1110	D	D	D	D	D	D	D	A	VDD	VSS	1 / 0
1111	D	D	D	D	VREF+	VREF-	D	A	AN3	AN2	1 / 2

A = Analog input D = Digital I/O

C/R = # of analog input channels / # of A/D voltage references

ADCON1

We Want
A **VDD** **VSS**



bit 3-0 PCFG3:PCFG0: A/D Port Configuration Control bits

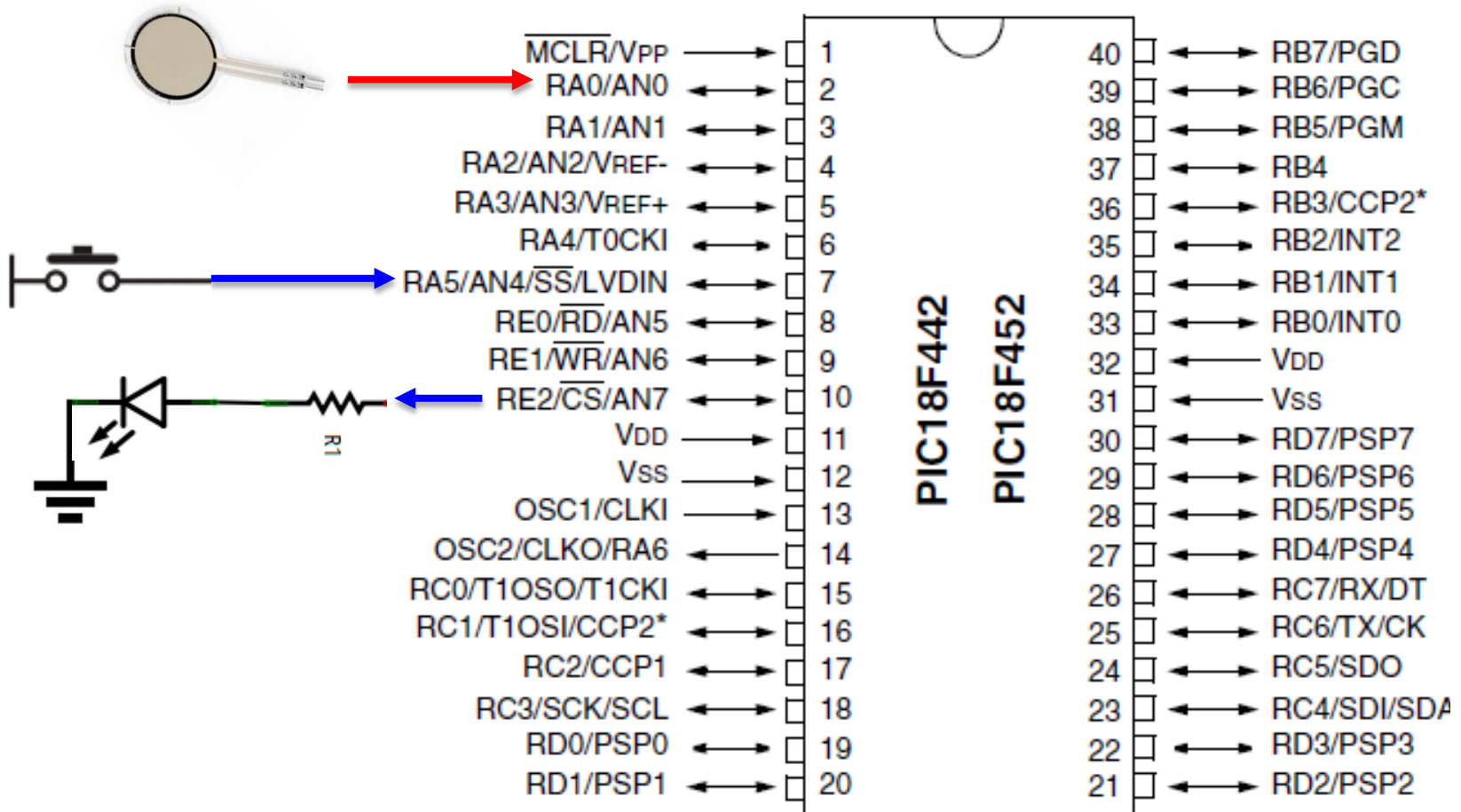
PCFG <3:0>	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0	VREF+	VREF-	C / R
0000	A	A	A	A	A	A	A	A	VDD	VSS	8 / 0
0001	A	A	A	A	VREF+	A	A	A	AN3	VSS	7 / 1
0010	D	D	D	A	A	A	A	A	VDD	VSS	5 / 0
0011	D	D	D	A	VREF+	A	A	A	AN3	VSS	4 / 1
0100	D	D	D	D	A	D	A	A	VDD	VSS	3 / 0
0101	D	D	D	D	VREF+	D	A	A	AN3	VSS	2 / 1
011x	D	D	D	D	D	D	D	D	—	—	0 / 0
1000	A	A	A	A	VREF+	VREF-	A	A	AN3	AN2	6 / 2
1001	D	D	A	A	A	A	A	A	VDD	VSS	6 / 0
1010	D	D	A	A	VREF+	A	A	A	AN3	VSS	5 / 1
1011	D	D	A	A	VREF+	VREF-	A	A	AN3	AN2	4 / 2
1100	D	D	D	A	VREF+	VREF-	A	A	AN3	AN2	3 / 2
1101	D	D	D	D	VREF+	VREF-	A	A	AN3	AN2	2 / 2
1110	D	D	D	D	D	D	D	A	VDD	VSS	1 / 0
1111	D	D	D	D	VREF+	VREF-	D	A	AN3	AN2	1 / 2

A = Analog input D = Digital I/O

C/R = # of analog input channels / # of A/D voltage references



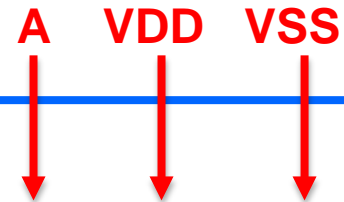
Which AN Channels should be **Analog (input)** or **Digital (I/O)**?



ADCON1

We Want

A VDD VSS



bit 3-0 PCFG3:PCFG0: A/D Port Configuration Control bits

PCFG <3:0>	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0	VREF+	VREF-	C / R
0000	A	A	A	A	A	A	A	A	VDD	VSS	8 / 0
0010	D	D	D	A	A	A	A	A	VDD	VSS	5 / 0
0100	D	D	D	D	A	D	A	A	VDD	VSS	3 / 0
1001	D	D	A	A	A	A	A	A	VDD	VSS	6 / 0
1110	D	D	D	D	D	D	D	A	VDD	VSS	1 / 0

A = Analog input D = Digital I/O

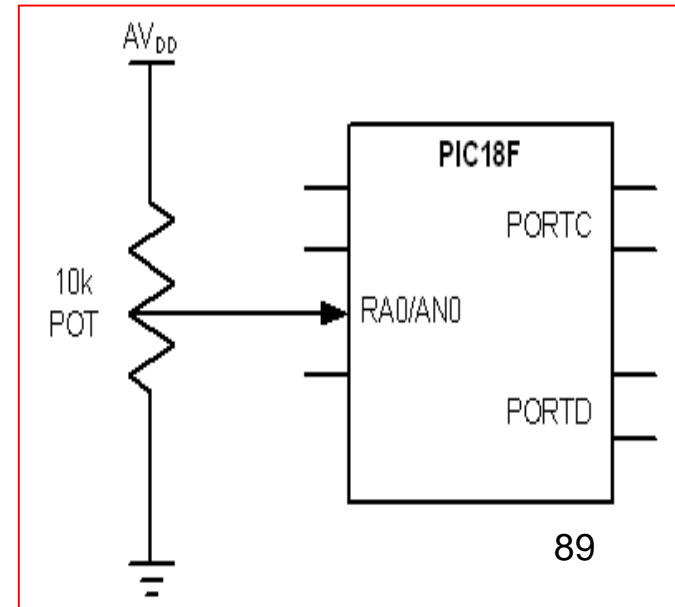
C/R = # of analog input channels / # of A/D voltage references



Programming the PIC18 ADC

```
unsigned char adLow = 0;
unsigned char adHigh = 0;
int fullAD = 0;
TRISAbits.TRISA0 = 1; //A0 is an input
ADCON0 = 0b01000001; //FOSC/16, channel 0, A/D module turned ON
ADCON1 = 0b11001110; //right justified (FOSC/16) AN0=analog
```

```
while(1)
{
    //Delay >= TACQ //Datasheet pg. 185
    ADCON0bits.GO = 1;
    while(ADCON0bits.DONE == 1) ;
    adLow = ADRESL;
    adHigh = ADRESH;
    //combine into a single 10-bit number
    DELAY(250); // set your sampling rate (in ms)
}
```





ADC

Questions?