Panel 1

Prior to Le16

Analog-to-Digital Conversions

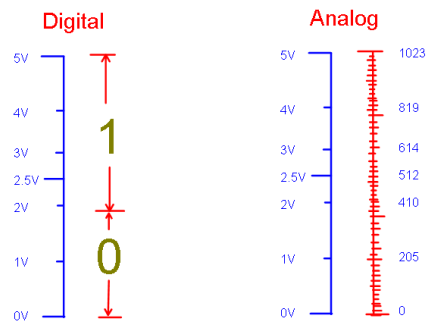
ME430 Mechatronics

1

Panel 2

Analog-to-Digital Conversions (ADC)

The ADC hardware peripheral allows you to read an analog voltage

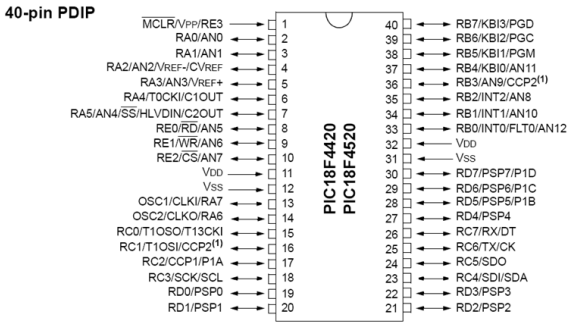


Digital pins can be set as inputs or output using TRIS. Analog pins are ONLY inputs

2

Panel 3

There are 13 pins that can be used for Analog inputs
but really only one can be used at a time



3

Panel 4

Syntax is completely different

Digital	Analog

4

Panel 5

2.2 A/D CONVERTER FUNCTIONS

The A/D peripheral is supported with the following functions:

TABLE 2-1: A/D CONVERTER FUNCTIONS

Function	Description
BusyADC	Is A/D converter currently performing a conversion?
CloseADC	Disable the A/D converter.
ConvertADC	Start an A/D conversion.
OpenADC	Configure the A/D convertor.
ReadADC	Read the results of an A/D conversion.
SetChanADC	Select A/D channel to be used.

Excellent, easy to use functions

5

Panel 6

OpenADC
All Other Processors

Function: Configure the A/D convertor.

Include: adc.h

Prototype:
```
void OpenADC(unsigned char config,
             unsigned char config2 ,
             unsigned char portconfig);
```

Arguments:

config
A bitmask that is created by performing a bitwise AND operation ('&') with a value from each of the categories listed below. These values are defined in the file adc.h.

A/D clock source:

ADC_FOSC_2	Fosc / 2
ADC_FOSC_4	Fosc / 4
ADC_FOSC_8	Fosc / 8
ADC_FOSC_16	Fosc / 16
ADC_FOSC_32	Fosc / 32
ADC_FOSC_64	Fosc / 64
ADC_FOSC_RC	Internal RC Oscillator

A/D result justification:

ADC_RIGHT_JUST	Result in Least Significant bits
ADC_LEFT_JUST	Result in Most Signifcant bits

A/D acquisition time select:

ADC_0_TAD	0 Tad
ADC_2_TAD	2 Tad
ADC_4_TAD	4 Tad
ADC_6_TAD	6 Tad
ADC_8_TAD	8 Tad
ADC_12_TAD	12 Tad
ADC_16_TAD	16 Tad
ADC_20_TAD	20 Tad

config2
A bitmask that is created by performing a bitwise AND operation ('&') with a value from each of the categories listed below. These values are defined in the file adc.h.

Channel:

ADC_CH0	Channel 0
ADC_CH1	Channel 1
ADC_CH2	Channel 2
ADC_CH3	Channel 3
ADC_CH4	Channel 4
ADC_CH5	Channel 5
ADC_CH6	Channel 6
ADC_CH7	Channel 7
ADC_CH8	Channel 8
ADC_CH9	Channel 9
ADC_CH10	Channel 10
ADC_CH11	Channel 11
ADC_CH12	Channel 12
ADC_CH13	Channel 13
ADC_CH14	Channel 14
ADC_CH15	Channel 15

6

Panel 7

Looks complex, but it isn't...

```
// configure A/D convertor
// config 1 = Setup the timing to a conservative value (you don't need to ever change this)
// config 2 = Use channel 0, not interrupts off, use the power and ground as referrences
// portconfig = 0x0E setup only analog 0 as a possible analog input pin

OpenADC(ADC_FOSC_8 & ADC_RIGHT_JUST & ADC_12_TAD,
        ADC_CH0 & ADC_INT_OFF & ADC_REF_VDD_VSS,
        0x0E);
```

7

Panel 8

REGISTER 19-2: ADCON1 REGISTER

U-0	U-0	R/W-0	R/W-0	R/W-0(1)	R/W(1)	R/W(1)	R/W(1)
—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 7-6 **Unimplemented:** Read as '0'

bit 5 **VCFG1:** Voltage Reference Configuration bit (VREF- source)
1 = VREF- (AN2)
0 = VSS

bit 4 **VCFG0:** Voltage Reference Configuration bit (VREF+ source)
1 = VREF+ (AN3)
0 = VDD

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

PCFG3: PCFG0	AN12	AN11	AN10	AN9	AN8	AN7(2)	AN6(2)	AN5(2)	AN4	AN3	AN2	AN1	AN0
0000(1)	A	A	A	A	A	A	A	A	A	A	A	A	A
0001	A	A	A	A	A	A	A	A	A	A	A	A	A
0010	A	A	A	A	A	A	A	A	A	A	A	A	A
0011	D	A	A	A	A	A	A	A	A	A	A	A	A
0100	D	D	A	A	A	A	A	A	A	A	A	A	A
0101	D	D	D	A	A	A	A	A	A	A	A	A	A
0110	D	D	D	D	A	A	A	A	A	A	A	A	A
0111(1)	D	D	D	D	D	A	A	A	A	A	A	A	A
1000	D	D	D	D	D	D	A	A	A	A	A	A	A
1001	D	D	D	D	D	D	D	A	A	A	A	A	A
1010	D	D	D	D	D	D	D	D	A	A	A	A	A
1011	D	D	D	D	D	D	D	D	D	A	A	A	A
1100	D	D	D	D	D	D	D	D	D	D	A	A	A
1101	D	D	D	D	D	D	D	D	D	D	D	A	A
1110	D	D	D	D	D	D	D	D	D	D	D	D	A
1111	D	D	D	D	D	D	D	D	D	D	D	D	D

A = Analog input D = Digital I/O

8

Panel 9

ADC is a four step program

2.2 A/D CONVERTER FUNCTIONS

The A/D peripheral is supported with the following functions:

TABLE 2-1: A/D CONVERTER FUNCTIONS

Function	Description
BusyADC	Is A/D converter currently performing a conversion?
CloseADC	Disable the A/D converter.
ConvertADC	Start an A/D conversion.
OpenADC	Configure the A/D convertor.
ReadADC	Read the results of an A/D conversion.
SetChanADC	Select A/D channel to be used.

9

Panel 10

Four step program:

1. Select the Analog channel (the pin you want to read)
2. Start the ADC conversion
3. Wait for it to finish
4. Read the ADC result (and store it to a variable)

```
SetChanADC( ADC_CH0 );        // Select the pin
ConvertADC();                 // Start conversion
while( BusyADC() );           // Wait for completion
RA0result = ReadADC();        // Read result
```

10

Panel 11

Circle what will change and explain...

```
// configure A/D convertor
// config 1 = Setup the timing to a conservative value (you don't need to ever change this)
// config 2 = Use channel 0, not interrupts off, use the power and ground as referrences
// portconfig = 0x0E setup only analog 0 as a possible analog input pin

OpenADC(ADC_FOSC_8 & ADC_RIGHT_JUST & ADC_12_TAD,
        ADC_CH0 & ADC_INT_OFF & ADC_REF_VDD_VSS,
        0x0E);
```

```
SetChanADC( ADC_CH0 );        // Select the pin
ConvertADC();                 // Start conversion
while( BusyADC() );           // Wait for completion
RA0result = ReadADC();        // Read result
```

11

Panel 12

What are we gonna do in class?

Much like the Timers example code is on-line

Steps:

Watch the program run

Modify the code to monitor RB0 as an analog input
(even though it's a button you can read it analog)

You could use interrupts, but I rarely do so...
We're gonna skip interrupts with the ADC

12