Data Acquisition and Manipulation

Chapter 11
Sections 1 - 3

Dr. Iyad Jafar
Outline

- Analog and Digital Quantities
- The Analog to Digital Converter
- Features of Analog to Digital Converter
- The Data Acquisition System
- The 16F873 ADC
- Summary
Analog and Digital Quantities

- Most signals that are produced by transducers are analog; continuously variable in time and can take infinite range of values.
- Digital signals are *discrete representation* for the analog signals *in time and value*.
- Digital signals perform better and are easier to work with.
- Analog signals have to be converted into digital form in order to be processed by the microcontroller.
- The device that performs this conversion is called Analog to Digital Converter (ADC).
## Analog and Digital Quantities

<table>
<thead>
<tr>
<th>Property</th>
<th>Analog</th>
<th>Digital</th>
</tr>
</thead>
<tbody>
<tr>
<td>Representation</td>
<td>Continuous voltage or current</td>
<td>Binary Number</td>
</tr>
<tr>
<td>Precision</td>
<td>Infinite range of values</td>
<td>Only fixed number of digits combination are available</td>
</tr>
<tr>
<td>Resistance to</td>
<td>Suffers from drift, attenuation, distortion, interference. Recovery is hard</td>
<td>Tolerant to most forms of signal degradation. Error checking can be included for complete recovery</td>
</tr>
<tr>
<td>Degradation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Processing</td>
<td>Processing using op amps and other sophisticated circuits. Limited, complex, and suffers from distortion</td>
<td>Powerful computer-based techniques</td>
</tr>
<tr>
<td>Storage</td>
<td>Analog storage for any length of time is almost impossible</td>
<td>All semiconductor memory techniques are digital</td>
</tr>
</tbody>
</table>
The Analog to Digital Converter

- Conversion to digital form requires two steps
  - Sampling
  - Quantization
Features of Analog to Digital Converter

- **Conversion Characteristics**
  - The ADC accepts a voltage that is infinitely variable and converts it to one of a fixed number of output values.
Features of Analog to Digital Converter

- Conversion Characteristics

Quantization Error

The Magnitude of the Error Ranges from Zero to 1 LSB
Features of Analog to Digital Converter

- **Reference voltages** \([V_{\text{min}}, V_{\text{max}}]\)
  - Determine the acceptable range of input analog voltage
  - Out of range input values are clipped
  - *Unipolar or bipolar*
  - Should be stable and accurate for proper operation
  - **Input range** \(V_r = V_{\text{max}} - V_{\text{min}}\)

- **Resolution**
  - The amount by which the input voltage has to change to go from one output value to another
  - The more the output bits the more the output steps and finer is the conversion
  - **Resolution** = \(V_r / 2^n\)

- **Quantization error** \(Q = \text{resolution} / 2\)
Features of Analog to Digital Converter

- Conversion Characteristic

<table>
<thead>
<tr>
<th>$n$</th>
<th>No. of quantisation levels</th>
<th>Max. quantisation error as % of range</th>
<th>Quantisation error for range of 5 V</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>8</td>
<td>6.25</td>
<td>312.50 mV</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>3.13</td>
<td>156.25 mV</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>1.56</td>
<td>78.13 mV</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>0.781</td>
<td>39.06 mV</td>
</tr>
<tr>
<td>8</td>
<td>256</td>
<td>0.195</td>
<td>9.77 mV</td>
</tr>
<tr>
<td>10</td>
<td>1 024</td>
<td>0.0488</td>
<td>2.44 mV</td>
</tr>
<tr>
<td>12</td>
<td>4 096</td>
<td>0.0122</td>
<td>0.61 mV</td>
</tr>
<tr>
<td>16</td>
<td>65 536</td>
<td>0.00076</td>
<td>38.1 μV</td>
</tr>
</tbody>
</table>
Features of Analog to Digital Converter

- **Conversion Speed**
  - Time for the ADC to do the conversion
  - Slow ADCs are used with low frequency signals
  - High accuracy ADCs take longer to complete conversion

- **Digital Interface**
  - Made up of control signals and data outputs
  - Data outputs – serial or parallel
The Analog to Digital Converter

• ADC Types

  • Dual Ramp ADC
    • Slow but with high accuracy

  • Flash Converter ADC
    • Fast but less accuracy
    • Used with high speed signals such as video and radar

  • Successive Approximation ADC
    • Medium speed and accuracy
    • Used in general-purpose industrial applications
    • Commonly found in embedded systems
The Data Acquisition System

Elements of data acquisition system

- **Transducer**
  - Generates signal
  - Amplify and offset
  - Amplifies signal and adds DC offset to match ADC input range

- **Filter**
  - Removes unwanted signal components, usually for anti-aliasing purposes

- **Multiplexer**
  - Selects which input channel is connected to its output

- **Sample and hold**
  - Samples its input signal and holds that voltage as a steady value at its output

- **ADC**
  - Converts its analog input to a digital output

- **Voltage reference**

- **CPU control**

- **Digital output**
The Data Acquisition System

Elements of data acquisition system

- **Amplification**
  - Most sensors produce low voltages
  - Need to amplify to exploit the input range of the ADC
  - Voltage level shifting might be needed for bipolar signals

- **Filtering**
  - Pick the actual signal and restrict its frequency content to the sampling rate of the ADC to avoid aliasing
  - Remove unwanted signals

- **Analog multiplexer**
  - Used when working with multiple inputs instead of using multiple ADCs
  - Semiconductor switches
The Data Acquisition System

Elements of data acquisition system

- **Sample and Hold**
  - ADCs are unable to convert accurately a changing signal
  - We need to capture the sample value and hold it for the duration of the conversion process
  - Acquisition time!
The Data Acquisition System

Elements of data acquisition system

- Sample and Hold

Acquisition time increase as we increase the resolution of the ADC
The Data Acquisition System

Typical Timing Requirements for Analog to Digital Conversion

1. Configure and enable ADC
2. Select multiplexer input
3. 'Sample' input signal
4. Delay for signal acquisition
5. 'Hold' input signal
6. Start conversion
7. Delay for conversion to complete
8. Read data

These stages merge if multiplexer forms part of S&H
Data Acquisition in Microcontroller Environment

- Embedded systems need ADCs; usually they are integrated within the MC as 8 or 10 bit ADCs

- Integration is not easy!
  - Proper operation of ADCs demands clean power supply and ground and freedom of interference
  - This is not easily available in digital devices

- Compromise accuracy of integrated ADCs!
# The PIC 16F87x'A ADC Module

<table>
<thead>
<tr>
<th>Device</th>
<th>Pins</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>16F873A</td>
<td>28</td>
<td>3 parallel ports,</td>
</tr>
<tr>
<td>16F876A</td>
<td></td>
<td>3 counter/timers,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 capture/compare/PWM,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 serial,</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>5 10-bit ADC,</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 comparators</td>
</tr>
<tr>
<td>16F874A</td>
<td>40</td>
<td>5 parallel ports,</td>
</tr>
<tr>
<td>16F877A</td>
<td></td>
<td>3 counter/timers,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 capture/compare/PWM,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 serial,</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>8 10-bit ADC,</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 comparators</td>
</tr>
</tbody>
</table>
The PIC 16F87xA ADC Module
The PIC 16F87xA ADC Module

Related Registers

• Operation is controlled by two SFRs
  • ADCON0  0x1F
  • ADCON1  0x9F

• Conversion result (10-bit) is placed in two SFRs
  • ADRESL  0x9E
  • ADRESH  0x1E

• ADC interrupt enable and flag are available in
  • PIE1    0x8C
  • PIR1    0x0C

• Related registers
  • TRISA   0x85
  • TRISE   0x89  (in 40-pin devices)
The PIC 16F87xA ADC Module

Controlling the ADC

(1) **Switching on**
- The ADC is switched on/off by setting/clearing ADON bit (ADCON0<0>)
- It is preferred to turn the ADC off when it is not needed as it offers some power saving

(2) **Setting Conversion Speed**
- Operation of the ADC is governed by a clock with period $T_{AD}$
- For correct conversions, $T_{AD}$ must be 1.6 us at least
- The ADC clock can be selected by software ($2T_{OSC}$, $4T_{OSC}$, $8T_{OSC}$, $16T_{OSC}$, $32T_{OSC}$, $64T_{OSC}$, or internal RC 2-4 us)
- Selection of ADC clock source is through ADCS2 (ADCON1<6>), ADCS1:ADCS0 (ADCON0<7:6>)
- If the system clock is fast (>500KHz), use it to derive the ADC clock. Otherwise, use the internal RC.
The PIC 16F87xA ADC Module

Controlling the ADC

**Setting Conversion Speed**

- A full 10-bit conversion requires $12 \ T_{AD}$
The PIC 16F87x A ADC Module

Controlling the ADC

(3) Configuring Inputs and Voltage Reference

- The ADCON1 and TRIS registers control the operation of the A/D port pins
- Inputs AN7 to AN0 can be configured as analog inputs or digital inputs.
- AN3 (RA3) and AN2 (RA2) can be used as the inputs for the external reference voltages separately
- Configuration is made through PCFG3:PCFG0 (ADCON1<3:0>)

(4) Channel Selection

- We can select one out of five (or eight channels) as the analog input using the bits CHS2:CHS0 (ADCON0<5:3>)
- Selection of the input channel closes the sampling switch.
**The PIC 16F87xA ADC Module**

**Controlling the ADC**

**(5) Starting Conversion and Flagging its End**

- Conversion can be started by setting the GO/DONE’ (ADCON0<2>) bit. **This opens the sampling switch.**

- Once the conversion is complete, this bit is cleared to indicate the end of conversion.

- **The GO/DONE’ bit should not be set using the same instruction that turns on the A/D.**
The PIC 16F87xA ADC Module

Controlling the ADC

(6) **Formatting the result**

- The ADC result is 10-bit data that is placed in ADRESH and ADCRESL (0x 1E and 0x9E respectively)
- The result can be left justified or right justified
- Selection of desired format is through the ADFM (ADCON1<7>) bit
The PIC 16F87xA ADC Module

ADCON0 Register 0x1F

<table>
<thead>
<tr>
<th>R/W-0</th>
<th>R/W-0</th>
<th>R/W-0</th>
<th>R/W-0</th>
<th>R/W-0</th>
<th>R/W-0</th>
<th>U-0</th>
<th>R/W-0</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADCS1</td>
<td>ADCS0</td>
<td>CHS2</td>
<td>CHS1</td>
<td>CHS0</td>
<td>GO/DONE</td>
<td>—</td>
<td>ADON</td>
</tr>
</tbody>
</table>

- **bit 7**
  - **ADCS1:ADCSC0**: A/D Conversion Clock Select bits (ADCON0 bits in **bold**)

<table>
<thead>
<tr>
<th>ADCON1 &lt;ADCSC2&gt;</th>
<th>ADCON0 <a href="">ADCS1:ADCSC0</a></th>
<th>Clock Conversion</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>00</td>
<td>Fosc/2</td>
</tr>
<tr>
<td>0</td>
<td>01</td>
<td>Fosc/8</td>
</tr>
<tr>
<td>0</td>
<td>10</td>
<td>Fosc/32</td>
</tr>
<tr>
<td>0</td>
<td>11</td>
<td>FRC (clock derived from the internal A/D RC oscillator)</td>
</tr>
<tr>
<td>1</td>
<td>00</td>
<td>Fosc/4</td>
</tr>
<tr>
<td>1</td>
<td>01</td>
<td>Fosc/16</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
<td>Fosc/64</td>
</tr>
<tr>
<td>1</td>
<td>11</td>
<td>FRC (clock derived from the internal A/D RC oscillator)</td>
</tr>
</tbody>
</table>

- **bit 6-5**
  - **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)

- **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
# The PIC 16F87xA ADC Module

## ADCON1 Register 0x9F

<table>
<thead>
<tr>
<th>R/W-0</th>
<th>R/W-0</th>
<th>U-0</th>
<th>U-0</th>
<th>R/W-0</th>
<th>R/W-0</th>
<th>R/W-0</th>
<th>R/W-0</th>
<th>bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ADFM</td>
<td>ADCS2</td>
<td>—</td>
<td>—</td>
<td></td>
</tr>
</tbody>
</table>

### 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 shaded area and in **bold**)

### bit 5-4 Unimplemented: Read as ‘0’

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

<table>
<thead>
<tr>
<th>PCFGx&lt;3:0&gt;</th>
<th>AN7</th>
<th>AN6</th>
<th>AN5</th>
<th>AN4</th>
<th>AN3</th>
<th>AN2</th>
<th>AN1</th>
<th>AN0</th>
<th>VREF+</th>
<th>VREF-</th>
<th>C/R</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>D</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>VDD</td>
<td>VSS</td>
<td>8/0</td>
</tr>
<tr>
<td>0001</td>
<td>D</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>VREF+</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>VDD</td>
<td>5/0</td>
</tr>
<tr>
<td>0010</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>VDD</td>
<td>VSS</td>
<td>7/1</td>
</tr>
<tr>
<td>0011</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>A</td>
<td>VREF+</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>VDD</td>
<td>VSS</td>
<td>3/0</td>
</tr>
<tr>
<td>0100</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>A</td>
<td>A</td>
<td>D</td>
<td>A</td>
<td>A</td>
<td>VDD</td>
<td>VSS</td>
<td>4/1</td>
</tr>
<tr>
<td>0101</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>A</td>
<td>VREF+</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>AN3</td>
<td>VSS</td>
<td>2/1</td>
</tr>
<tr>
<td>011x</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>VREF+</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>AN3</td>
<td>AN2</td>
<td>0/0</td>
</tr>
<tr>
<td>1000</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>VREF+</td>
<td>VREF-</td>
<td>A</td>
<td>A</td>
<td>AN3</td>
<td>AN2</td>
<td>6/2</td>
</tr>
<tr>
<td>1001</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>A</td>
<td>A</td>
<td>VDD</td>
<td>A</td>
<td>A</td>
<td>VSS</td>
<td>AN2</td>
<td>6/0</td>
</tr>
<tr>
<td>1010</td>
<td>D</td>
<td>D</td>
<td>A</td>
<td>A</td>
<td>VREF+</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>AN3</td>
<td>VSS</td>
<td>5/1</td>
</tr>
<tr>
<td>1011</td>
<td>D</td>
<td>D</td>
<td>A</td>
<td>A</td>
<td>VREF+</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>AN3</td>
<td>AN2</td>
<td>4/2</td>
</tr>
<tr>
<td>1100</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>A</td>
<td>VREF+</td>
<td>VREF-</td>
<td>A</td>
<td>A</td>
<td>AN3</td>
<td>AN2</td>
<td>3/2</td>
</tr>
<tr>
<td>1101</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>A</td>
<td>VREF+</td>
<td>VREF-</td>
<td>A</td>
<td>A</td>
<td>AN3</td>
<td>AN2</td>
<td>2/2</td>
</tr>
<tr>
<td>1110</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>VREF+</td>
<td>VREF-</td>
<td>A</td>
<td>A</td>
<td>VDD</td>
<td>VSS</td>
<td>1/0</td>
</tr>
<tr>
<td>1111</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>VREF+</td>
<td>VREF-</td>
<td>A</td>
<td>A</td>
<td>AN3</td>
<td>AN2</td>
<td>1/2</td>
</tr>
</tbody>
</table>

* A = Analog input  D = Digital I/O  
* C/R = # of analog input channels/# of A/D voltage references
The PIC 16F87xA ADC Module

Related Registers

<table>
<thead>
<tr>
<th>Address</th>
<th>Name</th>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
<th>Value on POR, BOR</th>
<th>Value on MCLR, WDT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0Bh, 0Bh, 0Ch</td>
<td>INTCON</td>
<td>GIE</td>
<td>PEIE</td>
<td>TMR0IE</td>
<td>INTE</td>
<td>RBIE</td>
<td>TMR0IF</td>
<td>INTF</td>
<td>RBIF</td>
<td>0000 0000x</td>
<td>0000 0000u</td>
</tr>
<tr>
<td>0Ch</td>
<td>PIR1</td>
<td>PSPIF</td>
<td>ADIF</td>
<td>RCIF</td>
<td>TXIF</td>
<td>SSPIF</td>
<td>CCP1IF</td>
<td>TMR2IF</td>
<td>TMR1IF</td>
<td>0000 0000</td>
<td>0000 00000</td>
</tr>
<tr>
<td>8Ch</td>
<td>PIE1</td>
<td>PSPIE</td>
<td>ADIE</td>
<td>RCIE</td>
<td>TXIE</td>
<td>SSPIE</td>
<td>CCP1IE</td>
<td>TMR2IE</td>
<td>TMR1IE</td>
<td>0000 0000</td>
<td>0000 00000</td>
</tr>
<tr>
<td>1Eh</td>
<td>ADRESH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>xxxxx xxxxx</td>
<td>uuuuu uuuuu</td>
</tr>
<tr>
<td>9Eh</td>
<td>ADRESL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>xxxxx xxxxx</td>
<td>uuuuu uuuuu</td>
</tr>
<tr>
<td>1Fh</td>
<td>ADCON0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>xxxxx xxxxx</td>
<td>uuuuu uuuuu</td>
</tr>
<tr>
<td>9Fh</td>
<td>ADCON1</td>
<td>ADFM</td>
<td>ADCS2</td>
<td></td>
<td></td>
<td>PCFG3</td>
<td>PCFG2</td>
<td>PCFG1</td>
<td>PCFG0</td>
<td>00-- 0000</td>
<td>00-- 00000</td>
</tr>
<tr>
<td>85h</td>
<td>TRISA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>PORTA</td>
<td>Data</td>
<td>Direction</td>
<td>Register</td>
<td>--11 1111</td>
<td>--11 1111</td>
</tr>
<tr>
<td>05h</td>
<td>PORTA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>PORTA</td>
<td>Data</td>
<td>Latch when</td>
<td>written: PORTA</td>
<td>pins when read</td>
<td>--0x 0000</td>
</tr>
<tr>
<td>89h(1)</td>
<td>TRISE</td>
<td>IBF</td>
<td>OBF</td>
<td>IBOV</td>
<td>PSPMODE</td>
<td></td>
<td>PORTE</td>
<td>Data Direction</td>
<td>bits</td>
<td>0000 -111</td>
<td>0000 -111</td>
</tr>
<tr>
<td>09h(1)</td>
<td>PORTE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RE2</td>
<td>RE1</td>
<td>RE0</td>
<td></td>
<td>----- -xxx</td>
<td>----- -uuu</td>
</tr>
</tbody>
</table>

(1) Indicates a register that is affected by the configuration of the device.
The PIC 16F87xA ADC Module

- **Steps for using the A/D module**
  1. Configure the A/D module
     a. Select analog pins/voltage reference and digital I/O (ADCON1)
     b. Select the A/D channel (ADCON0)
     c. Select the conversion clock (ADCON0)
     d. Turn the A/D module on (ADCON0)
  2. Configure interrupts (if desired)
     1. Clear ADIF (PIR1<6>) and set ADIE (PIE1<6>)
     2. Set PEIE (INTCON<6>) then set GIE (INTCON<7>)
  3. Wait the required acquisition time
  4. Start conversion by setting the GO/DONE’ bit
  5. Wait for conversion complete
  6. Read the A/D result register pair ADRESH:ADRESL
The PIC 16F87xA ADC Module

- The analog input model
The PIC 16F87xA ADC Module

- **Calculating conversion speed (Qerror is ½ LSB)**

  \[ A/D \text{ Total Time} = \text{Acquisition Time} + A/D \text{ Conversion time} \]

  \[ = T_{ACQ} + 12 \cdot T_{AD} \]

  \[ T_{ACQ} = \text{Amplifier settling time} \]

  + Hold capacitor charging time

  + Temperature coefficient

  \[ T_{ACQ} = T_{AMP} + T_{HOLD} + T_{COFF} \]

  \[ T_{HOLD} = -(R_{IC}+R_{SS}+R_S) \cdot C_{HOLD} \cdot \ln(1/2^{(n+1)}) \]

  \[ = -(R_{IC}+R_{SS}+R_S) \cdot 120 \text{ pF} \cdot \ln(1/2048) \]

  \[ = 7.6 \cdot R \cdot C \text{ us} \]

  \[ A/D \text{ Total Time} = 2 \mu s + 7.6RC + (\text{Temperature} - 25^\circ C)(0.05 \mu s/\circ C) + 12 T_{AD} \]
The PIC 16F87x A ADC Module

• Calculating conversion speed example

\[
R_{SS} = 7k\Omega \ (V_{DD} = 5V), \ R_{IC} = 1k\Omega, \ R_S = 0, \\
\text{Temp} = 35 \degree C, \ T_{AD} = 1.6 \mu s
\]
\[
t_{ac} = 2 \mu s \\
+ 7.6(7k\Omega + 1k\Omega + 0)(120pF) \\
+ (35 - 25)(0.05 \mu s/\degree C)
= 2 + 7.3 + 0.5 = 9.8 \mu s
\]

Total time = \( t_{ac} + 12T_{AD} = 9.8 + 19.2 \mu s = 29 \mu s \)

Maximum sampling rate \( \sim = 34.5 \) KHz
The PIC 16F87xA ADC Module

- **Repeated Conversions**
  - When a conversion is complete, the converter waits a period of 2*TAD before it is available to start a new conversion
  - This time has to be added to the conversion time!

- **Trading off conversion speed and resolution**
  - If resolution is not an issue, then we can start the conversion with correct clock then we switch it to higher clock
  - Consider only bits produced before switching the clock
Example: use the ADC in PIC 16F877A to obtain one sample of an analog signal that is connected RA0. Assume the ADC clock to be Fosc/8 and reference voltage to be internal. The PIC is operating with Fosc = 4 MHz, VDD = 5 v, and temperature 25 C. The result should be right justified.

Setup:
1) set RA0 as analog input
2) select the clock
3) generate appropriate delays \( T_{\text{acq}} = 2 + 7.6 \times (1K + 7K) \times 120 \text{ pF} = 9.3 \text{ us} \approx 10 \text{ us} \)
Example

```assembly
#include p16F877A.inc ; include the definition file for 16F77A
org  0x0000 ; reset vector
goto START

ISR
org  0x0004 ; define the ISR
goto ISR

START
org  0x0006 ; Program starts here

bsf STATUS, RP0 ; select bank 1
movlw B'00000001' ; set RA0 as input
movwf TRISA
movlw B'10001110' ; select RA0 as analog input, result right justified, and internal reference voltage
movwf ADCON1
bcf STATUS, RP0 ; select bank 0
movlw B'10001110' ; select RA0 as analog input, result right justified, and internal reference voltage
movwf ADCON1
movlw B'00000001' ; turn on ADC, clock Fosc/8, select channel 0
movwf ADCON0
```
Example

; start the conversion
call    delay10us ; acquisition time delay
bsf    ADCON0, GO ; start conversion
btfsc   ADCON0, GO_DONE ; wait for conversion to complete
goto    $-1

done    goto    done

delay10us
movlw   D'2'
movwf   0x20 ; counter for delay loop
more
nop
decfsz  0x20,1
goto    more
return

end
Summary

- Most signals produced by transducers are analog in nature, while all processing done by a microcontroller is digital.

- Analog signals can be converted to digital form using an analog-to-digital converter (ADC).

- The 16F873A has a 10-bit configurable ADC module.

- Data values, once acquired, are likely to need further processing, including offsetting, scaling and code conversion.