Analog to Digital Converter (ADC) and Digital to Analog Converter (DAC)
Analog I/O

• Analog inputs – convert to digital using an Analog to Digital converter (A/D or ADC)
• Analog output – convert digital output to analog using a Digital to Analog converter (D/A or DAC)
• A/D outputs and D/A inputs can be attached to digital I/O ports
• Design issues to consider – number of bits of accuracy, conversion time delay, and sample rate needed by application
The MCP3008 10-bit 200Ksps A/D chip used in Phidget modules has an SPI interface.
Off-the-shelf ADC

• Resolution – smallest distinguishable change in input
• Precision – number of distinguishable inputs
• Accuracy – the absolute error of the entire system
• Monotonic – no missing codes
• Linear – constant resolution
• Speed – time to convert
Analog-digital interface
Processing analog signal
Analog input signal
Analog input signal

- For periodic waveforms, the duration of the waveform before it repeats is called the period of the waveform.
- The rate at which a regular vibration pattern repeats itself (frequency = 1/period).
Frequency of a Waveform

• The unit for frequency is cycles/second, also called Hertz (Hz).

• The frequency of a waveform is equal to the reciprocal of the period.

\[
\text{frequency} = \frac{1}{\text{period}}
\]
Frequency of a Waveform

• Examples:
  frequency = 10 Hz
  period = .1 (1/10) seconds

  frequency = 100 Hz
  period = .01 (1/100) seconds

  frequency = 261.6 Hz (middle C)
  period = .0038226 (1/ 261.6) seconds
Waveform Sampling (Quantization)

• To represent waveforms on digital computers, we need to digitize or sample the waveform.

• side effects of digitization:
  – introduces some noise
  – limits the maximum upper frequency range
Sampling rate

• The sampling rate (SR) is the rate at which amplitude values are digitized from the original waveform.
  – CD sampling rate (high-quality):
    SR = 44,100 samples/second
  – medium-quality sampling rate:
    SR = 22,050 samples/second
  – phone sampling rate (low-quality):
    SR = 8,192 samples/second
Sampling rate

- Higher sampling rates allow the waveform to be more accurately represented.
Nyquist Theorem

• **Nyquist Theorem:**
  We can digitally represent only frequencies up to half the sampling rate.

  – **Example:**
    CD: SR=44,100 Hz
    Nyquist Frequency = SR/2 = 22,050 Hz

  – **Example:**
    SR=22,050 Hz
    Nyquist Frequency = SR/2 = 11,025 Hz
Nyquist Theorem

Sampling rate (SR) > 2 \( f_{\text{max}} \)

\( f_{\text{max}} \) is the largest signal frequency of interest
Nyquist Theorem and Aliasing

- **Graphical Example 1a:**
  - SR = 20,000 Hz
  - Nyquist Frequency = 10,000 Hz
  - f = 2,500 Hz (no aliasing)
Nyquist Theorem and Aliasing

• **Graphical Example 1b:**
  - SR = 20,000 Hz
  - Nyquist Frequency = 10,000 Hz
  - f = 5,000 Hz (no aliasing)
Nyquist Theorem and Aliasing

• **Graphical Example 2:**
  - SR = 20,000 Hz
  - Nyquist Frequency = 10,000 Hz
  - f = 10,000 Hz (no aliasing)
Nyquist Theorem and Aliasing

• **Graphical Example 2:**
  - BUT, if sample points fall on zero-crossings the sound is completely cancelled out
Nyquist Theorem and Aliasing

- **Graphical Example 3:**
  - SR = 20,000 Hz
  - Nyquist Frequency = 10,000 Hz
  - f = 12,500 Hz, f' = 7,500
Nyquist Theorem and Aliasing

• **Graphical Example 3:**
  
  – Fitting the simplest sine wave to the sampled points gives an aliased waveform (dotted line below):

![Graph showing an aliased sine wave](image-url)
Sample of sine wave at different freq.

- **a. Analog frequency = 0.0 (i.e., DC)**
- **b. Analog frequency = 0.09 of sampling rate**
- **c. Analog frequency = 0.31 of sampling rate**
- **d. Analog frequency = 0.95 of sampling rate**
Processing analog signal
Low pass filter

- Allow only low frequency value to pass
- Prevent aliasing
Layout of ADC

![Diagram of ADC components: Sampler, Quantizer, Coder.](image)
ADC precision

• Number of ADC bit output (n):
  \[ n = \frac{\text{input range (r)}}{\text{input resolution} \Delta y} \]

E.g., input range of 1, and resolution of 0.0001

  \[ n = 10000 \text{ alternatives or 15 bits value} \]

Assume linear ADC
Sample and hold circuit

Diagram showing the components of a sample and hold circuit:
- Analog signal
- Semiconductor Switch
- Sampled signal
- Capacitor
- Control signal

Graph showing the sampled signal over time.
Sample and hold circuit

- Using op-amp to hold signal strength
- Unity gain buffer
Sample and hold signal
Hold circuit output

b. Sampled analog signal

Amplitude (in volts)

Time

0 5 10 15 20 25 30 35 40 45 50

3.000 3.005 3.010 3.015 3.020 3.025
Quantized output
# Convert analog value to digital value

<table>
<thead>
<tr>
<th>Bipolar codes</th>
<th>Offset binary</th>
<th>2s binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>+5.00</td>
<td>1111</td>
<td>0111</td>
</tr>
<tr>
<td>+2.50</td>
<td>1100</td>
<td>0100</td>
</tr>
<tr>
<td>+0.04</td>
<td>1000</td>
<td>0000</td>
</tr>
<tr>
<td>+0.00</td>
<td>1000</td>
<td>0000</td>
</tr>
<tr>
<td>+0.00</td>
<td>1000</td>
<td>0000</td>
</tr>
<tr>
<td>-2.50</td>
<td>0100</td>
<td>1100</td>
</tr>
<tr>
<td>-5.00</td>
<td>0000</td>
<td>1000</td>
</tr>
</tbody>
</table>
ADC types

- Flash ADC
  - fast
- Successive Approximation ADC
  - most popular
- Sigma Delta ADC
  - highest output precision
Flash ADC

• Use reference voltage and differential op-amp to generate digital output
• Fast conversion
• To increase the number of bits, it requires larger hardware support
# Two-bit flash ADC

<table>
<thead>
<tr>
<th>Vin</th>
<th>X3</th>
<th>X2</th>
<th>X1</th>
<th>Z1</th>
<th>Z0</th>
</tr>
</thead>
<tbody>
<tr>
<td>V&lt;2.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2.5≤V&lt;5.0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>5.0≤V&lt;7.5</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>V≥7.5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

![Diagram of a two-bit flash ADC](attachment:diagram.png)
Flash converter

Fig. 18.12 Flash Converter
256R ladder network
Successive Approximation ADC

• Similar to Flash ADC, but use only one set of hardware
• Require longer time
• Fixed size of hardware
• Widely used
Successive Approximation ADC

Vin

5V

Digital output

N bit ADC

N bit Successive Approximation

Done

Go

Clock
The MCP3008 10-bit 200Ksps A/D chip used in Phidget modules has an SPI interface.
Successive approximation counting
Block diagram of internal AD converter

† Multiplexed with port inputs
AD converter
3-bit AD converter
Delta-Sigma ADC

• Quantization noise is uniformly distributed among frequency spectrum

• Using over-sampling technique to reduce the impact of quantization noise

• Results in higher bit-precision

• Slow
Frequency domain

SNR = 6.02N + 1.76dB for an N-bit ADC
Oversampling by K times
Delta-Sigma circuit

Fig. 1 - Block Diagram Delta Sigma ADC

Typical Waveforms
Signal pin of AD converter
ARM Cortex ADC support

• 12 bit ADC is successive approximation
• 18 multiplexed channel (16 external and 2 internal)
• The result is stored in a left-aligned or right aligned 16 bit register
• Analog watchdog feature allows the application to detect if input voltage goes outside user defined high or low threshold
• ADC input clock is generated from PCLK2
ADC for STM32F4xx

• ADC can be power-on by setting ADON bit in ADC_CR1 control register. It will wake up ADC from power down mode.

• Then, we need to set ADON bit again to start analog conversion.
Single ADC block diagram
# ADC pins

<table>
<thead>
<tr>
<th>Name</th>
<th>Signal type</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V_{REF+}$</td>
<td>Input, analog reference positive</td>
<td>The higher/positive reference voltage for the ADC, $1.8 , V \leq V_{REF+} \leq V_{DDA}$</td>
</tr>
<tr>
<td>$V_{DDA}$</td>
<td>Input, analog supply</td>
<td>Analog power supply equal to $V_{DD}$ and $2.4 , V \leq V_{DDA} \leq V_{DD}$ (3.6 V) for full speed, $1.8 , V \leq V_{DDA} \leq V_{DD}$ (3.6 V) for reduced speed</td>
</tr>
<tr>
<td>$V_{REF-}$</td>
<td>Input, analog reference negative</td>
<td>The lower/negative reference voltage for the ADC, $V_{REF-} = V_{SSA}$</td>
</tr>
<tr>
<td>$V_{SSA}$</td>
<td>Input, analog supply ground</td>
<td>Ground for analog power supply equal to $V_{SS}$</td>
</tr>
<tr>
<td>ADCx_IN[15:0]</td>
<td>Analog input signals</td>
<td>16 analog input channels</td>
</tr>
</tbody>
</table>
Timing diagram

- ADC_CLK
- ADON
- SWSTART/JSWSTART
- ADC
- EOC
- Start 1st conversion
- ADC conversion
- Conversion time (total conv. time)
- Start next conversion
- Next ADC conversion
- tSTAB
- Software clears the EOC bit
Analog watchdog
Temperature sensor and Vref diagram
Example of the code

```c
int main(void){
    // System Clocks Configuration
    RCC_Configuration();

    // NVIC Configuration
    NVIC_Configuration();

    // Configure the GPIO for ADC1
    ADC1GPIOInit();

    // Init ADC1
    ADC1Init();

    // Configure the GPIO for LEDs
    LEDsGPIOInit();

    VoltageMeter();
}
```
void ADC1GPIOInit(void)
{
    GPIO_InitTypeDef GPIO_InitStructure;

    // Enable ADC1 clock
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_ADC1, ENABLE);

    // Config PA1 as analog input
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_1;
    GPIO_InitStructure.GPIO_Speed = (GPIOSpeed_TypeDef)0;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AIN;
    GPIO_Init (GPIOA, &GPIO_InitStructure);
}
void ADC1Init(void) {
    ADC_InitTypeDef ADC_InitStructure;

    /* ADC1 is configured as follow:
       - Single channel, single conversion mode
       - Output data aligned left */
    ADC_InitStructure.ADC_Mode = ADC_Mode_Independent;
    ADC_InitStructure.ADC_ScanConvMode = DISABLE;
    ADC_InitStructure.ADC_ContinuousConvMode = DISABLE;
    ADC_InitStructure.ADC_ExternalTrigConv = ADC_ExternalTrigConv_None;
    ADC_InitStructure.ADC_DataAlign = ADC_DataAlign_Left;
    ADC_InitStructure.ADC_NbrOfChannel = 1;

    // Do it from scratch
    ADC_DeInit(ADC1);
    ADC_Init(ADC1, &ADC_InitStructure);
}
// Enable the ADC
ADC_Cmd(ADC1, ENABLE);

// ADC calibration
ADC_ResetCalibration(ADC1); // Enable ADC1 reset calibration
while(ADC_GetResetCalibrationStatus(ADC1) == SET);
ADC_StartCalibration(ADC1);  // Start ADC calibration
while(ADC_GetCalibrationStatus(ADC1) == SET);
// Configure channel to sampling rate of 55.5 cycle
ADC-RegularChannelConfig(ADC1, 1, 1, ADC_SampleTime_55Cycles5);
void LEDsGPIOInit(void)
{
    GPIO_InitTypeDef GPIO_InitStructure;
    // Initial LED PB[8..15]
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOB, ENABLE);
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out_PP;
    GPIO_InitStructure.GPIO_Pin =
        GPIO_Pin_8 | GPIO_Pin_9 | GPIO_Pin_10 | GPIO_Pin_11 |
        GPIO_Pin_12 | GPIO_Pin_13 | GPIO_Pin_14 | GPIO_Pin_15;
    GPIO_Init(GPIOB, &GPIO_InitStructure);
}
void VoltageMeter(void)
{
    u16 currentReading = 0;
    u8 first8bits = 0;
    while(1){
        currentReading = GetADC1Channel1();
        // Get only higher byte
        first8bits = currentReading >> 8;
        OutputLEDs(first8bits);
    }
}

u16 GetADC1Channel1(void)
{
    // Configure channel
    ADC-RegularChannelConfig(ADC1, 1, 1, ADC_SampleTime_55Cycles5);
    // Start the conversion
    ADC-SoftwareStartConvCmd(ADC1, ENABLE);
    // Wait until conversion completion
    while(ADC_GetFlagStatus(ADC1, ADC_FLAG_EOC) == RESET);
    // Get the conversion value
    return ADC_GetConversionValue(ADC1);
}
Simulink

• ADC block can be used to generate analog signal
Configurations
Configurations

• ADC modules: There are 3 modules available
• Output data type: double/single/raw(int16)
• ADC Prescaler : ADC clock prescaler

The block outputs digital value between 0 to 4095.
Example
Digital to Analog Converter (DAC)
Layout of DAC
Analog of digital conversion

![Graph](image-url)
Analog voltage of zero order

c. Zeroth-order hold
Reconstruction

f. Reconstructed analog signal
Summing Amplifier

Since negative amplifier input is at virtual ground,

\[ i_1 = \frac{v_1}{R_1}, \quad i_2 = \frac{v_2}{R_2}, \quad i_3 = -\frac{v_o}{R_3} \]

Since \( i_- = 0 \), \( i_3 = i_1 + i_2 \),

\[ v_o = -\frac{R_3}{R_1} v_1 - \frac{R_3}{R_2} v_2 \]

- Scale factors for the 2 inputs can be independently adjusted by proper choice of \( R_2 \) and \( R_1 \).
- Any number of inputs can be connected to summing junction through extra resistors.
- This is an example of a simple digital-to-analog converter.
Binary weight register method

\[ V_0 = -R_f \left( S_3 \frac{V}{R} + S_2 \frac{V}{2R} + S_1 \frac{V}{4R} \right) \]

\[ = -\frac{R_f}{4R} V(4S_3 + 2S_2 + S_1) \]

<table>
<thead>
<tr>
<th>S3</th>
<th>S2</th>
<th>S1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ V_0 \]
DA converter
DA pin signal
Connection of DAC
Analog interfacing network
Analog-digital-analog path

Signal chain
STMF4
Digital to Analog converter

- Can be configured to 8 or 12 bit mode
- The data could be left or right aligned
- DAC has two output channels
  - Both output can be independently or simultaneously
DAC channel block diagram
## DAC pins

<table>
<thead>
<tr>
<th>Name</th>
<th>Signal type</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V_{REF+}$</td>
<td>Input, analog reference positive</td>
<td>The higher/positive reference voltage for the DAC, $1.8, V \leq V_{REF+} \leq V_{DDA}$</td>
</tr>
<tr>
<td>$V_{DDA}$</td>
<td>Input, analog supply</td>
<td>Analog power supply</td>
</tr>
<tr>
<td>$V_{SSA}$</td>
<td>Input, analog supply ground</td>
<td>Ground for analog power supply</td>
</tr>
<tr>
<td>DAC_OUTx</td>
<td>Analog output signal</td>
<td>DAC channelx analog output</td>
</tr>
</tbody>
</table>
Simulink - DAC

• DAC block can be used to read data
Configurations

[Image of a configuration settings window for a DAC (Digital to Analog Converter) module, showing input type, DAC selection, input voltage reference, DAC output buffer enable, sample time, and custom port labels.]
Configurations

- Digital to analog block to generate analog signals (with the maximum of 2 channels)
  - Channel1 = pin A4
  - Channel2 = pin A5

- Analog output voltage = Vref x Data/ 4095
Example

Note the use of a Rate Transition Block to adjust rate from continuous-time output (black color) to discrete-time (red color) signals.

Users can view simulated signals before they are physically generated.
Hardware Setup
Configuring

Configure the Repeating Sequence Block to generate triangular waveform.

Parameters
- Time values:
  - [0 0.001]
- Output values:
  - [0 2]
Configuring

Configure the Sine Wave block to generate sinusoidal waveform.
Notice the staircase appearance of the generated signal. A smoother (higher resolution) waveform can be generated by reducing the sample time of DAC block, i.e. outputting signal at a faster rate. However, there is a finite limit to this number because a MCU can only perform a limited number of computations within one sample time.

Triangular waveform
0-2V 1kHz

Sinusoidal waveform
0.5-2.5V 1kHz
Question?

- Write a program to report error if the analog reading ADC1 pin1 is lower than 10 or analog reading ADC2 pin 2 is higher than 1000 using data align right.
Questions?