Analog Input/Output Subsystem Design

Reference:
STM32F4xx Reference Manual
(ADC, DAC chapters)
Analog input subsystem

Property being measured

- Input transducer
  - Convert “property” to electrical voltage/current

- Signal conditioning
  - Produce convenient voltage/current levels over range of interest

- Sample & hold
  - Hold value during conversion

- Analog to digital conv.
  - Digital value to CPU
Analog output subsystem

Digital value from CPU

- Convert binary code to an analog voltage/current

- Produce convenient voltage/current levels over range of interest

- Convert electrical signal to mechanical or other property

Property being controlled
Typical analog input subsystem

- Property 1
  - Input transducer
  - Signal conditioning
  - Mux
  - Sample & hold
  - Analog to digital conv.
  - Digital value

- Property 2
  - Input transducer
  - Signal conditioning

- Property N
  - Input transducer
  - Signal conditioning

STM32L1xx
---------------
16 channels, 12-bit ADC

- Convert "property" to electrical voltage/current
- Produce convenient voltage/current levels over range of interest
- Select channel
- Hold value during conversion
- Convert analog value to digital #
Analog subsystem properties

- **Accuracy**: degree to which measured value differs from true value
- **Resolution/precision**: degree to which two conditions can be distinguished
  - Related to #bits in digital value
- **Range**: minimum to maximum “useful” value
- **Linearity**: \( y = Ax + B \) (correction req’d if not linear)
  - piecewise linear approximation over different ranges
- **Repeatability**: same measurement for a given value
  - affected by hysteresis or other phenomena
- **Stability**: value changes other than due to the property being measured (e.g., \( T \) affecting \( P \))
Analog to digital conversion errors

May need to correct in software

Offset error

Gain error

Quantization error:
Difference between digital & analog values
Usually want ± ½ LSB

Nonlinearity error -
Unequal distances between transition points
Transducers

- Convert physical quantity to electrical signal
  - Self-generating – generates voltage/current signal
  - Non-self-generating – other property change (ex. R)

Examples:
- Force/stress (strain gage)
- Temperature (thermocouple, thermistor, semicond.)
- Pressure
- Humidity (gypsum block)
- Smoke
- Light (phototransistor, photoconductive cell)
- Acceleration (accelerometer)
- Flow
- Position (potentiometer, displacement)
Temperature sensors

- **Thermocouple** – “Seeback EMF produced by heating junction of dissimilar metals (μV)

- **Thermistor** – mix of materials in ceramic

  \[ R_t = R_0 e^{\beta \left[ \frac{1}{T} - \frac{1}{T_0} \right]} \]

  - Negative temperature coefficient: \( R^\uparrow \) with \( T_v \)
  - Linear over small range

- **Metal conductor:**

  \[ R_t = R_0 \left[ 1 + \alpha (T - T_0) \right] \]

  - Positive temp. coefficient: \( R^\uparrow \) with \( T^\uparrow \)
Semiconductor temperature sensor

Base-emitter voltage approximately proportional to $T$

$$V_{BE} \approx V_{BE}$$

$$V_{BE} = \frac{kT}{q} \ln \left[ \frac{I_c}{I_s} \right]$$

$$V_{BE} \propto T$$
Analog Devices AD590 Temperature Transducer

- IC generates current proportional to temperature
- Generated current $I_T$ is linear: $1 \mu A/\circ K$

Example:
Design a temperature monitor with output in the range [0v..4v] over temperature range [-20°C .. +60°C]
(Use summing amplifier)
Strain Gage

- Measure stress by measuring change or resistance of a conductor due to change of its length/area

- Compression: $L$ decreases, $A$ increases
- Elongation: $L$ increases, $A$ decreases
- “Gage factor” (sensitivity): $S = \frac{\Delta R / R}{\Delta L / L}$

\[ R = \rho \left( \frac{L_0}{A_0} \right) \]
Wheatsone bridge

- Measure small resistance changes

\[ V_o = V_{ref} \left( \frac{R}{R + R} \right) - V_{ref} \left( \frac{R_s}{R + R_s} \right) \]

\[ = V_{ref} \left[ \frac{1}{2} - \frac{R_s}{R + R_s} \right] \]

“Balanced”: \( V_o = 0 \) when \( R = R_s \)

Some pressure sensors use bridge with all 4 R’s variable
Signal conditioning

- Produce noise-free signal over “working” input range
  - Amplify voltage/current levels
  - Bias (move levels to desired range)
  - Filter to remove noise
  - Isolation/protection (optical/transformer)
  - Common mode rejection for differential signals
  - Convert current source to voltage
- Conditioning often done with op amp circuits
Operational amplifiers

- Amplifier types:
  - Inverting amplifier
  - Non-inverting amplifier
  - Summing amplifier
  - Differential amplifier
  - Instrumentation amplifier

- Tradeoffs
  - Inverting/noninverting
  - High input impedance
  - Defined gain
  - Common mode rejection
Basic op amp configurations

Inverting amplifier

\[
V_i = -\frac{V_o}{R_1}
\]

\[
V_o = -\frac{R_2}{V_i}
\]

Noninverting amplifier

\[
V_i = V_o \left(\frac{R_2}{R_1 + R_2}\right)
\]

\[
\frac{V_o}{V_i} = \frac{R_1 + R_2}{R_2}
\]

Noninverting version has high input impedance
Summing amplifier

\[ \frac{V_1}{R_1} + \frac{V_2}{R_2} = -\frac{V_o}{R_3} \]

\[ V_o = -R_3\left(\frac{V_1}{R_1} + \frac{V_2}{R_2}\right) \]

Potential application:
V1 = input voltage
V2/R2 provide an “offset” to V1/R1
(ex. to produce Vo=0 at some V1 value)
Differential amplifier

Eliminates “common mode” voltage (noise, etc.)

\[ \frac{V_1 - V_x}{R_1} = \frac{V_x - V_o}{R_2} \]

\[ V_x = \frac{R_1 V_0 + R_2 V_1}{R_1 + R_2} \]

\[ V_o = \frac{R_2}{R_1} (V_2 - V_1) \]

Choose R1 to set input impedance; R2 to set gain
Instrumentation amplifier

\[ V_o = (V_2 - V_1) \left[ 1 + \frac{2R_2}{R_1} \right] \left( \frac{R_4}{R_3} \right) \]

- High input impedance, common mode rejection
- Can match R2, R3, R4 on chip and use external R1 to set gain
Sample-and-_hold

Required if A/D conversion slow relative to frequency of signal:

- Close switch to “sample” Vin (charge C to Vin)
  - Aperture (sampling) time = duration of switch closure
- Open switch to “hold” Vin

Figure 17-4  Aperture time error.
Analog to digital conversion

Given: continuous-time electrical signal
\( v(t), \ t \geq 0 \)

Desired: sequence of discrete numeric values that represent the signal at selected sampling times:

\[ v(0), v(T), v(2T), \ldots v(nT) \]

- \( T \) = “sampling time”: \( v(t) \) “sampled” every \( T \) seconds
- \( n \) = sample number
- \( v(nT) \) = value of \( v(t) \) measured at the \( n^{th} \) sample time and quantized to one of \( 2^k \) discrete levels
A/D conversion process

Sampled data sequence:
\[ n = 1, 2, 3, 4, 5, 6, 7 \]
\[ d = 10, 10, 10, 10, 11, 11, 11 \]

Binary values of \( d \), where
\[ v(nT) = (d/4)V_{\text{ref}} \]
A/D conversion parameters

- **Sampling rate, F** (sampling interval \( T = 1/F \))
  - *Nyquist rate* \( \geq 2 \times \) (highest frequency in the signal)
    - to reproduce sampled signals
    - CD-quality music sampled at 44.1KHz
      (ear can hear up to about 20-22KHz)
    - Voice in digital telephone sampled at 8KHz

- **Precision (# bits in sample value)**
  - \( k = \) # of bits used to represent sample values
  - “precision”: each step represents \((1/2^k) \times V_{\text{range}}\)
    - Ex. Temperatures [-20°C…+60°C]: if \( k=8 \), precision = 80°C/256 = 0.3125°C
  - “accuracy”: degree to which converter discerns proper level
    (error when rounding to nearest level)
Analog to digital conversion

- More difficult than D/A conversion
- Tradeoffs:
  - Precision (# bits)
  - Accuracy
  - Speed (of conversion)
  - Linearity
  - Unipolar vs. bipolar input
  - Encoding method for output
  - Cost
- Often built around digital to analog converters
Digital to analog conversion

Number = \( b_n b_{n-1} \ldots b_1 b_0 = b_n \cdot 2^n + b_{n-1} \cdot 2^{n-1} + \ldots + b_1 \cdot 2^1 + b_0 \cdot 2^0 \)

R-2R Ladder Network

\[ V_o = V_R \sum_{k=0}^{n} b_k \left( \frac{1}{2^k} \right) \]

Equivalent resistance = \( R \)
STM32F4xx D/A converter

- 8 or 12-bit modes
- 2 DACs/channels
  - Left/Right channel
  - Concurrent conversions
- Sample triggers:
  - SW trigger
  - Timer triggers
  - EXTI trigger
- DMA support
  - Memory -> DHRx

Diagram description:
- Triggers
- Control logic
- Data Holding Reg
- Data Output Reg
- Digital-to-analog converter
- External Reference (internal reference also available)
DAC data formats

**Single DAC Channel**

- 8-bit right aligned
- 12-bit left aligned
- 12-bit right aligned

**DHR name (x = 1/2 channel):**
- DAC_DHR8Rx
- DAC_DHR12Lx
- DAC_DHR12Rx

**Dual DAC Channels**

- 8-bit right aligned
- 12-bit left aligned
- 12-bit right aligned

**DHR name:**
- DAC_DHR8RD
- DAC_DHR12LD
- DAC_DHR12RD
DAC data conversion

Write data to DAC_DHRx register (trigger disabled: TEN=0):

\[ \text{DACoutput} = V_{\text{REF}} \times \frac{\text{DOR}}{4095} \]
## DAC control/status registers

DAC_CR (Upper half = channel 2; Lower half = channel 1)

<table>
<thead>
<tr>
<th>Channel</th>
<th>Enable</th>
<th>Trigger</th>
<th>Enable*</th>
<th>Trigger</th>
<th>Select</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>DMAU DRIE1</td>
<td>DMA EN1</td>
<td>MAMP1[3:0]</td>
<td>WAVE1[1:0]</td>
<td>TSEL1[2:0]</td>
</tr>
</tbody>
</table>

* If TEN=0, start when DHR written

**DAC_SWTRIGR** = Software trigger – start when bit set by SW (reset by HW)

<table>
<thead>
<tr>
<th>Reserved</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
</tr>
<tr>
<td>Reserved</td>
</tr>
<tr>
<td>SWTRIG2</td>
</tr>
<tr>
<td>w</td>
</tr>
</tbody>
</table>

**DAC_SR** = Status Register – Indicates DMS underrun (no data before trigger)
Flash A/D conversion

- N-bit result requires $2^n$ comparators and resistors:

$$ V = V_{ref} \left( \frac{2^n - 1}{2^n R} \right) $$

Comparator output = 1 if $V_{in} > V_{ref} \times \left( \frac{N}{2^n} \right)$

($N = 1, 2, \ldots, 2^{n-1}$)

"Thermometer code" – bottom $k$ bits = 1, upper $2^{n-1}-k$ bits = 0
Dual-slope conversion

- Use counter to measure time required to charge/discharge capacitor (relatively low speed).
- Charging, then discharging eliminates non-linearities (high accuracy).
- Relatively low cost
1. SW1 connects Vin for fixed time T
   - C charges with current = \( \text{Vin}(t)/R \)

\[
\text{Vo}(t) = -\frac{1}{C} \int_{0}^{T} i_c(t)dt = -\frac{1}{RC} \int_{0}^{T} \text{Vin}(t)dt = -\frac{T}{RC} \text{Vin}
\]
Dual-slope conversion steps

2. SW1 connects –Vref until Vo discharges to 0.
   - C discharges with constant current = -Vref/R
     \[ Vo(T + t_1) = -\frac{1}{RC} \int_0^T Vin(t)dt + \frac{1}{RC} \int_T^{T+t_1} V_{ref} dt \]
   - When \( Vo(T+t_1) = 0 \):
     \[ \frac{1}{RC} \int_0^T Vin(t)dt = \frac{1}{RC} \int_T^{T+t_1} V_{ref} dt \]

\[ Vin = \left( \frac{t_1}{T} \right) V_{ref} \]

Use a counter to measure \( t_1 \).
Successive approximation analog to digital converter (ADC)

- Determine one bit at a time, from MSB to LSB
  Used in most microcontrollers (low cost)

1. **Successive Approximation Register (SAR)** sets $D_{N-1} = 1$
2. SAR outputs $D_{N-1} \ldots D_0$, converted by DAC to analog $V_{DAC}$
3. $V_{DAC}$ is compared to $V_{IN}$
4. **Comparator** output resets $D_{N-1}$ to 0 in SAR if $V_{DAC} < V_{IN}$
5. Repeat 1-4 for $D_{N-2} \ldots D_0$ (one clock period per bit)

- Final SAR value $D_{N-1} \ldots D_0$ is digital representation of $V_{IN}$
12-bit successive approximation A/D converter
- Programmable precision: 6-8-10-12 bits
- Conversion time = #bits + 3 clock cycles
  - 1.2 Msamp/sec @\(V_{DDA}=1.8\text{-}2.4\text{v}\)
  - 1.4 Msamp/sec @\(V_{DDA}=2.4\text{-}3.6\text{v}\)

“Regular” and “Injected” channel groups
- Injected channels processed after, or between, regular channels

19 multiplexed input channels
- 16 external sources
- 3 internal sources: \(V_{BAT}, V_{REFINT}, \) temperature sensor

External trigger option (16 sources)

Multiple conversion modes
- Single, continuous, scan, discontinuous

DMA and/or interrupts are supported
- DMA often used in “scan” mode, to unload the single data register
STM32 ADC block diagram

- **ADC Inputs**: ADCx_IN0, ADCx_IN1, ..., ADCx_IN15
- **Regular Channel Triggers**: TIM1_TRGO, TIM2_TRGO, ..., TIM8_TRGO
- **Injected Channel Triggers**: TIM1_TRGO, TIM2_TRGO, ..., TIM8_TRGO
- **Reference Voltage**: VREF, VREFINT, VBAT
- **ADC**: Analog to digital converter
- **Injected Data Registers**: 4 x 16 bits
- **Regular Data Register**: 16 bits
- **DMA Request**
- **Clock**: Prescaled \( f_{APB1} \)
- **Interrupts**
- **Analog Watchdog**
- **Results**: Injected, Regular

Additional labels:
- **DMA Overrun**
- **End of Conversion**
- **End of Injected Conversion**
- **Analog Watchdog Event**
- **OVR**
- **EOC**
- **EOE**
- **AEW**
- **AWDIE**
- **OVRIE**
- **EOCIE**
- **EOEIE**
- **AWDIE**
**ADC clocking**

- **Analog circuitry clock: ADCCLK**
  - Derive from APB2 clock ÷ prescale
    - \( f_{\text{ADC}} = \frac{f_{\text{PCLK2}}}{2, 4, 6, 8} \) (bits ADCPRE in ADC_CCR)
    - \( f_{\text{ADC}} \) required range = 0.6MHz – 18MHz (\( V_{\text{DDA}} = 1.8 \) to 2.4v)
      - = 0.6MHz – 36MHz (\( V_{\text{DDA}} = 2.4 \) to 3.6v)
  - Sample time (ts) = 3 to 480 clock cycles (8 choices)
    - ts = 0.10\( \mu \)s to 16\( \mu \)s @\( f_{\text{ADC}} = 30\)MHz
    - Set for *each channel* in ADC_SMPR1, ADC_SMPR2
  - Conversion time = ts + n (#data bits) = 9 to 492 clocks
    - 0.50\( \mu \)s to 16.40\( \mu \)s for 12-bit data @\( f_{\text{ADC}} = 30\)MHz
    - \( f_{\text{s}} \leq 2 \) Msamples/sec @\( f_{\text{ADC}} = 30\)MHz, ts = 3 ADC cycles
- Enable HSI clock in RCC->CR, which runs ADC conversions
  - RCC->CR |= RCC_CR_HSION;  //HSION = bit 0 of RCC->CR
- Digital interface clock (register read/write)
  - Enable APB2 clock in RCC_APB2ENR (clock enable register)
Conversion modes

- **Single conversion** (default: SCAN=0 in CR1, CONT=0 in CR2)
  - Select an input channel (SQ1 field in in ADC1->SQR5)
  - Start the conversion (software start or hardware trigger)
  - EOC sets when conversion is complete
  - Read the result in the DR

- **Scan mode** (enable with SCAN=1 in CR1)
  - Perform a **sequence** of conversions of designated input channels
    - Define sequence length in ADC1->SQR1
    - Select channels in ADC1->SQR1…ADC1->SQR5 (channels can be in any order)
  - Start the conversion sequence (software start or hardware trigger)
  - EOC sets after each conversion (EOCS = 0) or after the entire sequence is complete (EOCS = 1). EOCS is in ADC1->CR2

- **Continuous mode** (enable with CONT=1 in CR2)
  - Start 1st conversion/sequence (software start or hardware trigger)
  - Next conversion/sequence starts automatically after a conversion/sequence completes
Scan mode

- Convert multiple channels in a “sequence”
  - Enable via SCAN bit in ADC_CR1
  - Repeat if CONT bit set in ADC_CR2
  - EOC bit set in ADC_SR at end of sequence or after each conversion (select via EOCS bit)
    - Regular channel data to ADC_DR
    - Injected channel data to ADC_JDR1 – ADC_JDR4
  - Configure sequence via sequence registers
    - ADC_SQR1 – seq. length and channel #s for conversions 13-16
    - ADC_SQR2 – channel #s for conversions 7-12
    - ADC_SQR3 – channel #s for conversions 1-6
    - ADC_JSQR – seq. length and channel #s for up to 4 injected channels
  - If JAUTO=1 (in ADC_CR1),
    - Injected group is converted after regular group after regular trigger
    - Injected group interrupts regular group after injection trigger
Discontinuous mode

- Convert a subset of a sequence on each external trigger

  Regular group, on external trigger:
  - convert n (≤ 8) channels from a sequence
  - convert the next n channels on the next trigger
  - repeat until all channels in the sequence are done
  - restart the sequence on the next trigger

- Injected group:
  - Similar, but only 1 channel per external trigger
**STM32 ADC control register 1 (ADC_CR1)**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RES</td>
<td>Resolution (00=12 bit, 01=10-bit, 10=8-bit, 11=6-bit)</td>
</tr>
<tr>
<td>SCAN</td>
<td>Enable scan mode (channel #s in ADC_SQRx, ADC_JSQRx)</td>
</tr>
<tr>
<td>JAUTO</td>
<td>Enable automatic injected group conversion after regular group</td>
</tr>
<tr>
<td>EOCIE/JEOCIE</td>
<td>On end of conversion (regular/injected channel)</td>
</tr>
<tr>
<td>OVRIE</td>
<td>On overrun</td>
</tr>
</tbody>
</table>

**Discontinuous mode:**
- **DISCEN/JDISCEN**: Enable on regular/injected channels
- **DISCNUM**: # channels to convert after trigger (1-8)

**Analog Watchdog:**
- **AWDEN/JAWDEN**: Enable on regular/injected channels
- **AWDCH**: Analog watchdog channel selection
- **AWDSGL**: Enable watchdog on single channel in scan mode
- **AWDIE**: Enable interrupt on analog watchdog
### STM32 ADC control register 2 (ADC_CR2)

<table>
<thead>
<tr>
<th>Bit 31</th>
<th>Bit 30</th>
<th>Bit 29</th>
<th>Bit 28</th>
<th>Bit 27</th>
<th>Bit 26</th>
<th>Bit 25</th>
<th>Bit 24</th>
<th>Bit 23</th>
<th>Bit 22</th>
<th>Bit 21</th>
<th>Bit 20</th>
<th>Bit 19</th>
<th>Bit 18</th>
<th>Bit 17</th>
<th>Bit 16</th>
</tr>
</thead>
<tbody>
<tr>
<td>reserved</td>
<td>SWSTART</td>
<td>EXTEN</td>
<td>EXTFSEL[3:0]</td>
<td>reserved</td>
<td>JSWSTART</td>
<td>JEXTEN</td>
<td>JEXTSEL[3:0]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
</tr>
<tr>
<td>15</td>
<td>14</td>
<td>13</td>
<td>12</td>
<td>11</td>
<td>10</td>
<td>9</td>
<td>8</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>reserved</td>
<td>ALIGN</td>
<td>EOCS</td>
<td>DDS</td>
<td>DMA</td>
<td>Reserved</td>
<td>CONT</td>
<td>ADON</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### ADON: 1=enable ADC, 0=disable ADC and power down

#### CONT: 1 = continuous conversions, 0 = single conversion

#### ALIGN: data alignment in 16-bit data register (0=right, 1=left)

#### EOCS: end of conversion selection

- 0=set EOC at end of sequence, 1=set EOC at end of each conversion

#### DMA: DMA enable

#### DDS: DMA disable selection

- 0=no new DMS request after last channel,
- 1=continue DMA requests as long as DMA=1

#### SWSTART/JSWSTART: start conversion of regular/injected channels

#### EXTEN/JEXTEN: external trigger event

- 00=disable, 01=rising edge, 10=falling edge, 11=both edges

#### EXTFSEL/JEXTSEL[3:0]: select external event for trigger (regular/injected)

- different sets of 16 sources for regular and injected mode
ADC status register ADC_SR

<table>
<thead>
<tr>
<th>Bit 15</th>
<th>Bit 14</th>
<th>Bit 13</th>
<th>Bit 12</th>
<th>Bit 11</th>
<th>Bit 10</th>
<th>Bit 9</th>
<th>Bit 8</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>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>OVR</td>
<td>STRT</td>
<td>JSTRT</td>
<td>JEOC</td>
<td>EOC</td>
<td>AWD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rc_w0</td>
<td>rc_w0</td>
<td>rc_w0</td>
<td>rc_w0</td>
<td>rc_w0</td>
<td>rc_w0</td>
<td>rc_w0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**OVR**: overrun flag (set if data has been lost)

**STRT/JSTRT**: regular/injected channel conversion started flag

**EOC/JEOC**: end of conversion flag (regular/injected channel)

End of sequence (if EOCS=1) or one conversion (EOCS=0)

**AWD**: analog watchdog flag

“event: if voltage crosses values in ADC_LTR and ADC_HTR

All flags set by HW and cleared by SW
Sigma Delta ADC

- High resolution (16 or more bits)
- High integration
- Reasonable cost
- Often used to sample CD-quality audio
  - 16-bit resolution @ 44.1Ksamples/sec
- Oversampling used to spread noise over wider frequency range
- Digital filtering eliminates the noise
  - Gives good dynamic range with simple ADC
Sigma-Delta A/D Converter

Figure 6-4  Block Diagram of First-Order Sigma-Delta A/D converter
Sigma-Delta ADC

Density of 1’s at modulator output proportional to the input signal.

Filtering extracts info from serial data stream. (lower rate)
Modulator operation

- Slope of integrator output depends on magnitude of Vin
  - “sigma” => summing/integration

- Compare integrator output to 0v, producing “1” if positive and “0” if negative (1-bit ADC)
  - “delta” = difference

- Density of 1’s in the bitstream proportional to magnitude of input voltage Vin
Example

ΔΣ converter operation with 0 volt analog input

Flip-flop output

0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1

Integrator output

ΔΣ converter operation with small negative analog input

Flip-flop output

0 1 0 1 0 1 1 0 0 1 0 1 0 1 0 1

Integrator output

ΔΣ converter operation with medium negative analog input

Flip-flop output

0 1 0 1 1 0 1 0 1 1 0 1 1 0 1 1 0

Integrator output

ΔΣ converter operation with large negative analog input

Flip-flop output

0 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1

Integrator output

Filtering determines average voltage (density of 1s) in bitstream
Maxim MAX1402 Sigma-Delta ADC
# ADC converter characteristics

<table>
<thead>
<tr>
<th>Type</th>
<th>Need SHA?</th>
<th>Cycles/conversion</th>
<th>Advantages</th>
<th>Disadvantages</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flash</td>
<td>No</td>
<td>1</td>
<td>Fastest</td>
<td>Expensive, power</td>
<td>6-bit @ 400MHz</td>
</tr>
<tr>
<td>Successive Approximation</td>
<td>Yes</td>
<td>&gt;= 2</td>
<td>Fast, cheap</td>
<td>Slower than flash</td>
<td>8-bit @ 20 MHz</td>
</tr>
<tr>
<td>Integrating</td>
<td>Yes</td>
<td>Varies</td>
<td>Precise</td>
<td>Slow</td>
<td>22-bit @ 20Hz</td>
</tr>
<tr>
<td>Sigma-Delta</td>
<td>No</td>
<td>Many</td>
<td>Mostly digital, linear, high resolution</td>
<td>Complex digital circuit</td>
<td>16-bit @ 100 KHz</td>
</tr>
</tbody>
</table>
ADC converter comparison

The diagram compares different ADC (Analog-to-Digital Converter) methods based on resolution (bit) versus samples per second. The methods shown are integrating, sigma-delta, successive approximation, half-flash, and flash. Each method is represented by a shaded area on the graph, indicating their performance characteristics.
ADC selection (Analog Devices, Inc.)