L10: Analog Building Blocks
(OpAmps, A/D, D/A)

Acknowledgement: Dave Wentzloff
Introduction to Operational Amplifiers

**DC Model**

- Typically very high input resistance ~ 300KΩ
- High DC gain (~$10^5$)
- Output resistance ~75Ω

\[ V_{out} = a(f) \cdot V_{in} \]

**LM741 Pinout**

- +10 to +15V
- -10 to -15V

-20dB/decade

10Hz

$10^5$
The Inside of a 741 OpAmp

**Differential Input Stage**

**Current Source for biasing**

**Additional Gain Stage**

**Output Stage**

Gain is Sensitive to Operating Condition (e.g., Device, Temperature, Power supply voltage, etc.)

Output devices provide large drive current

Bipolar version has small input Bias current

MOS OpAmps have ~0 input current
Simple Model for an OpAmp

- $i_+ \sim 0$
- $i_- \sim 0$
- $v_{id} \sim 0$
- $v_{id}$
- $v_{id} < -\varepsilon$
- $v_{id} > \varepsilon$
- $v_{out}$
- $v_{out}$
- $v_{out}$
- $v_{out}$
- $v_{out}$
- $v_{out}$
- $v_{out}$
- $V_{CC} = 10V$
- $V_{CC} = -10V$
- $\varepsilon = 100\mu V$
- $-100\mu V$

Linear Mode

- $v_{id}$
- $-v_{id}$
- $+a v_{id}$
- $v_{out}$

If $-V_{CC} < v_{out} < V_{CC}$

Negative Saturation

- $v_{id}$
- $v_{id}$
- $v_{id}$
- $-v_{CC}$
- $v_{out}$
- $-v_{out}$

$v_{id} < -\varepsilon$

Positive Saturation

- $v_{id}$
- $v_{id}$
- $v_{id}$
- $v_{id}$
- $+V_{CC}$
- $v_{out}$
- $v_{out}$

$v_{id} > \varepsilon$

Small input range for “Open” loop Configuration
The Power of (Negative) Feedback

\[ \frac{v_{in} + v_{id}}{R_1} + \frac{v_{out} + v_{id}}{R_2} = 0 \]

\[ v_{id} = \frac{v_{out}}{a} \]

\[ \frac{v_{in}}{R_1} = - \frac{v_{out}}{a} \left[ \frac{1}{R_1} + \frac{a}{R_2} + \frac{1}{R_2} \right] \]

\[ \frac{v_{out}}{v_{in}} = - \frac{R_2 a}{(1 + a)R_1 + R_2} \approx - \frac{R_2}{R_1} \quad (\text{if} \quad a \gg 1) \]

- Overall (closed loop) gain does not depend on open loop gain
- Trade gain for robustness
- Easier analysis approach: “virtual short circuit approach”
  - \( v_+ = v_- = 0 \) if OpAmp is linear
Basic OpAmp Circuits

Voltage Follower (buffer)

\[ v_{out} \approx v_{in} \]

Non-inverting

\[ v_{out} \approx \frac{R_1 + R_2}{R_1} v_{in} \]

Differential Input

\[ v_{out} \approx \frac{R_2}{R_1} (v_{in2} - v_{in1}) \]

Integrator

\[ v_{out} \approx -\frac{1}{RC} \int_{-\infty}^{t} v_{in} \, dt \]
Analog Comparator:

Is $V_+ > V_-$?
The Output is a DIGITAL signal

Analog Comparator: Analog to TTL

LM 311 Needs Pull-Up

LM311 is a single supply comparator
Data Conversion: Quantization Noise

A/D Conversion

D/A Conversion

- Quantization noise exists even with *ideal* A/D and D/A converters
Non-idealities in Data Conversion

**Offset** – a constant voltage offset that appears at the output when the digital input is 0

**Gain error** – deviation of slope from ideal value of 1

**Integral Nonlinearity** – maximum deviation from the ideal analog output voltage

**Differential nonlinearity** – the largest increment in analog output for a 1-bit change

**Non-monicity**
R-2R Ladder DAC Architecture

Note that the driving point impedance (resistance) is the same for each cell.

R-2R Ladder achieves large current division ratios with only two resistor values.

\[ V_{out} = \frac{1}{6} V_{ref} \left[ B_7 + \frac{1}{2} B_6 + \frac{1}{4} B_5 + \ldots + \frac{1}{128} B_0 \right] \]
DAC (AD 558) Specs

- 8-bit DAC
- Single Supply Operation: 5V to 15V
- Integrates required references (bandgap voltage reference)
- Uses a R-2R resistor ladder
- Settling time 1μs
- Programmable output range from 0V to 2.56V or 0V to 10V
- Simple Latch based interface

### Input Logic Coding

<table>
<thead>
<tr>
<th>Digital Input Code</th>
<th>Hexadecimal</th>
<th>Decimal</th>
<th>2.56 V Range</th>
<th>10.000 V Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000 0000</td>
<td>00</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0000 0001</td>
<td>01</td>
<td>1</td>
<td>0.010 V</td>
<td>0.039 V</td>
</tr>
<tr>
<td>0000 0010</td>
<td>02</td>
<td>2</td>
<td>0.020 V</td>
<td>0.078 V</td>
</tr>
<tr>
<td>0000 1111</td>
<td>0F</td>
<td>15</td>
<td>0.150 V</td>
<td>0.586 V</td>
</tr>
<tr>
<td>0001 0000</td>
<td>10</td>
<td>16</td>
<td>0.160 V</td>
<td>0.625 V</td>
</tr>
<tr>
<td>0111 1111</td>
<td>7F</td>
<td>127</td>
<td>1.270 V</td>
<td>4.961 V</td>
</tr>
<tr>
<td>1000 0000</td>
<td>80</td>
<td>128</td>
<td>1.280 V</td>
<td>5.000 V</td>
</tr>
<tr>
<td>1100 0000</td>
<td>C0</td>
<td>192</td>
<td>1.920 V</td>
<td>7.500 V</td>
</tr>
<tr>
<td>1111 1111</td>
<td>FF</td>
<td>255</td>
<td>2.55 V</td>
<td>9.961 V</td>
</tr>
</tbody>
</table>
Chip Architecture and Interface

Chip Architecture and Interface

Table I. AD558 Control Logic Truth Table

<table>
<thead>
<tr>
<th>Input Data</th>
<th>CE</th>
<th>CS</th>
<th>DAC Data</th>
<th>Latch Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>“Transparent”</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>“Transparent”</td>
</tr>
<tr>
<td>0</td>
<td>g</td>
<td>0</td>
<td>0</td>
<td>Latching</td>
</tr>
<tr>
<td>1</td>
<td>g</td>
<td>0</td>
<td>1</td>
<td>Latching</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>g</td>
<td>0</td>
<td>Latching</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>g</td>
<td>1</td>
<td>Latching</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
<td>X</td>
<td>Previous Data</td>
<td>Latched</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>1</td>
<td>Previous Data</td>
<td>Latched</td>
</tr>
</tbody>
</table>

NOTES
X = Does not matter.
g = Logic Threshold at Positive-Going Transition.

Outputs are noisy when input bits settles, so it is best to have inputs stable before latching the input data

Figure 6. AD558 Control Logic Function
Setting the Voltage Range

Very similar to a non-inverting amp

Strap output for different voltage ranges

Convert data to Offset binary

<table>
<thead>
<tr>
<th>Digital Input Code</th>
<th>Output Voltage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary</td>
<td>Offset Binary</td>
</tr>
<tr>
<td></td>
<td>Decimal</td>
</tr>
<tr>
<td>0000 0000</td>
<td>00</td>
</tr>
<tr>
<td>0000 0001</td>
<td>01</td>
</tr>
<tr>
<td>0000 0010</td>
<td>02</td>
</tr>
<tr>
<td>0000 1111</td>
<td>0F</td>
</tr>
<tr>
<td>0001 0000</td>
<td>10</td>
</tr>
<tr>
<td>0111 1111</td>
<td>7F</td>
</tr>
<tr>
<td>1000 0000</td>
<td>80</td>
</tr>
<tr>
<td>1100 0000</td>
<td>C0</td>
</tr>
<tr>
<td>1111 1111</td>
<td>FF</td>
</tr>
</tbody>
</table>
Another Approach: Binary-Weighted DAC

- Switch binary-weighted currents
- MSB to LSB current ratio is $2^N$

Analog Devices AD9768 uses two banks of ratioed currents

Additional current division performed by 750 Ω resistor between the two banks

$$v_{out} = -IR\left(b_3 + \frac{1}{2}b_2 + \frac{1}{4}b_1 + \frac{1}{8}b_0 \right)$$
Glitching is caused when switching times in a D/A are not synchronized.

Example: Output changes from 011 to 100 – MSB switch is delayed.

Filtering reduces glitch but increases the D/A settling time.

One solution is a thermometer code D/A – requires $2^N - 1$ switches but no ratioed currents.

\[
v_{out} = -IR(T_0 + T_1 + T_2)
\]
D/A converters are typically compact and easier to design. Why not A/D convert using a D/A converter and a comparator?
- D to A generates analog voltage which is compared to the input voltage
- If D to A voltage > input voltage then set that bit; otherwise, reset that bit
- This type of A to D takes a fixed amount of time proportional to the bit length

Example: 3-bit A/D conversion, 2 LSB < $V_{in}$ < 3 LSB
Serial conversion takes a time equal to $N(t_{D/A} + t_{comp})$
Successive-Approximation A/D (AD670)

Unipolar (BPO =0)

Bipolar (BPO =1)

<table>
<thead>
<tr>
<th>BPO/UPO</th>
<th>FORMAT</th>
<th>INPUT RANGE/OUTPUT FORMAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Unipolar/Straight Binary</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Bipolar/Offset Binary</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Unipolar/2s Complement</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Bipolar/2s Complement</td>
</tr>
</tbody>
</table>

2a. 0 V to 2.55 V (10 mV/LSB)

3a. ±1.28 V Range
**Single Write, Single Read Operation**
(see data sheet for other modes)

- **$t_w$** (write/start pulse width) = 300ns (min)
- $t_{DC}$ (delay to start conversion) = 700ns (max)
- $t_c$ (conversion time) = 10µs (max)
- $t_{TD}$ (Bus Access Time) = 250 (max)
- $t_{DT}$ (Output Float Delay) = 150 (max)

- Control bits CE and CS can be wired to ground if A/D is the only chip driving the bus
- Suggestion: tie CE and CS pins together and hardwire BPO and Format
Status should be synchronized: why?

 Courtesy of James Oey and Cemal Akcaba
module AD670 (clk, reset, sample, dataavail, r_wbar, cs_bar, status, state);

    // System Clk
    input clk;
    // Global Reset signal, assume it is synchronized
    input reset;

    // User Interface
    input sample;
    output dataavail;

    // A-D Interface
    input status;
    reg status_d1, status_d2;
    output r_wbar, cs_bar;
    output [3:0] state;

    // internal state
    reg [3:0] state;
    reg [3:0] nextstate;
    reg r_wbar_int, r_wbar;
    reg cs_bar_int, cs_bar;
    reg dataavail;

    // State declarations.
    parameter IDLE = 0;
    parameter CONV0 = 1;
    parameter CONV1 = 2;
    parameter CONV2 = 3;
    parameter WAITSTATUSHIGH = 4;
    parameter WAITSTATUSLOW = 5;
    parameter READDELAY0 = 6;
    parameter READDELAY1 = 7;
    parameter READCYCLE = 8;

    always @ (posedge clk or negedge reset)
    begin
        if (!reset) state <= IDLE;
        else state <= nextstate;

        status_d1 <= status;
        status_d2 <= status_d1;

        r_wbar <= r_wbar_int;
        cs_bar <= cs_bar_int;

        end
always @ (state or status_d2 or sample) begin
  // defaults
  r_wbar_int = 1; cs_bar_int = 1; dataavail = 0;

  case (state)
    IDLE: begin
      if(sample) nextstate = CONV0;
      else nextstate = IDLE;
      end
    CONV0: begin
      r_wbar_int = 0;
      cs_bar_int = 0;
      nextstate = CONV1;
      end
    CONV1: begin
      r_wbar_int = 0;
      cs_bar_int = 0;
      nextstate = CONV2;
      end
    CONV2: begin
      r_wbar_int = 0;
      cs_bar_int = 0;
      nextstate = WAITSTATUSHIGH;
      end
    WAITSTATUSHIGH: begin
      cs_bar_int = 0;
      if (status_d2) nextstate = WAITSTATUSLOW;
      else nextstate = WAITSTATUSHIGH;
      end
    WAITSTATUSLOW: begin
      cs_bar_int = 0;
      if (!status_d2) nextstate = READDELAY0;
      else nextstate = WAITSTATUSLOW;
      end
  endcase
end
Example A/D Verilog Interface (cont.)

```verilog
READDELAY0:
    begin
        cs_bar_int = 0;
        nextstate = READDELAY1;
    end

READDELAY1:
    begin
        cs_bar_int = 0;
        nextstate = READCYCLE;
    end

READCYCLE:
    begin
        cs_bar_int = 0;
        dataavail = 1;
        nextstate = IDLE;
    end

    default: nextstate = IDLE;
    endcase // case(state)
    end // always @(state or status_d2 or sample)
endmodule // adcInterface
```

---

5/5
On reset, present state goes to 0.

r_w_b must stay low for at least 3 cycles (@ 100ns period)

Sample pulse initiates data conversion.

Notice a one cycle delay since A/D control signal delayed through a register.

Status is synchronized – two register delays.

Wait for ~10µs for status to go low.

Enable read flip-flop.
Flash A/D Converter

- **Brute-force A/D conversion**
- **Simultaneously compare the analog value with every possible reference value**
- **Fastest method of A/D conversion**
- **Size scales exponentially with precision** (requires \(2^N\) comparators)

Can be implemented as OpAmp in open loop
**AD 775 – Flash Data Converter**

**TIMING SPECIFICATIONS**

<table>
<thead>
<tr>
<th>Specification</th>
<th>Symbol</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Conversion Rate</td>
<td></td>
<td>20</td>
<td>35</td>
<td></td>
<td>MHz</td>
</tr>
<tr>
<td>Clock Period</td>
<td>t_C</td>
<td>50</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Clock High</td>
<td>t_CH</td>
<td>25</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Clock Low</td>
<td>t_CL</td>
<td>25</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Output Delay</td>
<td>t_DD</td>
<td>18</td>
<td>30</td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Pipeline Delay (Latency)</td>
<td></td>
<td></td>
<td>2.5</td>
<td></td>
<td>Clock Cycles</td>
</tr>
<tr>
<td>Sampling Delay</td>
<td>t_DS</td>
<td>4</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Aperture Jitter</td>
<td></td>
<td>30</td>
<td></td>
<td></td>
<td>ps</td>
</tr>
</tbody>
</table>
High Performance Converters: Use Pipelining and Parallelism!

Pipelining (used in video rate, RF basestations, etc.)

Parallelism (use many slower A/D’s in parallel to build very high speed A/D converters)

[ISSCC 2003], Poulton et. al.

20Gsample/sec, 8-bit ADC from Agilent Labs
Analog blocks are integral components of any system. Need data converters (analog to digital and digital to analog), analog processing (OpAmps circuits, switched capacitors filters, etc.), power converters (e.g., DC-DC conversion), etc.

We looked at example interfaces for A/D and D/A converters
- Make sure you register critical signals (enables, R/W, etc.)

Analog design incorporate digital principles
- Glitch free operation using coding
- Parallelism and Pipelining!
- More advanced concepts such as calibration