L10: Analog Building Blocks

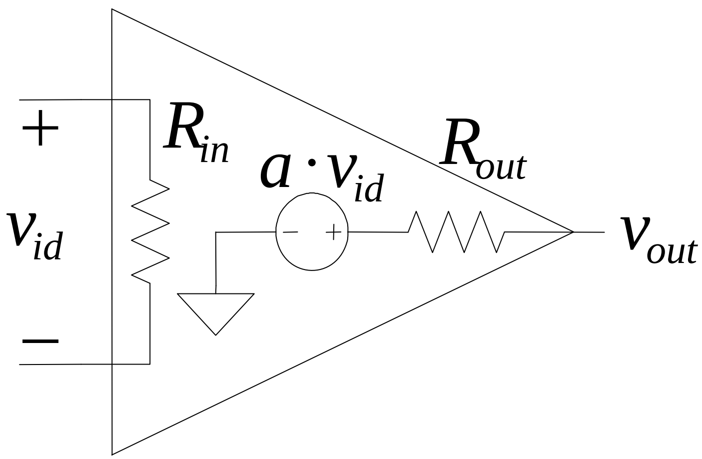
(OpAmps, A/D, D/A)

Acknowledgement: Dave Wentzloff

Lecture Notes prepared by Professor Anantha Chadrakasan

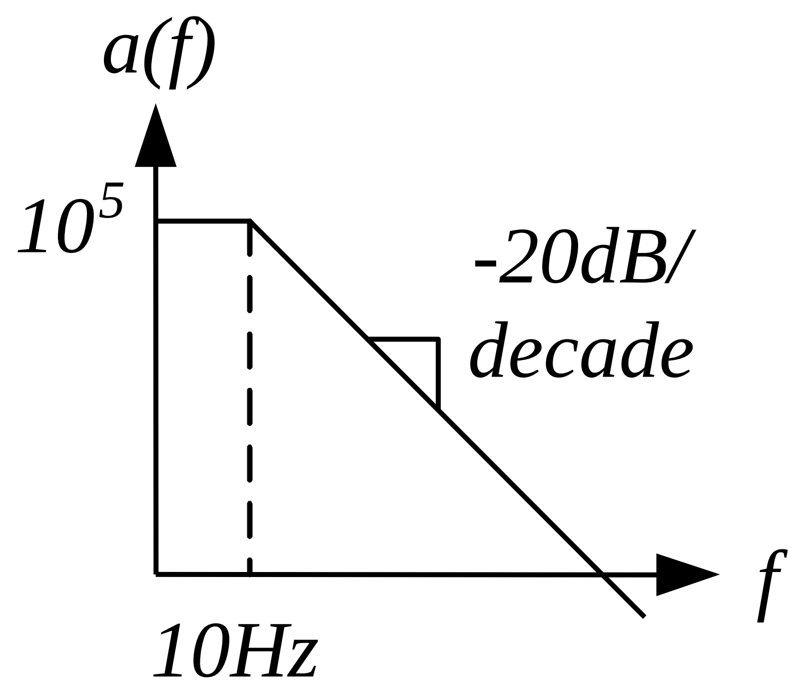
Introduction to Operational Amplifiers

DC Model

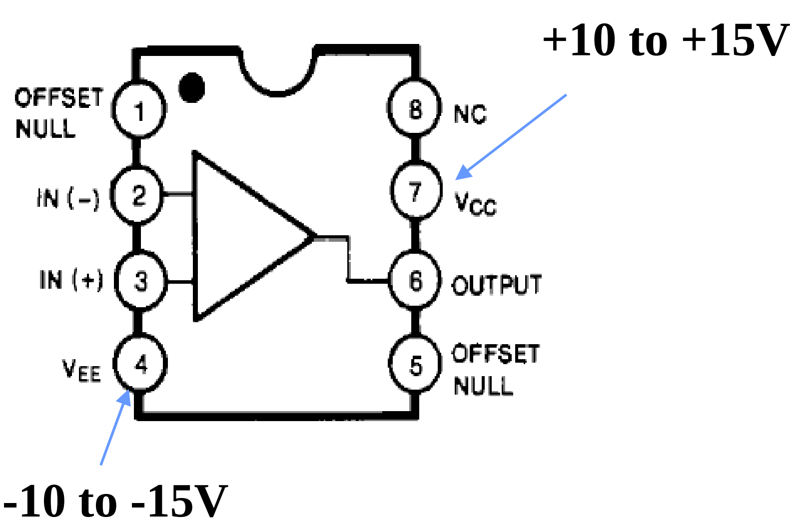


- Typically very high input resistance ~ 300KΩ
- High DC gain (~10^5)
- Output resistance ~75Ω

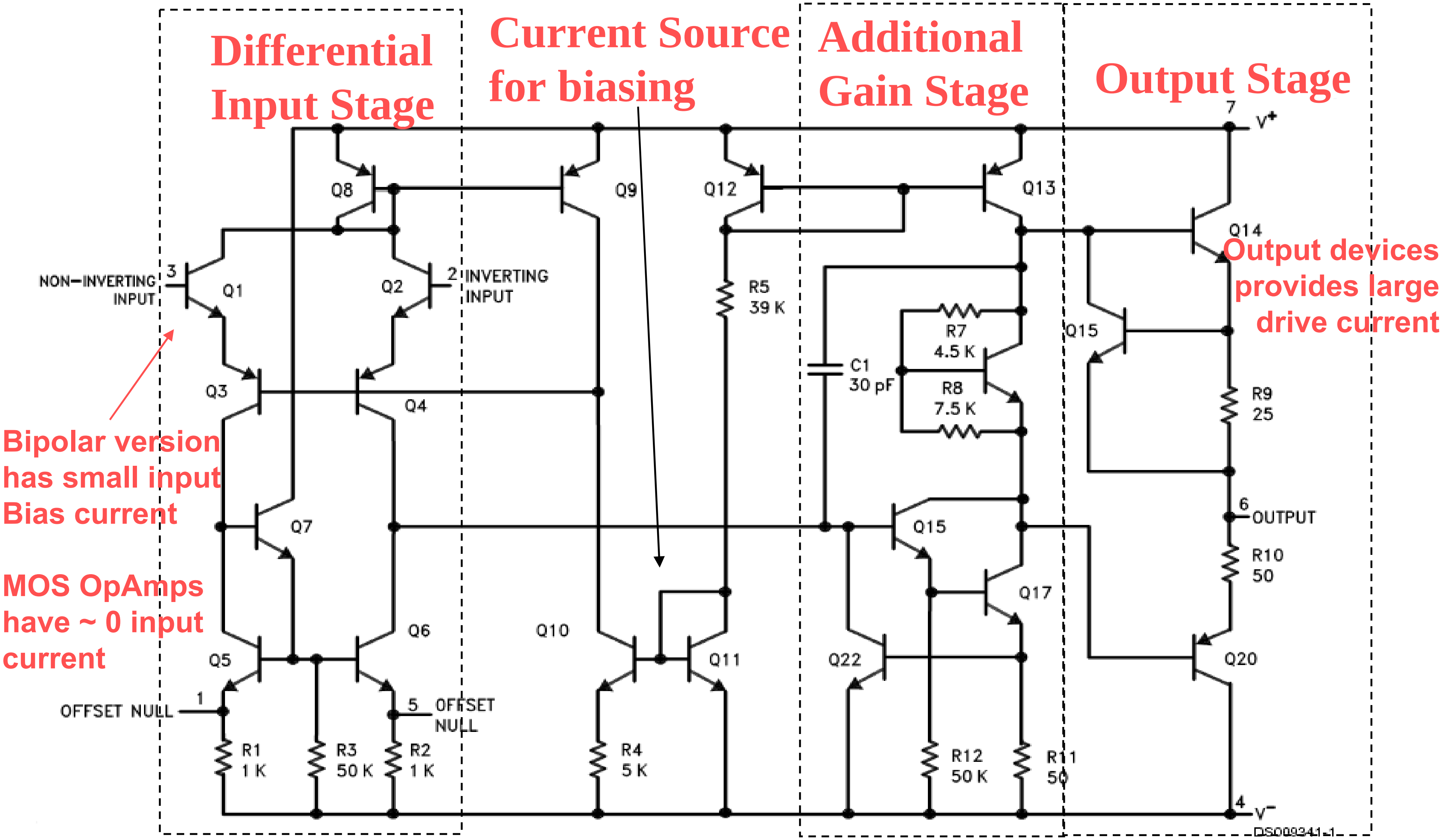
$$V_{out} = a(f) \cdot V_{in}$$



LM741 Pinout

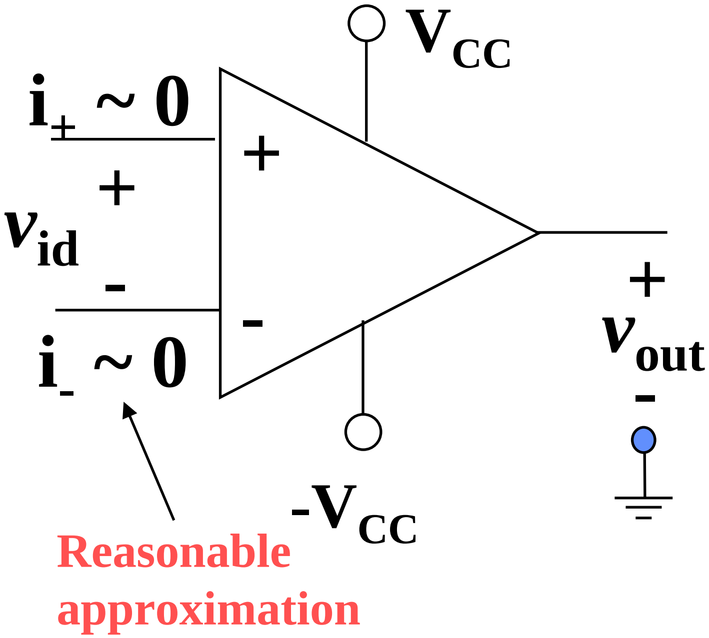


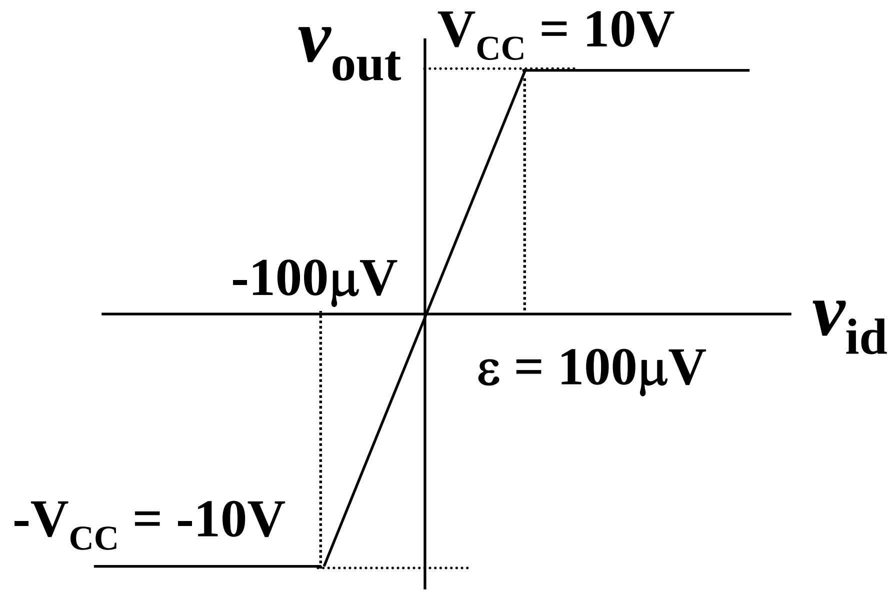
The Inside of a 741 OpAmp



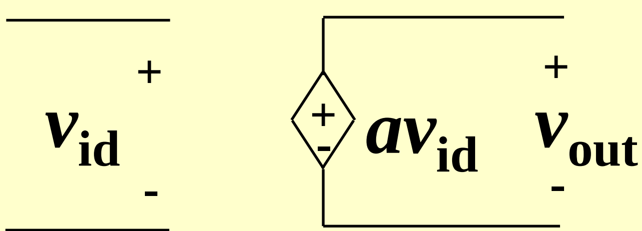
Gain is Sensitive to Operating Condition
(e.g., Device, Temperature, Power supply voltage, etc.)

Simple Model for an OpAmp



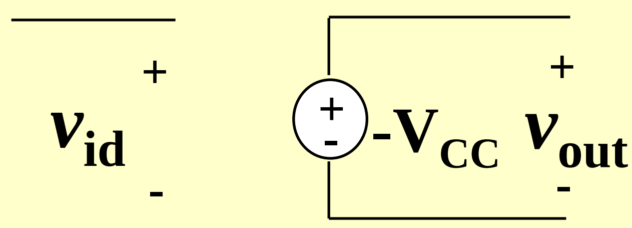


Linear Mode



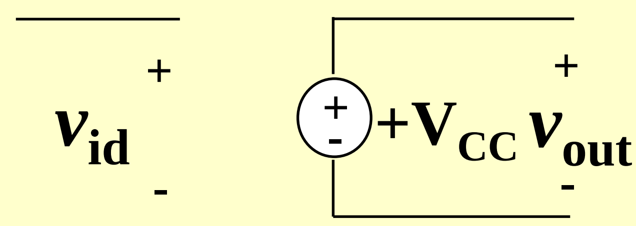
If $-V_{CC} < v_{out} < V_{CC}$

Negative Saturation



$v_{id} < -\varepsilon$

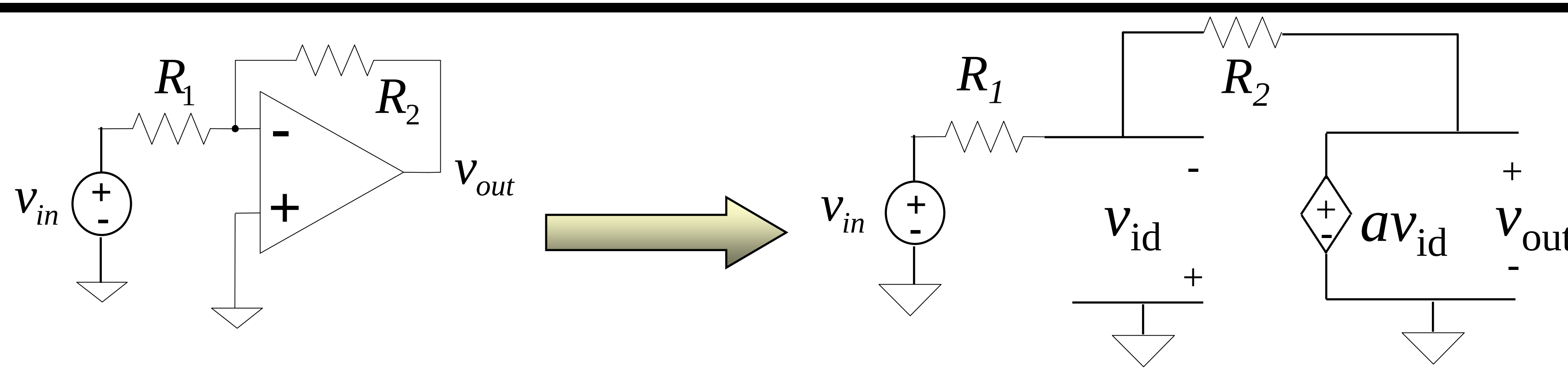
Positive Saturation



$v_{id} > \varepsilon$

Small input range for "Open" loop Configuration

The Power of (Negative) Feedback



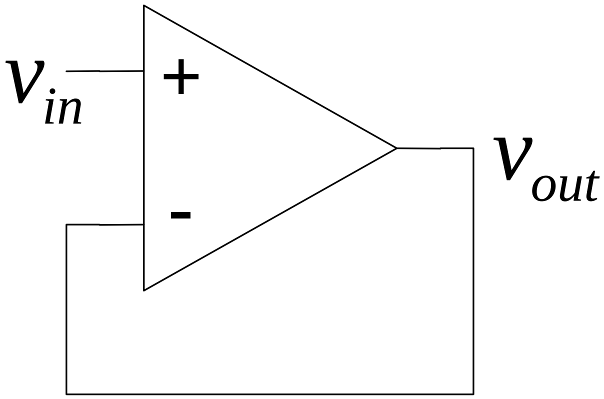
$$\frac{v_{in} + v_{id}}{R_1} + \frac{v_{out} + v_{id}}{R_2} = 0 \qquad v_{id} = \frac{v_{out}}{a} \qquad \frac{v_{in}}{R_1} = -\frac{v_{out}}{a}\left[\frac{1}{R_1} + \frac{a}{R_2} + \frac{1}{R_2}\right]$$

$$\frac{v_{out}}{v_{in}} = -\frac{R_2 a}{(1+a)R_1 + R_2} \approx -\frac{R_2}{R_1} \left(if \quad a >> 1\right)$$

- **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**

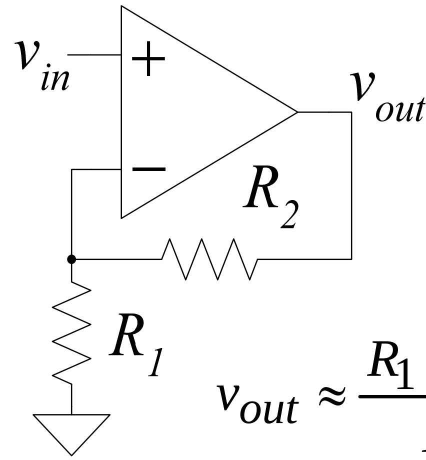
Basic OpAmp Circuits

Voltage Follower (buffer)



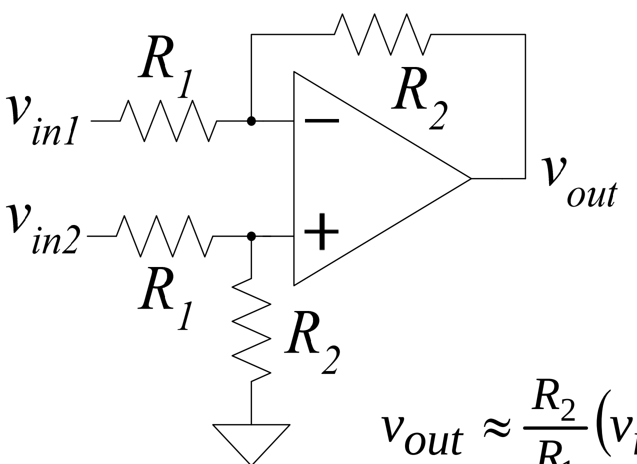
$$v_{out} \approx v_{in}$$

Non-inverting



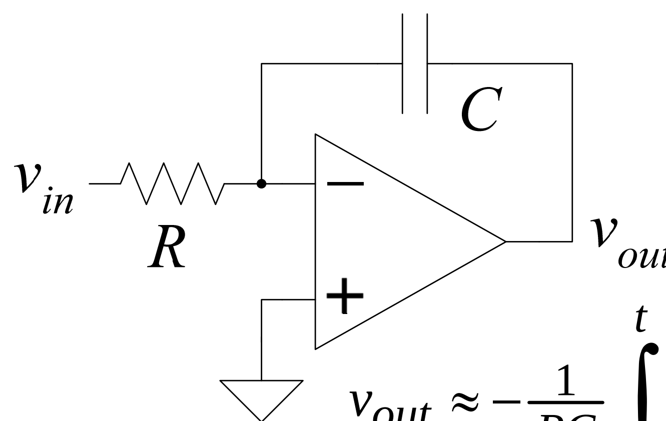
$$v_{out} \approx \frac{R_1 + R_2}{R_1} v_{in}$$

Differential Input



$$v_{out} \approx \frac{R_2}{R_1}\left(v_{in2} - v_{in1}\right)$$

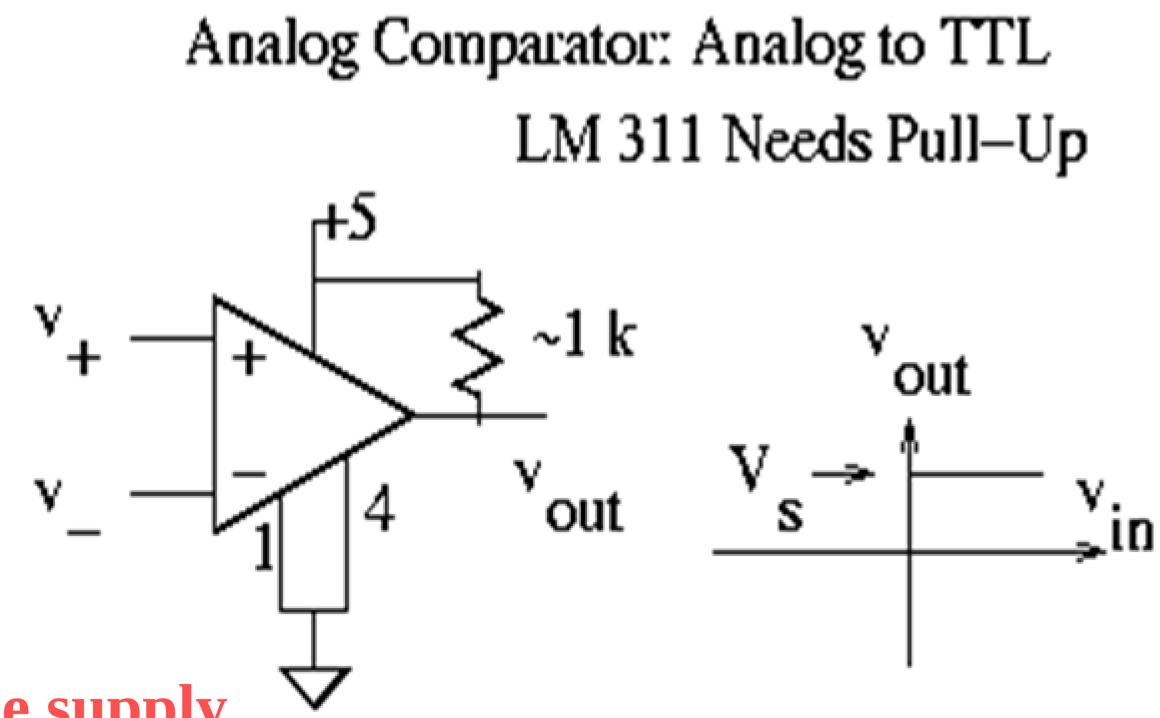
Integrator



$$v_{out} \approx -\frac{1}{RC}\int_{-\infty}^{t} v_{in}\,dt$$

Analog Comparator:

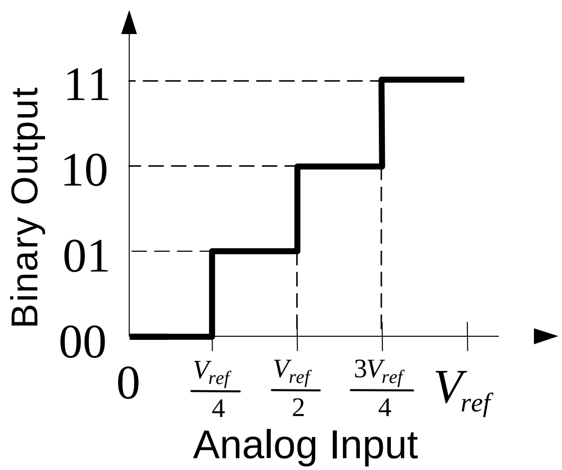
Is V+ > V- ?
The Output is a DIGITAL signal



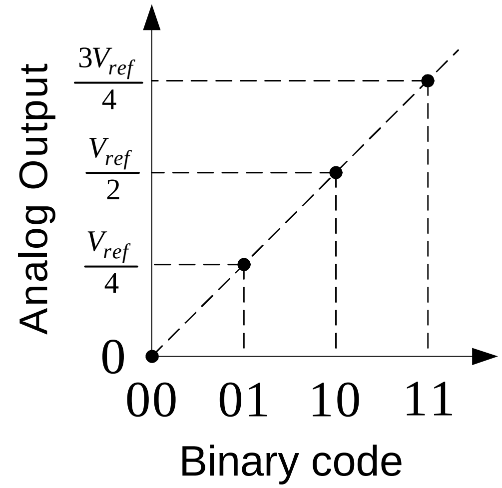
LM311 is a single supply comparator

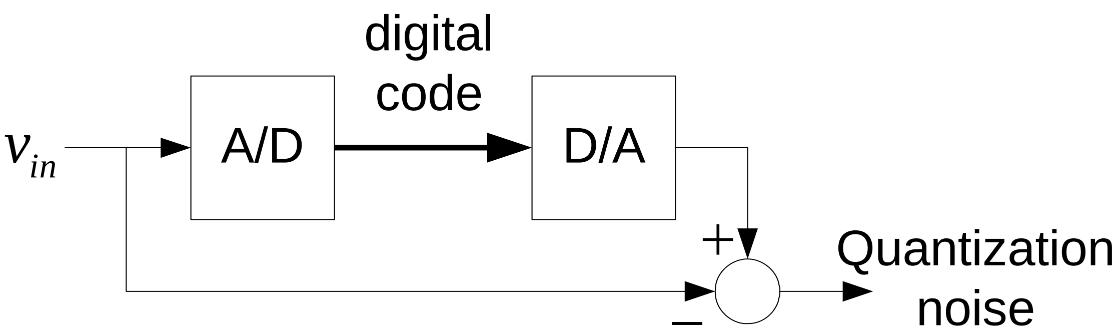
Data Conversion: Quantization Noise

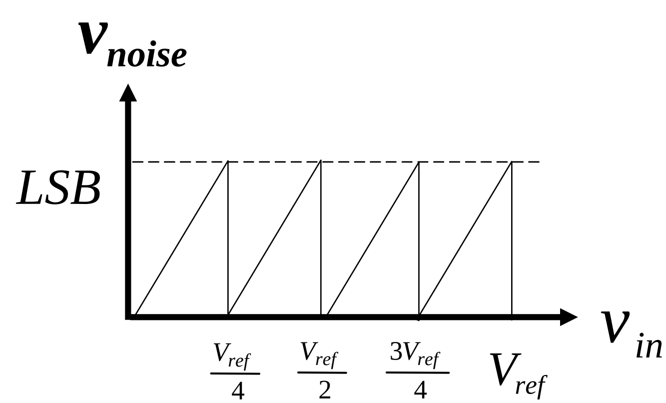
A/D Conversion



D/A Conversion



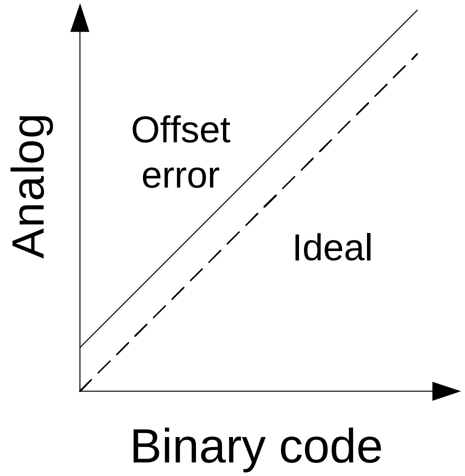




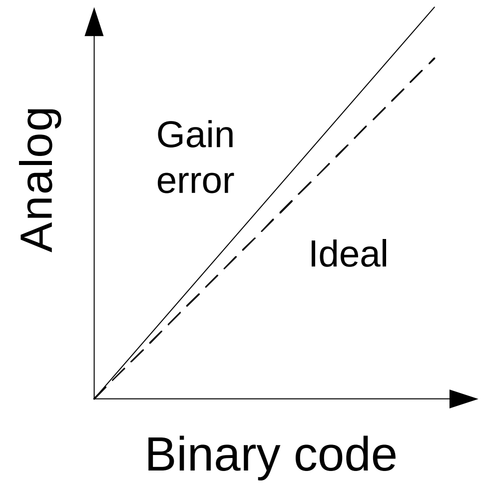
- **Quantization noise exists even with *ideal* A/D and D/A converters**

Non-idealities in Data Conversion

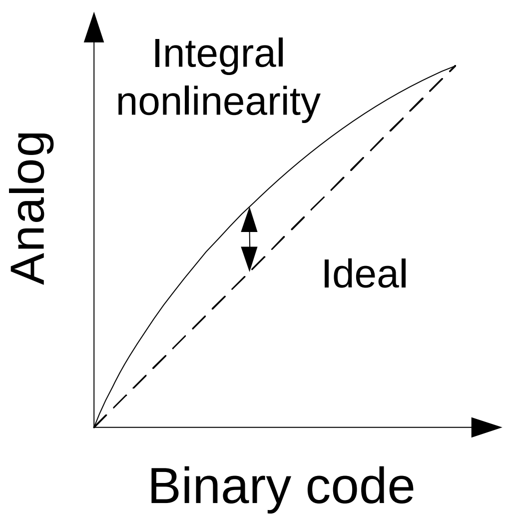
Offset – a constant voltage offset that appears at the output when the digital input is 0



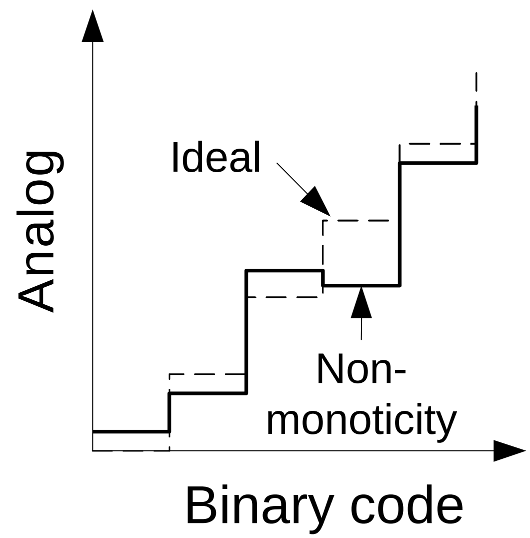
Gain error – deviation of slope from ideal value of 1



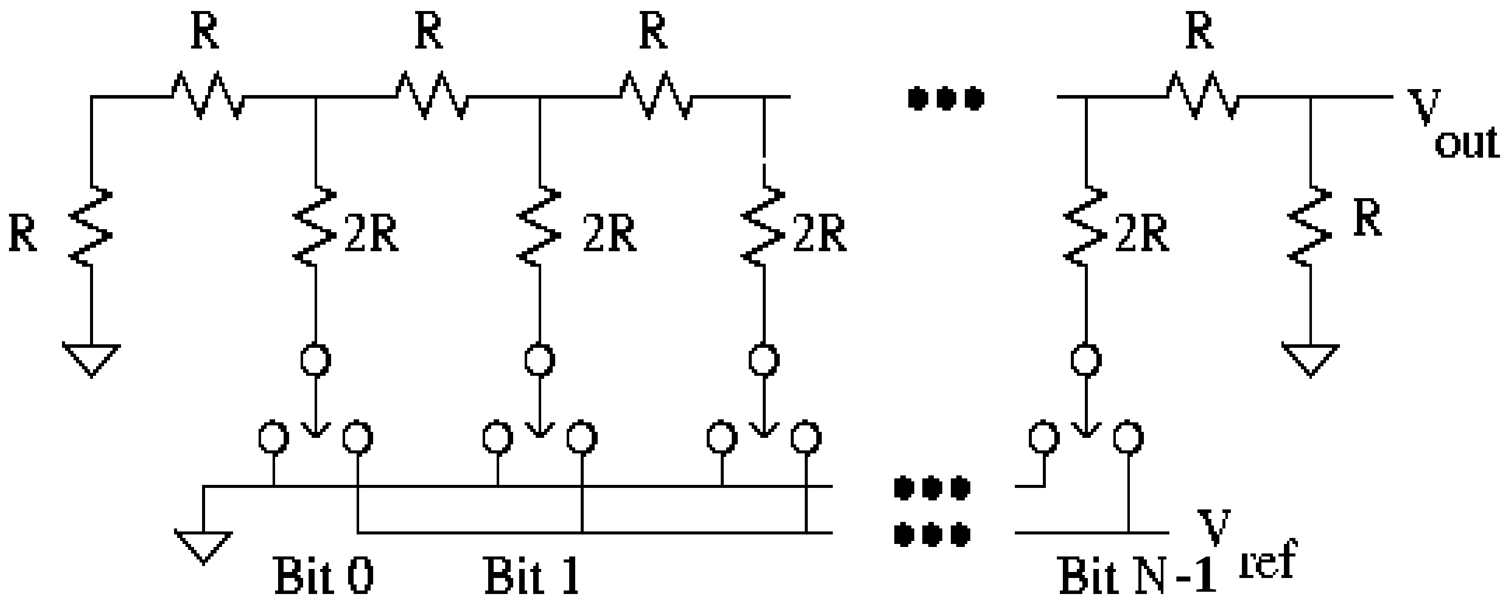
Integral Nonlinearity – maximum deviation from the ideal analog output voltage



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



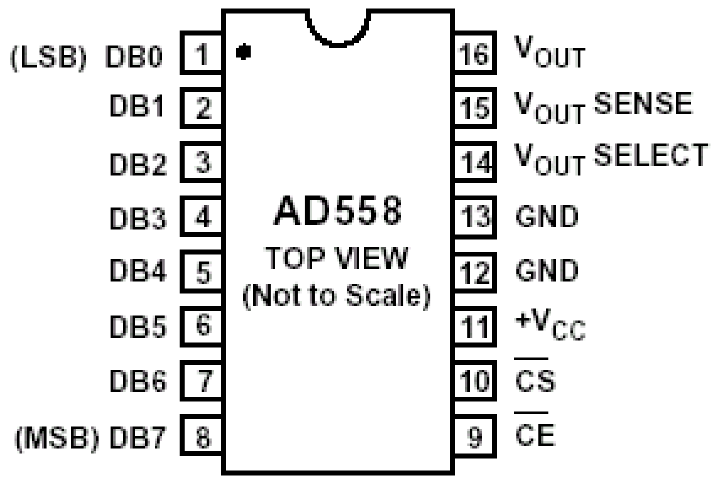
R-2R Ladder DAC Architecture



$$V_{out} = \frac{1}{6} V_{ref} \left[B_7 + \frac{1}{2} B_6 + \frac{1}{4} B_5 + \cdots + \frac{1}{128} B_0 \right]$$

- 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

DAC (AD 558) Specs



Input Logic Coding

Digital Input Code			Output Voltage	
Binary	**Hexadecimal**	**Decimal**	**2.56 V Range**	**10.000 V Range**
0000 0000	00	0	0	0
0000 0001	01	1	0.010 V	0.039 V
0000 0010	02	2	0.020 V	0.078 V
0000 1111	0F	15	0.150 V	0.586 V
0001 0000	10	16	0.160 V	0.625 V
0111 1111	7F	127	1.270 V	4.961 V
1000 0000	80	128	1.280 V	5.000 V
1100 0000	C0	192	1.920 V	7.500 V
1111 1111	FF	255	2.55 V	9.961 V

- 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

Chip Architecture and Interface

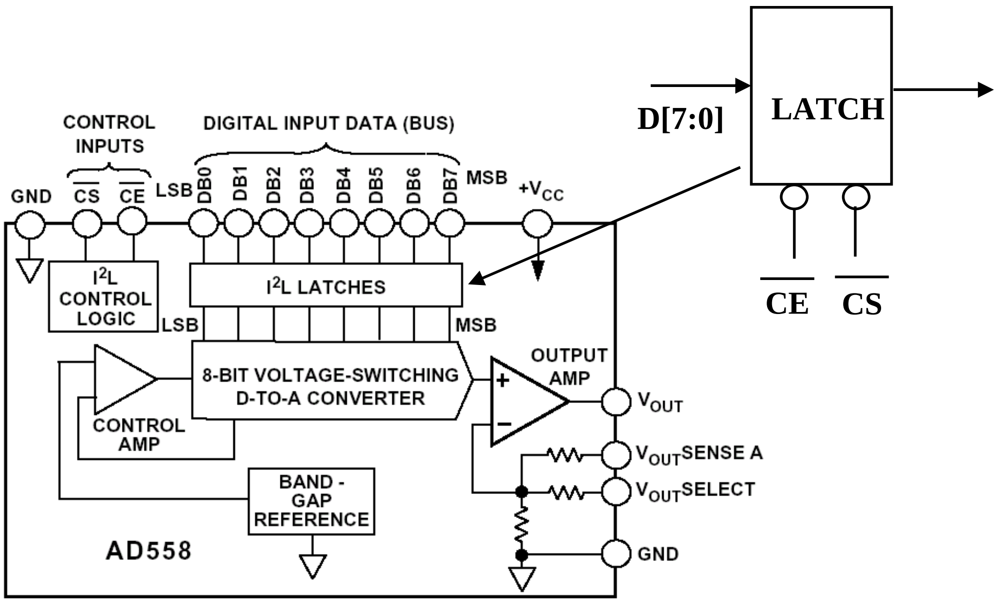
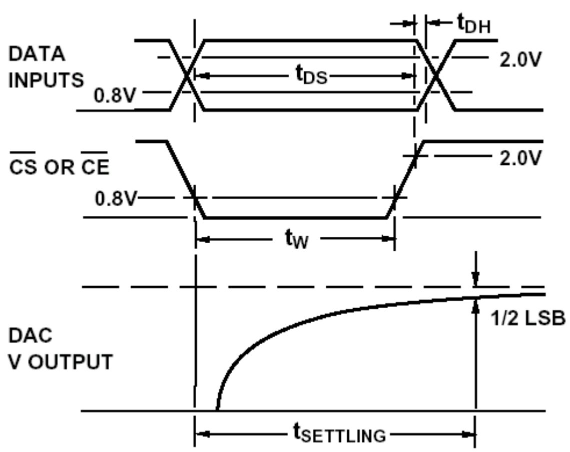


Table I. AD558 Control Logic Truth Table

Input Data	$\overline{CE}$	$\overline{CS}$	DAC Data	Latch Condition
0	0	0	0	"Transparent"
1	0	0	1	"Transparent"
0	g	0	0	Latching
1	g	0	1	Latching
0	0	g	0	Latching
1	0	g	1	Latching
X	1	X	Previous Data	Latched
X	X	1	Previous Data	Latched

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



t_W = STORAGE PULSE WIDTH = 200ns MIN
t_{DH} = DATA HOLD TIME = 10ns MIN
t_{DS} = DATA SETUP TIME = 200ns MIN
$t_{SETTLING}$ = DAC OUTPUT SETTLING TIME TO ±1/2 LSB

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

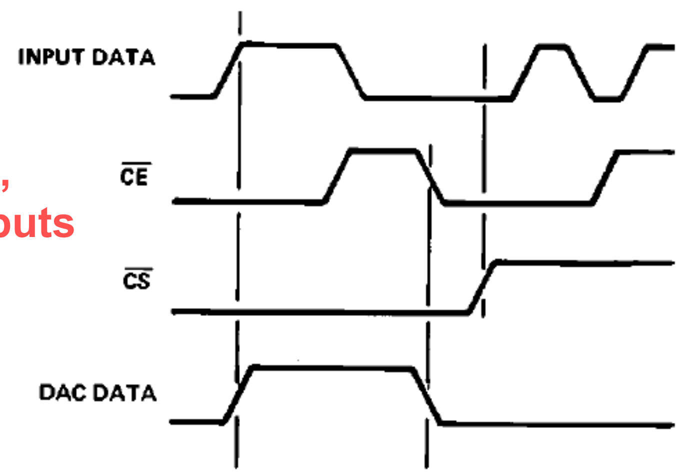
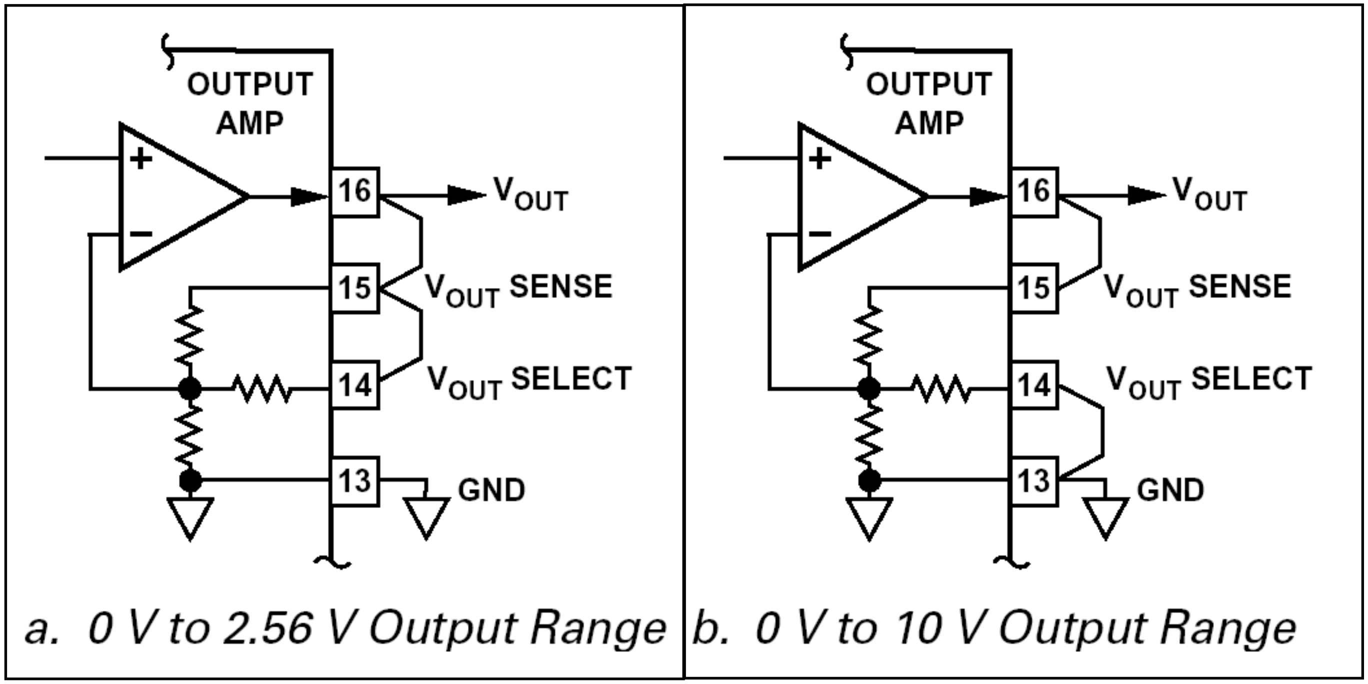


Figure 6. AD558 Control Logic Function

Setting the Voltage Range



Very similar to a non-inverting amp

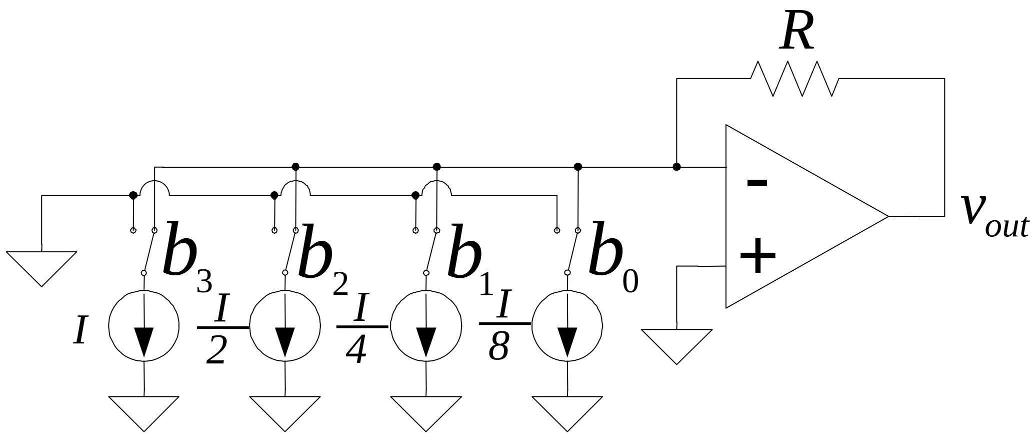
Strap output for different voltage ranges

Input Logic Coding

Digital Input Code			Output Voltage	
Binary	**Hexadecimal**	**Decimal**	**2.56 V Range**	**10.000 V Range**
0000 0000	00	0	0	0
0000 0001	01	1	0.010 V	0.039 V
0000 0010	02	2	0.020 V	0.078 V
0000 1111	0F	15	0.150 V	0.586 V
0001 0000	10	16	0.160 V	0.625 V
0111 1111	7F	127	1.270 V	4.961 V
1000 0000	80	128	1.280 V	5.000 V
1100 0000	C0	192	1.920 V	7.500 V
1111 1111	FF	255	2.55 V	9.961 V

Convert data to Offset binary

Another Approach: Binary-Weighted DAC

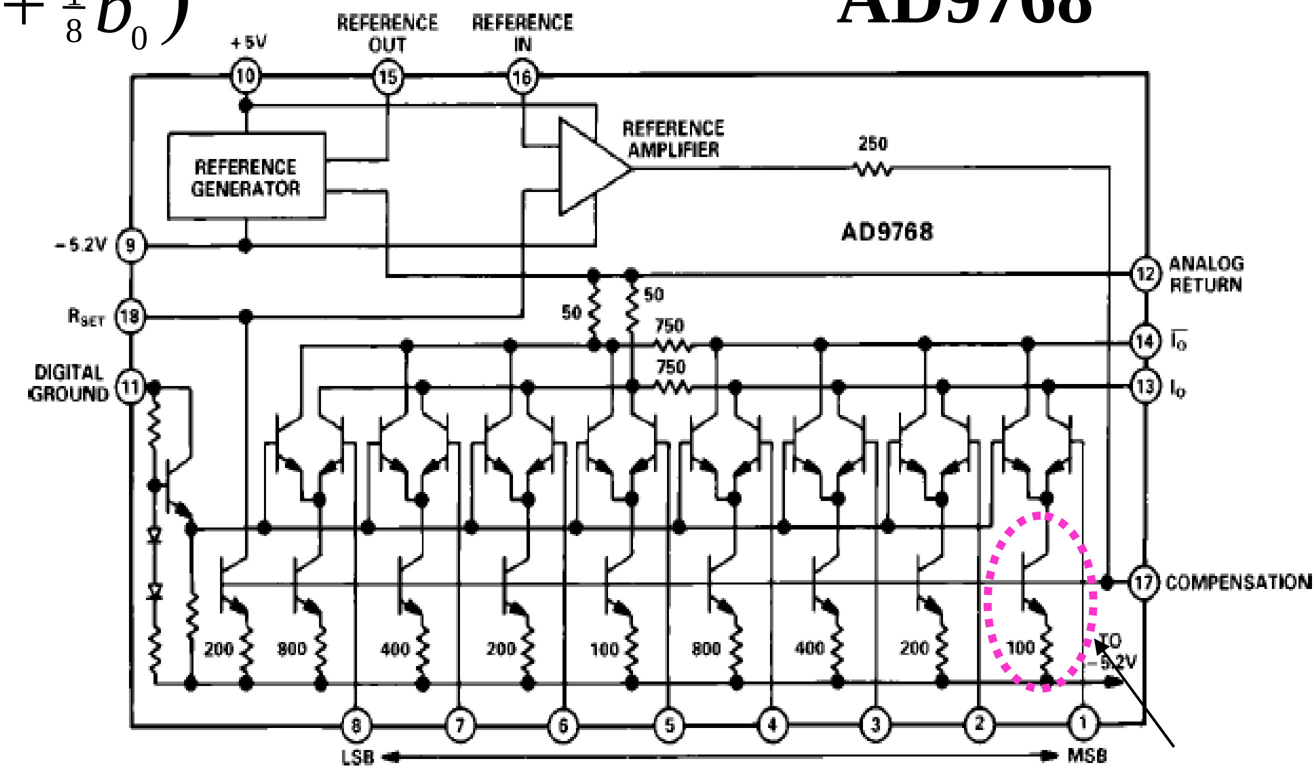


- Switch binary-weighted currents
- MSB to LSB current ratio is 2^N

$$v_{out} = -IR\left(b_3 + \tfrac{1}{2}b_2 + \tfrac{1}{4}b_1 + \tfrac{1}{8}b_0\right)$$

- Analog Devices AD9768 uses two banks of ratioed currents
- Additional current division performed by 750 Ω resistor between the two banks

AD9768

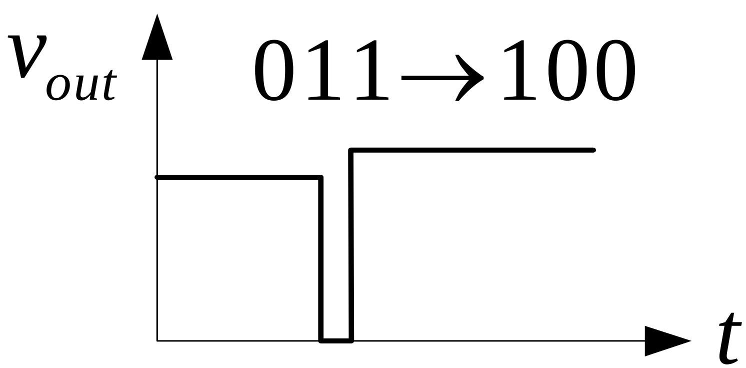


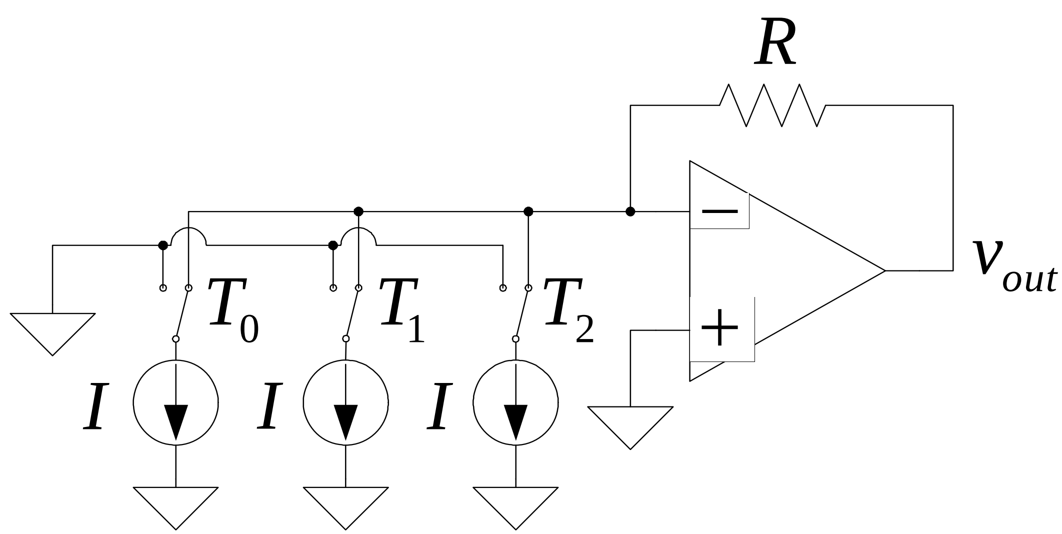
Reference current source

Glitching and Thermometer D/A

- **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**

Binary		Thermometer		
0	0	0	0	0
0	1	0	0	1
1	0	0	1	1
1	1	1	1	1

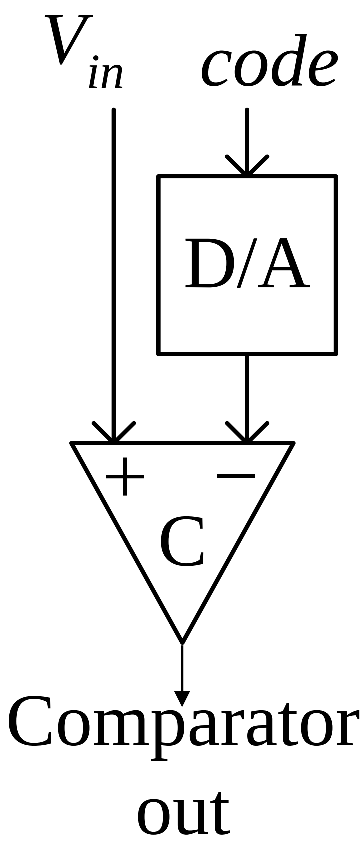


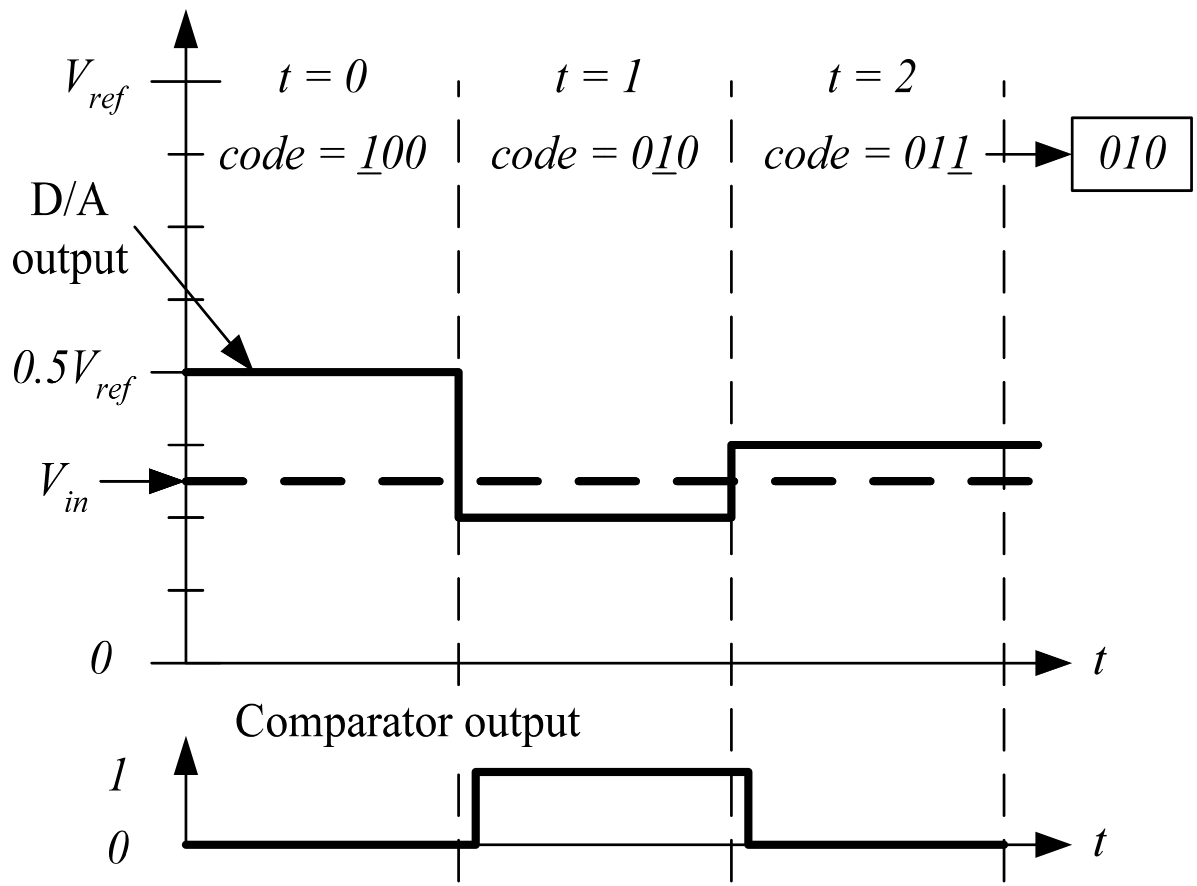


$$v_{out} = -IR(T_0 + T_1 + T_2)$$

Successive-Approximation A/D

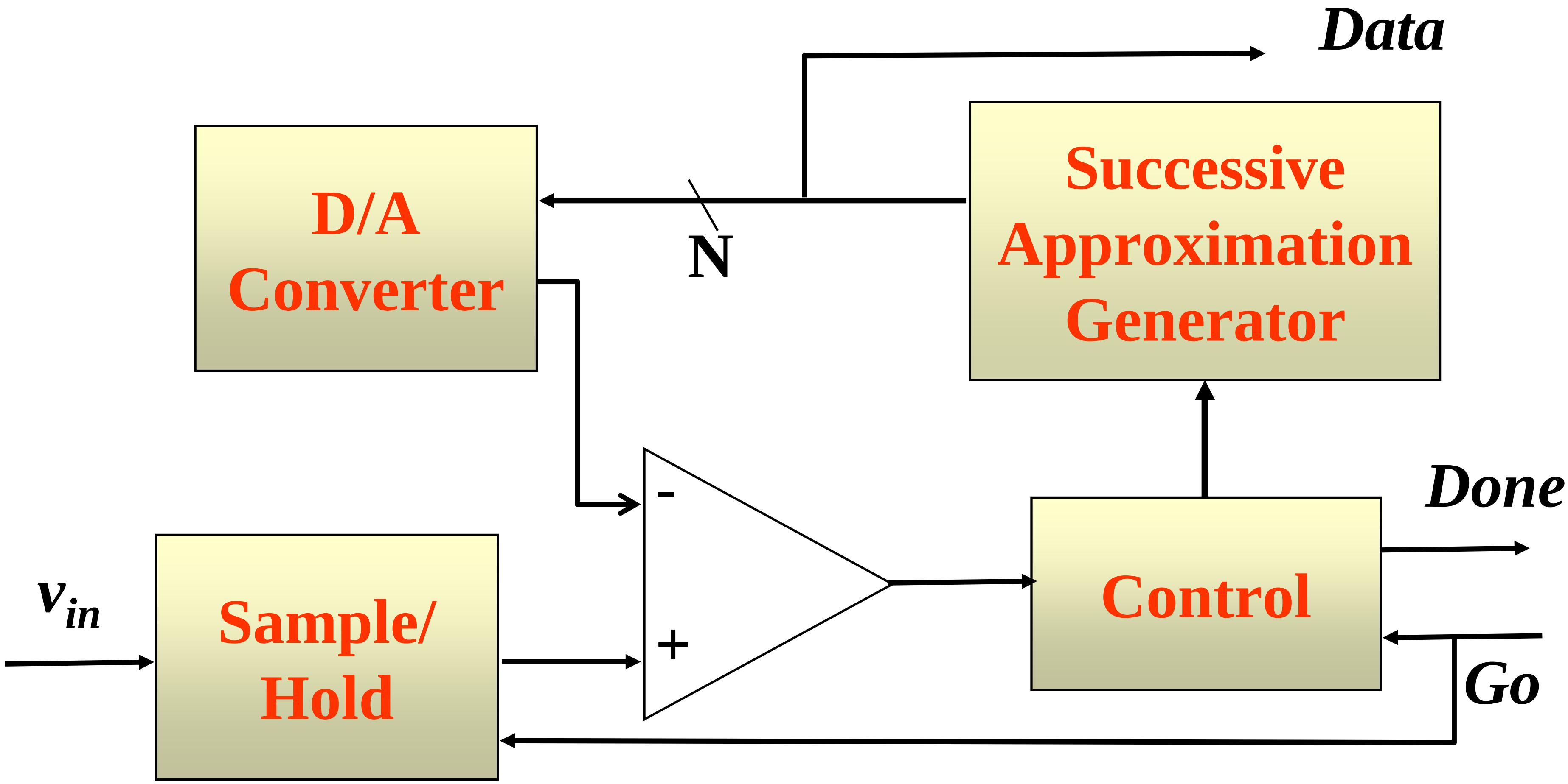
- D/A converters are typically compact and easier to design. Why not A/D convert using a D/A converter and a comparator?
- D to A generates analog voltage which is compared to the input voltage
- If D to A voltage > input voltage then set that bit; otherwise, reset that bit
- This type of A to D takes a fixed amount of time proportional to the bit length





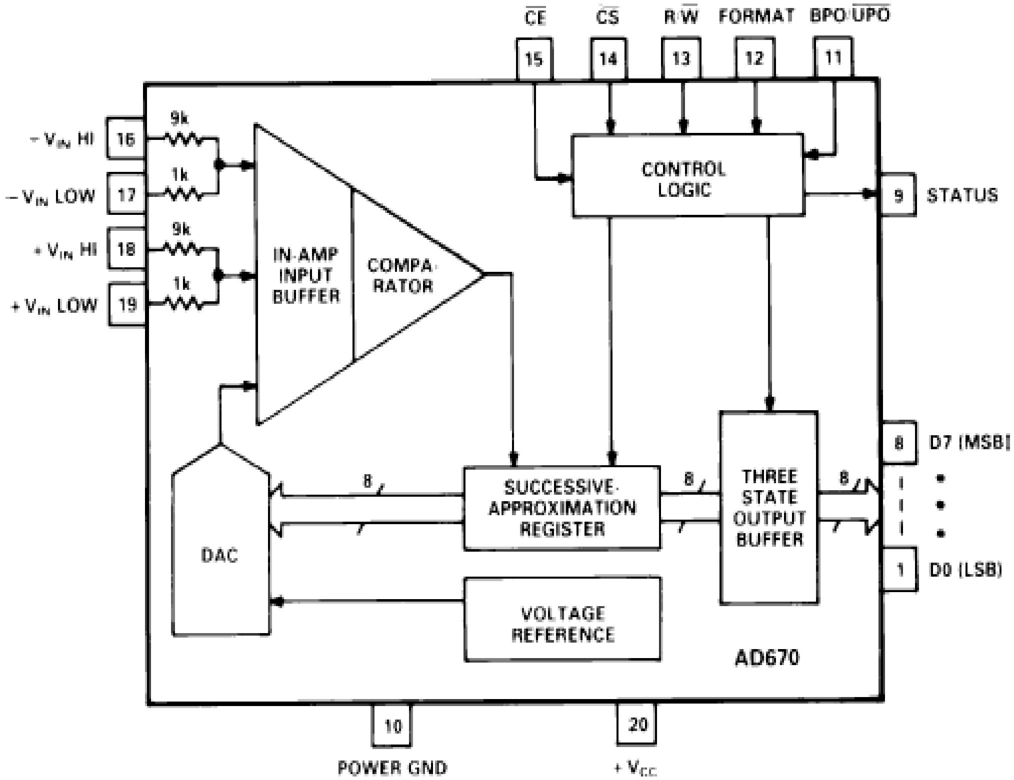
Example: 3-bit A/D conversion, 2 LSB < V_{in} < 3 LSB

Successive-Approximation A/D



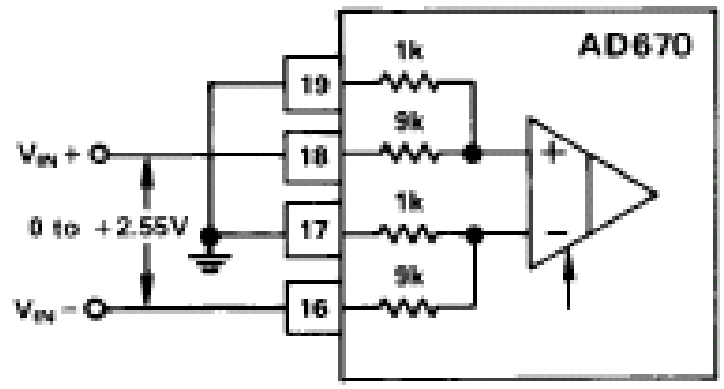
- **Serial conversion takes a time equal to $N(t_{D/A} + t_{comp})$**

Successive-Approximation A/D (AD670)



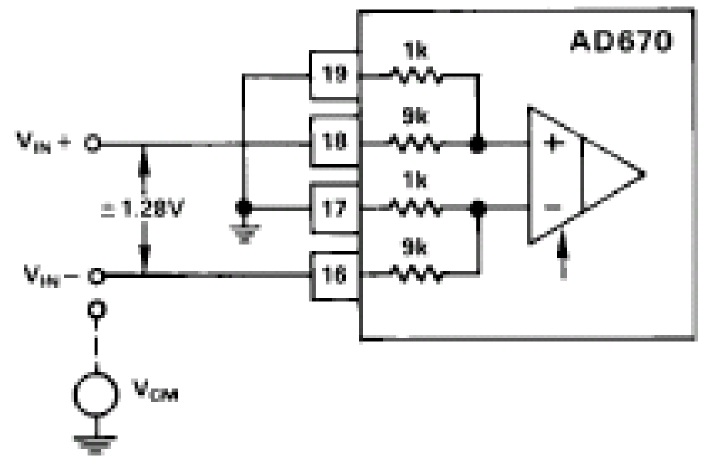
BPO/$\overline{\text{UPO}}$	FORMAT	INPUT RANGE/ OUTPUT FORMAT
0	0	Unipolar/Straight Binary
1	0	Bipolar/Offset Binary
0	1	Unipolar/2s Complement
1	1	Bipolar/2s Complement

Unipolar (BPO =0)



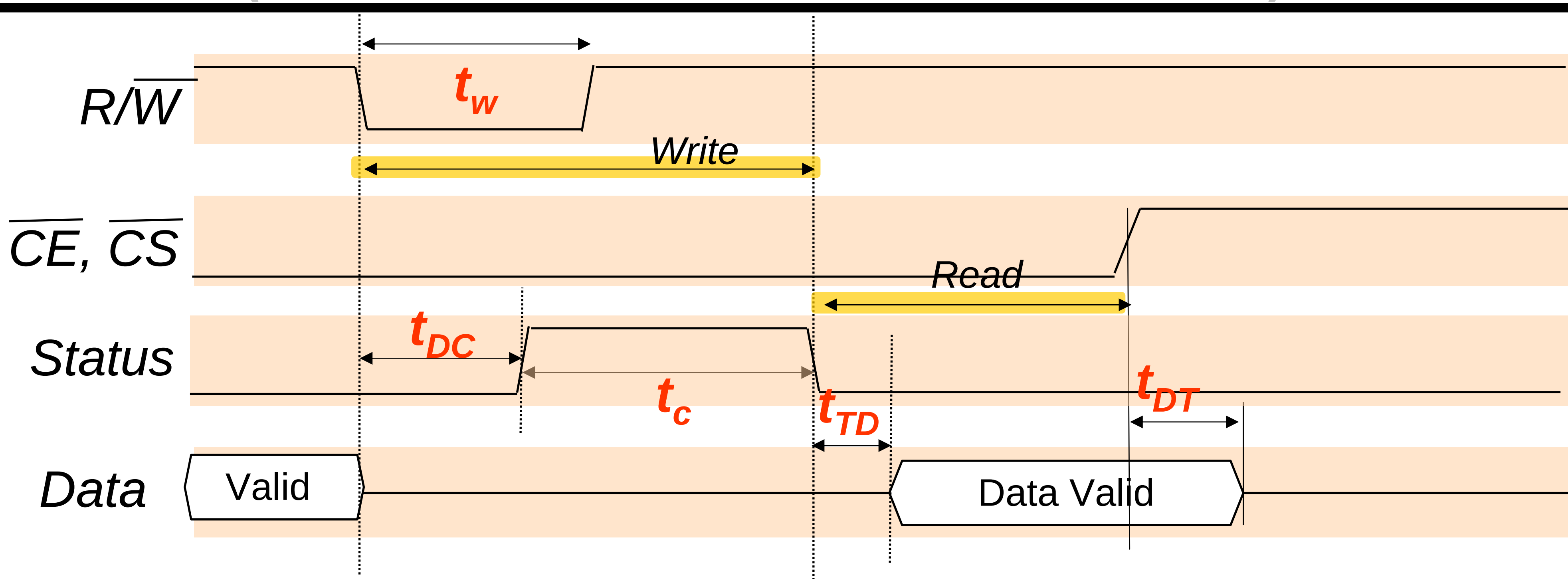
2a. 0 V to 2.55 V (10 mV/LSB)

Bipolar (BPO =1)



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)

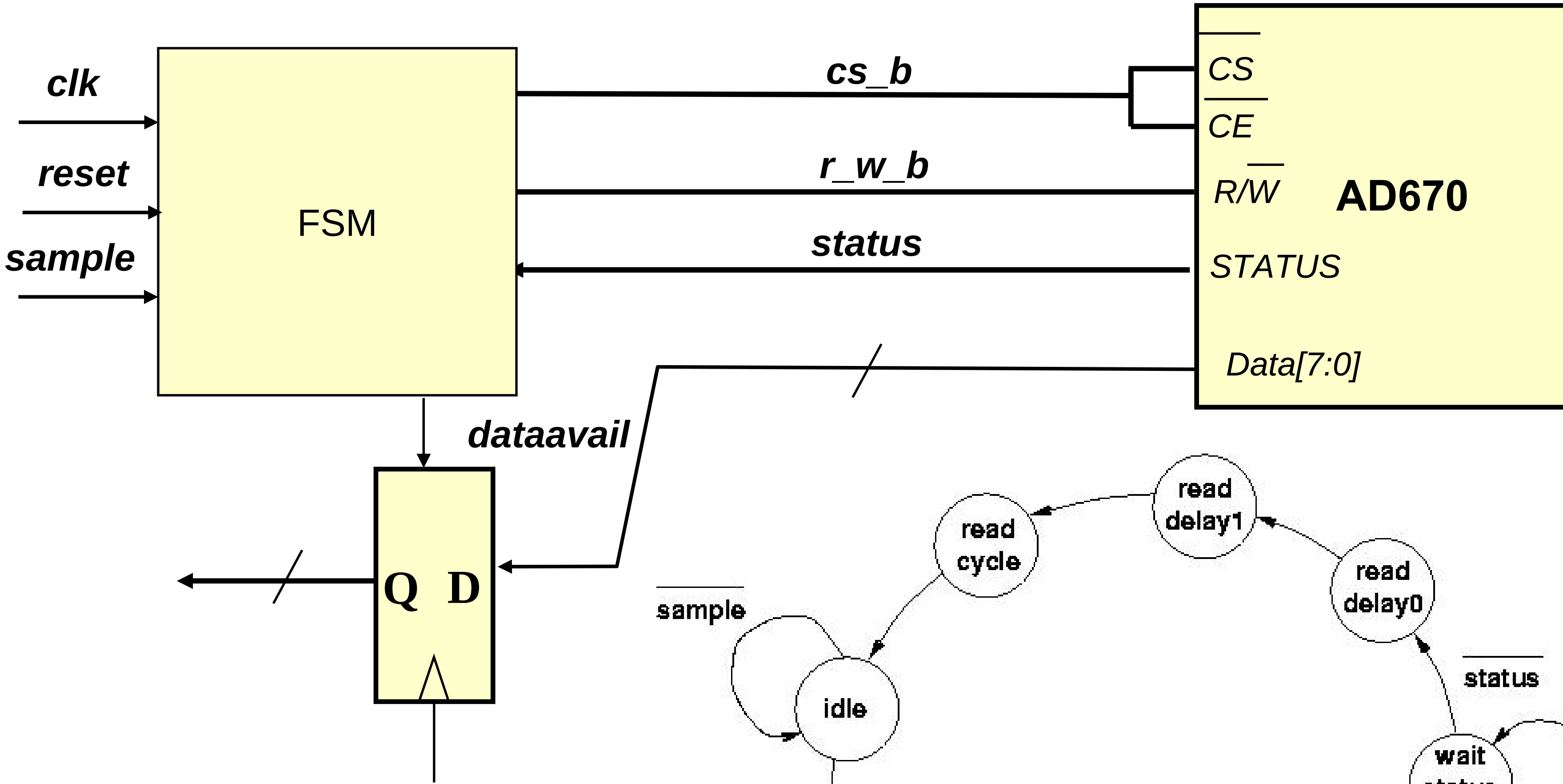
t_c (conversion time) = 10μs (max)

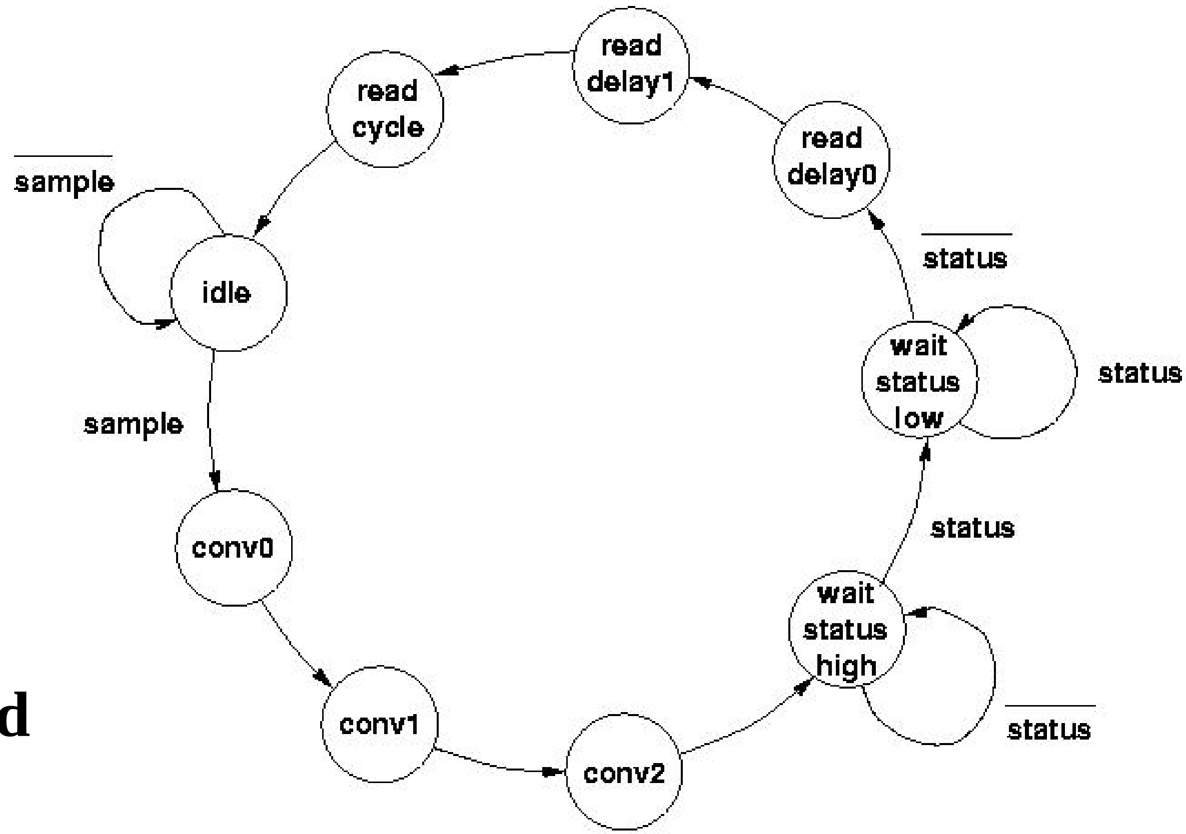
t_{TD} (Bus Access Time) = 250 (max)

t_{DT} (Output Float Delay) = 150 (max)

- Control bits $\overline{CE}$ and $\overline{CS}$ can be wired to ground if A/D is the only chip driving the bus
- Suggestion: tie $\overline{CE}$ and $\overline{CS}$ pins together and hardwire BPO and Format

Simple A/D Interface FSM





Status should be synchronized: why?

Courtesy of James Oey and Cemal Akcaba

Example A/D Verilog Interface

```
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;
```

1/5

```
// 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
```

2/5

Example A/D Verilog Interface (cont.)

```
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
```

3/5

```
            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
```

4/5

Example A/D Verilog Interface(cont.)

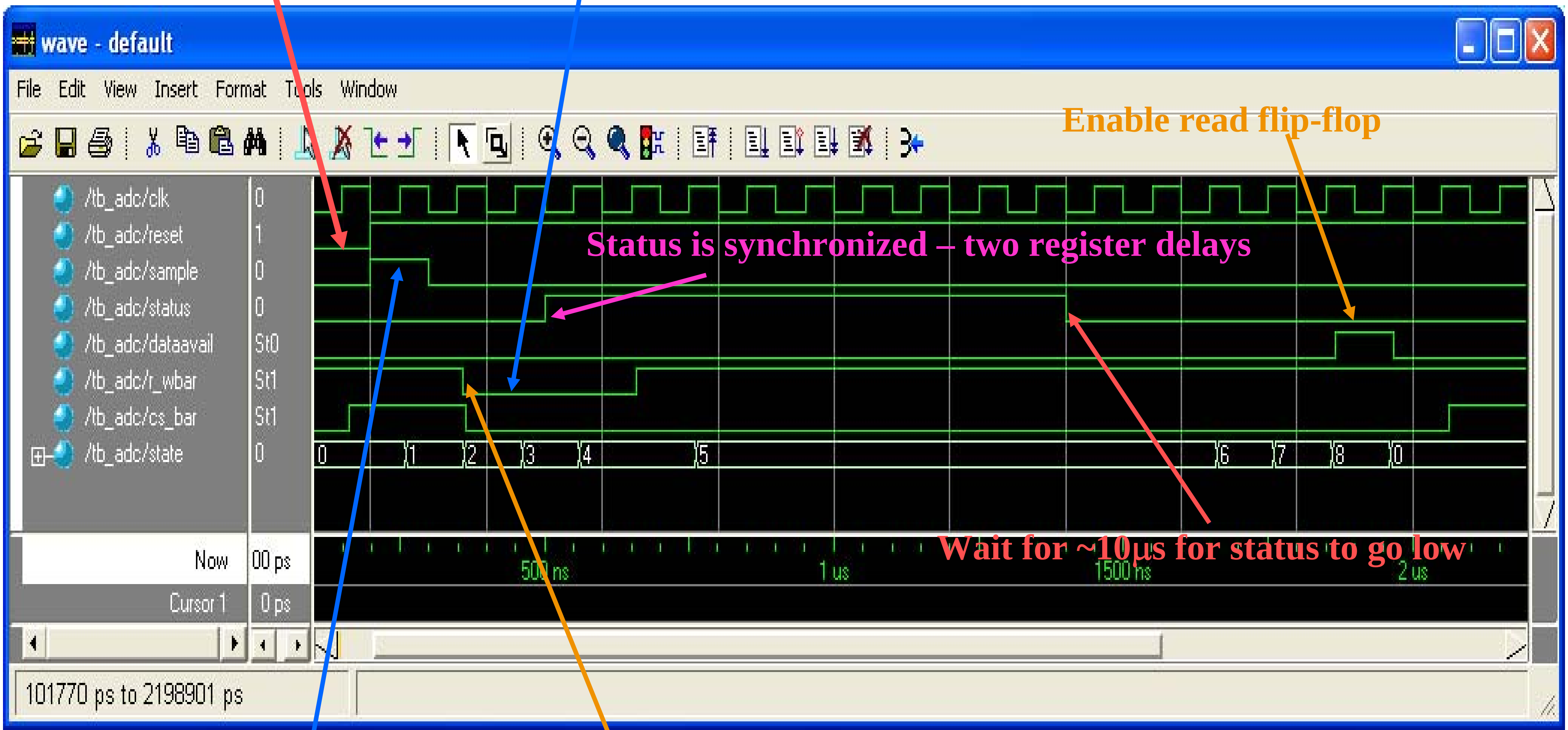
```
        READDELAY0:
          begin
            cs_bar_int = 0;
            nextstate = READDELAY1;
          end

        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

Simulation

On reset, present state goes to 0

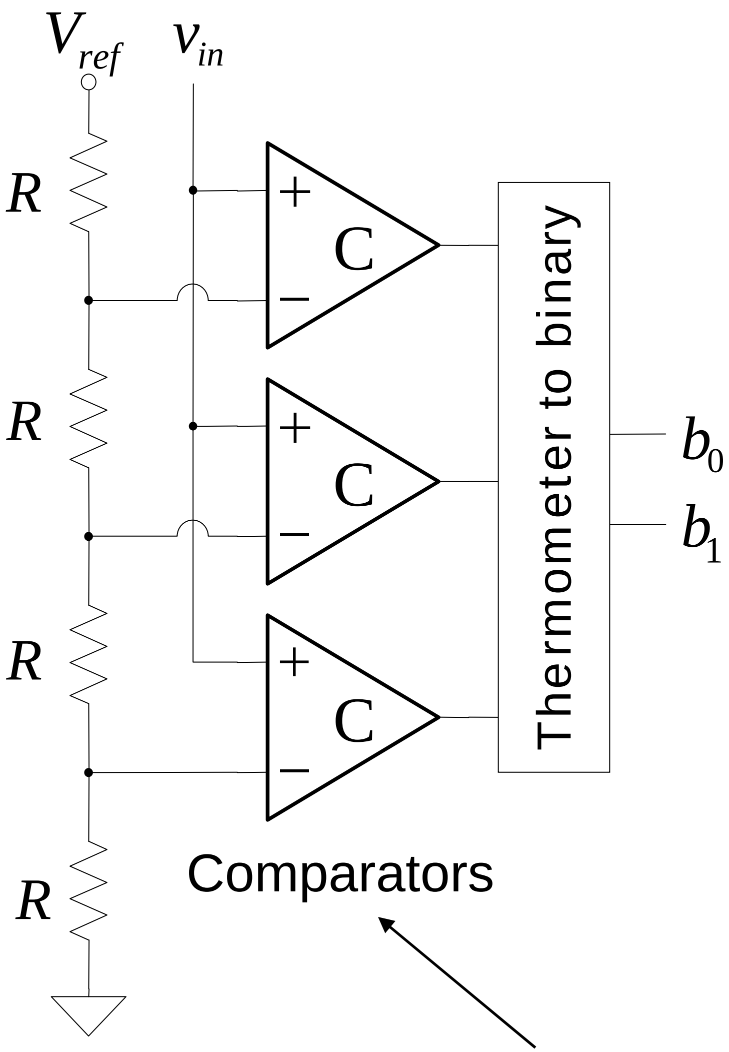
r_w_b must stay low for at least 3 cycles (@ 100ns period)



Sample pulse initiates data conversion

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

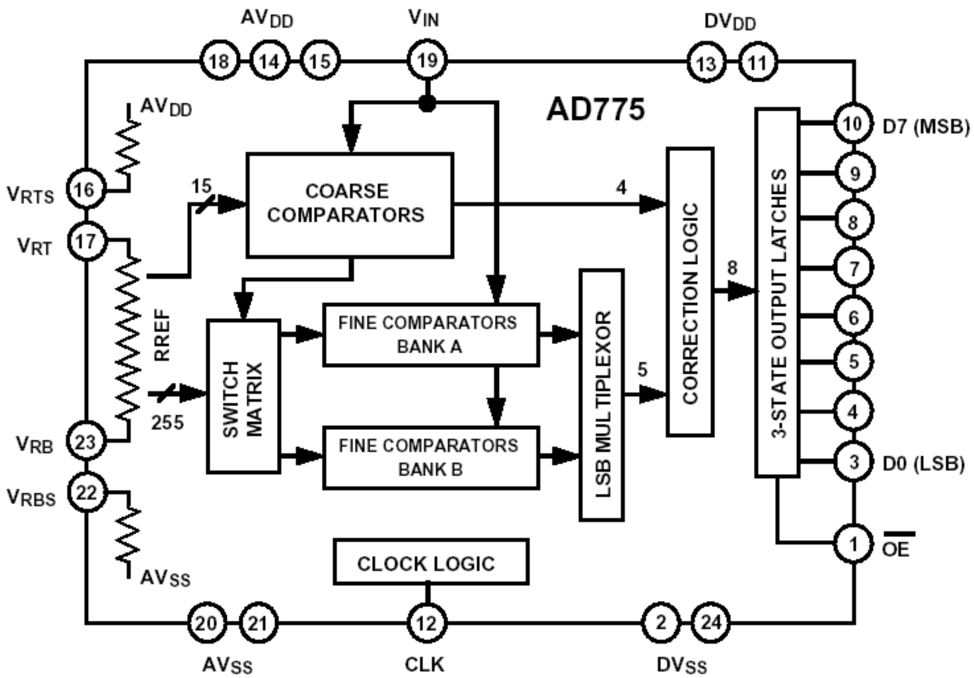
Flash A/D Converter

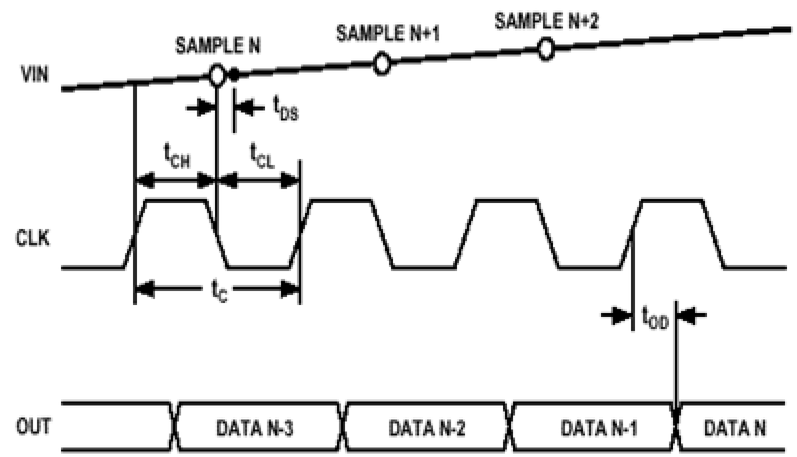


- Brute-force A/D conversion
- Simultaneously compare the analog value with every possible reference value
- Fastest method of A/D conversion
- Size scales exponentially with precision (requires 2^N comparators)

Can be implemented as OpAmp in open loop

AD 775 – Flash Data Converter



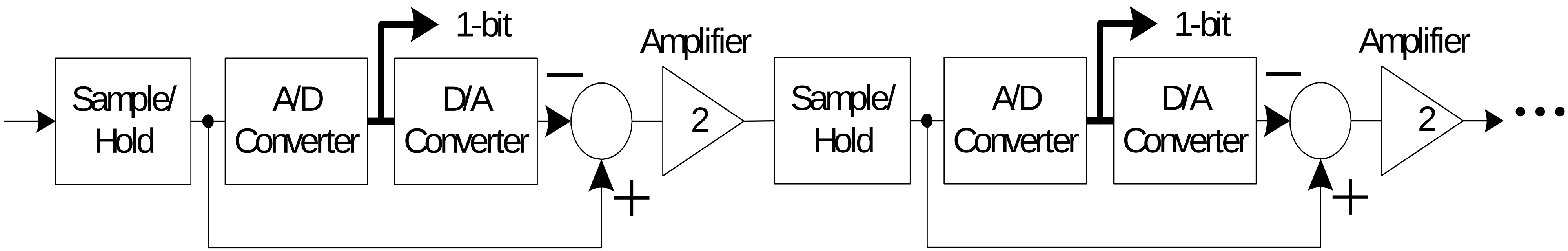


TIMING SPECIFICATIONS

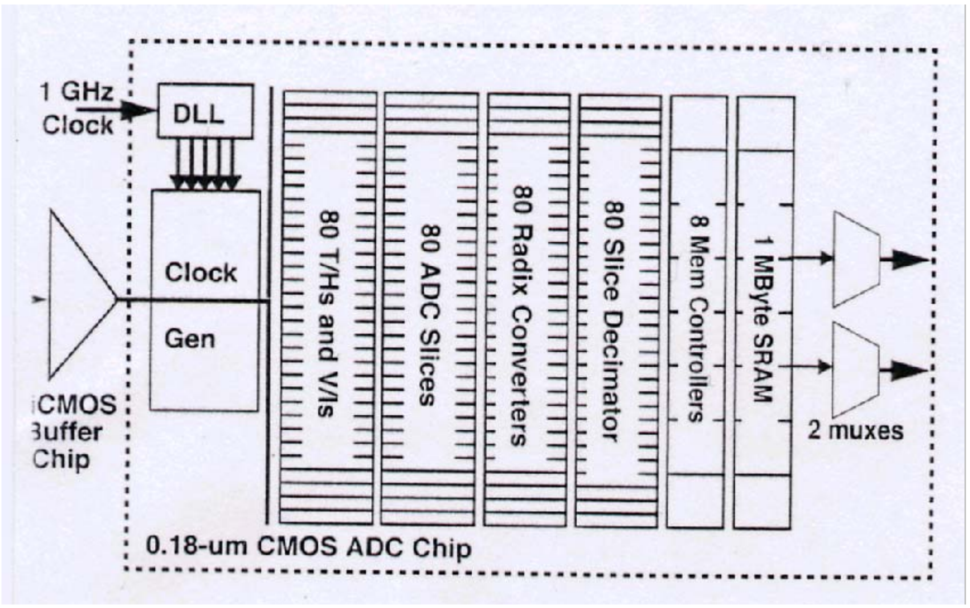
	Symbol	Min	Typ	Max	Units
Maximum Conversion Rate		20	35		MHz
Clock Period	t_C	50			ns
Clock High	t_{CH}	25			ns
Clock Low	t_{CL}	25			ns
Output Delay	t_{OD}		18	30	ns
Pipeline Delay (Latency)				2.5	Clock Cycles
Sampling Delay	t_{DS}		4		ns
Aperture Jitter			30		ps

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

Summary of Analog Blocks

- 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/$\overline{W}$, etc.)
- Analog design incorporate digital principles
  - Glitch free operation using coding
  - Parallelism and Pipelining!
  - More advanced concepts such as calibration