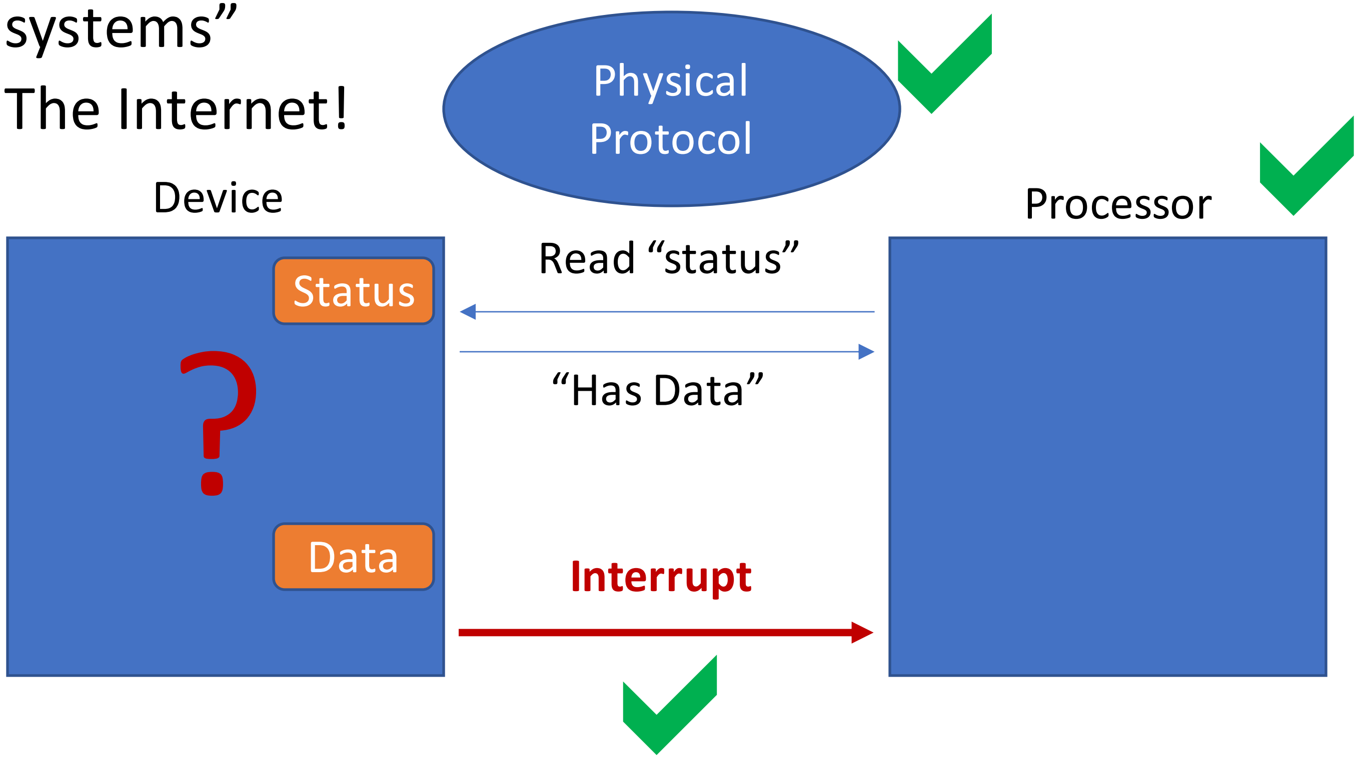
I/O Devices – ADC/DAC

EECS 388 – Fall 2022

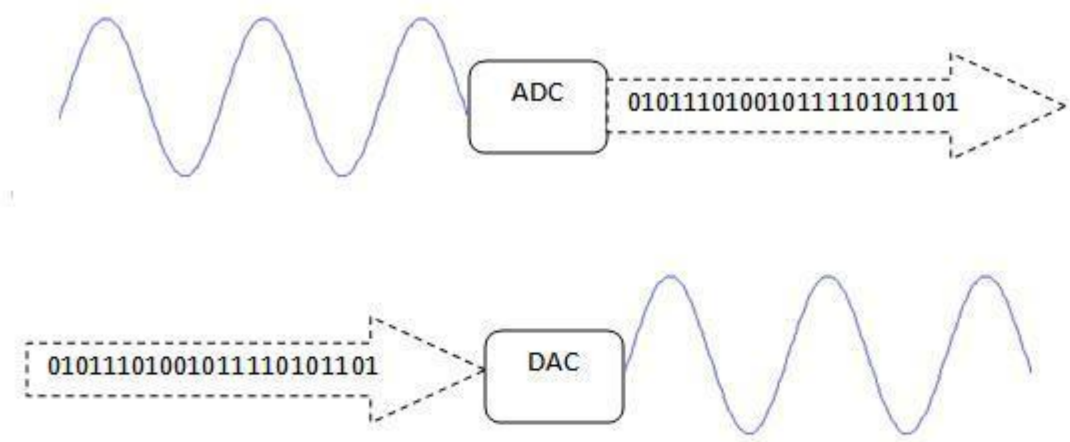
Context

- Recommended reading:
  - Datasheet of the devices
  - Chapter 13 of “AVR Microcontroller and Embedded systems”
  - The Internet!

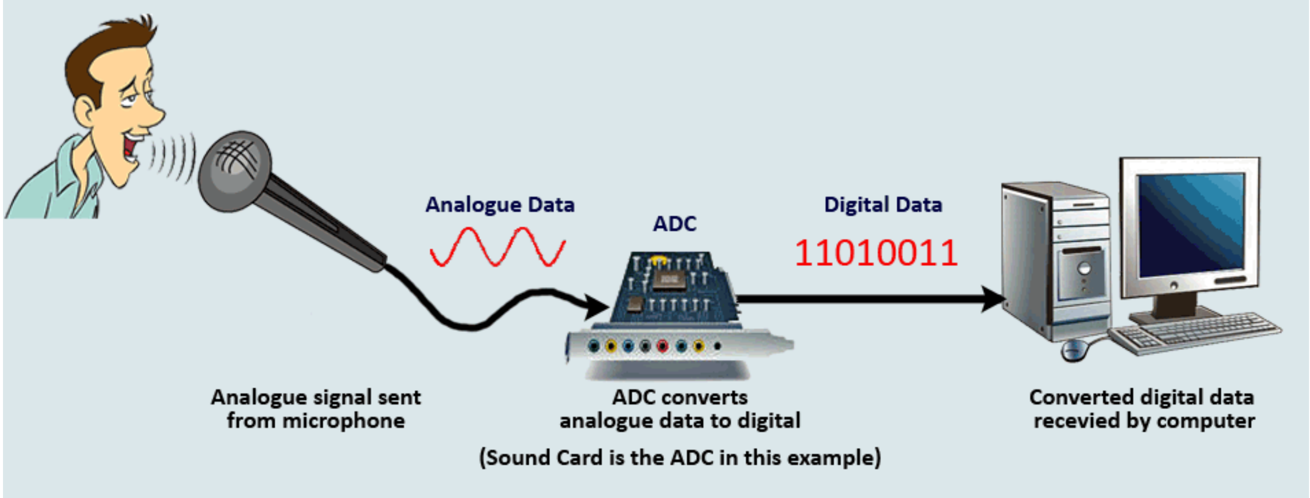


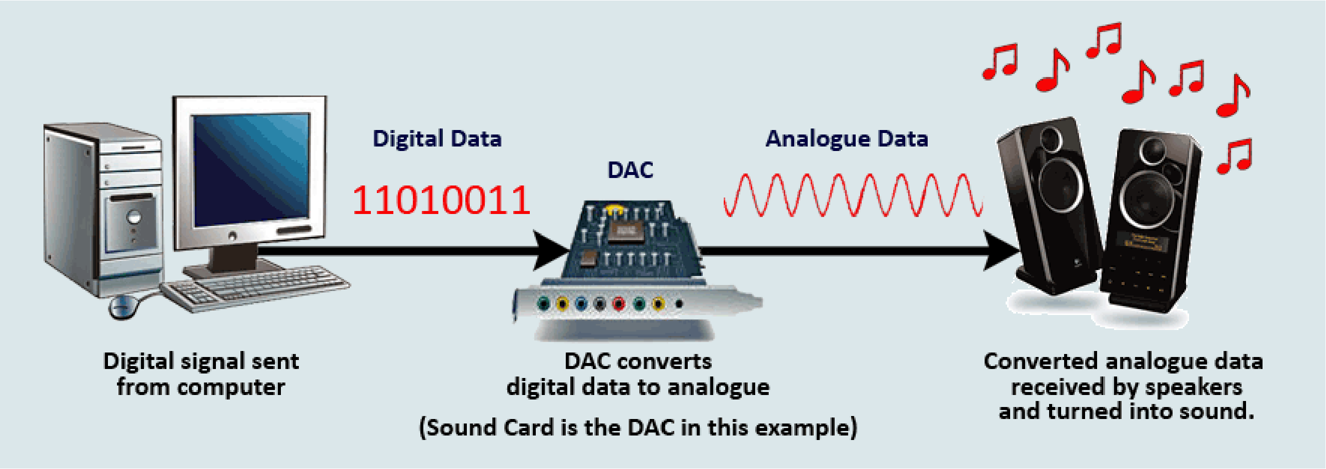
ADC & DAC

- ADC (analog-to-digital converter)
  - Convert an analog signal into a digital one
- DAC (digital-to-analog converter)
  - Convert a digital signal into an analog one



Analog to Digital, Digital to Analog





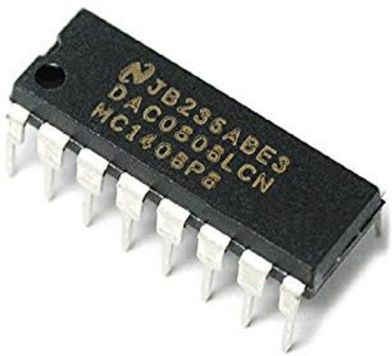
Source: http://samihaandniyatidatahandlingtask.weebly.com/analogue-and-digital-data.html

Digital to Analog Convertor (DAC)

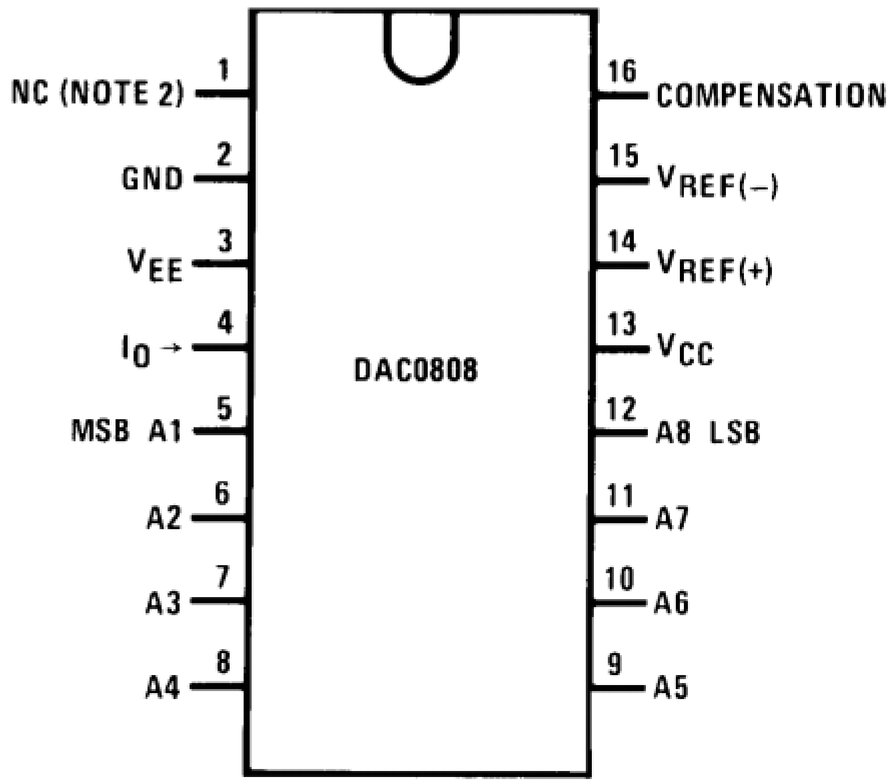
- The digital inputs are converted to current (I_{out})
- By connecting I_{out} to a negative feedback op-amp we convert current to voltage
- The current provided by I_{out} is a function of input binary numbers and the reference current
  - E.g., for an 8 bit DAC:

$$I_{out} = I_{ref}\left(\frac{A1}{2} + \frac{A2}{4} + \dots + \frac{A8}{256}\right)$$

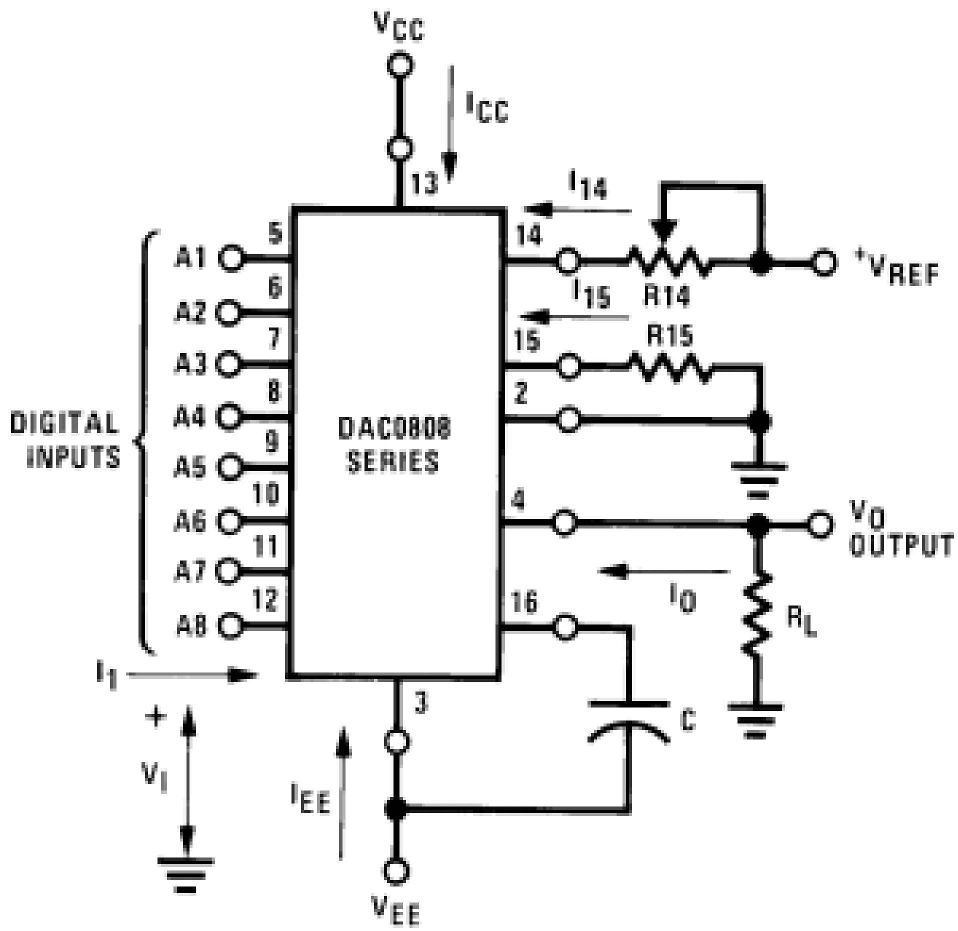
DAC0808: 8-Bit D/A Converter



Dual-In-Line Package



Source: DAC0808 datasheet



$$I_O = K\left(\frac{A1}{2} + \frac{A2}{4} + \frac{A3}{8} + \frac{A4}{16} + \frac{A5}{32} + \frac{A6}{64} + \frac{A7}{128} + \frac{A8}{256}\right)$$

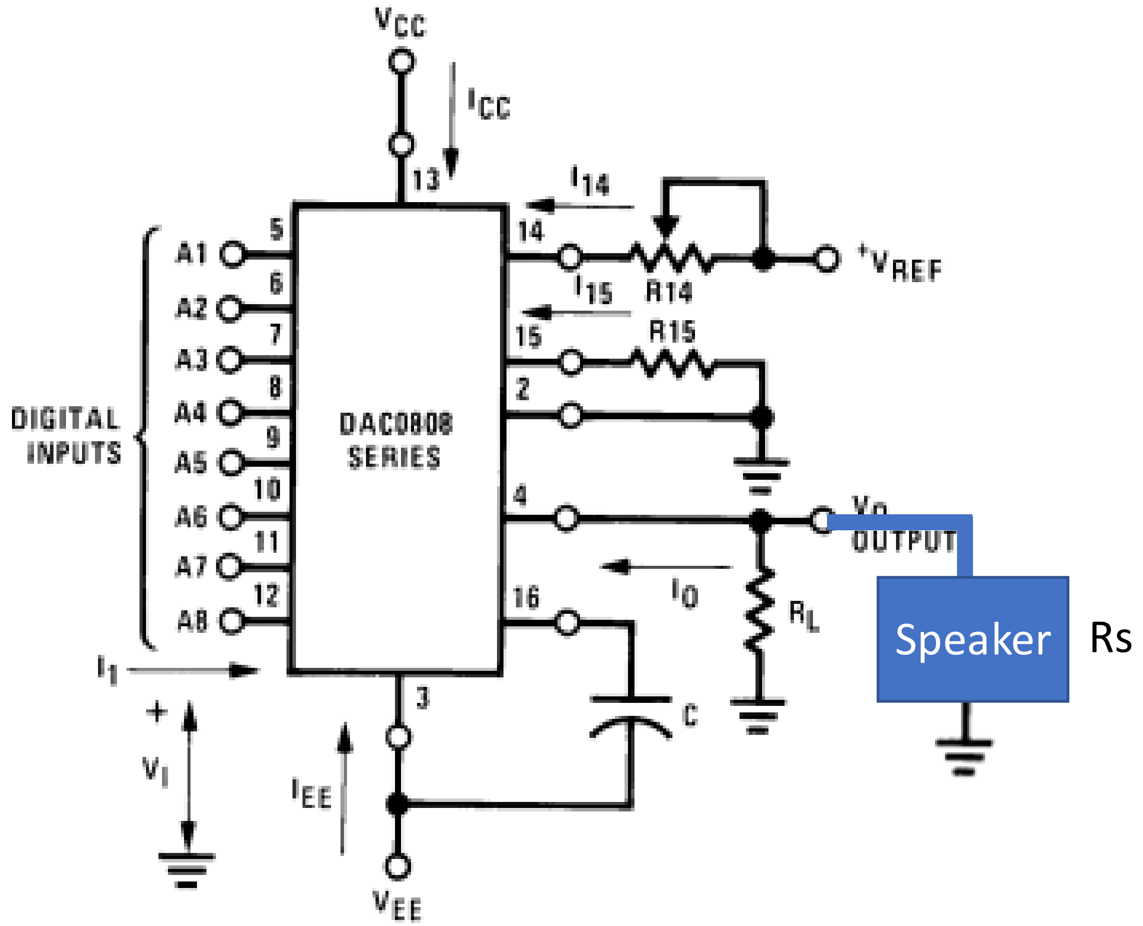
$$\text{where } K \cong \frac{V_{REF}}{R14}$$

Source: DAC0808 datasheet

But, V_{out} is not fixed!

- $V_o = I_O * \frac{R_S+R_L}{R_S*R_L}$

- How to convert the fixed I_o current to a fixed voltage?



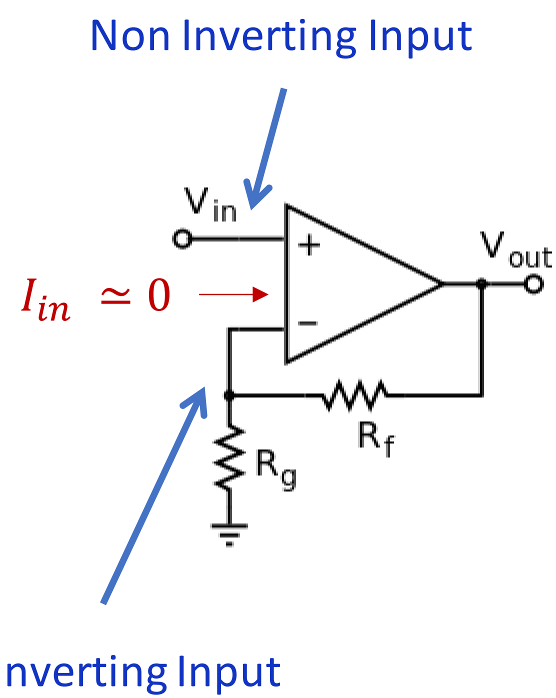
Operational amplifier (OpAmp)

- Negative feedback OpAmp

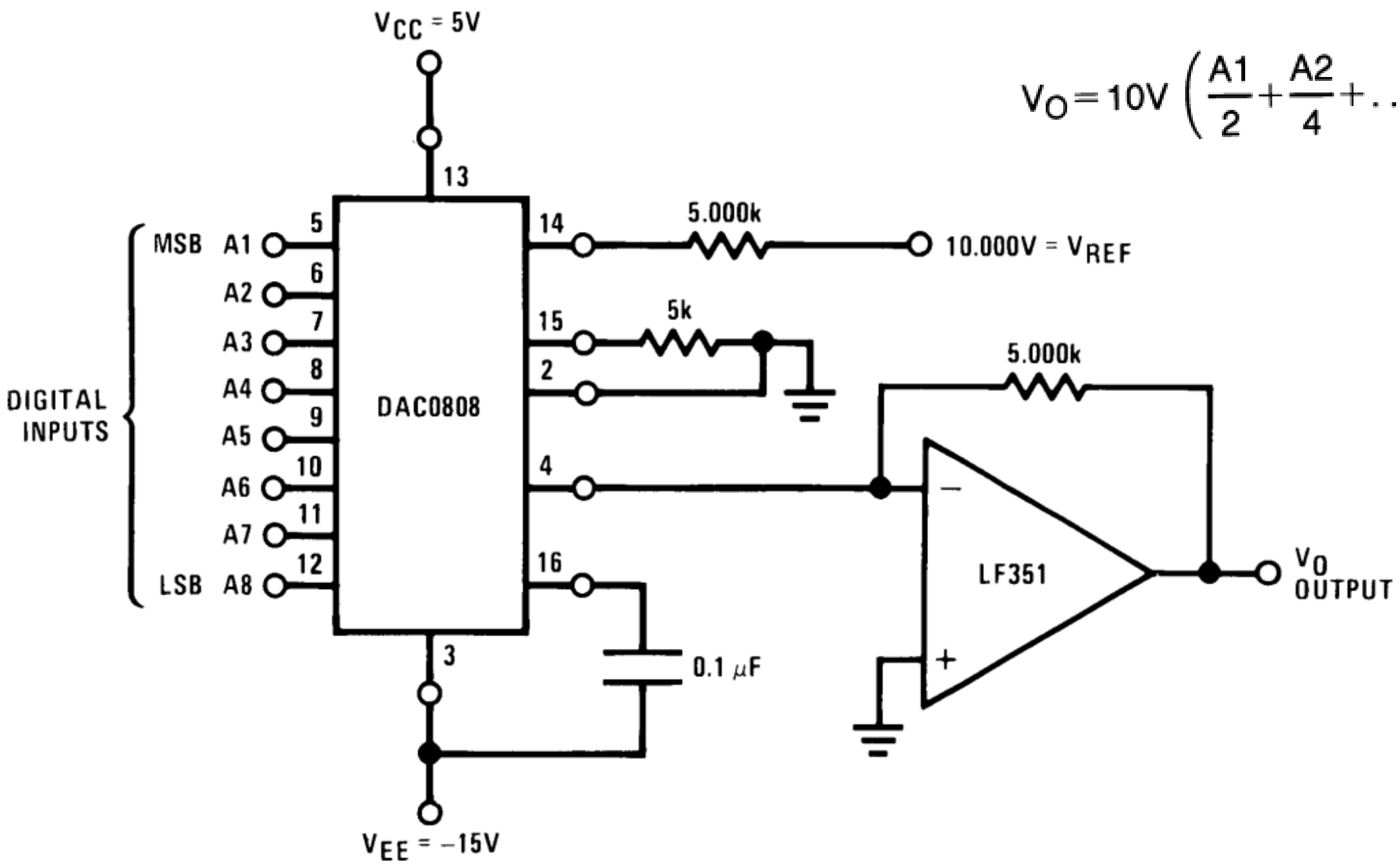
** OpAmp has high impedance at input and low impedance at output

- Behavior at equilibrium
  - “Equilibrium will be established when V_{out} is just sufficient to pull the inverting input to the same voltage as V_{in}”

  - **E.g., if Rg == Rf and Vin == 1V then Vout == 2V**



+10V Output Digital to Analog Converter



$$V_O = 10V\left(\frac{A1}{2} + \frac{A2}{4} + \ldots \frac{A8}{256}\right)$$

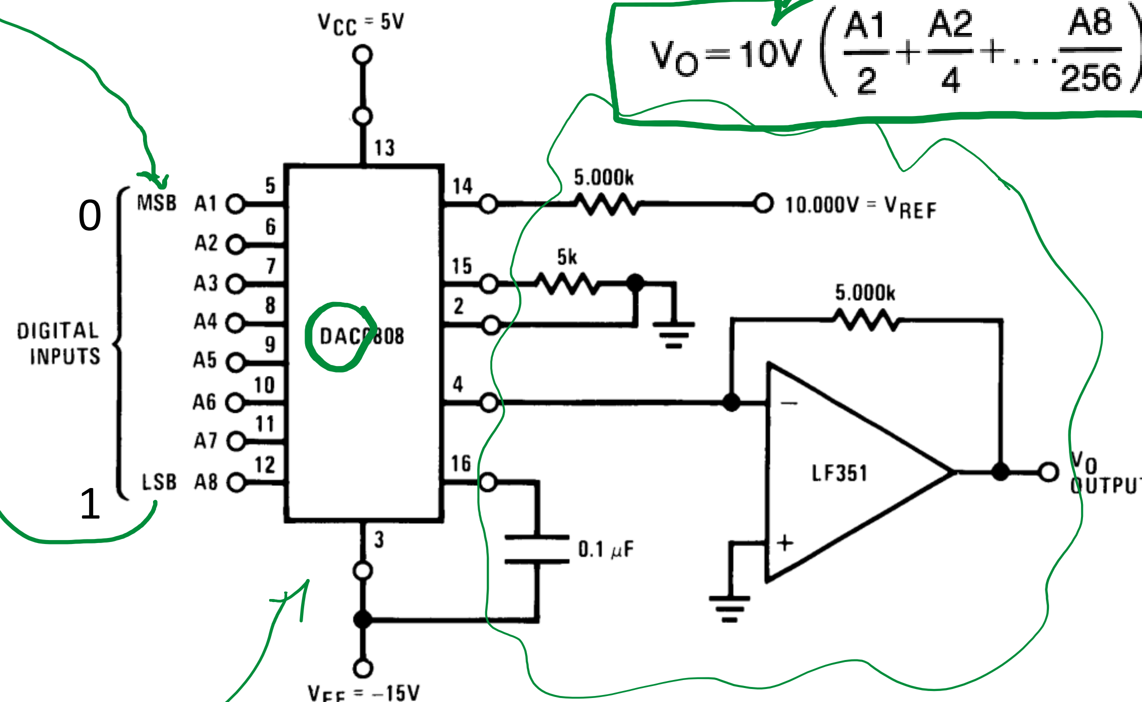
Source: DAC0808 datasheet

Example of Digital to Analog Conversion

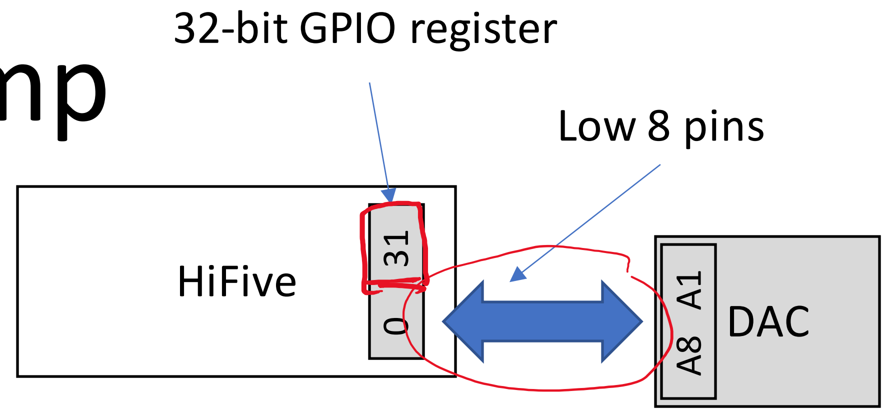
- Considering the following +10V DAC, fill in the table:

Digital In [MSB .. LSB]	Vo (V)
0x00	0
0x01	10/256
0x13=0b00010011	10 (1/2^4 + 1/2^7 + 1/2^8)
0xF0	10(1/2 + 1/4 + 1/8 + 1/16)
0xFF	

$$\boxed{V_O = 10V\left(\frac{A1}{2}+\frac{A2}{4}+\ldots\frac{A8}{256}\right)}$$

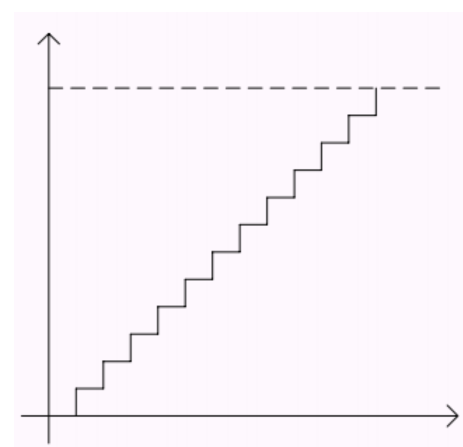


Example: Step Ramp

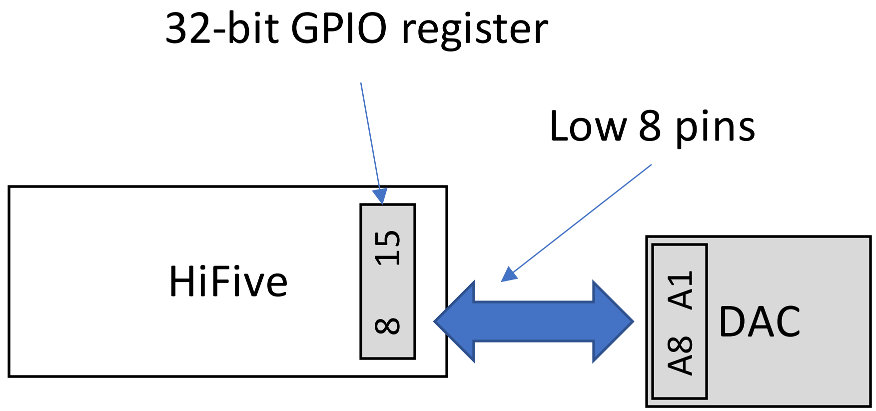



```
int step_ramp(int num_steps)
{
   unsigned int ramp = 0;
   for (ramp = 0; ramp < num_steps; i++ ) {
     uint32_t val = *(volatile uint32_t *) (GPIO_CTRL_ADDR + GPIO_OUTPUT_VAL);
     val &= 0xFFFFFF00;
     val |= ramp;
     *(volatile uint32_t *) (GPIO_CTRL_ADDR + GPIO_OUTPUT_VAL) = val;
   }
   return 0 ;
}
```

read



Example: in class




```
uint32_t dac = 0xFFFF40FF

uint32_t val = *(volatile uint32_t *) (GPIO_CTRL_ADDR + GPIO_OUTPUT_VAL);

                [15-8]
val = val & 0xFFFF00FF; //clear
val = val & dac;     //set

*(volatile uint32_t *) (GPIO_CTRL_ADDR + GPIO_OUTPUT_VAL) = val;
```

Bit Manipulation in C

- Setting a bit in a binary number
- Re-setting a bit a binary number
- Updating a bit a binary number

- Updating a range in a binary number

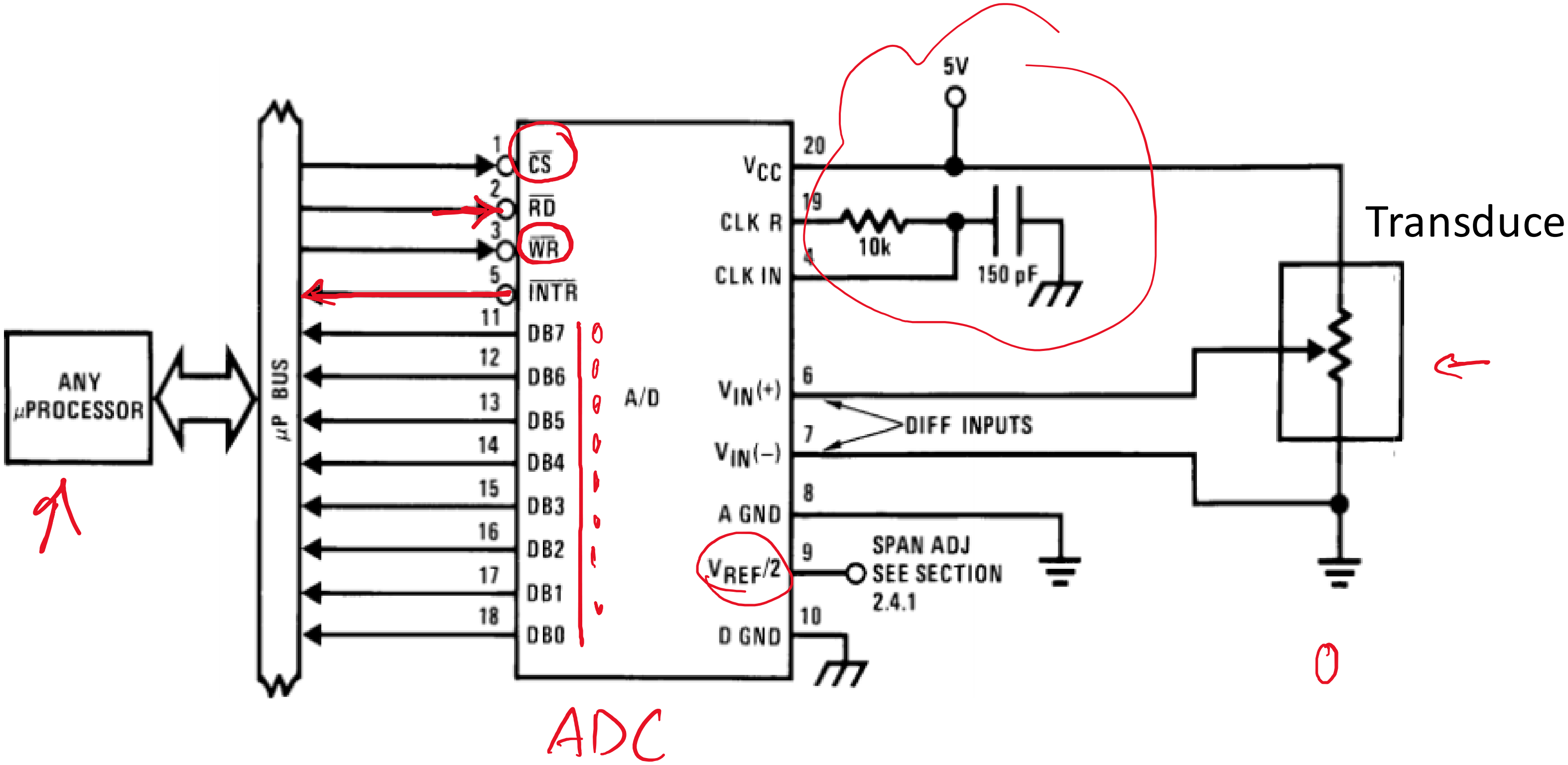
- Look at “Bonus Lecture - Bit Manipulation in C.pptx”

Analog to Digital Convertor

- Periodically samples the input and converts it to an unsigned number proportional to the input

Example: TI ADC080x 8-Bit

256



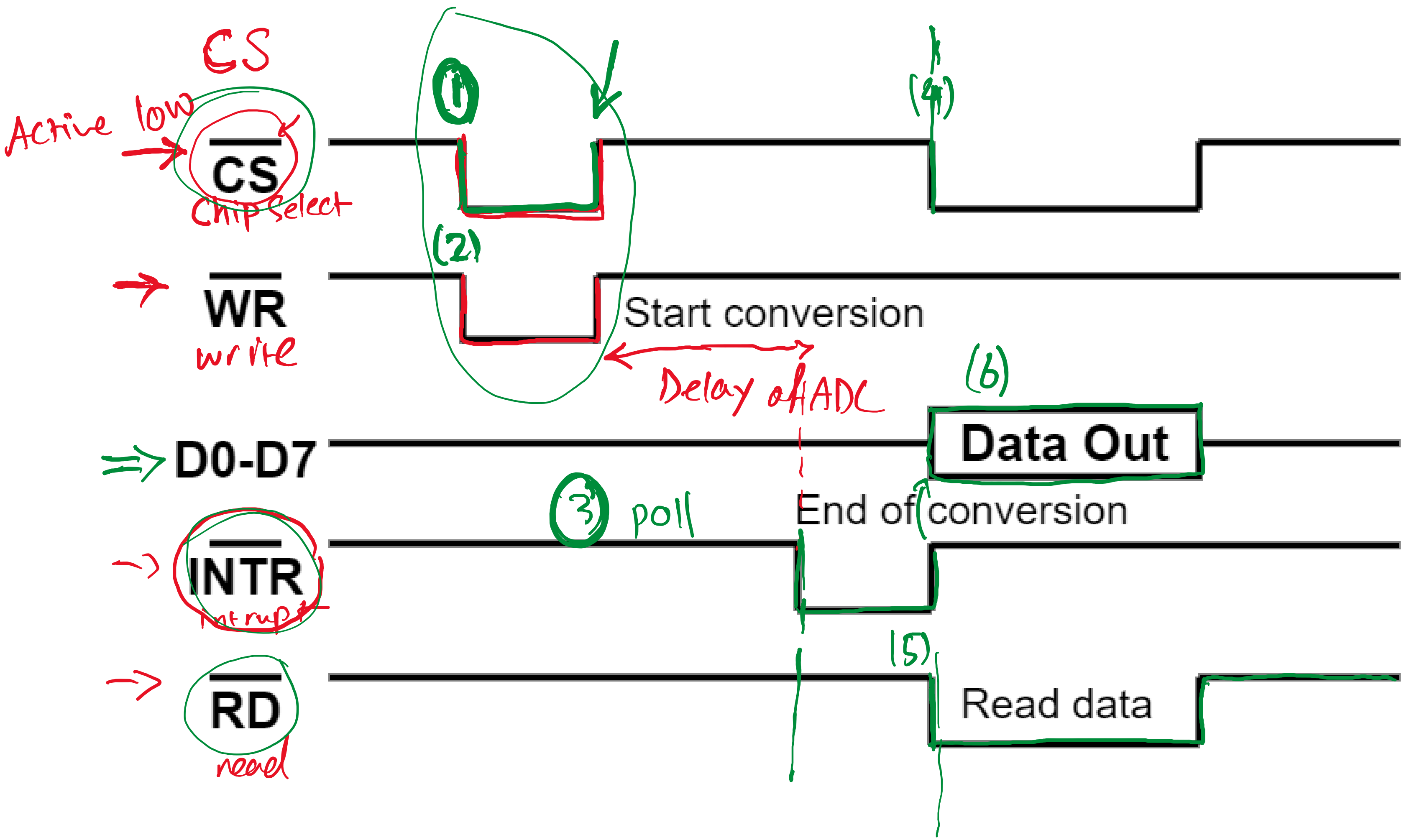
Source: ADC080x 8-Bit, μP-Compatible, Analog-to-Digital Converters datasheet

Vref/2 Relation to Vin Range

Vref/2 (V)	Vin (V)	Step Size (mV)
Not connected	0 to 5	5/256 = 19.53
2	0 to 4	4/256 = 15.62
1.5	0 to 3	3/256 = 11.71
1.28	0 to 2.56	2.56/256 = 10
1	0 to 2	2/256 = 7.81
0.5	0 to 1	1/256 = 3.90

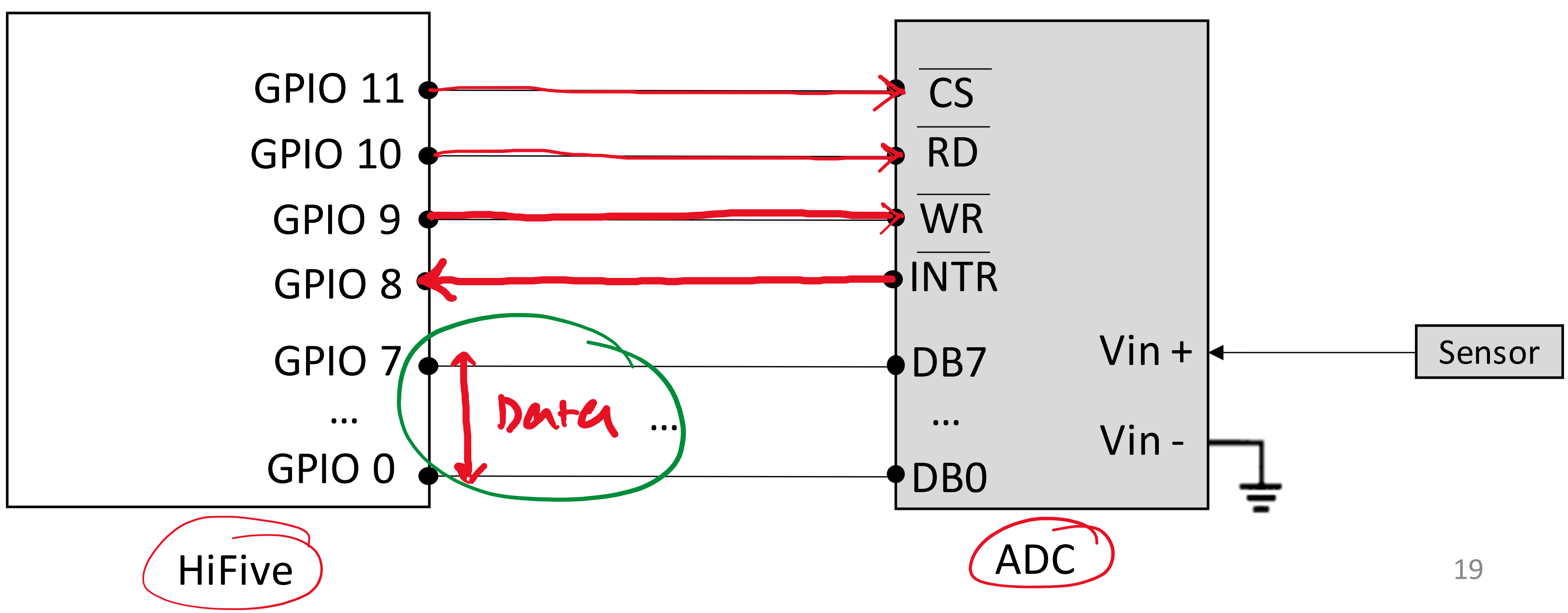
Timing

Hand shaking protocol



Example

- Write a C function that reads one digital sample from the connected sensor. Timing of the ADC is shown in the previous slide. Assume that GPIO pins 0 to 8 are already configured as input, and pins 9, 10, 11 are configured as output. Use `GPIO_CTRL_ADDR` and `GPIO_OUTPUT_VAL` parameters.



```
#define CS 11
#define RD 10
#define WR 9
#define INTR 8
uint8_t func(){
  uint32_t val = *(volatile uint32_t *) (GPIO_CTRL_ADDR +
GPIO_OUTPUT_VAL);
  val = val & ~(1 << CS); (1)
  val = val & ~(1 << WR); (2)
  *(volatile uint32_t *) (GPIO_CTRL_ADDR + GPIO_OUTPUT_VAL) = val;
  do {                (3)
     val = *(volatile uint32_t *) (GPIO_CTRL_ADDR + GPIO_OUTPUT_VAL);
  }
  while (val & 1 << INTR)
 val = val & ~(1 << CS); (4)
 val = val & ~(1 << RD); (5)
 *(volatile uint32_t *) (GPIO_CTRL_ADDR + GPIO_OUTPUT_VAL) = val;
  val = *(volatile uint32_t *) (GPIO_CTRL_ADDR + GPIO_OUTPUT_VAL);
 return (uint8_t) val;& 0x00 00 00 FF;
```