Imperial College London

Lecture 8

Digital-to-Analogue Conversion

Peter Cheung
Imperial College London

URL: www.ee.imperial.ac.uk/pcheung/teaching/EE2_CAS/
E-mail: p.cheung@imperial.ac.uk

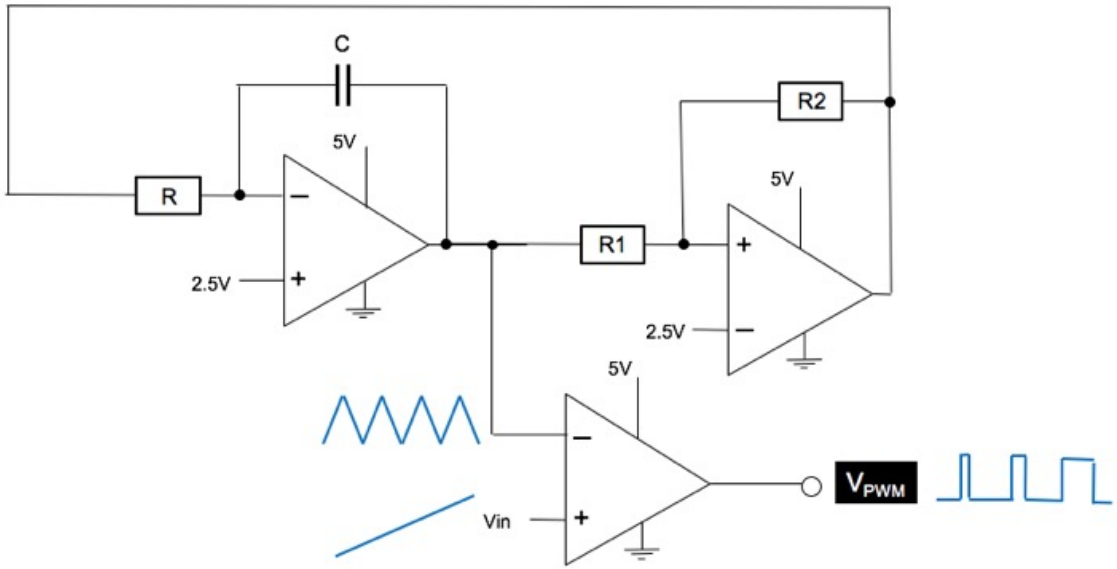
Lecture Objectives

- Understand **pulse-width modulated** (PWM) DAC
- Understand how a **weighted-resister DAC** can be used to convert numbers with binary or non-binary bit weightings
- Understand the meaning of the terms used to **specify DAC accuracy**
- Understand **resistor string based DAC** architecture
- Understand how an **R-2R ladder** can be used to convert both unsigned and signed binary numbers
- Understand **multiplying DAC**

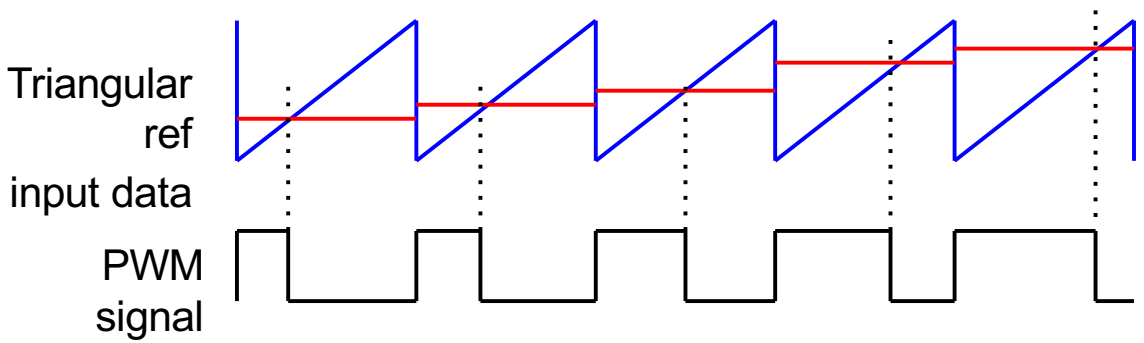
References:

- "Data Converter Architectures" in Data Conversion Handbook by Analog Devices

Analogue Pulse-width Modulated (PWM)

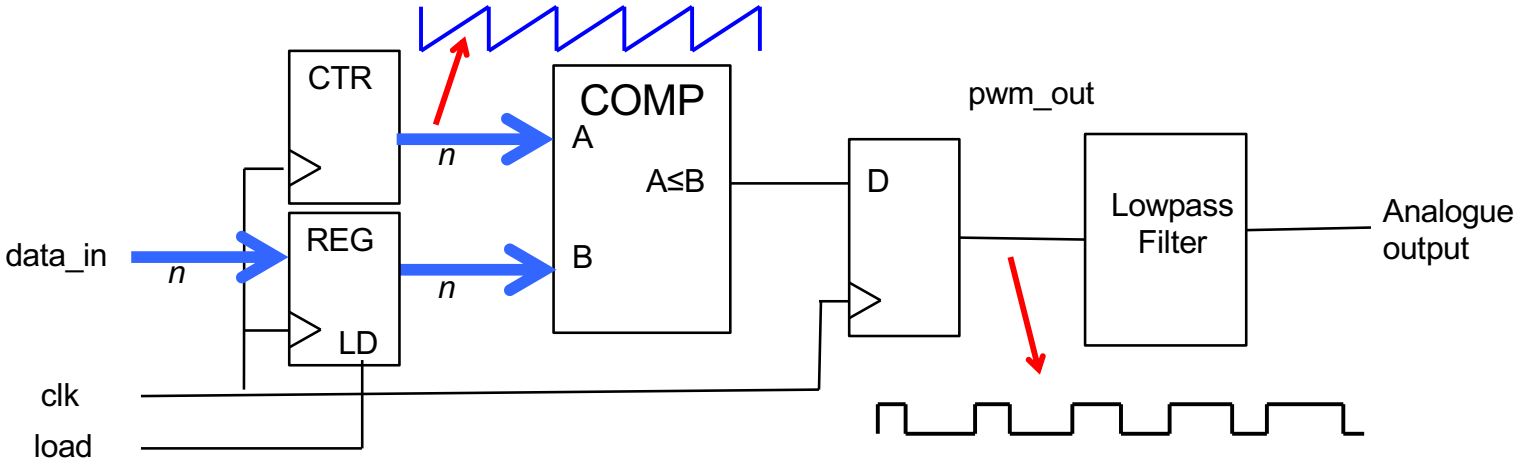


- Simple idea: PWM signal is generated by comparing a triangular reference signal with the input data value



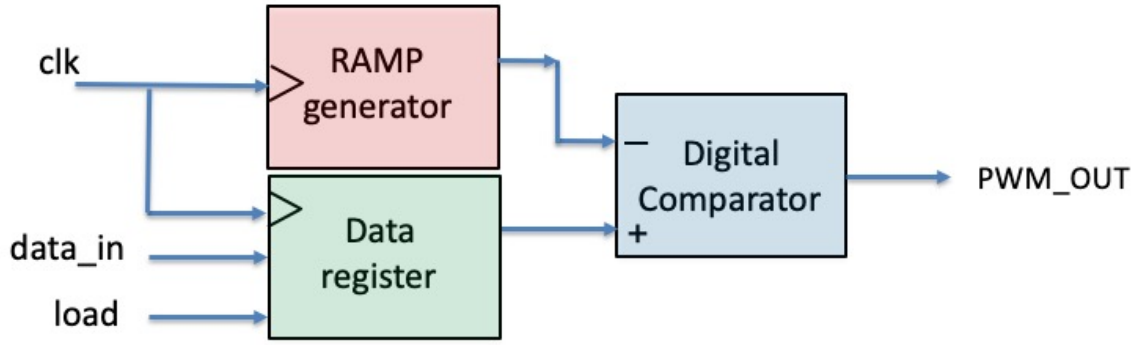
Digital Pulse-width Modulated (PWM) as DAC

- Sawtooth value generated by a wrap-around counter
- Load pulse latches **data_in** and stores it in register
- When input value is reached by counter, comparator output changes state (H to L)
- Lowpass filter provides analogue output voltage proportional to the duty cycle of PWM signal



PWM DAC in SystemVerilog

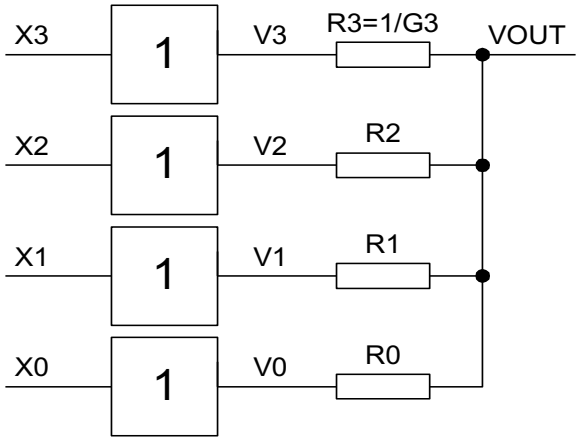
```
module pwm # (parameter WIDTH = 10)(
    input  logic                clk,        // system clock
    input  logic [WIDTH-1:0]    data_in,    // input data for conversion (limited to 10-bit)
    input  logic                load,       // high pulse to load new data
    input  logic [WIDTH-1:0]    max,        // maximum value of data_in
    output logic                pwm_out     // PWM output
);
    logic [WIDTH-1:0]       d;              // internal registe
    logic [WIDTH-1:0]       count;          // internal 10-bit
```




```
always_ff @ (posedge clk)
    if (load == 1'b1) d <= data_in;

initial count = {WIDTH{1'b0}};

always_ff @ (posedge clk) begin
    if (count == max)
        count <= {WIDTH{1'b0}};
    else
        count <= count + 1'b1;

    if (count >= d)
        pwm_out <= 1'b0;
    else
        pwm_out <= 1'b1;
    end
```

Simple DAC

- A DAC converts a binary number into a voltage proportional to its value:



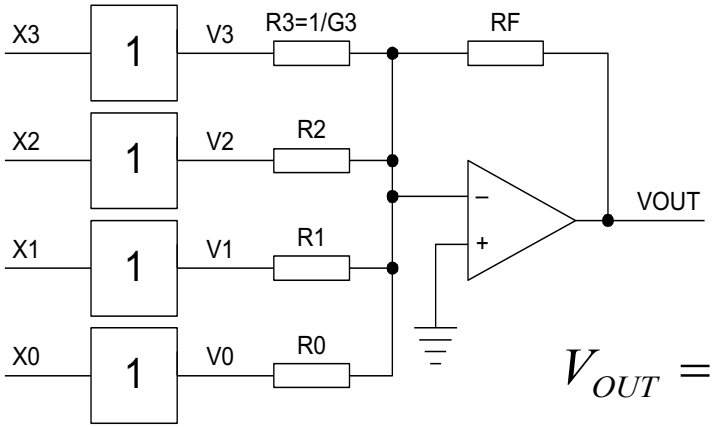
$$\left(V_3 - V_{OUT}\right)G_3 + \ldots + \left(V_0 - V_{OUT}\right)G_0 = 0$$

$$V_{OUT} = \frac{V_3G_3 + V_2G_2 + V_1G_1 + V_0G_0}{G_3 + G_2 + G_1 + G_0}$$

$$R_{Thevenin} = \frac{1}{G_3 + G_2 + G_1 + G_0}$$

- Hence V_{OUT} is a weighted sum of $V_3, \ldots, V_0$ with weights proportional to the conductances $G_3, \ldots, G_0$.
  - If X3:0 is a binary number we want conductances in the ratio 8:4:2:1.
  - Very fast: gate slew rate > 3 V/ns.
  - We can scale the resistors to give any output impedance we want.
- You do not have to use a binary weighting
  - By using other conductance ratios we can choose arbitrary output voltages for up to five of the sixteen possible values of X3:0. May need additional resistors from VOUT to the power supplies.

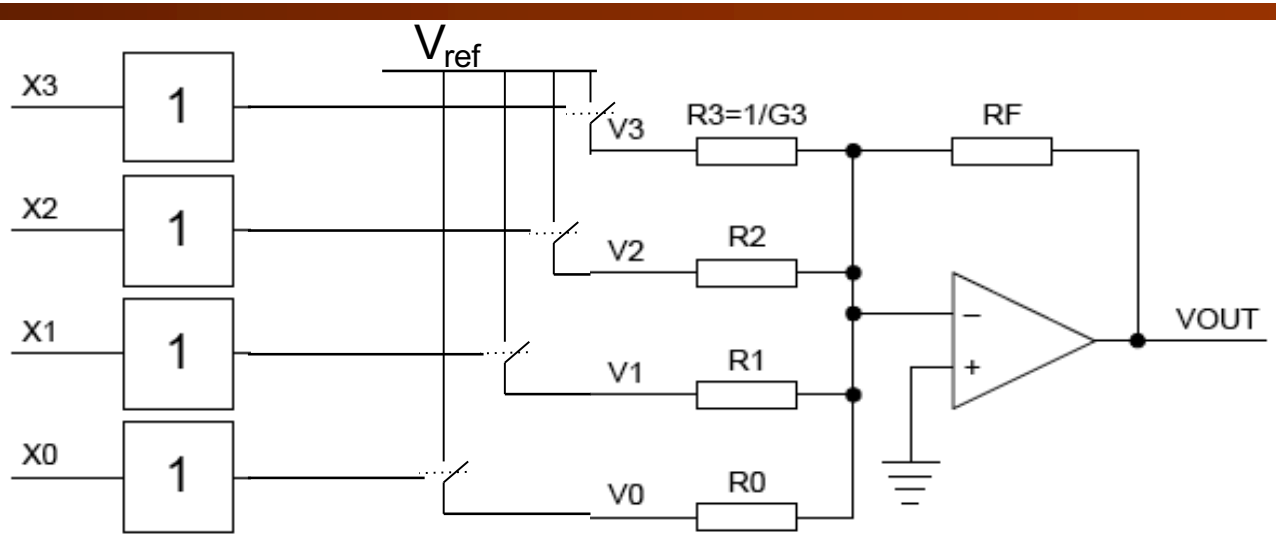
Improved DAC with Output Op-Amp



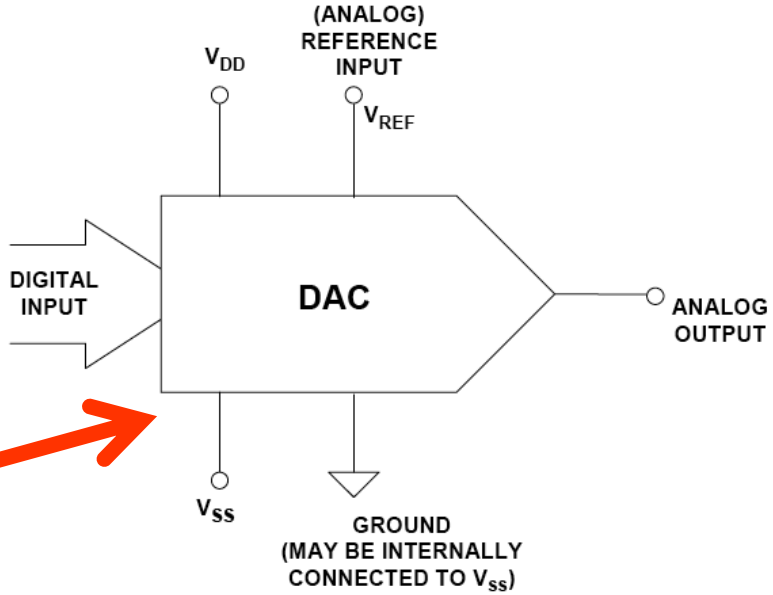
$$V_{OUT} = \frac{-R_F}{R_{Thévenin}} \times V_{Thévenin} = -R_F\left(V_3G_3 + V_2G_2 + V_1G_1 + V_0G_0\right)$$

- Adding an op-amp:
  - The voltage at the junction of all the resistors (the **virtue earth** node) is now held constant by the feedback
    - Hence current drawn from V_3 is independent of the other voltages V_2, …, V_0
    - Hence any gate non-linearity has no effect ⇒ more accurate
  - Lower output impedance
  - Much slower: op-amp slew rate ≈ 1 V/μs
- Hard to make accurate resistors covering a wide range of values in an integrated circuit
  - Weighted-resistor DAC is no good for converters with many bits

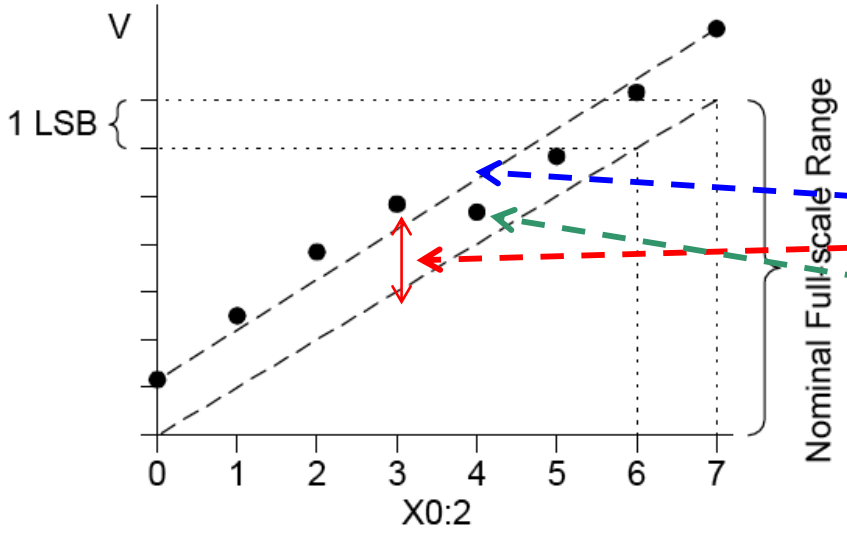
Further Improvement with reference voltage source



- Use digital signal to control analogue switches
- Switching Vref on/off the resistor network
- Clear separation between digital control and analogue voltages – much better accuracies
- General DAC block diagram:



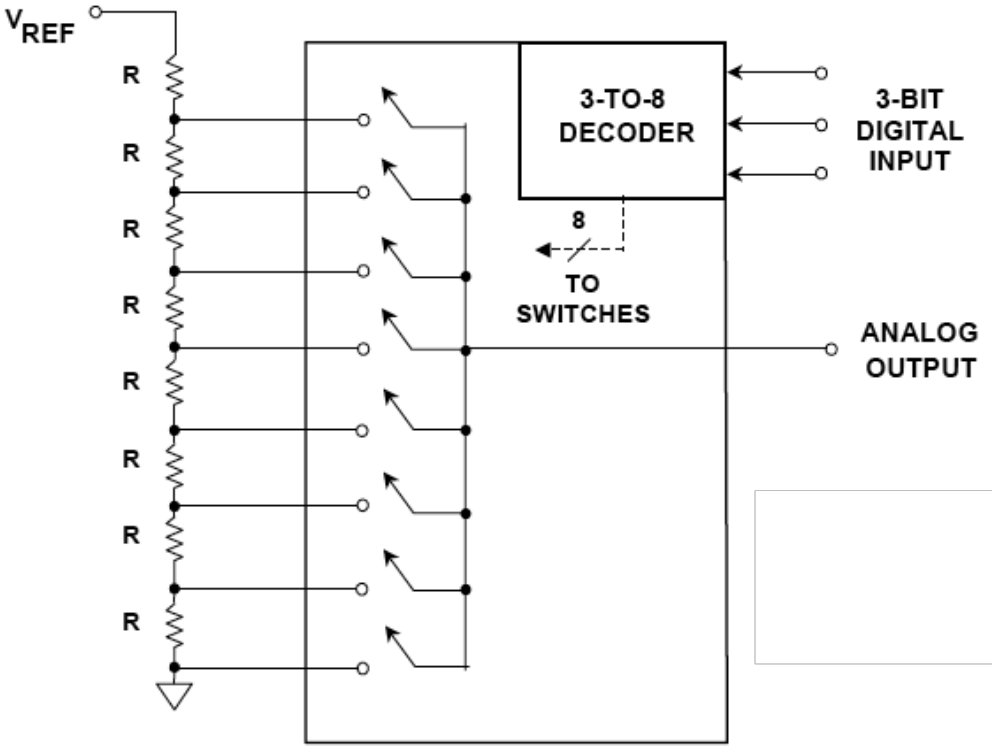
DAC Specification Jargon



Accuracy=1.8@X=3 Linearity=–0.7@X=4
Non-monotonic@3→4 Diff Linearity=–1.2@ 3→4
(all in units of LSB)

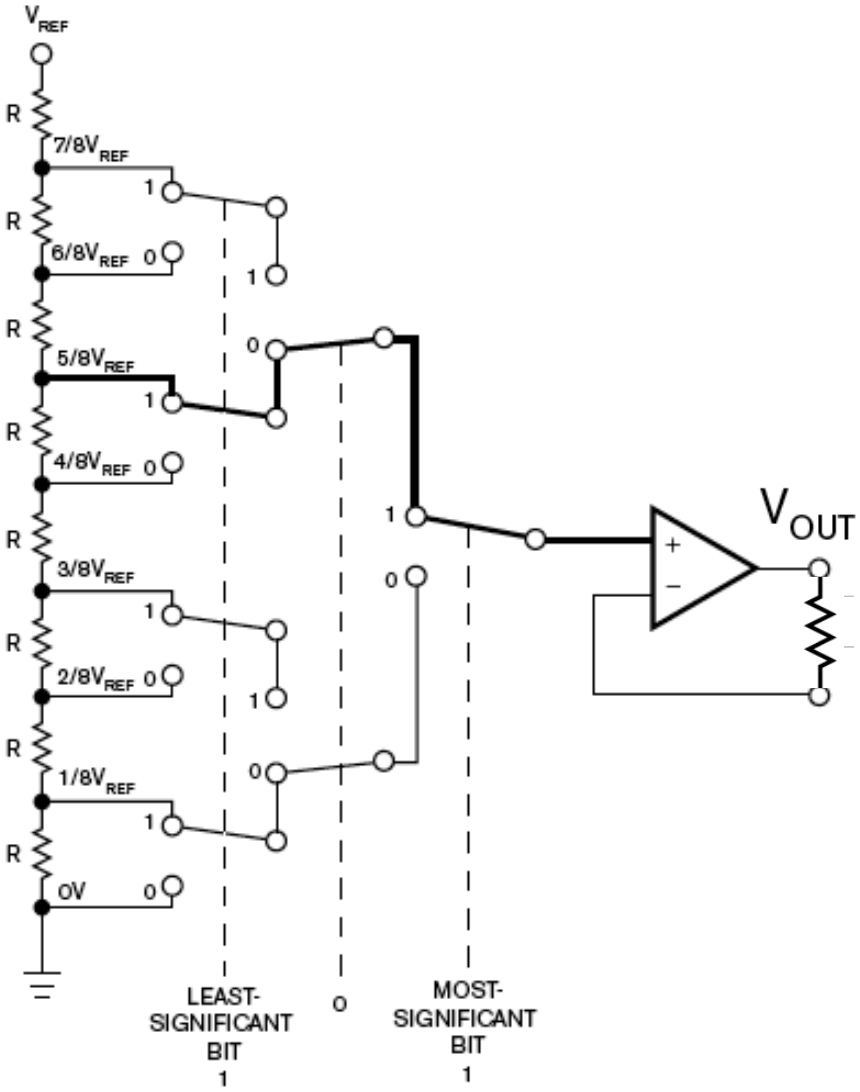
- **Resolution**
  - 1 LSB = ΔV when $X \rightarrow X+1$
    = Full-scale range ÷ (2^N-1)
- **Accuracy**
  - Worst deviation from nominal line
- **Linearity**
  - Worst deviation from line joining end points
- **Differential Linearity**
  - Worst error in ΔV when $X \rightarrow X+1$
  - measures smoothness
- **Monotonic**
  - At least ΔV always has the correct sign
- **Settling time**
  - Time taken to reach the final value to within some tolerance, e.g. ±½ LSB

Thermometer DAC using Resistor String



- ✓ Simple
- ✓ Inherently monotonic
- ✓ Needs only IDENTICAL resistors, good differential linearity
- ✓ Only two switches operate during a transition, low output glitch and fast settling
- ✓ Low power
- ✓ Widely use with modern technology with small feature sizes
- X Large number of resistors
- X Useful for low to medium resolution DAC
- X Large resistance – resulting in higher noise

Resistor String DAC with Op-Amp output

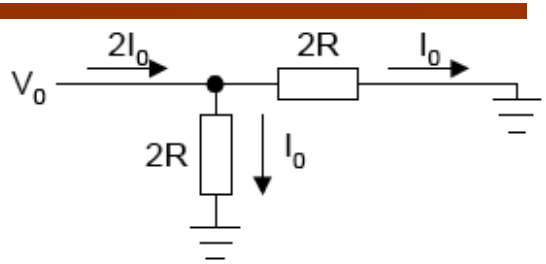


$$V_{OUT} = V_{REF} \Sigma (b_i / 2^i)$$

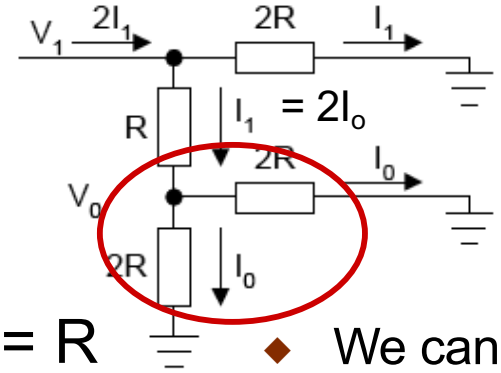
- Here the digital code is 3'b101
- Decodes to 5/8 VREF
- Output Op-amp isolate internal resistors from output load

DAC using R-2R Ladder

We want to generate currents $I_0, 2I_0, 4I_0, \ldots$

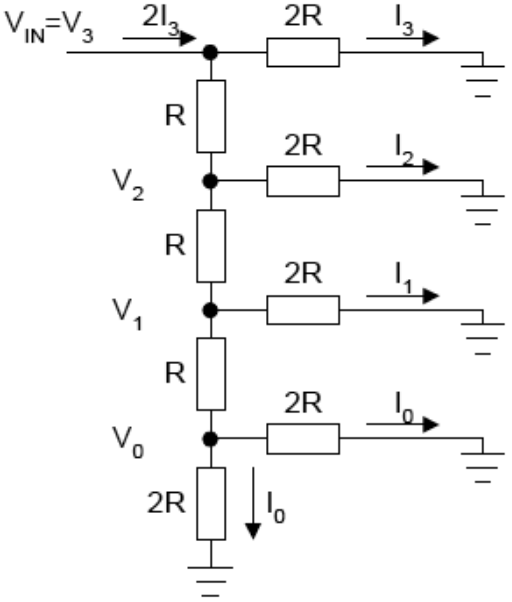


- Two $2R$ resistors in parallel means that the $2I_0$ current will split equally and equivalent resistance R



= R

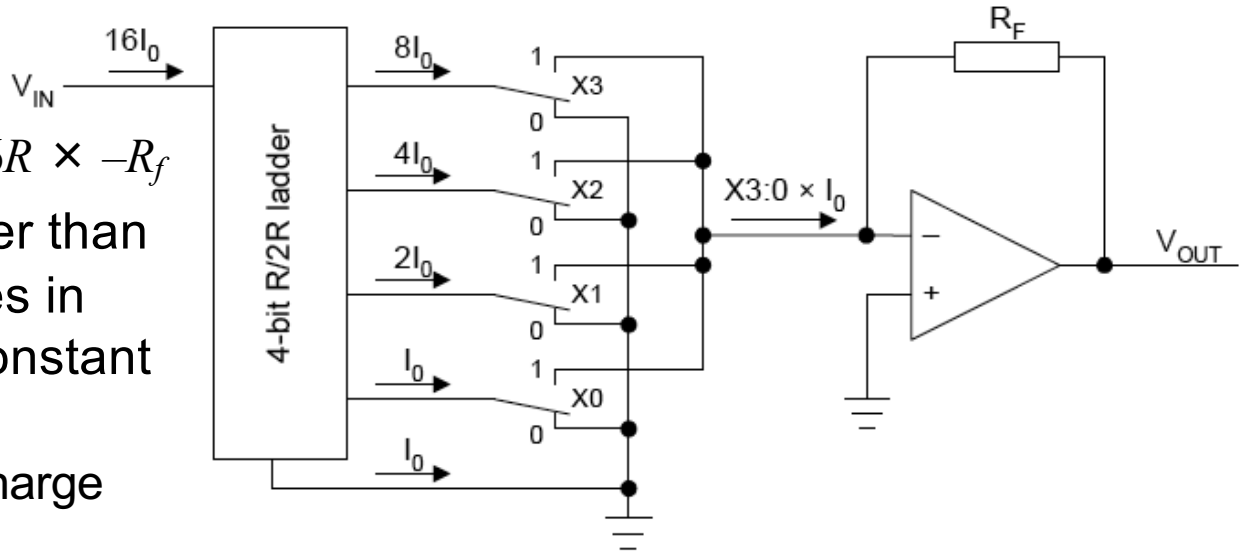
- The Thévenin resistances of the two branches at V_1 both equal $2R$ so the current into this node will split evenly
  - We already know that the current into node V_0 is $2I_0$, so it follows that I_1=$2I_0$



- We can repeat this process indefinitely and, using only two resistor values, can generate a whole series of currents where I_n=$2^n I_0$
  - From the voltage drop across the horizontal resistors, we see that $V_n = 2RI_n = 2^{n+1}RI_0$
  - For an N-bit ladder the input voltage is therefore $V_{in} = 2^N RI_0 \Rightarrow I_0 = 2^{-N} V_{in}/R$

Current-Switched DAC

- Total current into summing junction is $X3{:}0 \times I_0$
  - Hence $V_{out} = X3{:}0 \times V_{in}/16R \times -R_f$
  - We switch currents rather than voltages so that all nodes in the circuit remain at a constant voltage
    $\Rightarrow$ no need to charge/discharge node capacitances
    $\Rightarrow$ faster



- Use CMOS transmission gates as switches: adjust ladder resistors to account for switch resistance
  - Each 2-way switch needs four transistors

- As required by R/2R ladder, all the switch output terminals are at 0 V.
  - ladder outputs are always connected either to ground or to a virtual earth

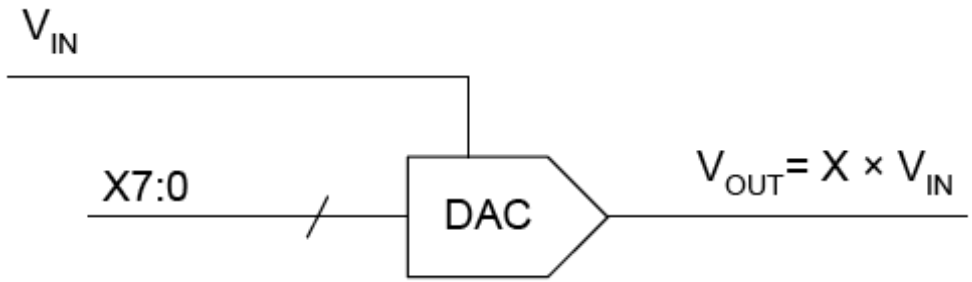
Programmable Attenuator (Amplifier)

- The output of the DAC is proportional to the ***product*** of an analog voltage (V_{in}) and a digital number ($\mathrm{X3:0}$)

$$V_{out} = \mathrm{X3:0} \times V_{\mathrm{in}} / 16R \times -R_f$$

- It is called a ***multiplying*** DAC

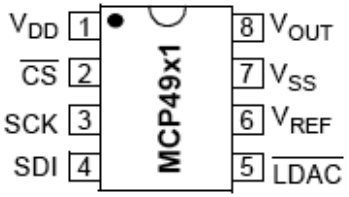
- Can be used as a digital attenuator:



- Here the digital number X7:0 controls the gain of the circuit

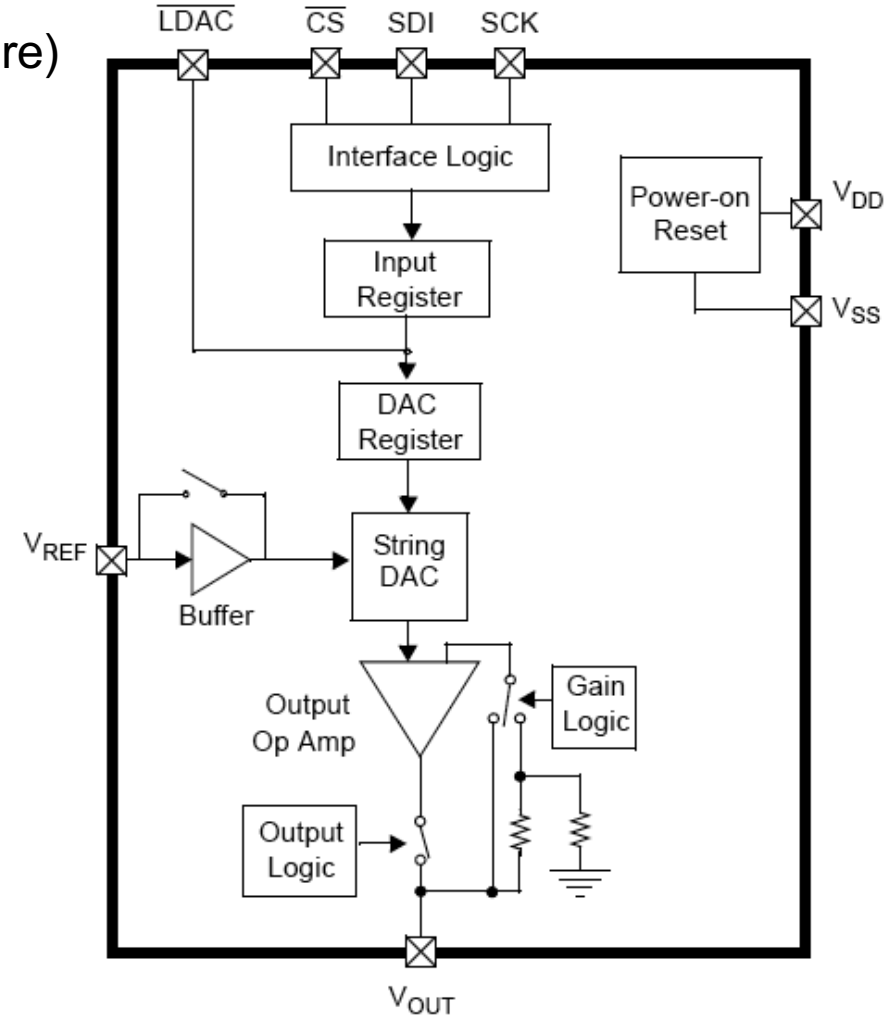
The MCP4921 DAC (used in Lab)

- **Microchip MCP4921** 12-bit DAC
- Uses **resistor string** architecture (earlier lecture)
- Serial Peripheral Interface (SPI)



- Rail-to-Rail Output
- SPI Interface with 20 MHz Clock Support
- Simultaneous Latching of the DAC Output with $\overline{\text{LDAC}}$ Pin
- Fast Settling Time of 4.5 µs
- Selectable Unity or 2x Gain Output
- External Voltage Reference Input
- External Multiplier Mode

Symbol	Description
V_{DD}	Supply Voltage Input (2.7V to 5.5V)
$\overline{CS}$	Chip Select Input
SCK	Serial Clock Input
SDI	Serial Data Input
$\overline{LDAC}$	DAC Output Synchronization Input. This pin is used to transfer the input register (DAC settings) to the output register (V_{OUT})
V_{REF}	Voltage Reference Input
V_{SS}	Ground reference point for all circuitry on the device
V_{OUT}	DAC Analog Output



Serial Peripheral Interface for DAC (SPI)

bit 15 0 = Write to DAC register
1 = Ignore this command

bit 14 **BUF:** V_{REF} Input Buffer Control bit
1 = Buffered
0 = Unbuffered

bit 13 **$\overline{GA}$:** Output Gain Selection bit
1 = 1x ($V_{OUT} = V_{REF} * D/4096$)
0 = 2x ($V_{OUT} = 2 * V_{REF} * D/4096$)

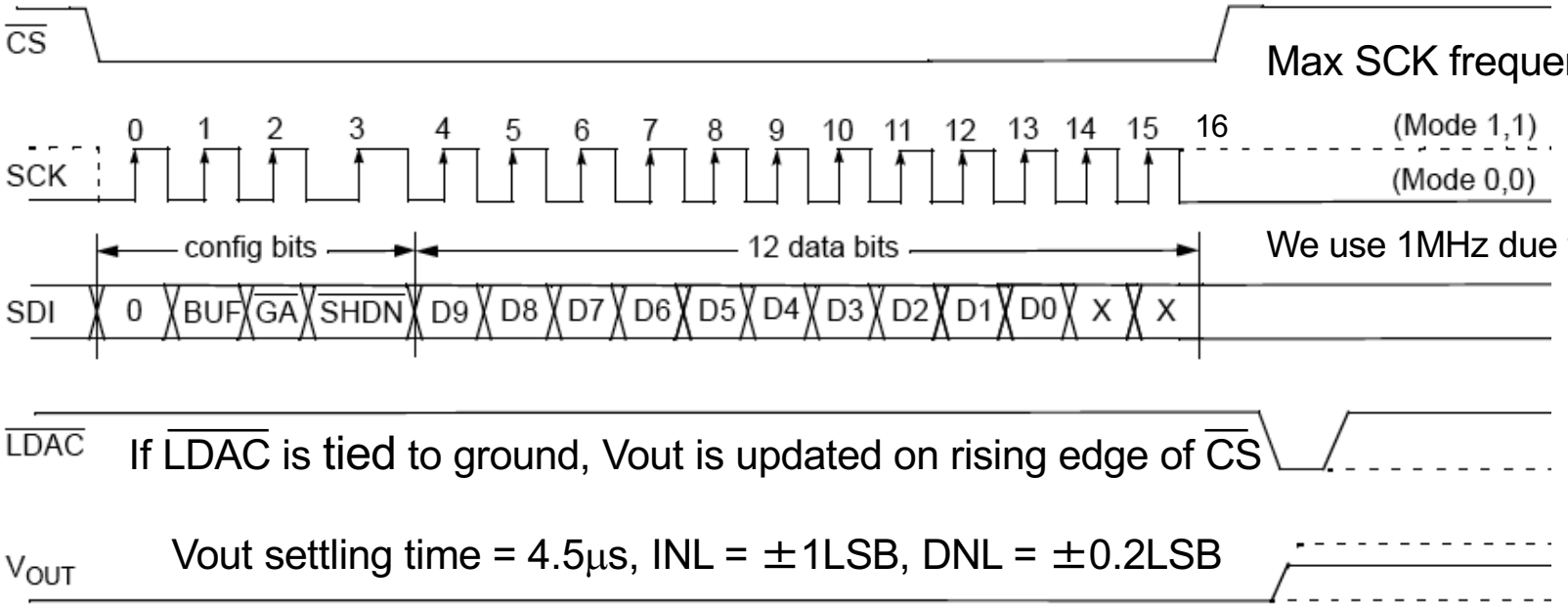
V_{REF} = 3.3V

bit 12 **$\overline{SHDN}$:** Output Shutdown Control bit
1 = Active mode operation. VOUT is available.
0 = Shutdown the device.

bit 11-0 **D11:D0:** DAC Input Data bits. Bit x is ignored.

bit 11-2 **D9:D0**: DAC input data bit

Vout = V_{REF} * (D[9:0]/1024)



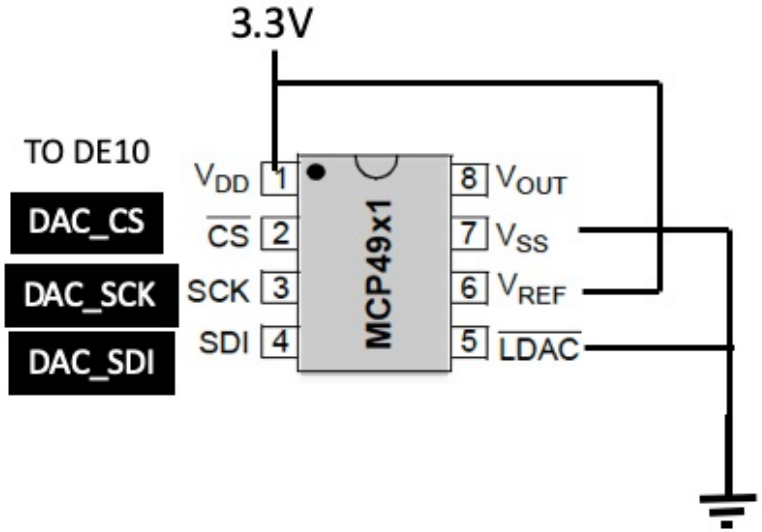
Max SCK frequency is 20MHz

We use 1MHz due to ADC limits.

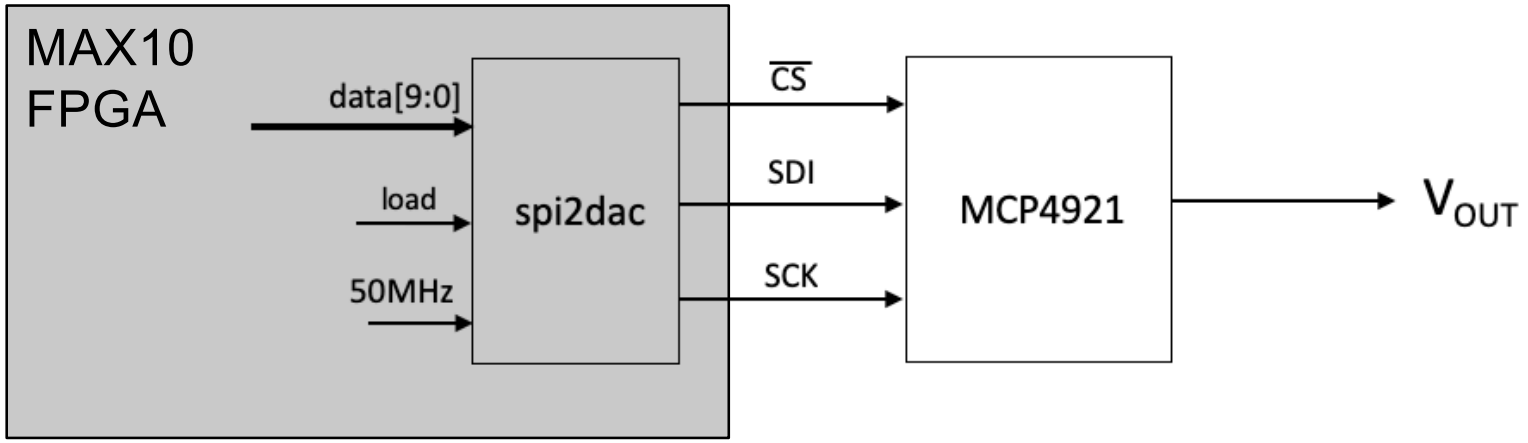
If $\overline{LDAC}$ is tied to ground, Vout is updated on rising edge of $\overline{CS}$

Vout settling time = 4.5μs, INL = ±1LSB, DNL = ±0.2LSB

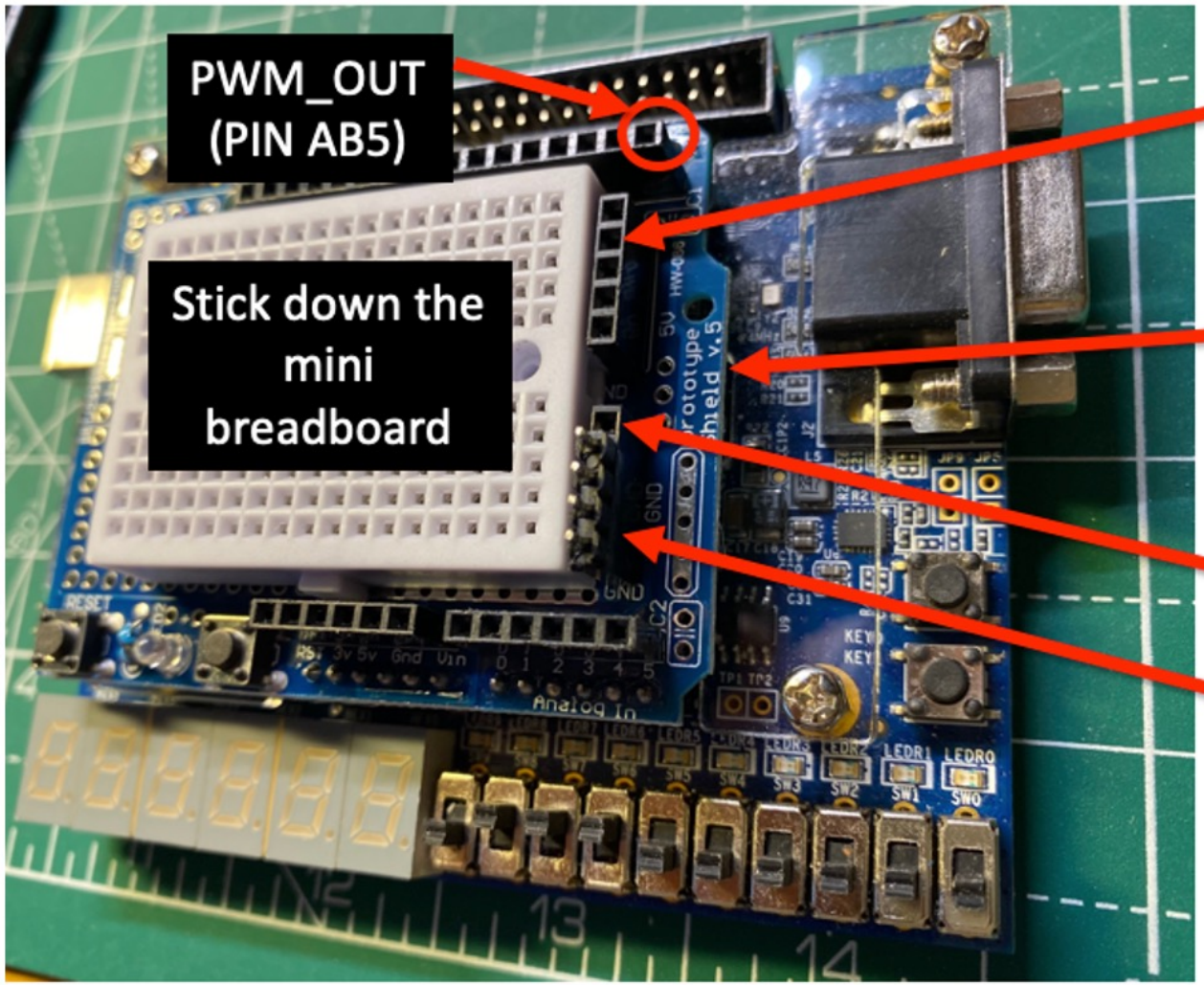
Interfacing the FPGA to the DAC



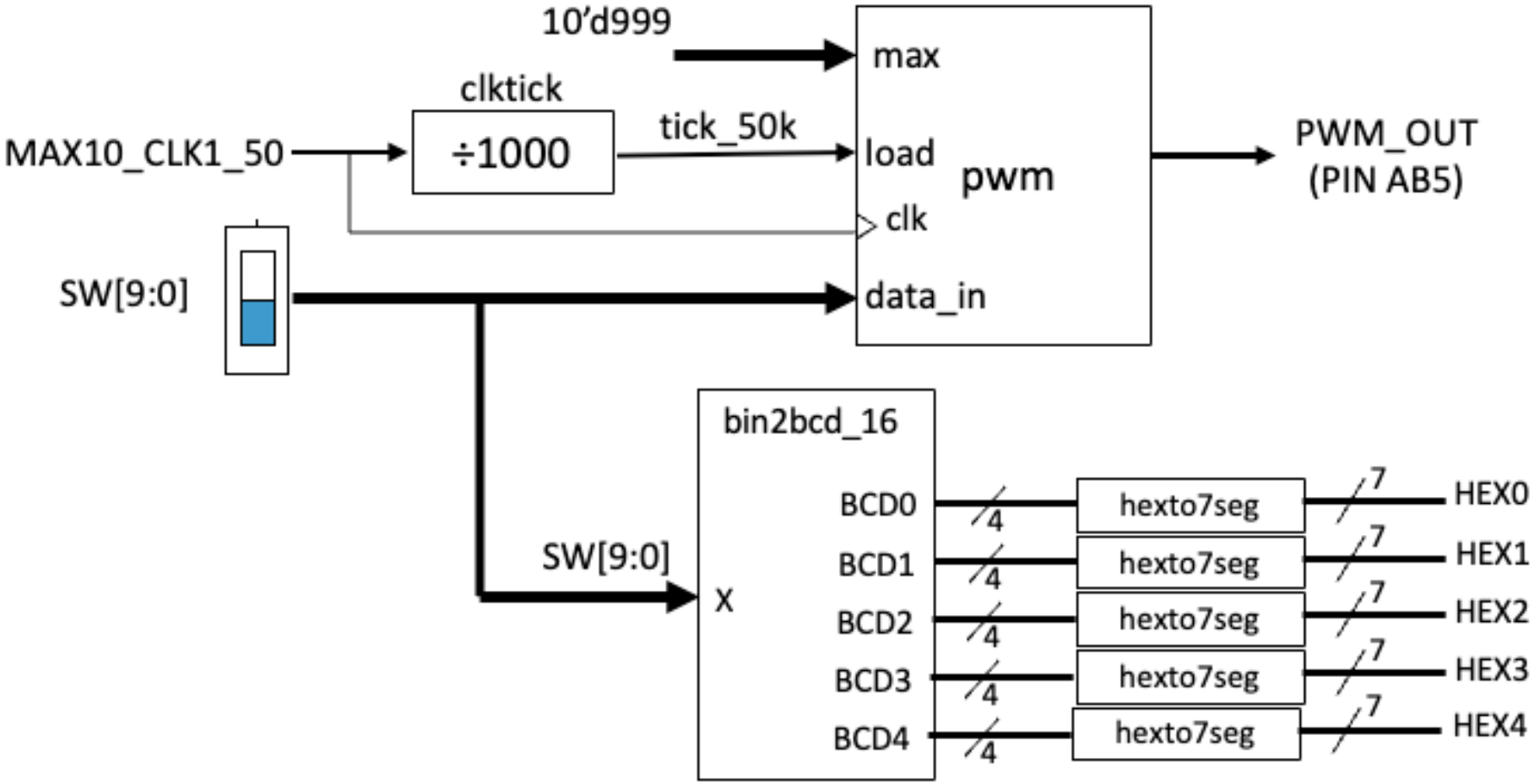
Symbol	Description
V_{DD}	Supply Voltage Input (2.7V to 5.5V)
$\overline{CS}$	Chip Select Input
SCK	Serial Clock Input
SDI	Serial Data Input
$\overline{LDAC}$	DAC Output Synchronization Input. This pin is used to transfer the input register (DAC settings) to the output register (V_{OUT})
V_{REF}	Voltage Reference Input
V_{SS}	Ground reference point for all circuitry on the device
V_{OUT}	DAC Analog Output



Lab 5 – DAC conversion – Prototype Shield



Lab 5 Task 1 PWM as a DAC converter



Lab 5 Task 2 Using the MCP4921 chip