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

This lecture focuses on digital to analogue conversion techniques. This is linked to Lab 5 of the practical experiment where you will be using two different methods of generating analogue voltages from digital signals.

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:

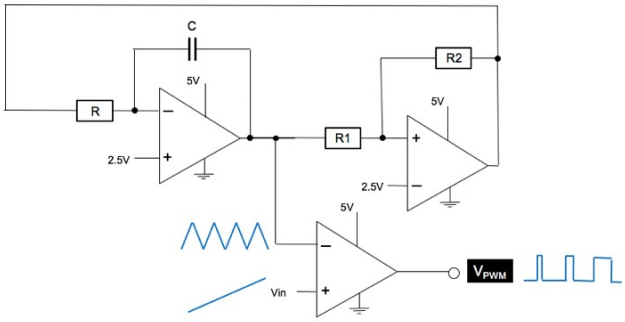
- "Data Converter Architectures" in Data Conversion Handbook by Analog Devices

Although digital technology dominates modern electronic systems, the physical world remains mostly analogue in nature. The most important components that link the analogue world to digital systems are analogue-to-digital and digital-to-analogue converters (ADCs and DACs).

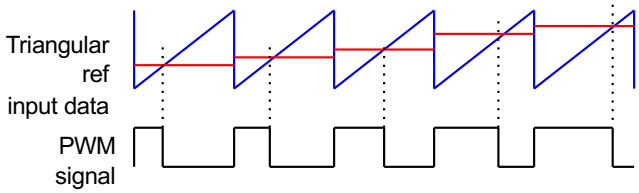
In this and a later lectures, we will consider how these converters work, their limitations and how to read their data sheets. Designing ADC and DAC requires both knowledge of analogue and digital designs. We are only interested in examining the basic principles of these converters and learn how to use them. We will NOT consider how they are designed. Detail ADC/DAC designs at transistor level will be considered in 3rd and 4th years on other course modules.

Analog Devices is a US company that has the largest range of converter products. They publish an excellent handbook which is available through the course webpage. Relevant to this lecture is the chapter on "Chapter 3: Data Converter Architectures".

Analogue Pulse-width Modulated (PWM)



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



PYKC 21 Nov 2023 EE2 Circuits & Systems Lecture 8 Slide 3

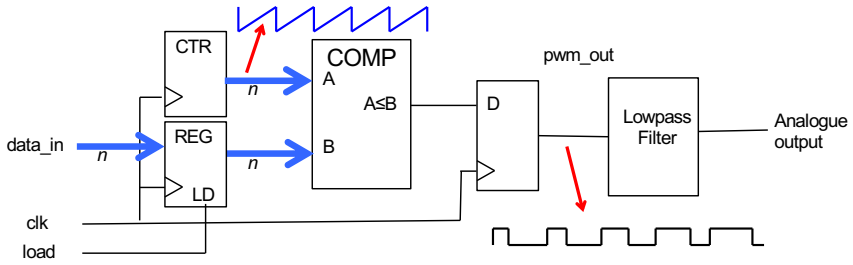
In Lecture 4, we explored the use of op-amps to build an analogue pulse-width modulator using an integrator, a comparator with hysteresis (also known as a Schmitt trigger circuit), and a simple analogue comparator.

This circuit takes an analogue input signal Vin and generates a digital PWM signal V_{PWM} - its duty cycle is proportional to Vin.

Instead of using a triangular signal, one could also use a ramp signal or a saw-tooth signal as shown here. However, it is not easy to produce such a saw-tooth signal using op-amp. The situation is different for a digital implementation.

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



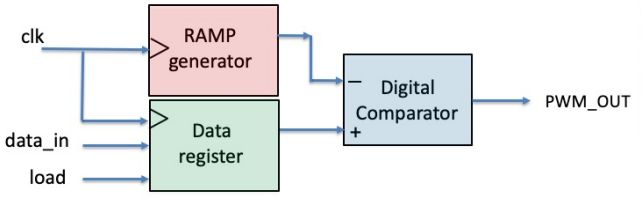
Here is a circuit schematic for a pulse-width modulated DAC. Here the counter is used to produce a count value A that ramps up linearly in a sawtooth manner. The digital value we want to convert to analogue value is data_in, which is stored as B in the input register.

A digital comparator circuit compares this input data with the counter value (which is ramping up). While A is less than B, the output of the comparator is high. As soon as A exceeds B, the output goes low. In this way, the pulse width is proportional to the value of B (or data_in) in a linear manner.

Passing this PWM signal through a lowpass filter will give an analogue output voltage that is proportional to digital value data_in. Hence this is a digital-to-analogue converter (DAC).

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
```



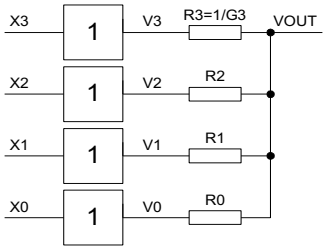

```
always_ff @ (posedge clk)
    if (load == 1'b1) d <= data_in;

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
```


Implementing a PWM DAC is extremely simple in SystemVerilog. Here is the code with colour coding for the three major blocks.

This module deliberately has reset signal missing. Instead the counter is initialized to zero in the SystemVerilog specification with “initial” keyword. This is possible for FPGA because initialization happens when you send the bitstream (sof file) to the FPGA during programming. If you were designing for an ASIC (Application Specific Integrated Circuit), which is not an FPGA, you must always use an explicit reset signal to reset all states in the chip.

Simple DAC

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



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

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

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

- 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.

The simplest DAC can be constructed using a number of resistors with binary weighted values. X[3:0] is the 4-bit digital value to be converter to an analogue voltage Vout. The 4-bit number is used as input to buffer circuits (the rectangular blocks labelled '1'). The outputs of the four buffers are V[3:0] respectively.

Using Kirchhoff current law, the current at node Vout sums to zero, and this gives the first equation. (G_0 is $1/R_0$ etc.) Rearranging the equation produces the equation for Vout.

The digital value X[3:0] can therefore be converted to an analogue voltage in the correct **binary weighting** if G3:G2:G1:G0 have the ratio of 8:4:2:1.

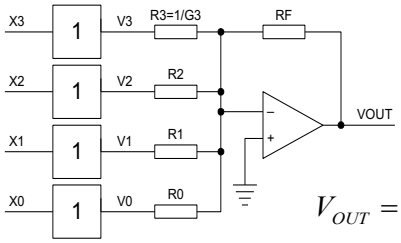
Since the digital buffer is very fast and the resistor network has no (or negligible) capacitance or inductance, this DAC can be very fast. However, this DAC has two problems:

1. The output impedance of the DAC is the Thevenin equivalent circuit resistance. Choosing too high a resistance value results in the DAC having a high output impedance; choosing too low a resistance value draws lots of current from the buffers and is inefficient on power.

2. It requires very large resistance ratio if the number of bits of X is large. For example, for a 10-bit DAC, the ratio is 1024:1. Such a DAC is difficult and expensive to manufacture.

Instead of only using binary weighting, it is possible for you to choose five arbitrary Vout values. If you add another resistor R4 connecting from Vout to the power supply, and set X[3:0] to 0000, 0001, 0010, 0100 and 1000, you can easily work out the required value of the resistances in order to give you the five arbitrary voltages.

Improved DAC with Output Op-Amp



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

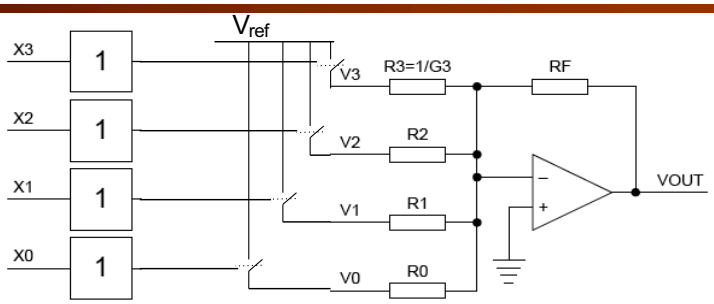
- Adding an op-amp:
  - The voltage at the junction of all the resistors (the **virtue earth** node) is now held constant by the feedback
    - Hence current drawn from V_3 is independent of the other voltages V_2, …, V_0
    - Hence any gate non-linearity has no effect ⇒ more accurate
  - Lower output 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

The high output impedance of the previous circuit can be circumvented using an operational amplifier. Shown here is a summing amplifier. Vout is given by this simple linear equation. The output impedance is that of the op amp and is very low.

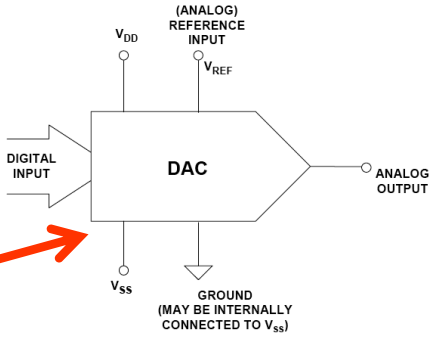
Unfortunately the output voltage of this circuit cannot change very fast. It is limited by the **slew rate** of the op amp. (Slew rate is a measure of how fast the output voltage can change, and it is in units of V/sec.)

Making binary weighted resistors is still difficult and expensive of the number of bits in the DAC is high.

Further Improvement with reference voltage source

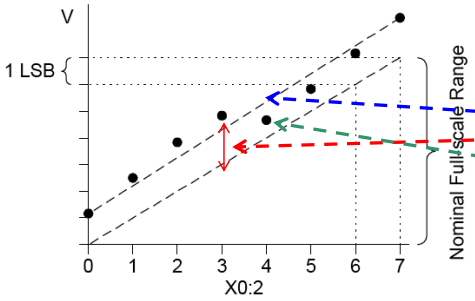


- Use digital signal to control analogue switches
- Switching Vref on/off the resistor network
- Clear separation between digital control and analogue voltages – much better accuracies
- General DAC block diagram:



Instead of driving the resistor network directly from the digital output, which is not very accurate, most DAC actually use the digital signal to control electronic switches which switch in or out a reference voltage Vref. This reference voltage can be made very accurate, thus providing accurate output voltage values.

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

PYKC 21 Nov 2023 EE2 Circuits & Systems Lecture 8 Slide 9

Here are the important specifications found in a datasheet that defines the performance of a DAC. Here we use the line from full range value to the origin as reference. We will express all voltage in terms of the delta-v corresponding to one LSB.

Resolution – the voltage step equivalent to one least significant bit (1 LSB) of the digital input. Assuming that the input is an N-bit number, then resolution of the DAC is the same as: **(full-scale voltage) / (2^N-1)**.

Accuracy – maximum error as compared to the perfect reference line (red).

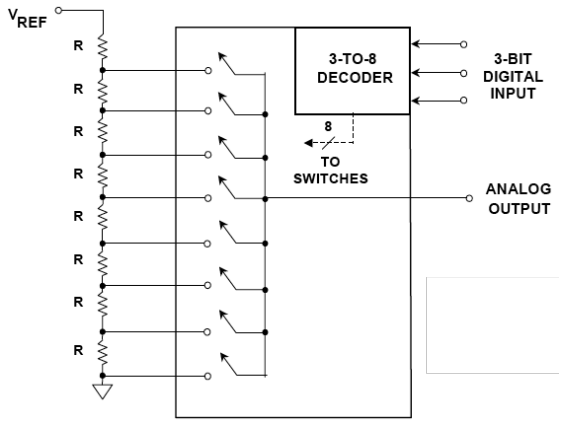
Linearity – Instead of using the reference line, we can join to max-point with the min-point to form another straight line. Linearity is the maximum deviation from this new line.

Differential Linearity – Worse case error as you step from X to X+1 for all values of X.

Monotonic DAC – One that always goes up as the input number X[3:0] increases.

Settling time – Time taken to reach final value within ±1 LSB as input changes.

Thermometer DAC using Resistor String



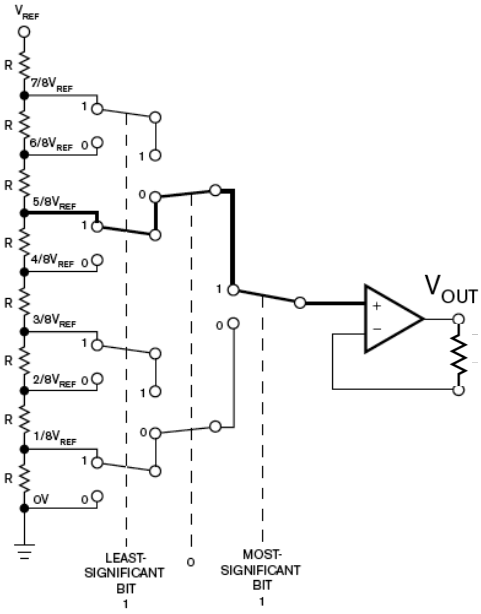
- ✓ Simple
- ✓ Inherently monotonic
- ✓ Needs only IDENTICAL resistors, good differential linearity
- ✓ Only two switches operate during a transition, low output glitch and fast settling
- ✓ Low power
- ✓ Widely use with modern technology with small feature sizes
- X Large number of resistors
- X Useful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

Instead of using binary weighted resistor network, we could use a series string of identical resistors as shown here. With this architecture, Vref to 0 is divided into 8 equal steps (including 0 value). The 3-bit digital input is decoded into 8 possible binary one-hot codes. For example, 000 results in the lowest switch being connected and 111 will switch the upper most switch on.

This DAC has the advantages listed here:

- It is simple, uses only one resistor value R everywhere, therefore easy to manufacture using semiconductor process.
- Only operating two switches at anyone time, so the glitches are smaller.
- It is low power and inherently monotonic.

Resistor String DAC with Op-Amp output



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

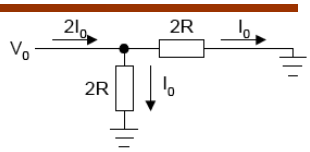
$$V_{OUT} = V_{REF} \Sigma (b_i / 2^i)$$

Instead of using a large number of switches, we can also use switches arranged in a tree structure as shown here. Here is an example showing the decoding of the digital value 3'b0101. Decoding is implicitly performed via the control of the switches using the three digital bits. The output op amp provides buffering of the DAC voltage.

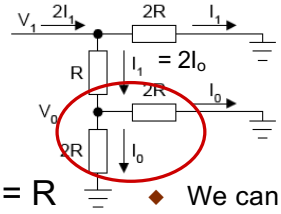
In this example, the 3'b101 digital value selects the 5/8 Vref tap of the resistor string to route to the op amp.

DAC using R-2R Ladder

We want to generate currents I_0, $2I_0$, $4I_0$, …

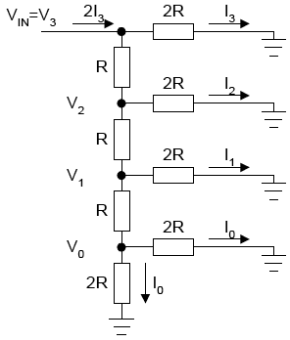


- Two $2R$ resistors in parallel means that the $2I_0$ current will split equally and equivalent resistance R



= 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^nI_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 $V_{in} = 2^N RI_0 \Rightarrow I_0 = 2^{-N} V_{in}/R$



String resistor network is good for, say, up to 10-bit DAC (requiring 1024 identical resistors). If you want a 16-bit DAC, you would need 65536 resistors! That is obviously not practical or too expensive. A better solution is to use R-2R Ladder network.

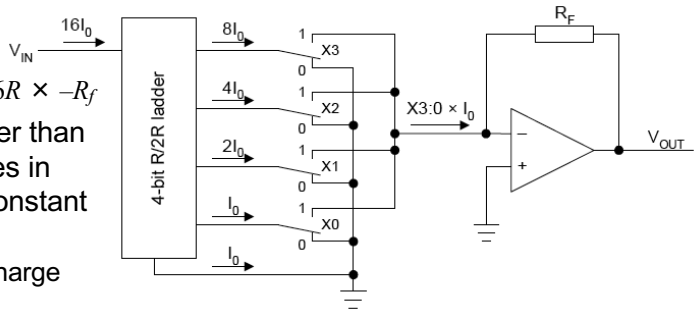
This circuit is very clever. The basic idea is to produce current `Io`, `2Io`, `4Io` etc, using only identical resistors connected in a special way.

The best way to understand the working of this R-2R network is to consider just two resistors both with values 2R. If the current flowing through each resistor is `Io`, then the total current at node Vo must be `I1 = 2Io`. The Thevenin equivalent resistance of these two resistor is 2R || 2R = R. Now we add an extra resistor R in series with these two 2R network. Together they form a resistance 2R. If we add the next step of the ladder as shown here, the total current at V1 is `2I1 = 4 Io`.

As you can see, adding each extra step of the ladder doubles the current. If the voltage drop across the horizontal resistors therefore also increases in ratios of 2 for each step.

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

For a practical DAC circuit, the R/2R ladder network is connected to the virtual earth of the op amp as shown here. The current is either sent to the virtual earth node if the digital value is '1', or switched to earth if it is '0'. In that way , the output voltage Vout is a converter analogue value of X[3:0].

Note that we switch current from one branch to another branch. It is known as **current steering**. Current steering is much faster than turning the current on and off.

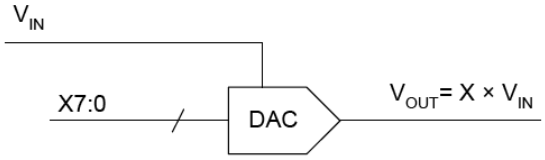
Programmable Attenuator (Amplifier)

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

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

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

- Can be used as a digital attenuator:

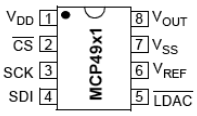


- Here the digital number X7:0 controls the gain of the circuit

Instead of using Vref, a fixed reference voltage, we could use an analogue input Vin (such as an audio signal), and then use the DAC as a digitally control amplifier or attenuator. This is also known as a multiplying DAC. The output is X multiplied by Vin.

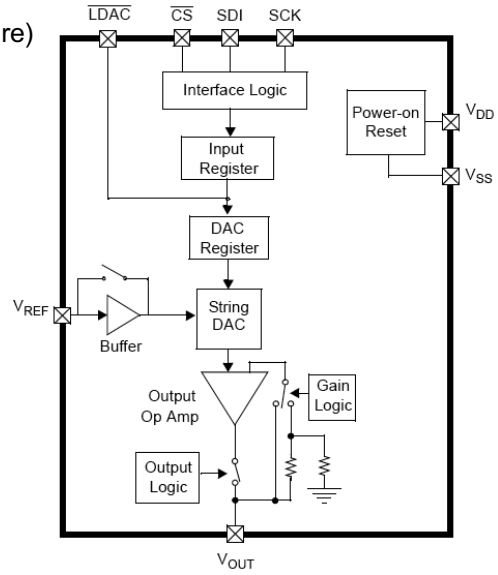
The MCP4921 DAC (used in Lab)

- **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



PYKC 21 Nov 2023 EE2 Circuits & Systems Lecture 8 Slide 15

The DAC used in the lab experiment is 12-bit. However, since the accuracy of the DAC is only up to 10-bit, only the top 10 bits are used throughout our lab sessions.

The functional block diagram of the DAC is shown in the slide. The DAC itself uses a resistor string architecture (i.e. just a bunch of 4096 series resistors of identical values). It has a selectable gain of 1X or 2X.

The DAC uses the Serial Peripheral interface (SPI) to receive the digital number to convert.

The SPI interface has four signals, which should be drive by either the microcontroller or the FPGA. In our case, LDAC is tied to GND and only the other three signals (CS, SDI and SCK) are used.

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 ($V_{OUT} = 2 * V_{REF} * D/4096$)

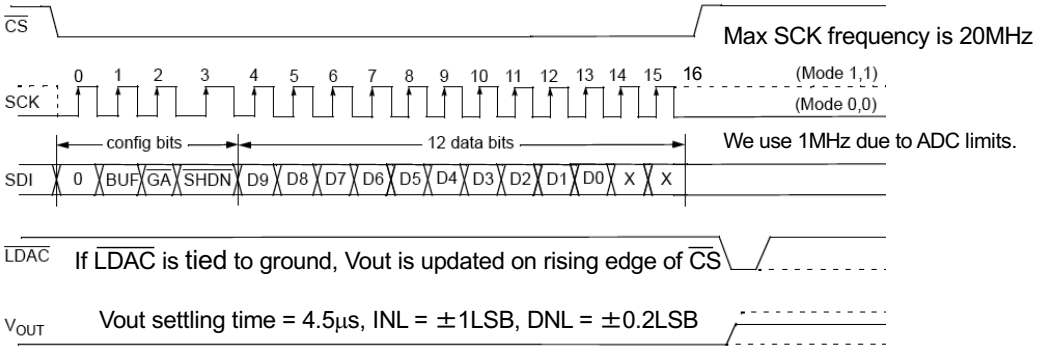
V_{REF} = 3.3V

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)



PYKC 21 Nov 2023 EE2 Circuits & Systems Lecture 8 Slide 16

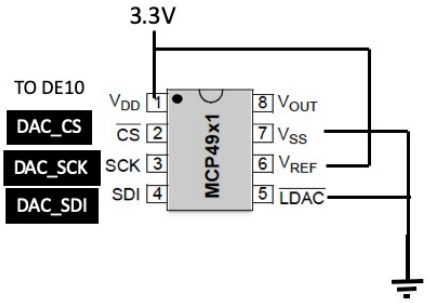
To send a value to the DAC to output (i.e. produce the analogue output Vout), a 16-bit value is sent to the DAC chip in a serial manner. The Chip Select (SC) signal going low indicate that this is the start of the data. This establishes the beginning of the data frame. First data bit (bit 15) is always 0. Bit 14 determines whether the reference voltage (Vreg) is buffered or not buffered (via an internal opamp). For our design, Vref is around 3.3V.

Bit 13 determines the gain of the DAC (x1 or x2). Bit 12 is set to 1 if you are using the DAC, and set to 0 if you want to shutdown the device to conserve power.

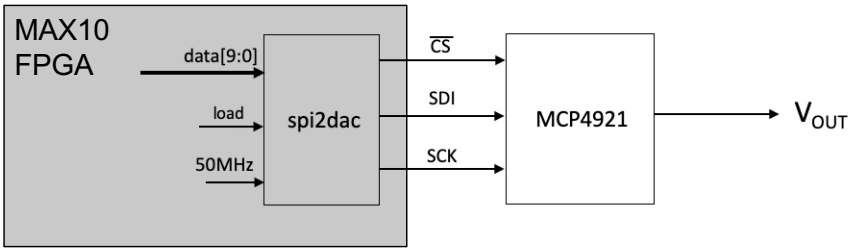
Bit 11 to 0 contains the 12-bit data D[11:0] to convert into analogue voltage Vout, MSB first. For our lab experiments, we extract the most significant 10-bits for various practical reasons.

The LDAC (low active) signal can be connected to ground or used a low active strobe signal to transfer the data to the DAC register (i.e. tell the DAC to update Vout). If LDAC is low, DAC update happens on rising edge of CS_bar.

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

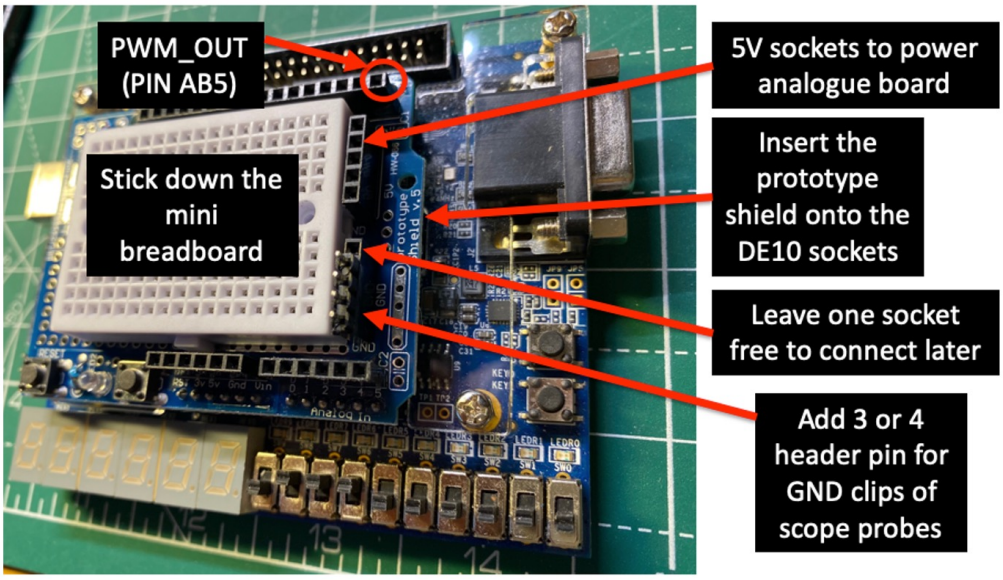


PYKC 21 Nov 2023 EE2 Circuits & Systems Lecture 8 Slide 17

This is a simplified diagram showing how the MAX10 FPGA is interfaced to the MCP4921 DAC.

The interface between the FPGA chip and the converters is through the SPI bus. You are provided with the interface module, spi2dac.v. How spi2dac module works will be explained in a later lecture.

Lab 5 – DAC conversion – Prototype Shield

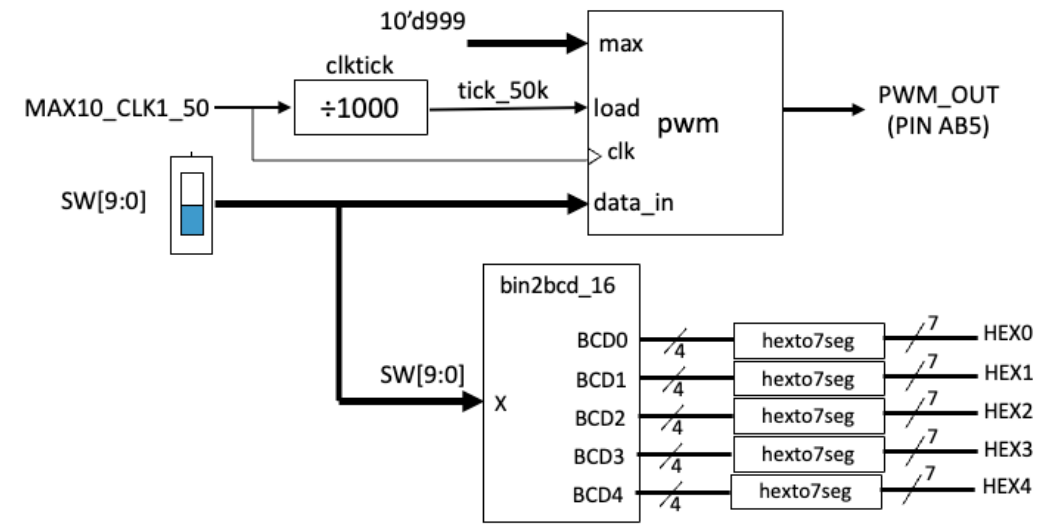


PYKC 21 Nov 2023 EE2 Circuits & Systems Lecture 8 Slide 18

You will have a chance to try out the contents of this Lecture when you do Lab 5. Before you start, you need to add a prototyping add-on board (a shield) to the DE10-Lite module as shown in the slide here. You will then add on the breadboard the MCP4921 digital-to-analogue converter chip. Later in Lab 6, you will add another chip for analogue-to-digital conversion.

Note that in Task 1 of Lab 5, you will connect the DE10 with the prototyping shield to your analogue circuit breadboard. The DE10 provides the 5V and GND supply (indirectly via the DE10's USB cable to your computer). Interface between the DE10 FPGA and the rest of your system is via the large 40-pin socket at the top of the DE10-Lite module.

Lab 5 Task 1 PWM as a DAC converter



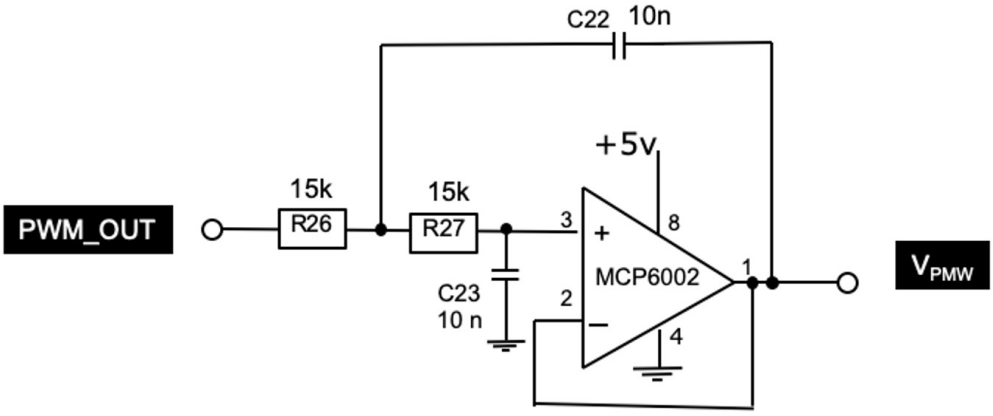
PYKC 21 Nov 2023 EE2 Circuits & Systems Lecture 8 Slide 19

Lab 5 Task 1 is to build and test a PWM as explained in this lecture. A digital value is supplied via SW[9:0]. The pwm.v entity produces a digital output signal PWM_OUT whose duty cycle is proportional to the digital input **data_in**.

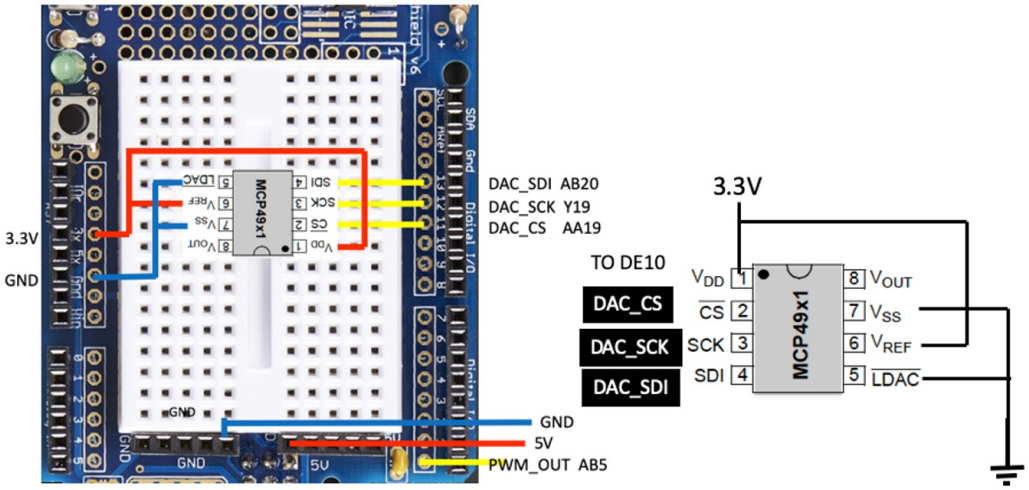
The clktic_16.v module (explained in Lecture 7) is used to provide a 50kHz sampling command signal (tick_50k) so that the PWM performs one conversion every 20 microseconds (period of 50kHz).

You can now use the lowpass filter you built in Lab 2 to produce an analogue voltage V_{PWM}, which is the converter anal

ogue voltage for the digital input!



Lab 5 Task 2 Using the MCP4921 chip



Lab 5 Task 2 requires you to add the DAC chip to the DE10 to produce directly the analogue output voltage from the digital input. The MCP4921 chip uses the SPI interface. A module spi2dac.v is provided. The detail working of spi2dac.v will be explained in a later lecture. For now, just used this as a given component.

In this task, you will be able to compare the analogue output of the DAC chip to that produced in task 1 using PWM + lowpass filter.