1. **Digital To Analogue Converter (DAC)**

A digital-to-analogue converter (DAC) is a chip or circuit that converts a number (digital) into a voltage or current (analogue). The DAC is a useful interface between a computer and an output transducer. For example, DACs are used to control devices that require a continuous range of control voltages or currents such as electro-acoustic transducers (speakers), some types of variable-speed motors, and many other applications where an analogue signal output is required. Another common application is to re-create waveforms from digital signals – for example in CD players.

![Figure 1: Example Digital-to-Analogue Converter](image-url)
### 1.1 Example: 4-bit DAC

An \( N \)-bit DAC can output \( 2^N \) different levels in \( 2^N - 1 \) steps so if we have an 4 bit DAC then this gives us \( 2^4 = 2^4 = 16 \) different output levels in 15 steps.

\[
V_{o} = V_{ref} \left( \frac{D_3}{2} + \frac{D_2}{4} + \frac{D_1}{8} + \frac{D_0}{16} \right) \quad \text{or} \quad V_{o} = V_{ref} \left( \frac{D}{2^4} \right) = V_{ref} \left( \frac{D}{16} \right)
\]

where \( D \) = decimal value of the digital input.

The Output voltage ranges from 0 to \( V_{ref}(15/16) \)

Table below shows the output voltage for a reference voltage of 10V and various digital inputs.

<table>
<thead>
<tr>
<th>A3</th>
<th>A2</th>
<th>A1</th>
<th>A0</th>
<th>Decimal</th>
<th>Vo</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.000</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0.625</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1.250</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>1.875</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>2.500</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>5</td>
<td>3.125</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>6</td>
<td>3.750</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>7</td>
<td>4.375</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>5.000</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>9</td>
<td>5.625</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>10</td>
<td>6.250</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>11</td>
<td>6.875</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>252</td>
<td>7.500</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>253</td>
<td>8.125</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>254</td>
<td>8.750</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>255</td>
<td>9.375</td>
</tr>
</tbody>
</table>

Note: The maximum output voltage of the DAC < \( V_{ref} \). As the number of bits increases the maximum output voltage -> \( V_{ref} \).

**Example 1**
What is \( V_{o} \) of a 4-bit DAC if the input is 0x7 and \( V_{ref} = 10V \)?

Answer 1: \( 0x7 = 0111 \) = 7 decimal. So \( V_{o}=10(7/2^4) = 70/16 = 4.374V \)

**Exercise 1**
What is the maximum output voltage \( V_{o} \) of a 4-bit DAC if \( V_{ref} = 5V \)?
1.2 Example: 8-bit DAC

An \(N\)-bit DAC can output \(2^N\) different levels in \(2^N-1\) steps so if we have an 8 bit DAC then this gives us \(2^8 = 256\) different o/p levels in \(2^7 = 128\) steps.

\[
V_o = V_{ref} \left( \frac{A_7}{2} + \frac{A_6}{4} + \frac{A_5}{8} + \frac{A_4}{16} + \frac{A_2}{32} + \frac{A_1}{64} + \frac{A_0}{128} \right)
\]

or

\[
V_o = V_{ref} \left( \frac{D}{2^8} \right) = V_{ref} \left( \frac{D}{256} \right)
\]

where \(D\) = decimal value of the digital input.

Table below shows the output voltage for a reference voltage of 10V and various digital inputs.

<table>
<thead>
<tr>
<th>A7</th>
<th>A6</th>
<th>A5</th>
<th>A4</th>
<th>A2</th>
<th>A1</th>
<th>A0</th>
<th>Decimal</th>
<th>Vo</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.000000</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0.039063</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0.078125</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>0.117188</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>0.156250</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>0.195313</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>6</td>
<td>0.234375</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>7</td>
<td>0.273438</td>
</tr>
</tbody>
</table>

Example 2

What is \(V_o\) if the input is 0x30 and \(V_{ref} = 10V\)?

Answer 1: 0x30 = 00110000b = 48 decimal So \(V_o=10(48/256) = 1.875V\)

Exercise 2

What is \(V_o\) if the input is 0x73 and \(V_{ref} = 10V\)?
2. Function Generation Using DAC

In this section we look at developing a simple C/C++ program to generate waveforms using an 8-bit DAC.

2.1 The Ramp Generator:

The program Ramp.c shown below generates a sawtooth waveform by inputting incrementing numbers from 0 to 255 to an 8-bit DAC connected to a PC via the parallel interface.

Example: ramp.cpp

```c
#include <conio.h>
void delay(int time);
#define DATA_PORT  0x378 // Address of Data Register
#define CONTROL_PORT 0x37A  // Address of Control Register

int main()
{
    _outp(CONTROL_PORT,0x01);   //configure the port to write out

    for(int count=0;count<=255;count++)
    {
        _outp(DATA_PORT,count);  //output the data to the port
        delay(100);    //add a delay to vary frequency
        if(count==255)
        {
            count=0;
        }
    }
    return 0;
}
// Rough delay function
void delay(int time)
{
    for (int i=0;i<time;i++)
    for (int j=time;j>0;j--);
}
```

Viewing the output of the DAC (assume Vref=10V) produces the following sawtooth waveform.

![Sawtooth Waveform](image)

A low pass filter can be used to smooth out the digital stairsteps.
Flowchart for Ramp.cpp program

2.2 In class Exercise: Triangular Waveform
Using the Ramp.cpp program as a template, write a C++ program that generates a triangular waveform.

```cpp
#include <conio.h>
void delay(int time);
#define DATA_PORT 0x378    // Address of Data Register
#define CONTROL_PORT 0x37A // Address of Control Register

int main()
{
    _outp(CONTROL_PORT,0x01);
    //configure the port to write out
    while(true)
    {
        for(int count=0;count<=255;count++)
        {
            _outp(DATA_PORT,count);  //output the data to the port
            delay(100);  //add a delay to vary frequency
        }
        for(count=254;count>=0;count --)
        {
            _outp(DATA_PORT,count);  //output the data to the port
            delay(100);  //add a delay to vary frequency
        }
    }
    return 0;
}
```

// Rough delay function
void delay(int time)
{
    for (int i=0;i<time;i++)
    {
        for (int j=time;j>0;j--);
    }
}

Draw a flowchart that describes a program that generates a triangular waveform.
2.3  **In Class Exercise: Sinusoidal Waveform**

- Using the `Ramp.cpp` program as a template, write a C++ program that generates a sinusoidal waveform.

- Draw a flowchart that describes a program which generates a sinusoidal waveform.