EXPERIMENT NO.-3

TITLE:
Waveform Generation using DAC

AIM: To Generate Different Types of Waveforms Using DAC 0808

OBJECTIVE:

1. To understand the concept waveform generation
2. Understand the interfacing diagram with microcontroller
3. Able to write programs foe generation of waveforms

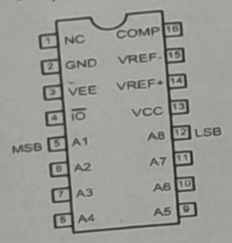
THEORY: The different types of waveforms can be generated using DAC 0808 when interfaced with 8051 Microcontroller. The DAC0808 is an 8-bit monolithic digital-to-analog converter (DAC) featuring a full scale output current settling time of 150 ns while dissipating only 33 mW with ±5V supplies No reference current (IREF) trimming is required for most applications since the full scale output current is typically ±1 LSB of 255 IREF/256. Relative accuracies of better than ±0.19% assure 8-bit monotonicity and linearity while zero level output current of less than 4 μA provides 8-bit zero accuracy for IREF³2 mA. The power supply current of the DAC0808 is independent of bit codes, and exhibits essentially constant device characteristics over the entire supply voltage range. The DAC0808 will interface directly with popular TTL, DTL or CMOS logic levels,

The DAC 0808 has following features

1. Relative accuracy: ±0.19% error maximum
2. Full scale current match: ±1 LSB typ
3. Fast settling time: 150 ns typ
4. Non inverting digital inputs are TTL and CMOS compatible
5. High speed multiplying input slew rate: 8 mA/μs

6. Power supply voltage range: ±4.5V to ±18V
7. Low power consumption: 33 mW @ ±5V

The pin out diagram for DAC 0808 is shown in figure below,



The pins are labeled A1 through A8, but note that A1 is the Most Significant Bit, and A8 is the Least Significant Bit (the opposite of the normal convention). Ground the two least significant bits. The D/A convertor has an output current, instead of an output voltage. The output pin should stay at about 0 volts. The op-amp on the "Typical Application" on the datasheet converts the current to a voltage. How does it do this? The output current from pin 4 ranges between 0 (when the inputs are all 0) to Imax *255/256 when all the inputs are 1. The current, Imax , is determined by the current into pin 14 (which is at 0 volts). Note: Since we are using 8 bits, the maximum value is IMAX *255/256. The output of the D/A convertor takes some time to settle. You may need to take this in consideration when planning the timing of the A/D conversion.

The D/A converter have an output current, instead of an output voltage. The output pin should stay at about 0 volts. It uses I to V converter at the output pin of DAC

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

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

$$Vo = R_f . I_o$$

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

ALGORITHM: Depending on type of waveform generated algorithm changes

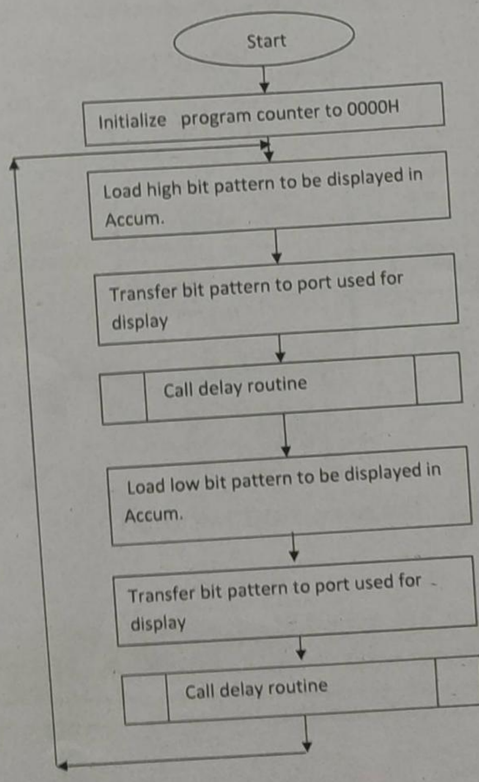
FOR square Wave: it can be generating on single port pin or at all the port pins

1. Load high[0ff] data into accumulator
2. Transfer it to port
3. Delay: Wait for some time depending on duty cycle

4. Output Low (00) to port
5. Delay:
6. Repeat Steps 1-5

Wave forms can be generated with change in reference point. Either by software or by Hardware

Flow chart



Interfacing diagram: The complete interface of DAC with 8051 is shown in fig (1) or fig (2)

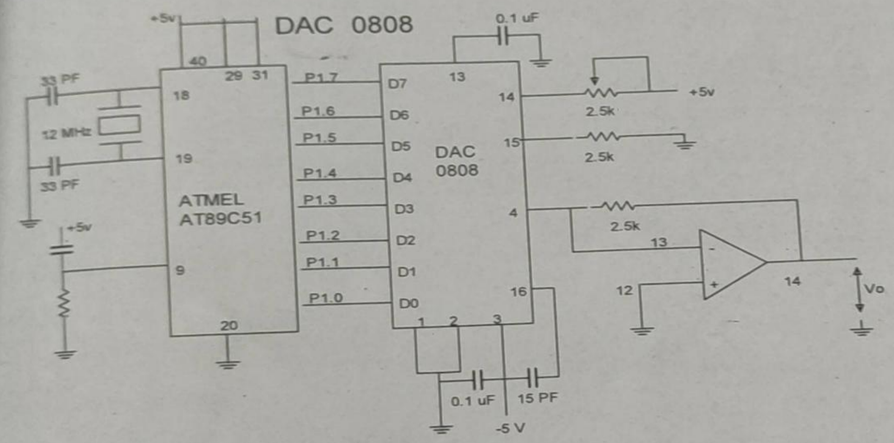


Fig (1): DAC interface with 8051

Conclusion:

In this practical we have studied about To generate different types of waveforms using DAC 0808

Experiment NO:03

```
//square waveform generation

#include<reg51.h>

void delay()
{
    int i,j;
  for(i=0;i<100;i++)
      for(j=0;j<100;j++);
}

void main()
{
while(1)
 {
P2=0x00          //logic0 of square wave.
  delay()
P2=0xff;         //logic1 of square wave.
delay();
    }
  }
```

```
//triangular waveform generation.

#include <reg51.h>

unsigned char d;
void main(void)
{
 while(1)
 {
   for(d=0; d<255; d++)
    {
      P2=d;
    }
 for(d=255; d>0; d--)
   {
      P2=d;
   }
 }
```