Course Name:	**Microprocessors and Microcontrollers Laboratory**	**Semester:**	**IV**
Date of Performance:	**08 / 04 / 2024**	**Batch No.:**	**A - 2**
Faculty Name:	**Kirti Sawlani**	**Roll No.:**	**16014022050**
Faculty Sign & Date:		**Grade / Marks:**	**___ / 25**

Experiment No.: 9

Title: Virtual Lab: MCU-DAC interfacing and generation of ramp wave

Aim and Objective of the Experiment:

Aim:

1. **Write Program to interface DAC with 8051 microcontroller and generate a ramp signal by programming the microcontroller.**

COs to be achieved:

CO1: Explain salient features of 8051 microcontroller and describe different components of the microcontroller and its operation.

CO2: Develop assembly language programs using instructions set of 8051 microcontrollers.

CO5: Write Embedded-C programs to interface different on-chip as well as off-chip peripherals with 8051/PIC18F452 microcontroller and explain different types of serial communication protocols of PIC18F452 microcontroller.

Tools Required:

Hardware:

MCU (AT89C51) , DAC0808, Counter (IC 79LS43), Resistors (10k, 8.2k), Capacitors(33pF, 10uF), Op-amp, Signal Generator, Oscilloscope, Power Supply

Software:

- Proteus
- Keil Microvision

Theory:

The digital-to-analog converter (DAC) is a device widely used to convert digital pulses to analog signals. There are two methods to create a DAC: binary weighted and R/2R ladder. Most of the integrated circuit DACs, including the MCl408 (DAC0808) used in this section, use the R/2R method because a much higher degree of precision can be achieved by it. The criterion for judging a DAC is its resolution, which is a function of the number of binary inputs. The common ones are 8, 10, and 12 bits. The number of data bit inputs decides the resolution of the DAC because the number of analog output levels is equal to `2^n`, where n is the number of data bit inputs.

Therefore, an 8-input DAC such as the DAC0808 provides 256 discrete voltage (or current) levels of output. Similarly, the 12-bit DAC provides 4096 discrete voltage levels. There are also 16-bit DACs, but they are very expensive. ln the MCl408 (DAC0808), the digital inputs are converted to current (`I_(out)`), and by connecting a resistor to the `I_(out)` pin, we convert the result to voltage.The total current provided by the `I_(out)` pin is a function of the binary numbers at the D0-D7 inputs of the DAC0808 and the reference current (`I_(ref)`), and is represented as: `I_(out)=I_(ref)( (D_7)/2 + (D_6)/4 + (D_5)/8 + (D_4)/6 + (D_3)/32 + (D_2)/64 + (D_1)/128 + (D_0)/256 )`, where `D_0` is the LSB, `D_7` is the MSB for the inputs, and `I_(ref)` is the input current that must be applied to pin 14. The `I_(ref)` current is generally set to 2.0 mA.

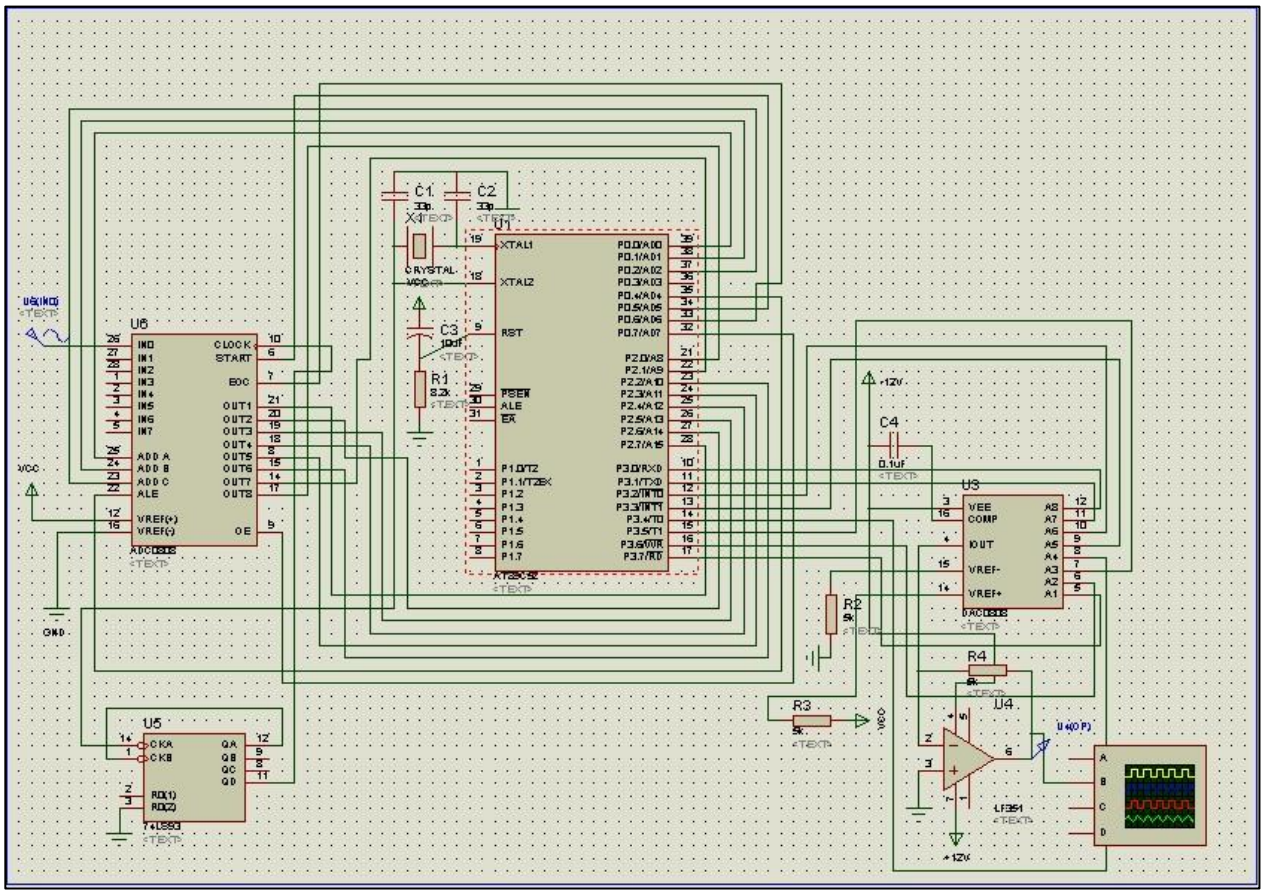
The generation of current reference (setting `I_(ref)` = 2 mA) is done by using the standard 5V power supply.

Ideally, we connect the output pin lout to a resistor to convert the current to voltage, and monitor the output on the scope. In real life, however, this can cause inaccuracy because the input resistance of the load will also affect the output voltage. For this reason, the `I_(ref)` current output is isolated by connecting it to an op-amp such as the 741 with the the feedback resistor Rf = 5 kilo-ohms. Assuming that R = 5 kilo-ohms, the output voltage changes with the change in the binary input.

Stepwise-Procedure:

1. Using Keil micro vision to generate hex file for the C code
2. Create the above block diagram using Proteus software.
3. Upload the hex file on microcontroller in Proteus.
4. Run the Proteus simulation.

Block Diagram:

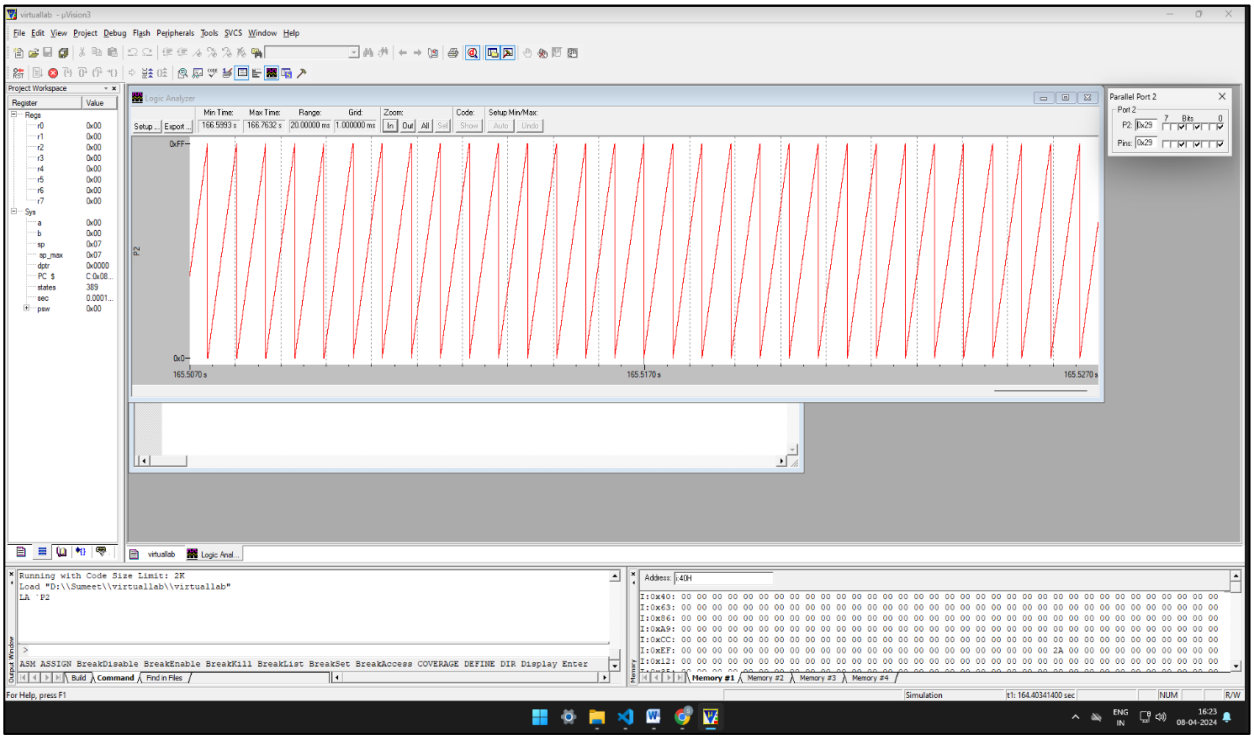


Code:

```
# include <reg51.h>
int main (void)
{
        unsigned char i = 0;      // Define a counter
        P2 = 0xFF;                // Make port P2 as output port
       while (1)                  // Do forever
                {
                  P2 = i;         // Copy i into port P2 to be converted
                   i++;           // Increment the counter
                 }
        return 0;
```

Output Screenshots:

Observe the ramp wave generated on the oscilloscope.



Post Lab Subjective/Objective type Questions:

1. **Explain the method to generate sine wave.**

   Generating a sine wave in a microprocessor or microcontroller involves using the computational capabilities of the device to calculate discrete points along the sine wave and then outputting these points to an external digital-to-analog converter (DAC) to generate the analog waveform. Here's a basic overview of the method:

   - Algorithm Selection:
     Choose an algorithm to generate discrete points along the sine wave. Some common algorithms include lookup tables, Taylor series, and direct digital synthesis (DDS).

   - Lookup Table Method:
     Pre-calculate a lookup table containing the amplitude values corresponding to different angles (or time intervals) of the sine wave. This table can be stored in the microcontroller's memory.
     During runtime, the microcontroller accesses the lookup table and retrieves the appropriate amplitude values based on the current angle (or time interval).

- Taylor Series Method:
  Implement a truncated version of the Taylor series expansion for the sine function.
  Calculate each term of the series up to a certain order to approximate the sine value.
  This method requires more computational resources but can be implemented without pre-calculated tables.

- Direct Digital Synthesis (DDS) Method:
  DDS generates sine waves by accumulating phase increments at a constant rate and converting the resulting phase value into a sine wave using a digital lookup table or algorithm.
  This method offers precise control over frequency and phase but may require more computational resources.

- Output to DAC:
  Convert the calculated amplitude values (digital representation of the sine wave) into analog voltage levels using a digital-to-analog converter (DAC).
  The DAC output is then filtered to remove any high-frequency components introduced by the digital-to-analog conversion process, resulting in a clean sine wave signal.

- Output Handling:
  The microcontroller continuously generates and outputs the discrete points of the sine wave to the DAC at regular intervals.
  The output waveform can be adjusted by changing parameters such as frequency, amplitude, and phase increment, allowing for dynamic control of the generated sine wave.

2. Write C code for generation of sine wave.

```
#include <math.h>
#include <stdint.h>

// Define parameters
#define SAMPLE_RATE     8000    // Sample rate in Hz
#define DURATION        5       // Duration of the sine wave in seconds
#define AMPLITUDE       127     // Amplitude of the sine wave

// Function to generate sine wave samples
void generateSineWave(int16_t *buffer, uint32_t num_samples) {
    double frequency = 440.0;  // Frequency of the sine wave in Hz (A4 note)
```

```
    double increment = 2 * M_PI * frequency / SAMPLE_RATE;
    double angle = 0.0;

    for (uint32_t i = 0; i < num_samples; i++) {
        buffer[i] = (int16_t)(AMPLITUDE * sin(angle)); // Calculate sine value
        angle += increment;
    }
}

int main() {
    uint32_t num_samples = SAMPLE_RATE * DURATION;
    int16_t samples[num_samples];

    // Generate sine wave samples
    generateSineWave(samples, num_samples);

    // The generated samples can now be used for further processing or output

    return 0;
}
```

Conclusion:

We successfully completed experiment 9 in which we learnt to interface DAC with a microcontroller and using it, generate a ramp and sine wave.

Signature of faculty in-charge with Date: