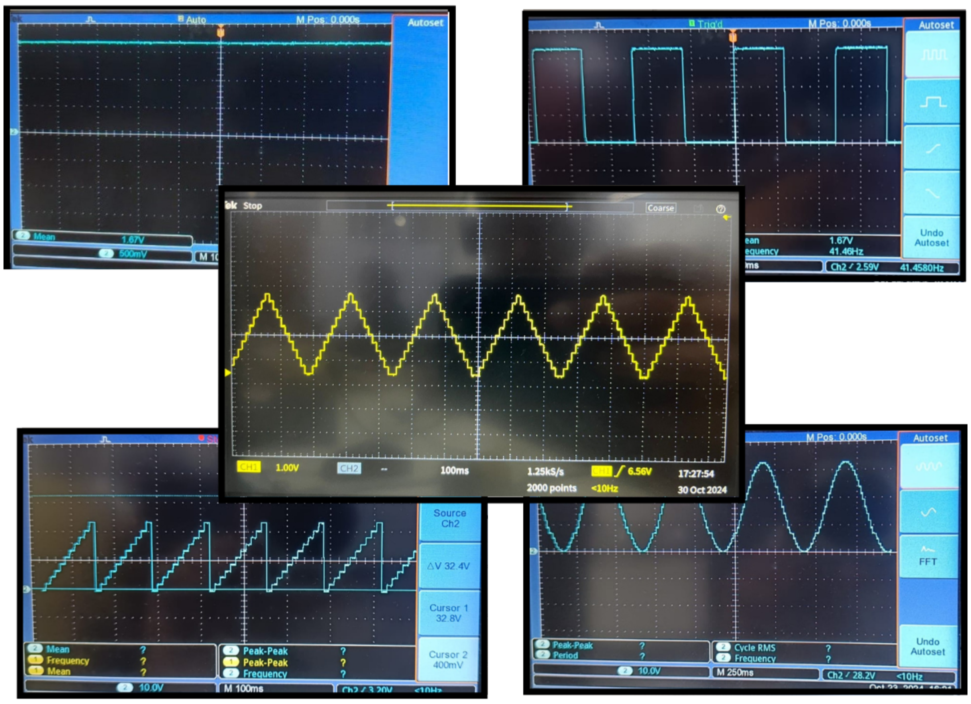


EE2016: Assignment-9

By Siva(EE23B151),
Bhavesh(EE23B017)
and Naveen(EE23B113)

Group 28

Date: 3/11/2024

Indian Institute of Technology Madras

Contents

1 Introduction

This report details various tasks involving the generation of different waveforms using an ARM-based microcontroller. The objective is to programmatically produce **Sine**, **DC**, **Triangular**, and **Staircase** signals, each with specific characteristics. We can produce such **continous** waveforms using **digital** values with the help of **DAC** circuits.

1.1 Digital-to-Analog Converter (DAC)

These are electronic devices or circuits that converts digital signals, typically binary data, into an analog voltage or current. They work by assigning specific analog output levels to digital input codes, with higher-resolution DACs providing finer granularity and more accurate signal representation.

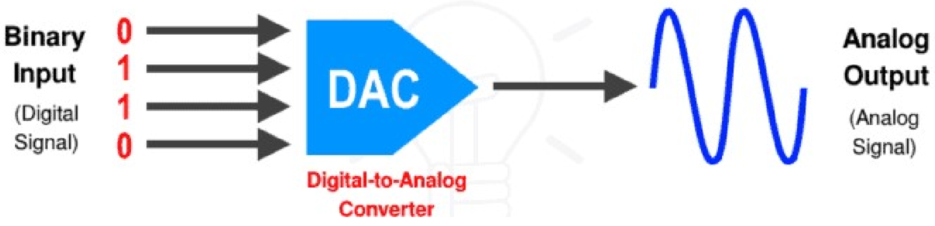


Figure 1: Input and Output of DAC circuit

1.2 DAC register in LPC2148

The DACR in this micro-controller is of 32 bits:

- Bits 31:17 and 5:0 are **Reserved** bits.
- Bits 15:6 are used to give Digital **Values** which are to be converted to Analog.
- Bit 16 is used for Bias, which is for initializing **Settling time** and **Current consumption**. (We go with Bias mode 1 in this expt)

The details of **DACR** register bits (**Reserved** bits, **Value** bits and **Bias** bit) are explained in this **Link**.

2 Tasks

For all the tasks below, the structure of C code is same:

- Initialize the DAC register. (done using `DACInit()`)

```c
#define DAC_BIAS 0x00010000      /* Bias mode 1 */
void DACInit(void){
    /* setup the related pin to DAC output */
    PINSEL1 &= 0xFFF3FFFF;
    PINSEL1 |= 0x00080000;       /* set p0.25 to DAC output */
    return;
}
```

- Looping through hard-coded values with some delay in between. (using `mydelay()`)

```c
void mydelay(int x){
    int j,k;
    for(j=0;j<=x;j++){
        for(k=0;k<=0xFF;k++);
    }
}
```

2.1 Task 1: Generating a Sine Wave

To create a sine wave, we programmed the LPC2148 microcontroller to convert digital signals to analog output. The resulting analog signal approximates a periodic sine wave with a peak-to-peak voltage of 2.38V, achieved by setting the DAC register (DACR) values as shown below:

```c
// Set maximum value for DACR to approximate peak voltage
DACR = 0x2E2;  // Max amplitude setting for 2.38V peak-to-peak
// Increment phase (0 x200 to 0 x3FF in steps)
// Decrement phase (0 x3FF back down to 0x000 in steps )
// Set minimum value for DACR to approximate trough voltage
DACR = 0x00;   // Min amplitude setting
```

Code for this task is in this **link**.

2.2 Task 2: Finding Maximum Amplitude of the Sine Wave

To determine the maximum possible peak-to-peak voltage, we modified the DACR values to utilize the full DAC range:

```c
// Maximum amplitude for full DAC range
DACR = 0x3FF;  // Max DACR value for highest peak

// Minimum amplitude for full DAC range
DACR = 0x00;   // Min DACR value for lowest trough
```

This ensures the waveform uses the entire DAC range, yielding the maximum peak-to-peak voltage.

2.3 Task 3: Generating a DC Signal

To create a stable DC signal, we set the DACR value to a constant and maintained it within an infinite loop. For a target output of 1.25V, the DACR value is set as follows:

```
// Set DACR to maintain 1.25V output
while(1){
    DACR = (0x184 << 6);
}
```

This configuration holds the output voltage steady at 1.25V. Code for this task is in this link.

2.4 Task 4: Generating a Triangular Wave

To generate a triangular wave, the DACR value is adjusted incrementally to form a linear ramp up and down:

```
// Repeat the above step incrementing by 0x066 until 0x3FF (Peak)
DACR = (0x3FF << 6);
mydelay(0x0F);

// Decrement DACR from 0x3FF to 0x000 in steps of 0x066 until 0x000 (Back to
    minimum)
DACR = (0x399 << 6);
mydelay(0x0F);
```

Adjusting the delay between DACR updates changes the frequency of the triangular wave. Code for this task is in this link.

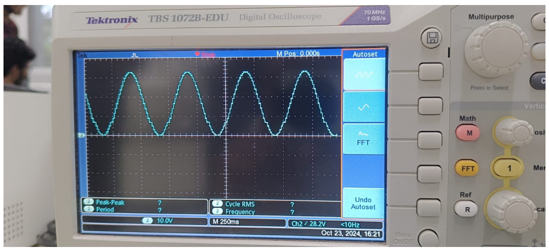
2.5 Task 5: Generating a Staircase Waveform

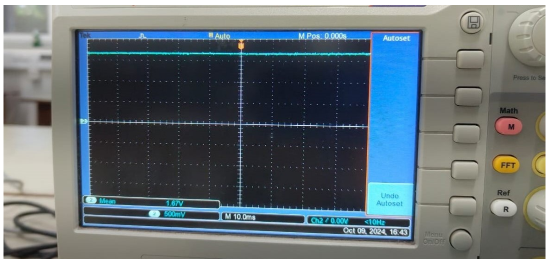
To produce a staircase waveform, the DACR is incremented in steps with two constant values per level, creating a stair-step pattern. The code snippet is as follows:

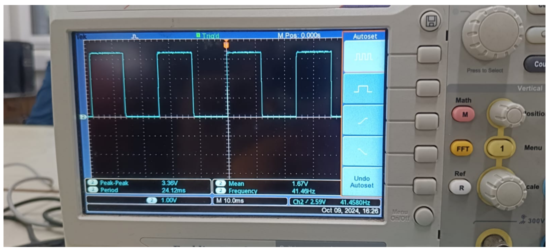
```
// Increment DACR from 0x000 to 0x2E2 in steps of 0x048
DACR = (0x000 << 6);
mydelay(0x0F);

// Repeat the above step incrementing by 0x048 until reaching 0x2E2 (Final
    step, maximum)
```

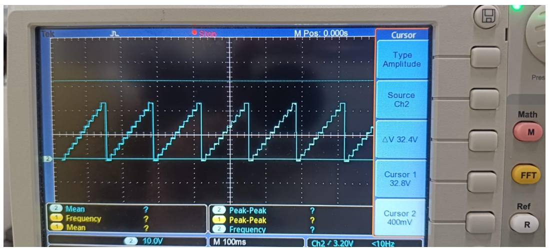
After reaching the maximum step, resetting DACR to zero creates a vertical drop, completing the staircase pattern. Code for this task is in this link.

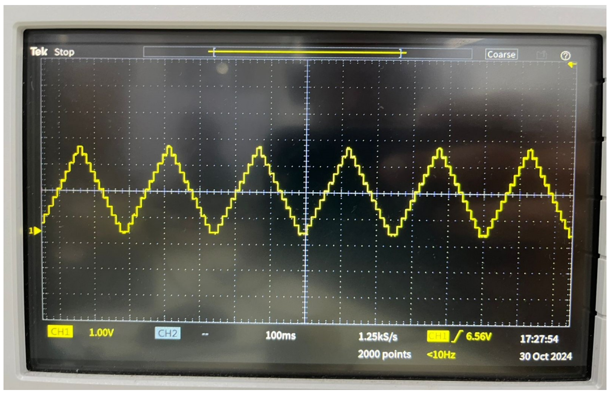
3 Outputs


(a) Sine wave


(b) DC voltage


(c) Square wave


(d) Stair-case wave


(e) Triangle wave