Pratyush Taiswal
18EE30021

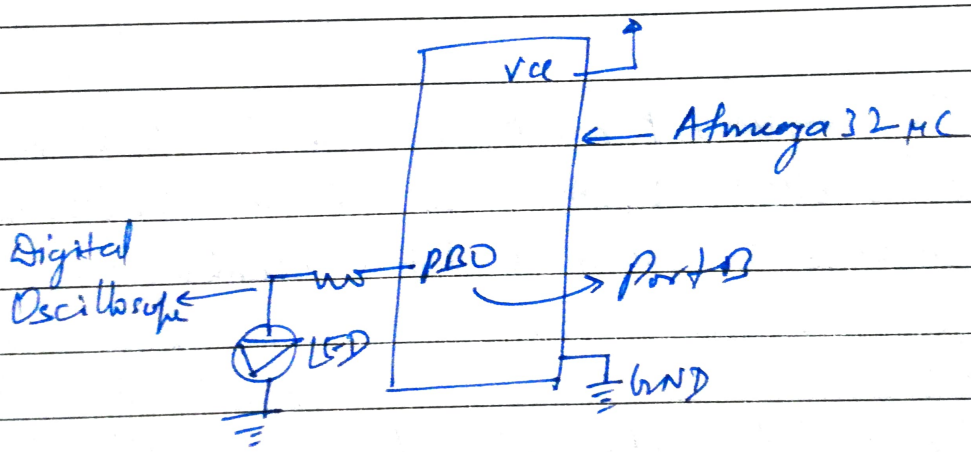
AIM: To implement an AVR code to blink an LED on Proteus Software on Atmega 32 uC and analyse it using a digital oscilloscope.

Procedure:
→ Algorithm to make LED blink.
→ AVR code written in atmel studio.
→ The hex file was obtained from the project developed to program the microcontroller in proteus
→ Then the circuit was made in proteus.
→ Output observed in a digital oscilloscope.
→ ~~Analysis~~ Timing Analysis was done for code written.

Algorithm Developed for LED blinking:
→ PORT was initialized as output & stack pointer also initialized.
→ A main loop developed for setting output pin High & low with continuous delays in between.
→ Delay subroutines defined and implemented as a nested loops where two registers were initialised & decreased to 0.
→ Branching statements were used to control sequence in delay subroutine.
→ Control returned to main loop after execution of delay subroutine.

Date: __/__/__
Page: _____

Circuit Diagram:



	CYCLES	N	Explanation
.INCLUDE "M32DEF.INC"	0	1	Adding board support package
.ORG 0x00	0	1	Directive to set PC at 0x00 start
LDI R16, HIGH(RAMEND)	1	1	load R16 SFR with address of last SRM location
OUT SPH, R16	1	1	Set upper byte of pointer of stack to SRAM end
LDI R16, LOW(RAMEND)	1	1	Load R16 SFR with address of penultimate SRM location.
OUT SPL, R16	1	1	Set lower byte of pointer to SRAM end.
LDI R16, 0x01	1	1	Load R16 with value 1
OUT DDRB, R16	1	1	Set PortB as output
Loop:	0	1	Main loop.
LDI R16, 0x01	1	1	Load SFR R16 with value 1.
OUT PORTB, R16	1	1	Set PB0 as output high
RCALL DELAY1	2	1	Transfer control to DELAY1 subroutine
LDI R16, 0x00	1	1	Load SFR R16 with value 0
OUT PORTB, R16	1	1	Set PB0 as low
RCALL DELAY1	2	1	Transfer Control to DELAY1
RJMP LOOP	2	1	Repeat main loop LOOP
DELAY1:	0	1	Delay1 subroutine
LDI R16, 0xFF	1	1	Set outer loop register as 0xFF.
LOOP1:	0	255	to define no. of loop1 executions
LDI R17, 0xFF	1	255	Set inner loop 2 counter register
LOOP2:	0	255x255	as 0xFF for 255 inner loop runs.
DEC R17	1	255x255	Decrease inner loop counter
BRNE LOOP2	0.5	(255x255+1)	Branch if inner loop counter is zero.
DEC R16	1	255	Decrease outer loop counter
BRNE LOOP1	0.5	(255+1)	Branch if outer loop counter is zero.
RET	4	1	Return to LOOP (main loop).

Date: __/__/____
Page: _____

Total Execution Time:

$$= \Sigma(\text{cycle count}) \times (\text{No. of executions in one run of loop})$$

$$= 6 + 10 + 2\times\left(1 + 4\times 1 + 255\times\left(1 + 255\times(1+0.5)+0.5\right)\right)$$

$$= \qquad\qquad \uparrow \text{Delay cycle Count (D)}$$

$$= 195868 \text{ clock cycles}$$

Time period $= \frac{1}{f} = \frac{1}{1\,MHz} = 1\,\mu s.$

Execution Time $= 195868 \times 1\,\mu s = 195.868\ ms.$

Delay time $= D \propto \frac{1}{f}$

$= 97926\,\mu s = 97.926\ ms.$

DISCUSSION:

→ In this experiment, programming a microcontroller was demonstrated with an example of blinking led on a microcontroller Atmega32.

→ The programming was done in Assembly Language.

→ Different Instruction sets were discussed.

→ How the stack pointer works and link between getting the location of memories & registers.

→ How the subroutines work & also the different timing analysis were done.

→ Finally, the output waveform was observed in a digital oscilloscope.

Pratyush Jaiswal
18EE30021

AIM: Implementation of Digital FIR filters.

Theory: All digital, linear, time-invariant (LTI) filters can be described a difference equation of the form:

$$\sum_{i=0}^{M} a_i \cdot y[n-i] = \sum_{j=0}^{N} b_j \cdot x[n-j]$$

$$y[n] = \sum_{j=0}^{N} b_j \cdot x[n-j] + \sum_{i=1}^{M} (-a_i) \cdot y[n-i].$$

If $x[n]$ is an impulse (1 for $n=0$ & 0 for $n \neq 0$), the output is called the filter's impulse response.

Procedure :-

~~The filter was~~

→ A low pass filter was implemented using fdatool in Matlab.

→ The order was set to 6 & the sampling frequency was set to 48000 Hz.

→ First the filter coefficients was exported to workspace in double-precision floating point & the output was plotted using a sine input.

→ Then the filter coefficients was changed to fixed point with numerator word length as 8. and again the output was plotted using a sine input.

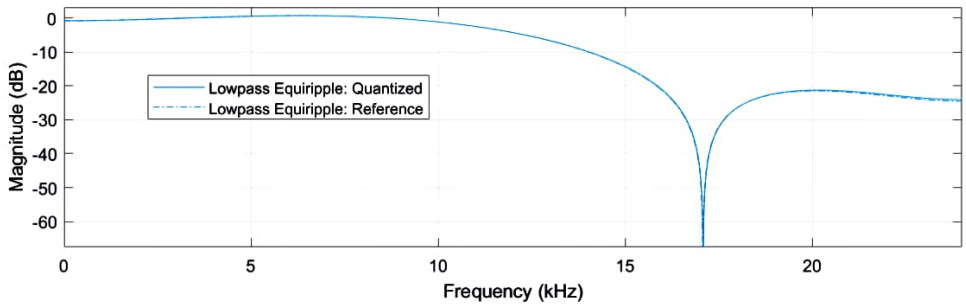
Current Filter Information

Structure: Direct-Form FIR
Order: 6
Stable: Yes
Source: Designed (quantized)

Store Filter ...

Filter Manager ...

Magnitude Response (dB)



Response Type

- (•) Lowpass
- ( ) Highpass
- ( ) Bandpass
- ( ) Bandstop
- ( ) Differentiator

Design Method

- ( ) IIR — Butterworth
- (•) FIR — Equiripple

Filter Order

- (•) Specify order: 6
- ( ) Minimum order

Options

Density Factor: 20

Frequency Specifications

Units: Hz

Fs: 48000

Fpass: 9600

Fstop: 16000

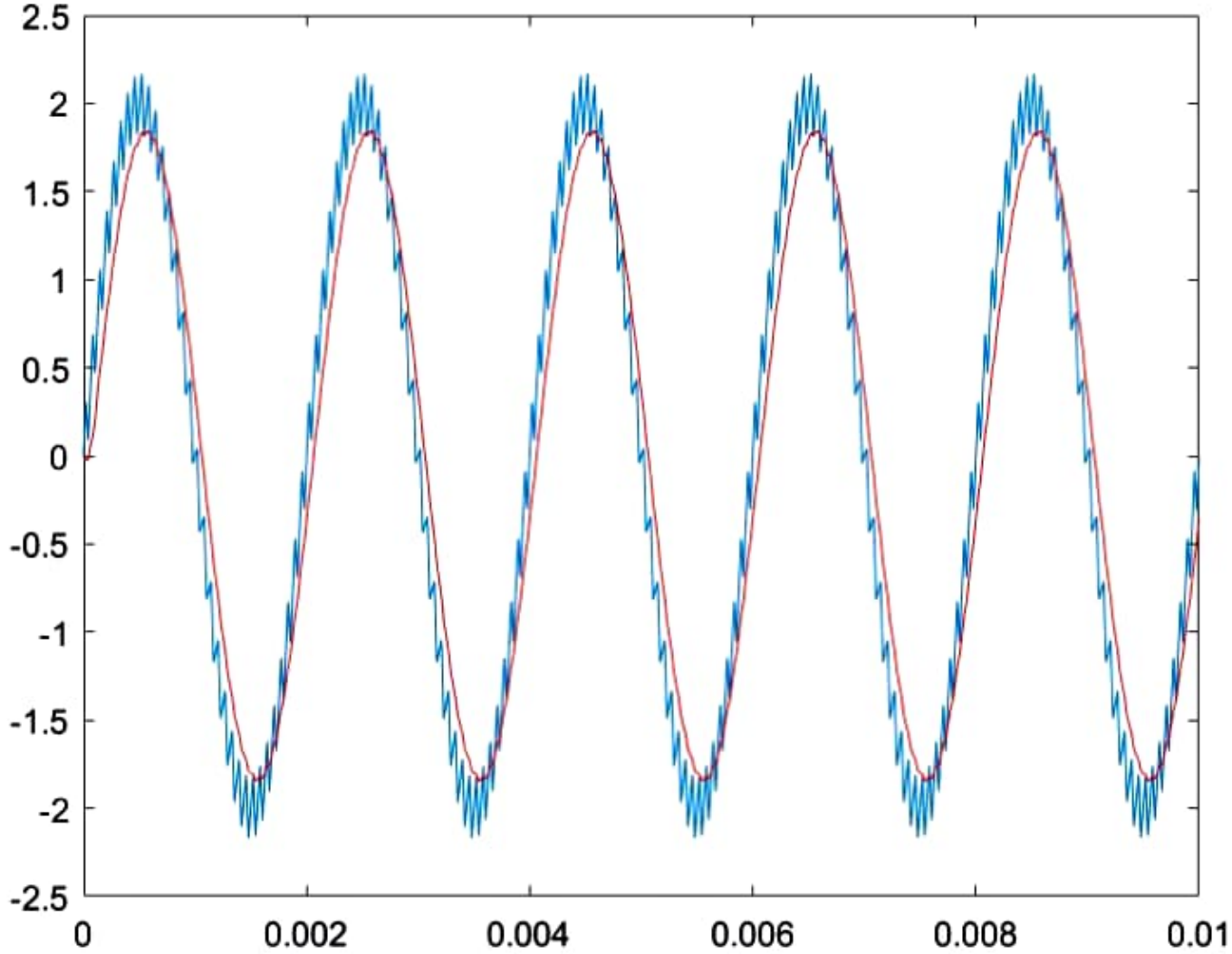
Magnitude Specifications

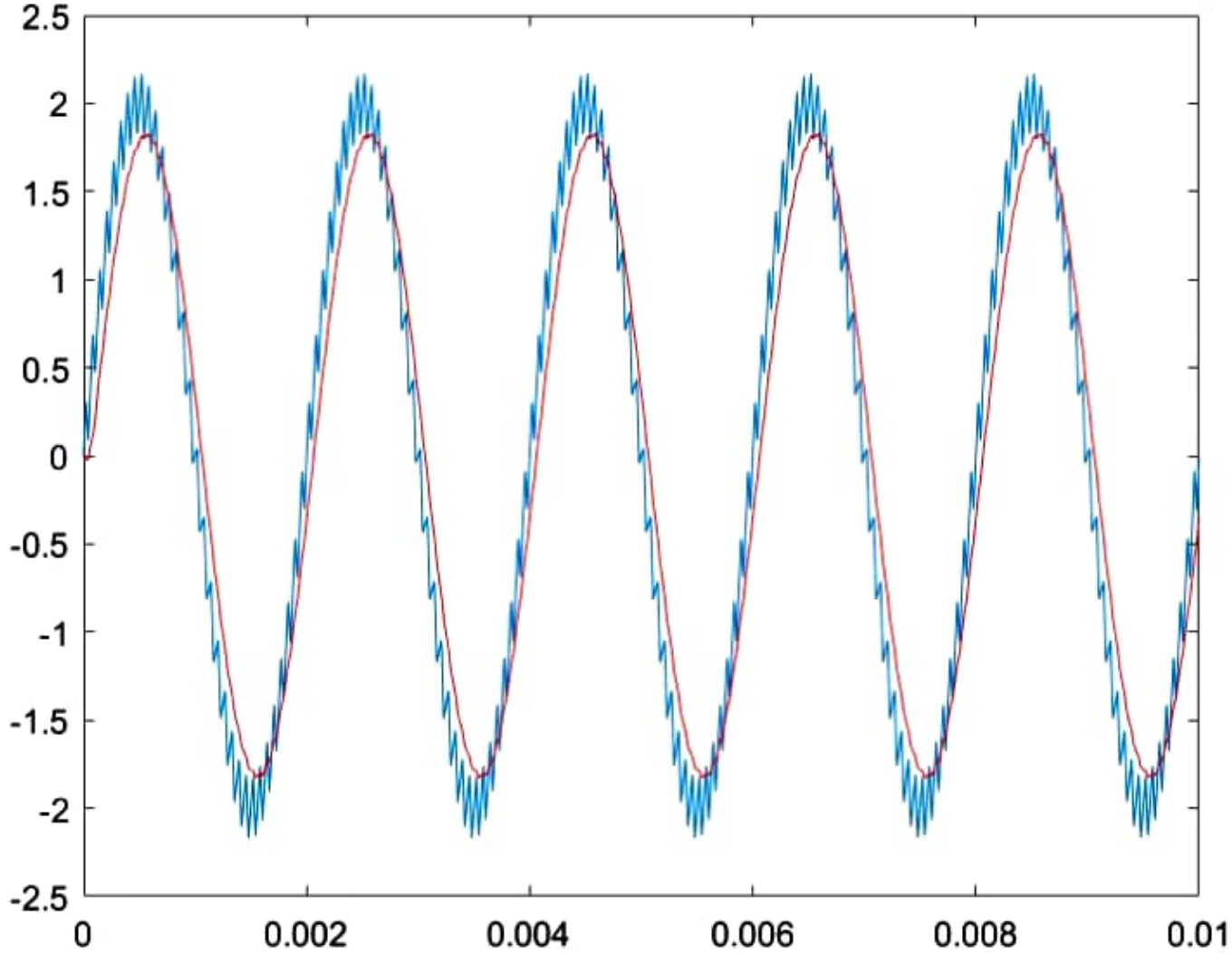
Enter a weight value for each band below.

Wpass: 1

Wstop: 1

```
clear;
fs = 48000;
T = 0.01;
ts = 1/fs;
t = 0:ts:T;
f1 = 500;
f2 = 16000;
% Input Signal
x = 2*sin(2*pi*f1*t)+0.2*sin(2*pi*f2*t);
% Filter Output with floating type filter arithmetic coefficients
figure(1);
plot(t,x);
hold;
floating_coeffs = [-0.0781393901070270,-0.0447899555785390,0.321833205219596,0.517757155393525,0.321833205219596,-0.0447899555785390,-0.0781393
y = filter(floating_coeffs,1,x);
plot(t,y,"r");
% Filter Output with fixed point filter arithmetic coefficients
figure(2);
plot(t,x);
hold;
quant_coeffs = [-0.0781250000000000,-0.0468750000000000,0.320312500000000,0.515625000000000,0.320312500000000,-0.0468750000000000,-0.0781250000
y = filter(quant_coeffs,1,x);
plot(t,y,"r");
```


2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2.5
0
0.002
0.004
0.006
0.008
0.01


2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2.5
0
0.002
0.004
0.006
0.008
0.01

Page:

Discussion :-

We are multiplying coefficients with 128 such that 7 bit range is covered because the coefficients have to be generated between -1 to 1.