LAB REPORT - 4

EMBEDDED SYSTEMS DESIGN

Shrinitihi Venkatesan
F’22

ECEN 5613 **Lab #4 Signoff Sheet** **Fall 2022**

You will need to obtain the signature of your TA on the following items in order to receive credit for your lab assignment. Print your name below, sign the honor code pledge, and then demonstrate your working hardware & firmware in order to obtain the necessary signatures.

Student Name: SHRINITHI VENKATESAN

Honor Code Pledge: "On my honor, as a University of Colorado student, I have neither given nor received unauthorized assistance on this work. **I have clearly acknowledged work that is not my own.**"

Student Signature: ______________________

Signoff Checklist

Part 1 Elements

- ☑ Pins and signals labeled and decoupling capacitors present on board
- ☑ C code for EEPROM functional, contents present after power cycle
- ☑ I^2C diagram/timing analysis

SShharade 11/18/22
TA signature and date

Part 2 Elements

- ☑ LCD functional, C code for basic LCD routines functional
- ☑ LCD control signal timing meets specifications (logic analyzer trace/diagram, analysis)
- ☑ Elapsed time stop, restart, reset to "00:00.0":
- ☑ Good integration with previous code, all functions work, no irregularities

SShharade 11/18/22

Part 3 Required and Supplemental Elements

- ☐ LCD Hex/DDRAM/CGRAM dumps, custom LCD characters, fun logo
- ☑ SPI interface, logic analyzer trace, compare with I^2C.
- ☑ ARM code development, 2 new features, ISR
- ☐ PCF8574 I^2C I/O Expander, input, output, ISR

11/19/22

FOR TA/INSTRUCTOR USE ONLY **Part 1 Elements**	Not Applicable	Poor/Not Complete	Meets Requirements	Exceeds Requirements	Outstanding
Schematics, SPLD code	☐	☐	☐	☑	☐
Hardware physical implementation	☐	☐	☐	☑	☐
Required Elements functionality	☐	☐	☐	☑	☐
Sign-off done without excessive retries		☐	☑		
Student understanding and skills	☐	☐	☐	✓ ☐	☐
Overall Demo Quality (Part 1 elements)	☐	☐	☐	☑	☐

FOR TA/INSTRUCTOR USE ONLY **Part 2 Elements**	Not Applicable	Poor/Not Complete	Meets Requirements	Exceeds Requirements	Outstanding
Schematics, SPLD code	☐	☐	☐	☑	☐
Hardware physical implementation	☐	☐	☐	☑	☐
Required Elements functionality	☐	☐	☐	✓ ☐	☐
Sign-off done without excessive retries		☐	☑		
Student understanding and skills	☐	☐	☐	✓ ☐	☐
Overall Demo Quality (Part 2 elements)	☐	☐	☐	☑	☐

FOR TA/INSTRUCTOR USE ONLY **Part 3 Elements**	Not Applicable	Below Expectation	Meets Requirements	Exceeds Requirements	Outstanding
Schematics, SPLD code	☐	☐	☐	☑	☐
Hardware physical implementation	☐	☐	☐	☐ ✓	☐
Required Elements functionality	☑	☐	☐	☐	☐
Supplemental Elements functionality	☐	☐	☐	☑	☐
Sign-off done without excessive retries		☐	☑		
Student understanding and skills	☐	☐	☐	☑	☐
Overall Demo Quality (Part 3 elements)	☐	☐	☐	☑	☐

TA/Instructor Comments ☐ ☐ ☐

Lab 4 Part 1 Signoff

(+) EEPROM writes should allow addresses from 0 - 7FF (Done)

(+) EEPROM Program functional.

(+) read, write, hexdump, reset working

(+) Contents retained after ~~program~~ power cycle.

(+) Timing analysis done

Lab 4 Part 2 Signoff

(+) ~~LD~~ LCD code functional.

(-) gotoxy not working properly

(+) gotoaddr, clear, putch, putstr working.

(+) Timer functions working. Timer is slower than actual time.

(+) Timing analysis done.

Lab 4 Part 3 Signoff

(+) STM32 features

→ UART DMA based PWM

→ WDT with WDT interrupt.

[+] SPI functional. Output expected in oscilloscope. Knows the key differences between SPI/I2C.

LAB3_PART1;

I2C_EEPROM

A well designed I2C driver was implemented with the feature that included bit banging. I have included write, read, reset and Hex dump functionality.

In the write feature, the user enters an EEPROM address and the respective function uses the address to store the user entered character. The address can be accessed through 8 pages (2048) bytes.

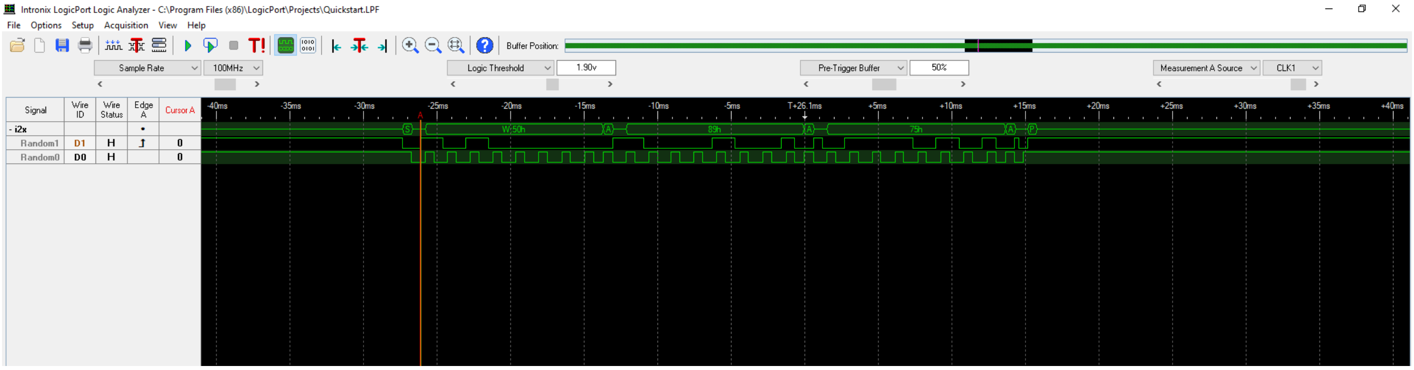
In the read feature, the user enters an EEPROM address from where the user can read the stored character from.

In the Hexdump function, the user enters a start address and an end address until which he can access the hexdump of.

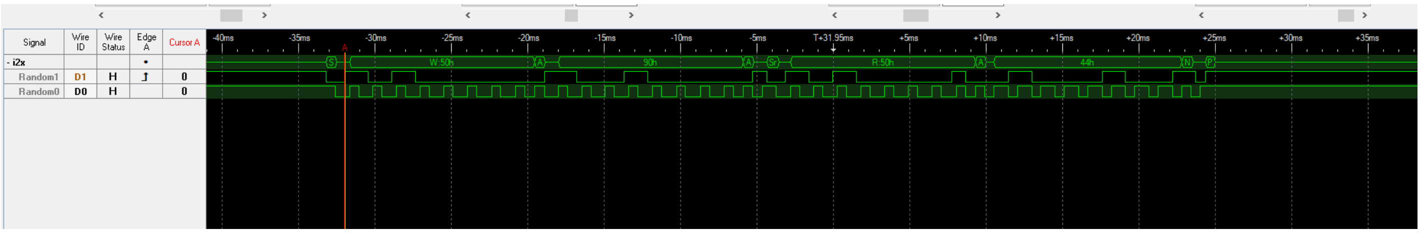
In the reset feature, the EEPROM undergoes a soft reset.

All these features were tested with logic implementation for various test cases and also a logic trace of the i2c signal was analyzed and reported to the TA.

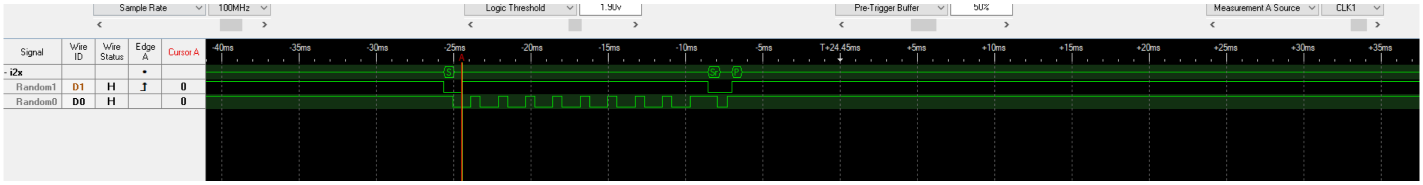
I2C Write :


Intronix LogicPort Logic Analyzer - C:\Program Files (x86)\LogicPort\Projects\Quickstart.LPF
File Options Setup Acquisition View Help
Buffer Position:
Sample Rate 100MHz
Logic Threshold 1.90v
Pre-Trigger Buffer 50%
Measurement A Source CLK1
Signal Wire ID Wire Status Edge A Cursor A
- i2x
Random1 D1 H 0
Random0 D0 H 0
T+26.1ms
S W 50h A 89h A 75h A P

I2C READ:


Signal Wire ID Wire Status Edge A Cursor A
- i2x
Random1 D1 H 0
Random0 D0 H 0
T+31.95ms
S W 50h A 90h A Sr R 50h A 44h N P

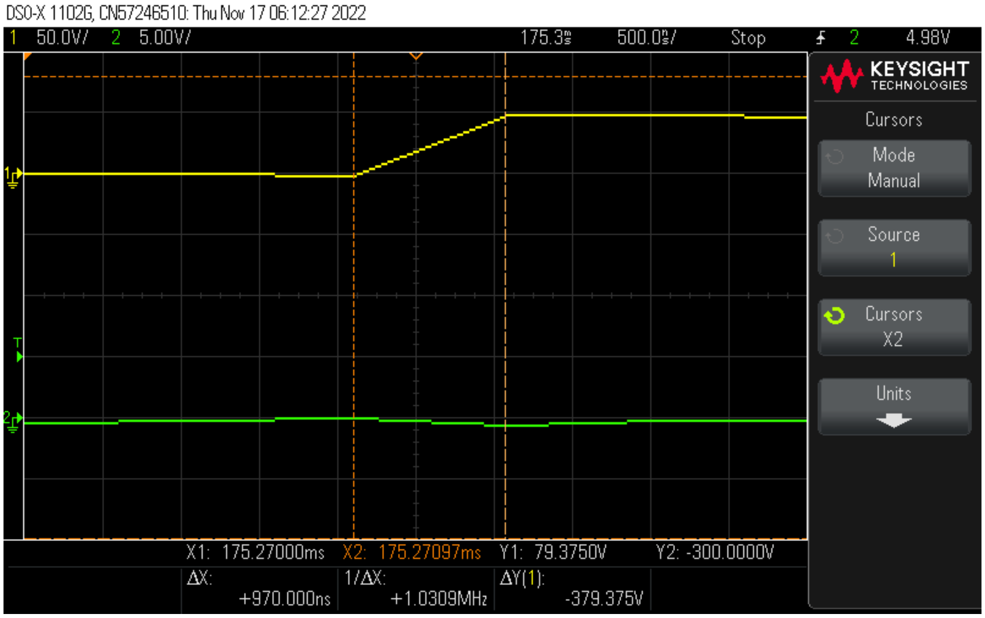
I2C RESET:



TIMING ANALYSIS:

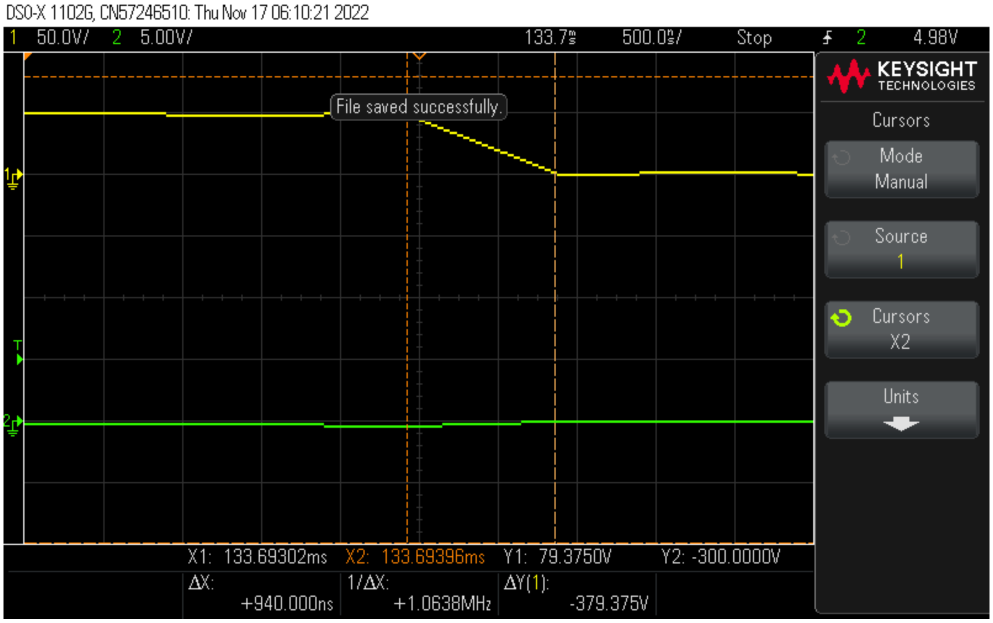
The I2c signal was analyzed and the timing analysis was done for the fall and rise times of the SDA and SCL was calculated.

SCL Rise Time: 970ns (ideal value should be max 300ns)



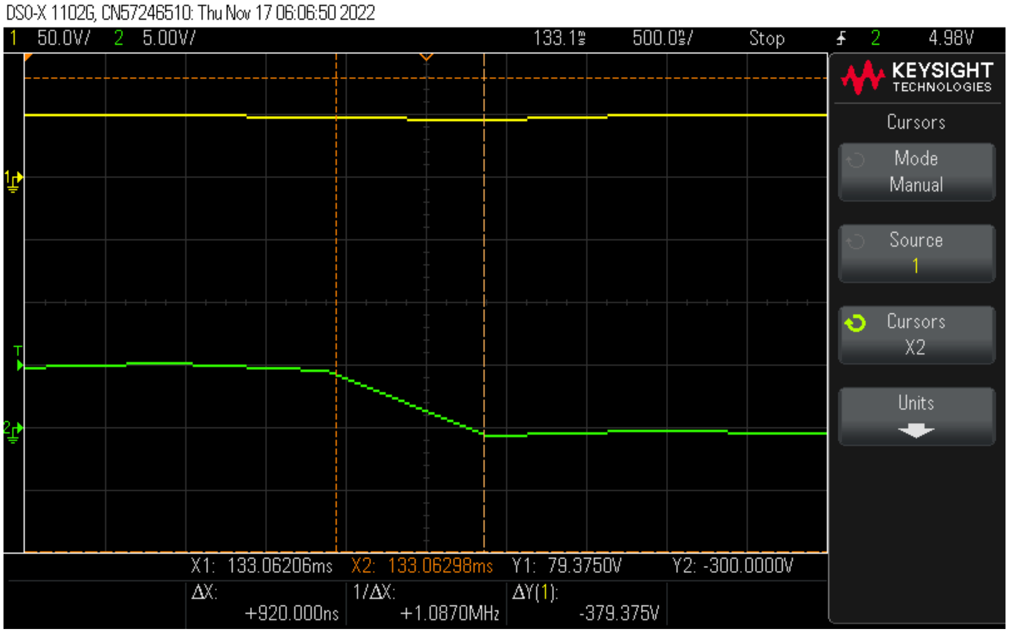
```
DSO-X 1102G, CN57246510: Thu Nov 17 06:12:27 2022
1  50.0V/  2  5.00V/        175.3ms   500.0ns/   Stop   2  4.98V
X1: 175.27000ms  X2: 175.27097ms  Y1: 79.3750V  Y2: -300.0000V
ΔX: +970.000ns   1/ΔX: +1.0309MHz   ΔY(1): -379.375V
```

SCL Fall Time: 940ns (ideal value should be max 300ns)



```
DSO-X 1102G, CN57246510: Thu Nov 17 06:10:21 2022
1  50.0V/  2  5.00V/        133.7ms   500.0ns/   Stop   2  4.98V
File saved successfully.
X1: 133.69302ms  X2: 133.69396ms  Y1: 79.3750V  Y2: -300.0000V
ΔX: +940.000ns   1/ΔX: +1.0638MHz   ΔY(1): -379.375V
```

SDA Rise Time: 920 ns (Ideal should be max 3us)



DSO-X 1102G, CN57246510: Thu Nov 17 06:06:50 2022

1 50.0V/ 2 5.00V/ 133.1ms 500.0us/ Stop 2 4.98V

X1: 133.06206ms X2: 133.06298ms Y1: 79.3750V Y2: -300.0000V

ΔX: +920.000ns 1/ΔX: +1.0870MHz ΔY(1): -379.375V

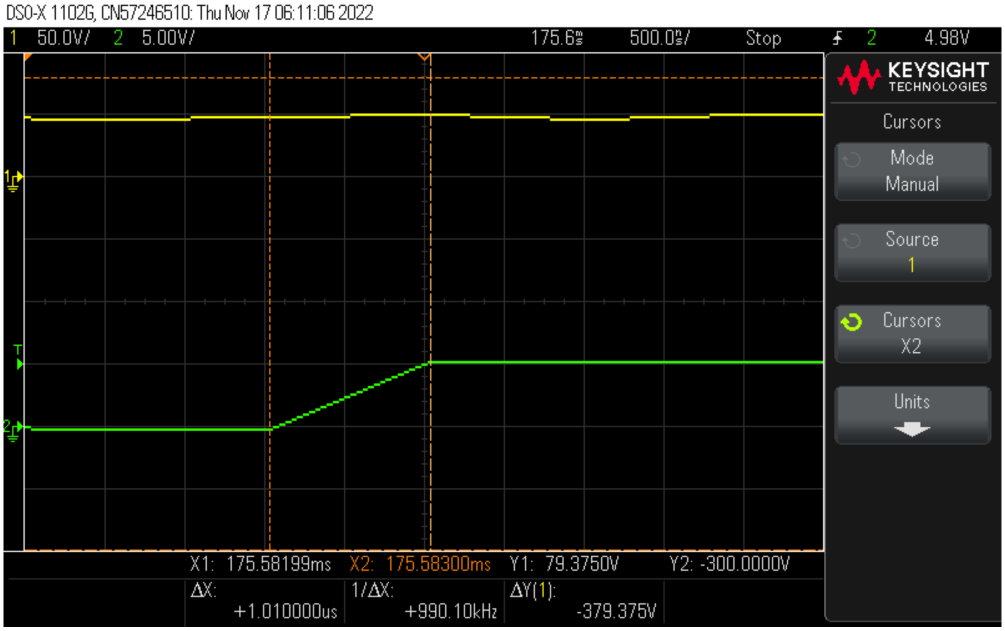
Cursors

Mode Manual

Source 1

Cursors X2

SDA Fall Time: 1.01 us (Ideal should be max 3us)



DSO-X 1102G, CN57246510: Thu Nov 17 06:11:06 2022

1 50.0V/ 2 5.00V/ 175.6ms 500.0us/ Stop 2 4.98V

X1: 175.58199ms X2: 175.58300ms Y1: 79.3750V Y2: -300.0000V

ΔX: +1.010000us 1/ΔX: +990.10kHz ΔY(1): -379.375V

Cursors

Mode Manual

Source 1

Cursors X2

Learning outcome and difficulties faced:

1. I learnt how to access different pages to make efficient EEPROM memory utilization.
2. I learnt in detail about the i2c protocol and the bit banging function.
3. Timing analysis was helpful to understand the i2c signal.

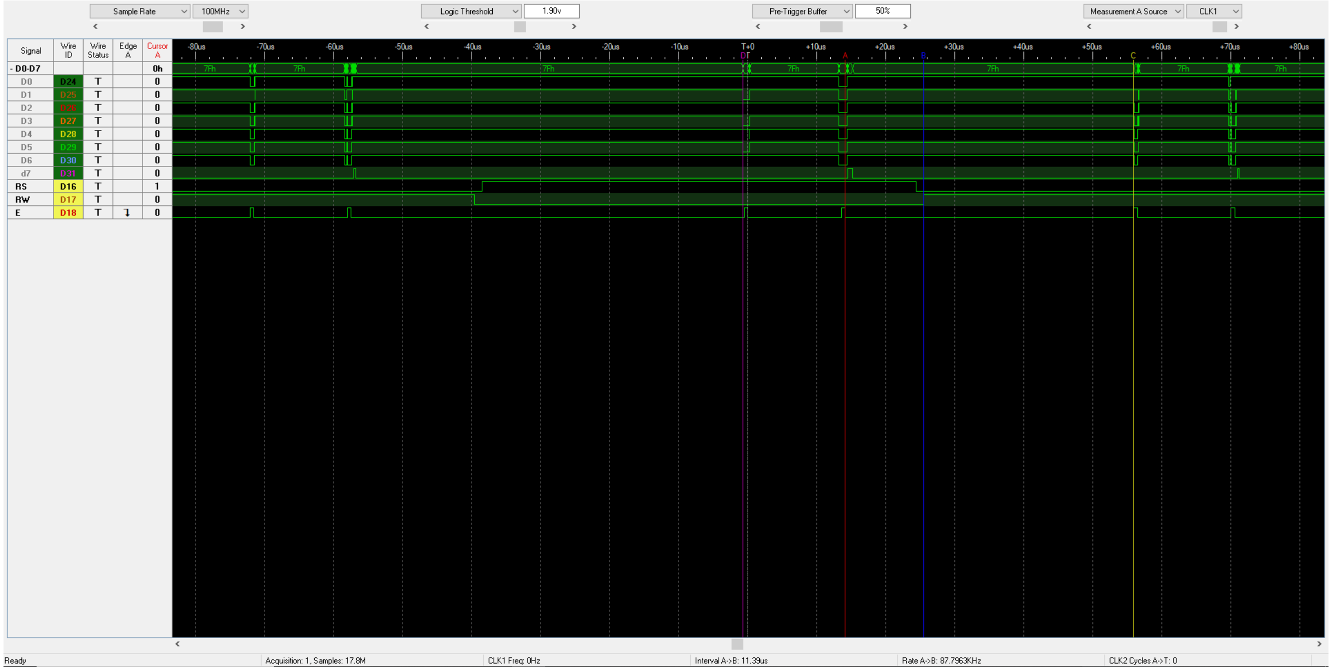
LCD:

An LCD driver for 16*4 display was implemented and verified for the following functionality:

1. Move to a particular address
2. Point cursor at a requested position
3. Write a character or a string to the LCD
4. Runs a real time clock.

Timing analysis:

TAH: 11.39us

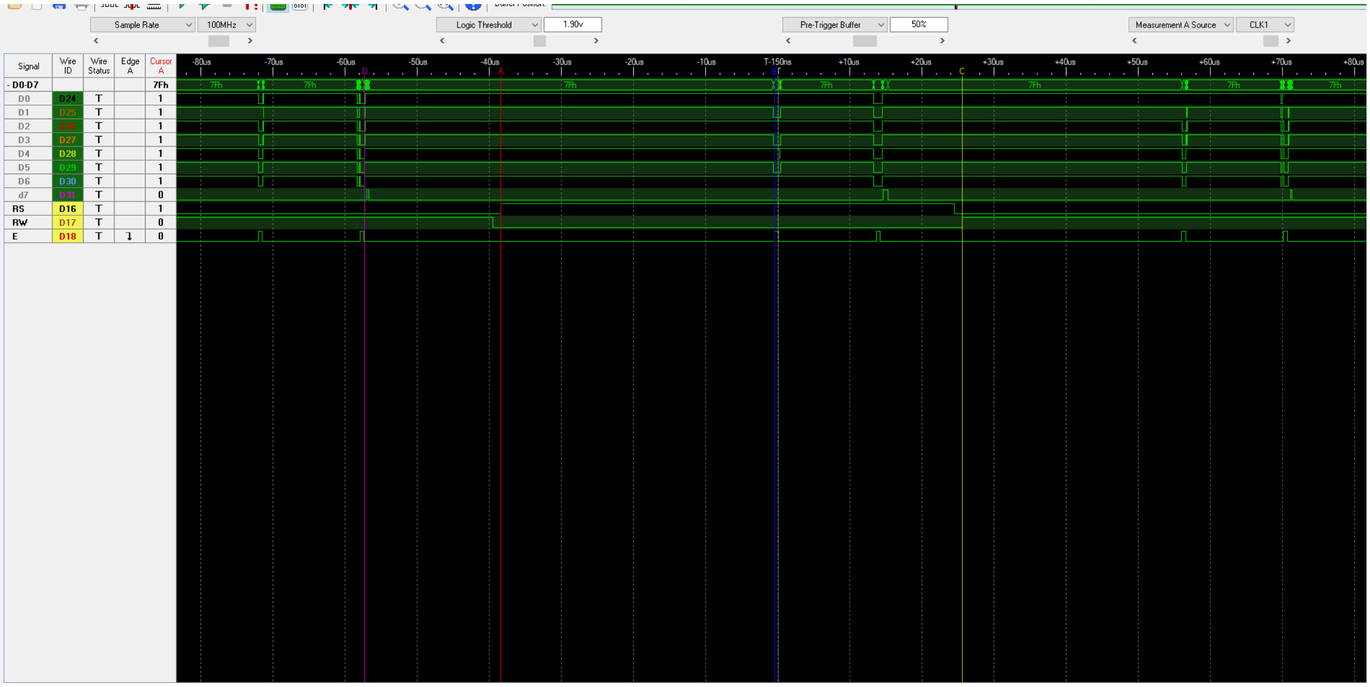


Sample Rate: 100MHz | Logic Threshold: 1.90v | Pre-Trigger Buffer: 50% | Measurement A Source: CLK1

Signal	Wire ID	Wire Status	Edge A	Cursor A
- D0-D7				0h
D0	D24	T		0
D1	D25	T		0
D2	D26	T		0
D3	D27	T		0
D4	D28	T		0
D5	D29	T		0
D6	D30	T		0
d7	D31	T		0
RS	D16	T		1
RW	D17	T		0
E	D18	T	↧	0

```
Ready    Acquisition: 1, Samples: 17.8M    CLK1 Freq: 0Hz    Interval A->B: 11.39us    Rate A->B: 87.7963KHz    CLK2 Cycles A->T: 0
```

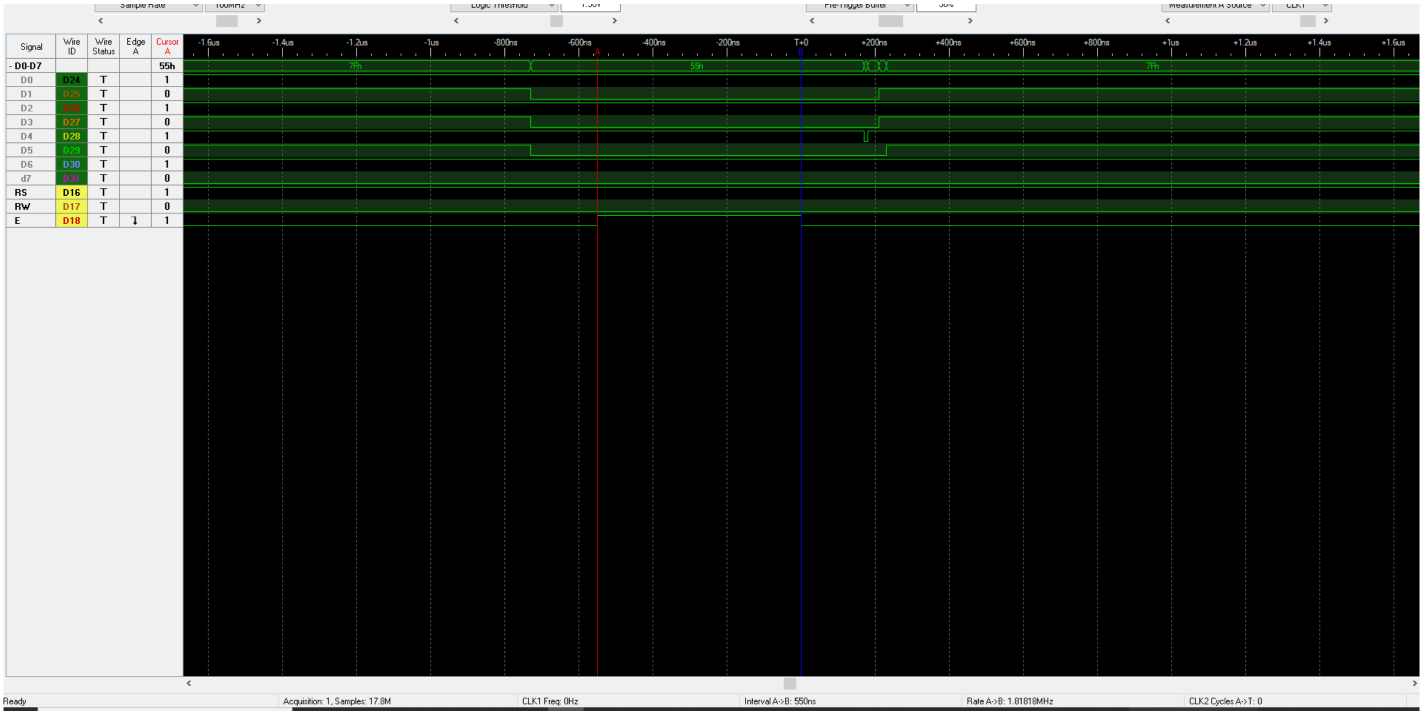
TAS: 36.85 us



Sample Rate: 100MHz | Logic Threshold: 1.90v | Pre-Trigger Buffer: 50% | Measurement A Source: CLK1

Signal	Wire ID	Wire Status	Edge A	Cursor A
- D0-D7				7Fh
D0	D24	T		1
D1	D25	T		1
D2	D26	T		1
D3	D27	T		1
D4	D28	T		1
D5	D29	T		1
D6	D30	T		1
d7	D31	T		0
RS	D16	T		1
RW	D17	T		0
E	D18	T	↧	0

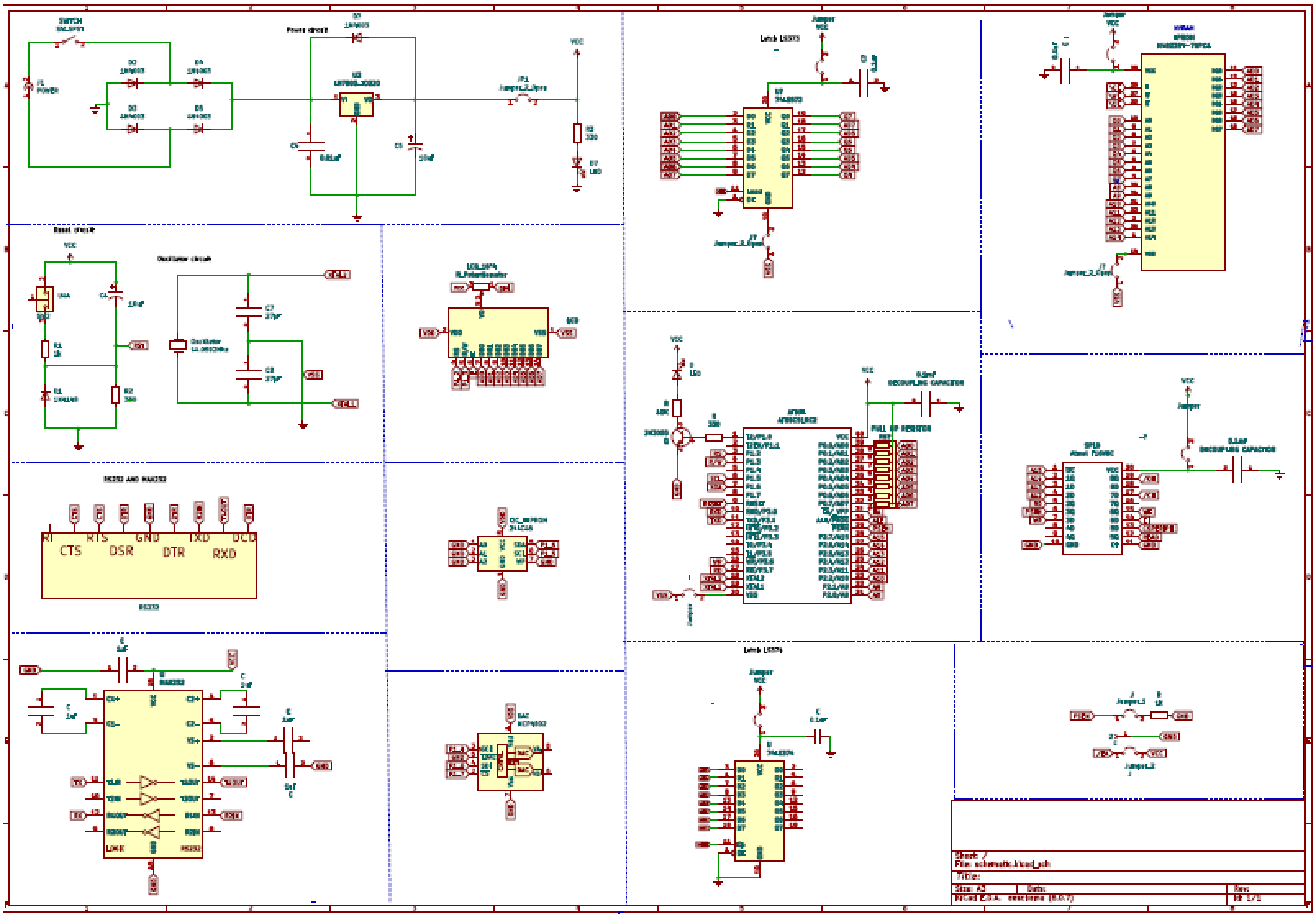
FeWH: 550ns



Signal	Wire ID	Wire Status	Edge A	Cursor A
- D0-D7				55h
D0	D24	T		1
D1	D25	T		0
D2	D26	T		1
D3	D27	T		0
D4	D28	T		1
D5	D29	T		0
D6	D30	T		1
d7	D31	T		0
RS	D16	T		1
RW	D17	T		0
E	D18	T	↓	1

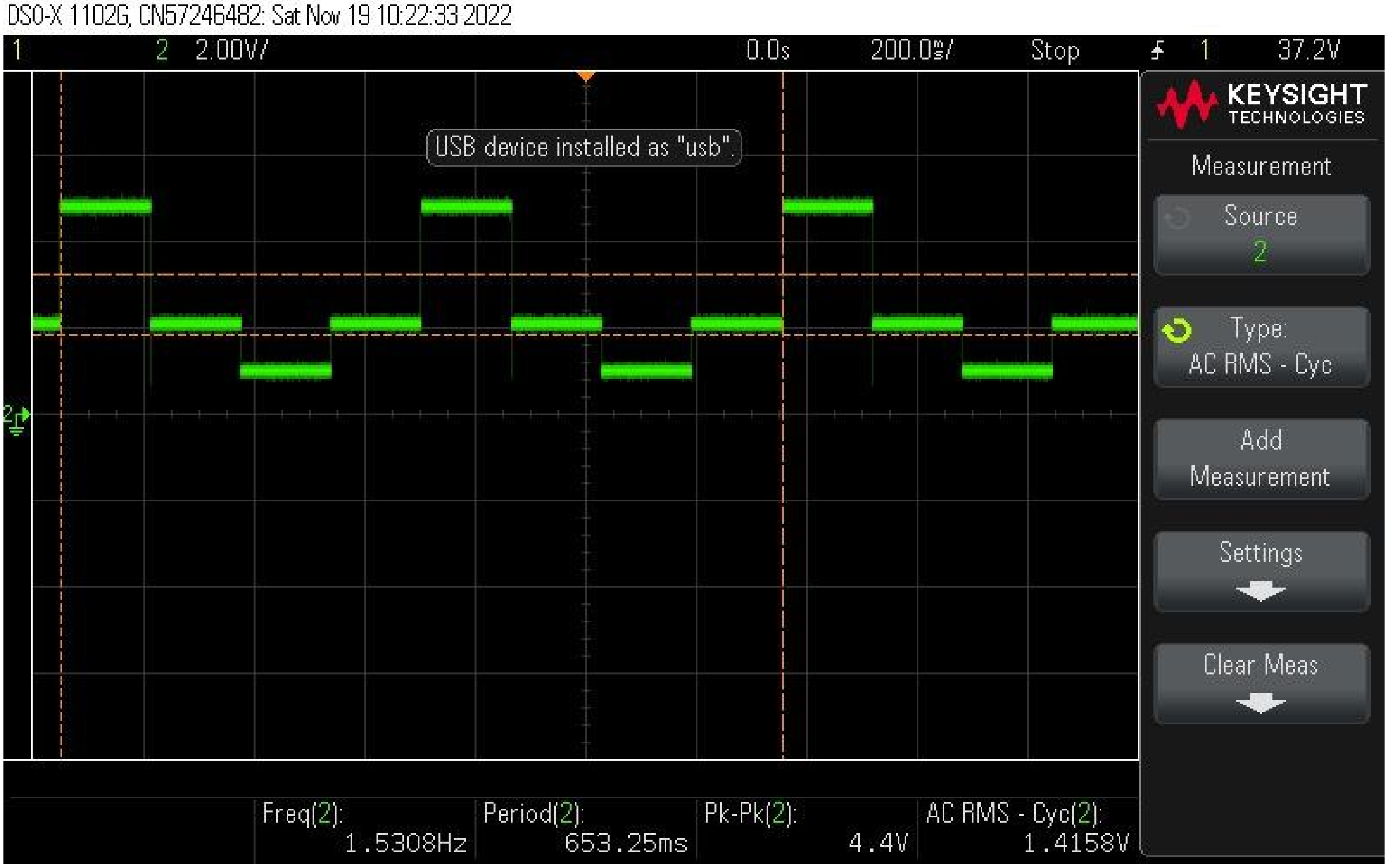
```
Ready    Acquisition: 1, Samples: 17.8M    CLK1 Freq: 0Hz    Interval A->B: 550ns    Rate A->B: 1.81818MHz    CLK2 Cycles A->T: 0
```

Schematic:



DAC interfacing with SPI protocol:

Using 8051, DAC was interfaced with SPI to create a step output . I have used the values 22,88,22,00 to get the output.



DSO-X 1102G, CN57246482: Sat Nov 19 10:22:33 2022

1 2 2.00V/ 0.0s 200.0ms/ Stop 1 37.2V

USB device installed as "usb".

Freq(2): 1.5308Hz

Period(2): 653.25ms

Pk-Pk(2): 4.4V

AC RMS - Cyc(2): 1.4158V

STM32:

WWDG and IWDG:

For the independent watchdog timer, 20s was calculated and whenever the HAL delay was set beyond that, the watchdog refresh was activated.

For the Window WatchDog Timer, window period between 13s to 20s was calculated and whenever the time frame condition was not satisfied, the watchdog timer will be refreshed.

DMA interrupt call using UART with PWM :

Configured the PWM through UART and the setting was interfaced with the LED to observe the brightness change. 25% PWM increase was interfaced with the ‘A’ character input and 25% decrease was interfaced with the ‘B’ character input.The UART was interfaced with the DMA interrupt.

Learning outcome:

1. Watchdog timers
2. DAC interfacing
3. SPI and I2C protocols.

QUESTIONS:

a) What operating system (including revision) did you use for your code development?

Windows 10

b) What compiler (including revision) did you use?

Small Device C Compiler (8051)

SDCC 4.2.0

c) What exactly (include name/revision if appropriate) did you use to build your code (what IDE, make/makefile, or command line)?

IDE : CodeBlocks 20.03

IDE for STM32: STM32CUBEMX

Command line: Putty (Release 0.77)

d) Did you install and use any other software tools to complete your lab assignment?

WinCUPL: For SPLD Code

PUTTY: Command line

KiCad 6.0 is a software suite used for Electronic Design Automation (EDA). I have developed schematic using the same.

FLIP: Flip helps in-system programming of flash devices through RS232, USB or CAN. I have flashed my .hex files to 8051 board using the same

e) Did you experience any problems with any of the software tools? If so, describe the problems

I could not fulfill the LCD custom character and RAM dump as it was difficult for me to finish within the time frame.