EMBEDDED SYSTEM DESIGN LAB ASSIGNMENT -3

STM8 Interfacing with LCD Module and Frequency Measurement using Timers

SUBMITTED BY- SANCHIT AGRAWAL (2016H140106)
ABHINAV GARG (2016H140103)

Objective – To design a frequency measurement instrument and to display the measured frequency on an alphanumeric LCD.

PROJECT WORK-

- Used IAR Embedded workbench with embedded C coding and used ST Visual Programmer for programming the STM8S micro controller
- In this project we are giving input waveform through Function Generator, used zero crossing detector method to measure the input frequency.

CRITICAL ISSUES-

- To print on the LCD by creating standard Library functions.
- Measuring Frequency through zero crossing detector method.

Methodology – We have tried to generate the Frequency using two approaches,

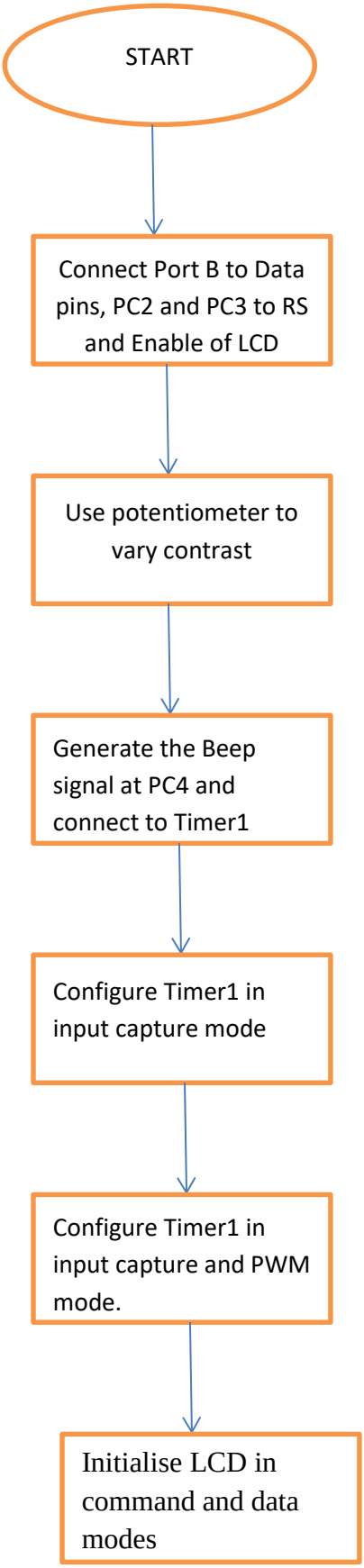
1) Through Internal Beep Module 2) Function Generator, the output for both is mapped to the PIN_C4. To measure the frequency we have tried to use the rising edge method describe later. We have interfaced LCD HD44780 to STM8S Micro Controller

HARDWARE DESIGN –

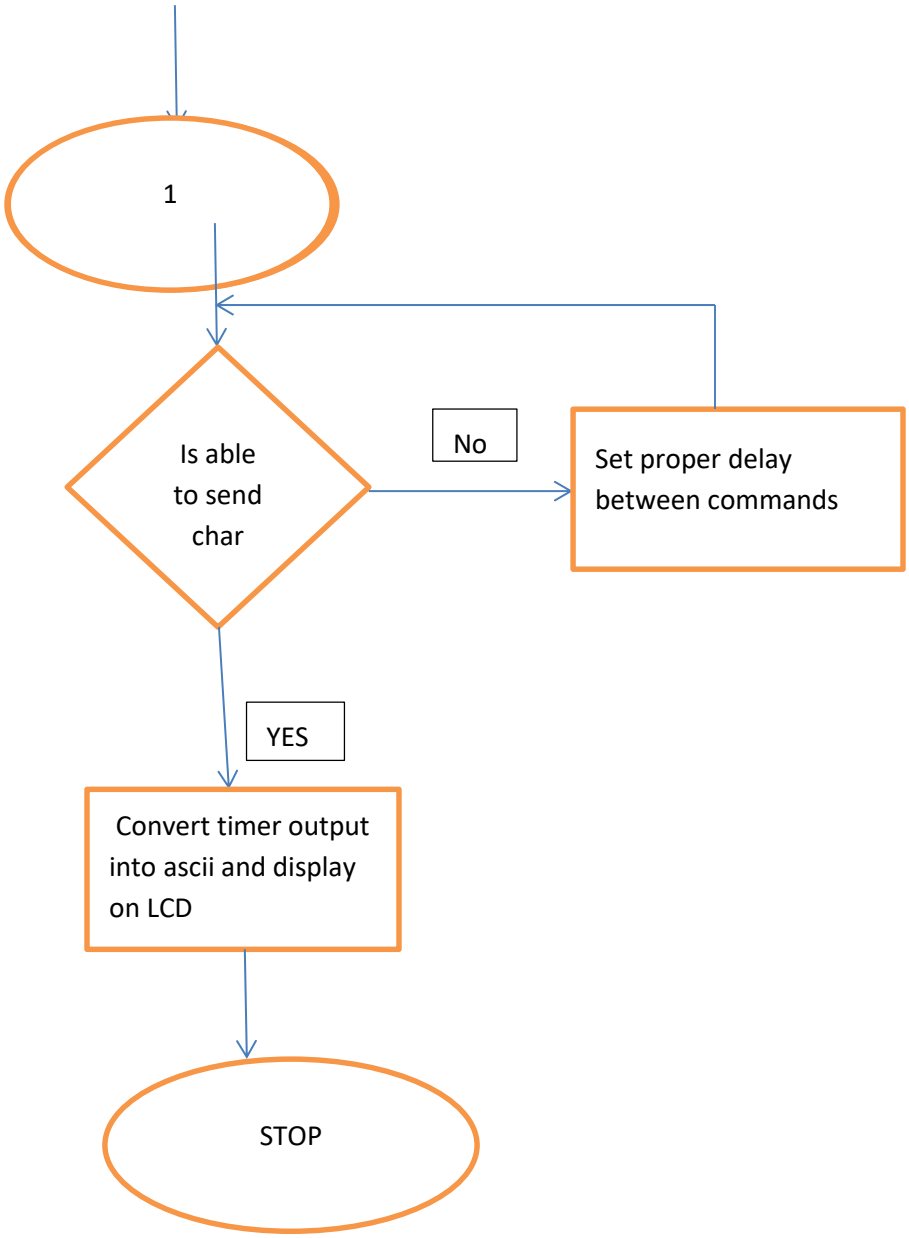
Used HD44770 LCD and we have used PORT D as LCD Data Port. We have grounded the Read/$write\bar{e}$. PORT A has been configured for Control Signals (RS and

EN) except for PIN.C4 which has been used for external interrupt. We have used Potentiometer for Display Control (For Varying Contrast).

FLOW CHART-



Continuing flow chart-


1
Is able to send char
No
Set proper delay between commands
YES
Convert timer output into ascii and display on LCD
STOP

SCHEMATIC-

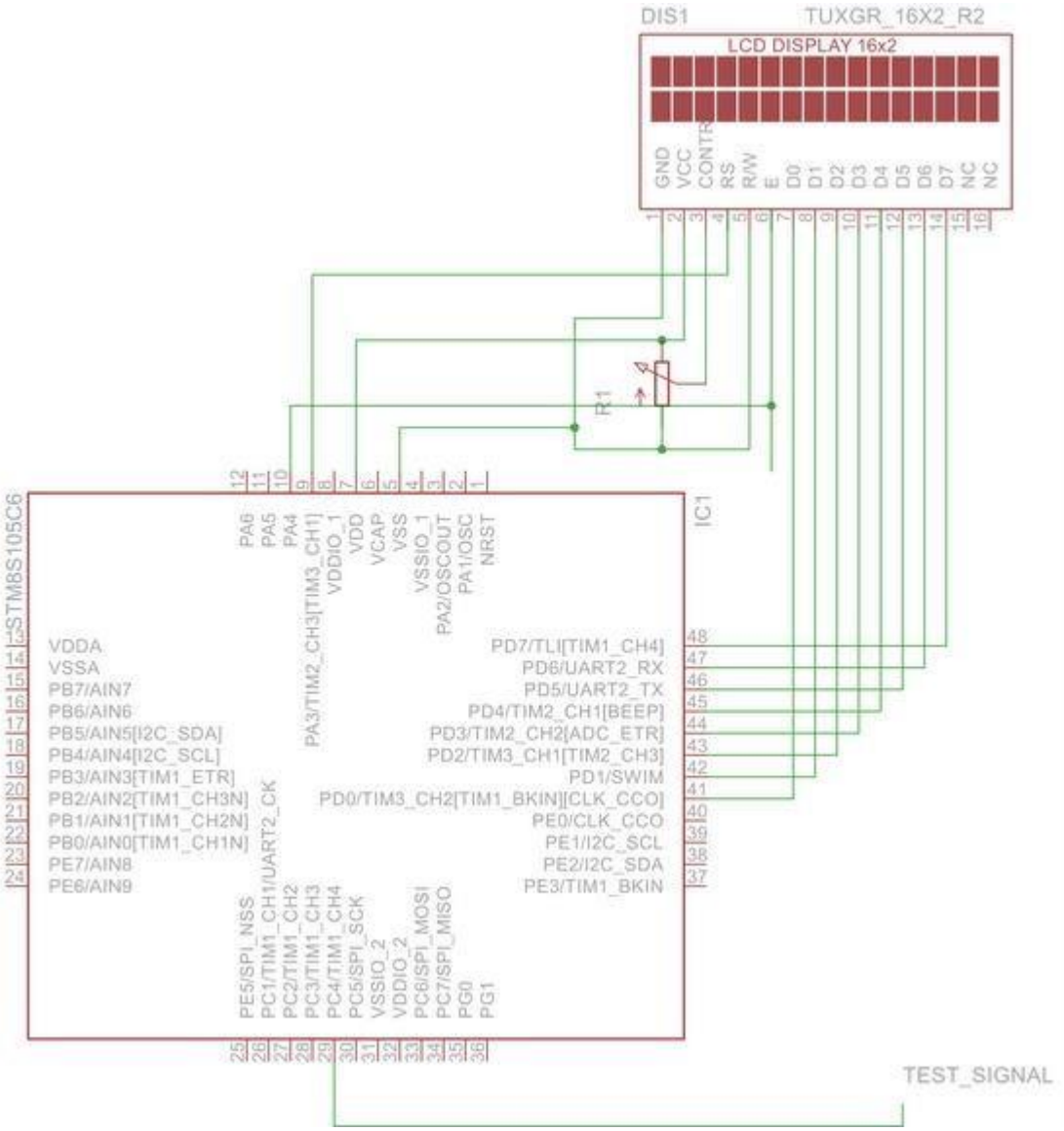


Figure2: schematic diagram of Frequency measurement

SOFTWARE DESIGN

– Used Basic LCD initialising instructions to display the measured frequency on the LCD.

C code for stated task is attached

EXPLANATION-

1. The Header file (iostm8S105C6.h) includes all the input-output functions. 2. Header file “intrinsics.h” includes intrinsic function definition such as __disable_interrupt(), __enable_interrupt (here “__” represents an intrinsic function).

3. Then, structure named PD_ODR_bit is used, structure must be defined in the header file to manipulate each bit of ODR register.
4. Global integer “data” and “F” is defined and initialized. Long int “dig” is defined to store the count value. Data is defined to to shoe lcd data.
5. The “**pragma**” directives control the behaviour of the compiler, for example how it allocates memory for variables and functions, whether it allows extended keyword, and whether it outputs warning messeges. The pragma directives are always enabled in the compiler.
6. **"__interrupt"** is a **function attribute**, it adds an attribute to the function that it neither receive nor pass any argument.
7. In the main program loop, **“__disable_interrupt()”** is used. The significance of “__” is **intrinsic function** calling (from intrinsic.h file) for fast execution.
8. The various function performed on setting the DDR, IDR, CR registers at various pin positions is shown in table below-

9. Next, we set our output ports where the LED’s are connected
10. CR 1 register is initialized which is responsible for push pull operation.
11. **PD_ODR_bit structure** is used. In general structure must be defined in header file to manipulate each bit of ODR register.
12. **Void sendcommand() is used to send commands to lcd i.e. this is basically initializing the LCD .**
13. Void set_lcd() is used to set the lcd for operation.here, 3C enables the 5x7 lcd matrix, 0E represents LCD on, 06 is for write cursor, 80 is for setting cursor at starting position.
14. During above function RS is kept low since we are sending commands.
15. Void write_lcd is used to write data on lcd, here RS=1 is used as we are writing data.

Note- #prgma vector=8, this statement tells the compiler which interrupt vector we are going to be writing. In our case, it is vector 8 which is EXTI_PORTD interrupt vector

FREQUENCY MEASUREMENT CONCEPT-

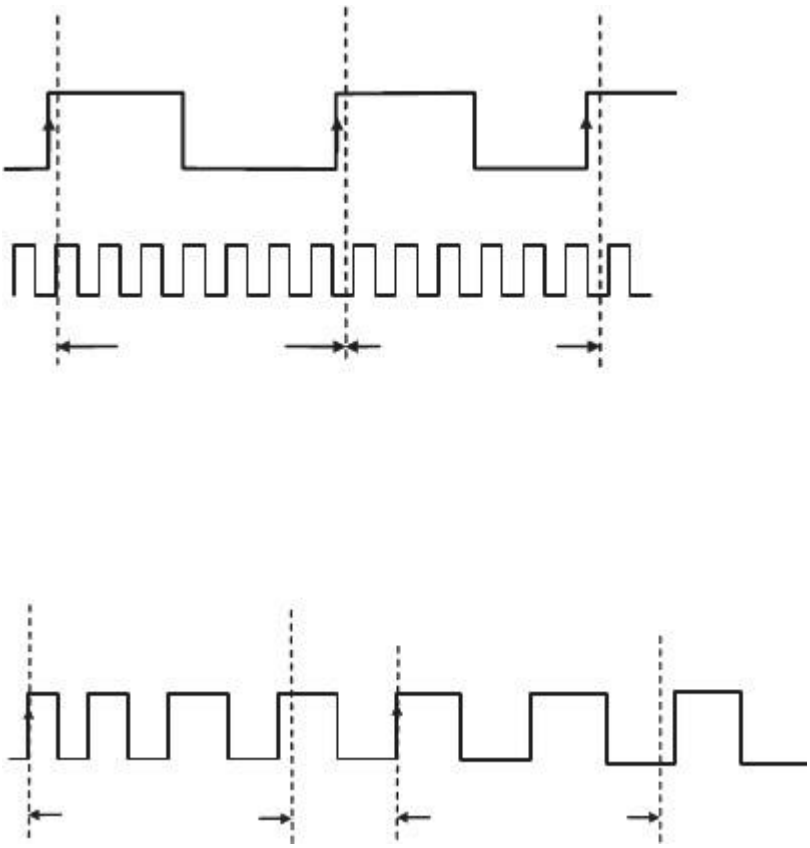


fig.-3.1

- Firstly, we defined Flag “F” which is set or reset depending on timer condition i.e. on or off state.
- Initially the flag is set to 0
- Now the concept is, on each rising edge of the clock, (refer fig.3.1) an external interrupt is generated which sets the timer on and following this the flag “F” is set to 1.
- Counter counts the no. of pulses in between the consequtive rising edge and the count value is stored in variable named “dig”.

- As soon as the next rising edge arrives, as flag F=1 this time,the external interrupt will set timer off and the count value will be stored in “dig”.
- Now again, the flag F will be set to 0, and this operation continues.

- Hence, **Frequency** will be-

1 / (count value)

OBSERVATION-

We are able to able to interface the LCD completely and timer interrupt was generated which actually finds the time period (in terms of no. of counts).

RESULT-

The LCD interfacing was done but we were not able to measure and display the input frequency on the LCD

REFERENCES-

https://www.google.co.in/search?q=&espv=2&biw=1366&bih=609&source=lnms&tbm=isch&sa &imgrc=UnCTpwSr_No4pM%3A

(above link referred for image)