EE663- Project 1 Report
Histogram of Signal Pulses

Table of Contents

Overview:

The aim of the project is to display the histogram of incoming signal pulses. Brief overview:

- Lower limit of the pulse is from 50ms – 9950ms and configurable.
- Bucket of 1000 measurement samples.

Cover Page:

Project title	:	Histogram of Incoming signal pulses.
Names	:	Deepak Siddharth Parthipan.
E-mail addresses of the authors	:	dp9040@rit.edu.
Date of submission	:	11-14-2016.

Areas of Focus:

Deepak Siddharth Parthipan : Problem Statement analysis, Functional Design, Developing Code, Test plan and Report.

Tanmay Shinde : Problem Statement analysis, Functional Design, Test plan.

Analysis / Design:

The idea of the project is to find the jitter of the STM32L476VGT6 development kit while it's running the software program and board at its crystal frequency. Histogram basically gives the total count of the sample frequencies points. Any latency in measurement is understood through the spread in the measurement.

Hardware configuration:

The STM32L476VGT6 kit is quite powerful one with 32-bit ARM Cortex 4 CPU, with a bunch of highly configurable peripherals. It is connected to the computer system via USB. This facilitates three basic functions: connection to IDE, power supply and serial communication. The Function generator is hooked up with the development kit at the TTL logic volts and incoming square pulses are capture by tuning the General-purpose IO PORTA Pin-0 as input capture mode at channel 1. The pulses generated are verified with the CRO. UART port pins TXD and RXD of the STM32 is used for serial communication with the terminal server. Putty is the host that enables the connection with the program.

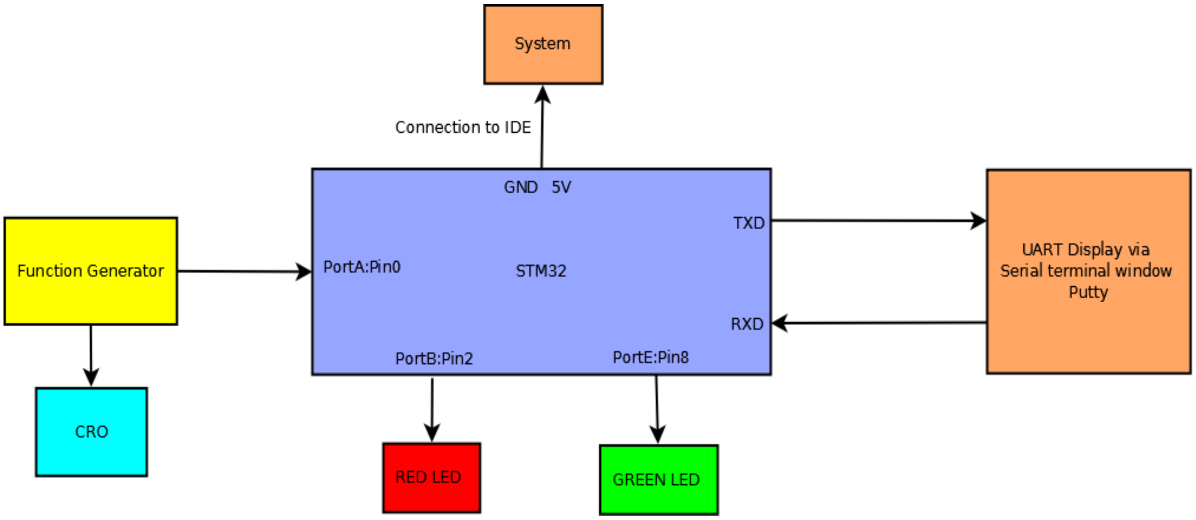


Figure1: Hardware block diagram of hierarchical components

Software Implementation:

Main software implementation involves configuring TIMER CAPTURE, UART, LED and SYSTEM CLOCK. The software code for UART, LED and SYSTEM CLOCK is reused from the demo code provided in the class. Since there is no Operating system the program must run in an infinite loop (while (1)).

Timer2 is used to capture the input signals. It's a 32-bit timer. The input signal is received at the GPIO PortA0 and configured as alternate function mode. The Alternate function is a special capability of for STM32 devices wherein we can configure each port to several possible options. Here its configured for Timer 2 channel 1 input capture operation. Once every time the rising edge pulse is detect at the pin the program polls for this information and gets the time stamp information from the TIM2 register. The time-period of the signal is the delta t difference between successive time stamps collected.

After the capture of the inputs its stored in the buffer of 1000 and processed in the histogram sorting algorithm. Which is basically nothing but a sorting sequence of the time-stamps in ascending order. Now if there are repeated time-stamps in the bucket the count value for the particular value of the frequency is incremented. Thus, after completing the measurement the spread is over 3-4 values of time frequencies that is obtained. The SW runs on the fly without a restart.

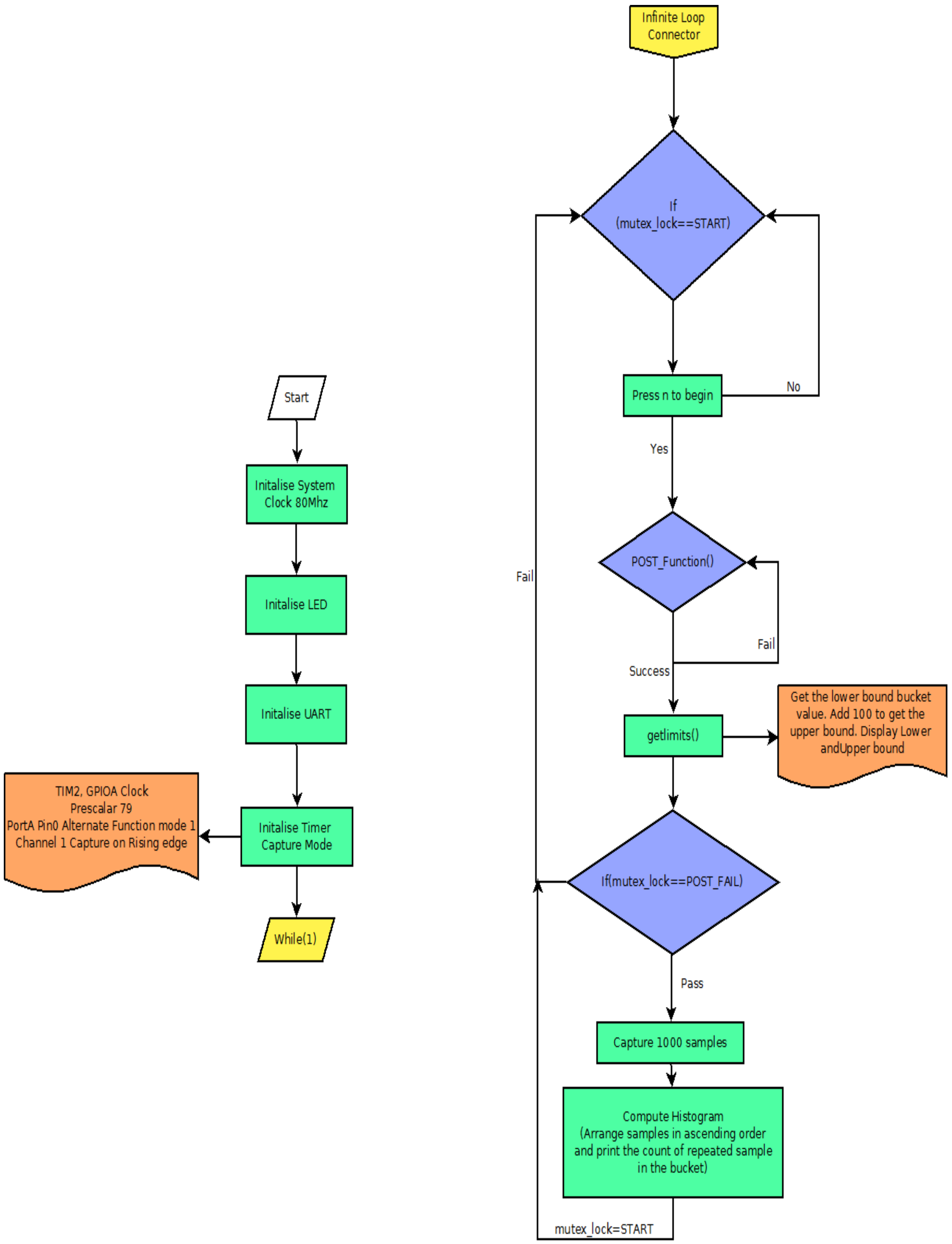

Start
Initalise System Clock 80Mhz
Initalise LED
Initalise UART
Initalise Timer Capture Mode
TIM2, GPIOA Clock
Prescalar 79
PortA Pin0 Alternate Function mode 1
Channel 1 Capture on Rising edge
While(1)
Infinite Loop Connector
If (mutex_lock==START)
Press n to begin
No
Yes
POST_Function()
Fail
Success
getlimits()
Get the lower bound bucket value. Add 100 to get the upper bound. Display Lower andUpper bound
If(mutex_lock==POST_FAIL)
Fail
Pass
Capture 1000 samples
Compute Histogram (Arrange samples in ascending order and print the count of repeated sample in the bucket)
mutex_lock=START

Figure2: Software Flow chart of the Histogram program

Test Plan:

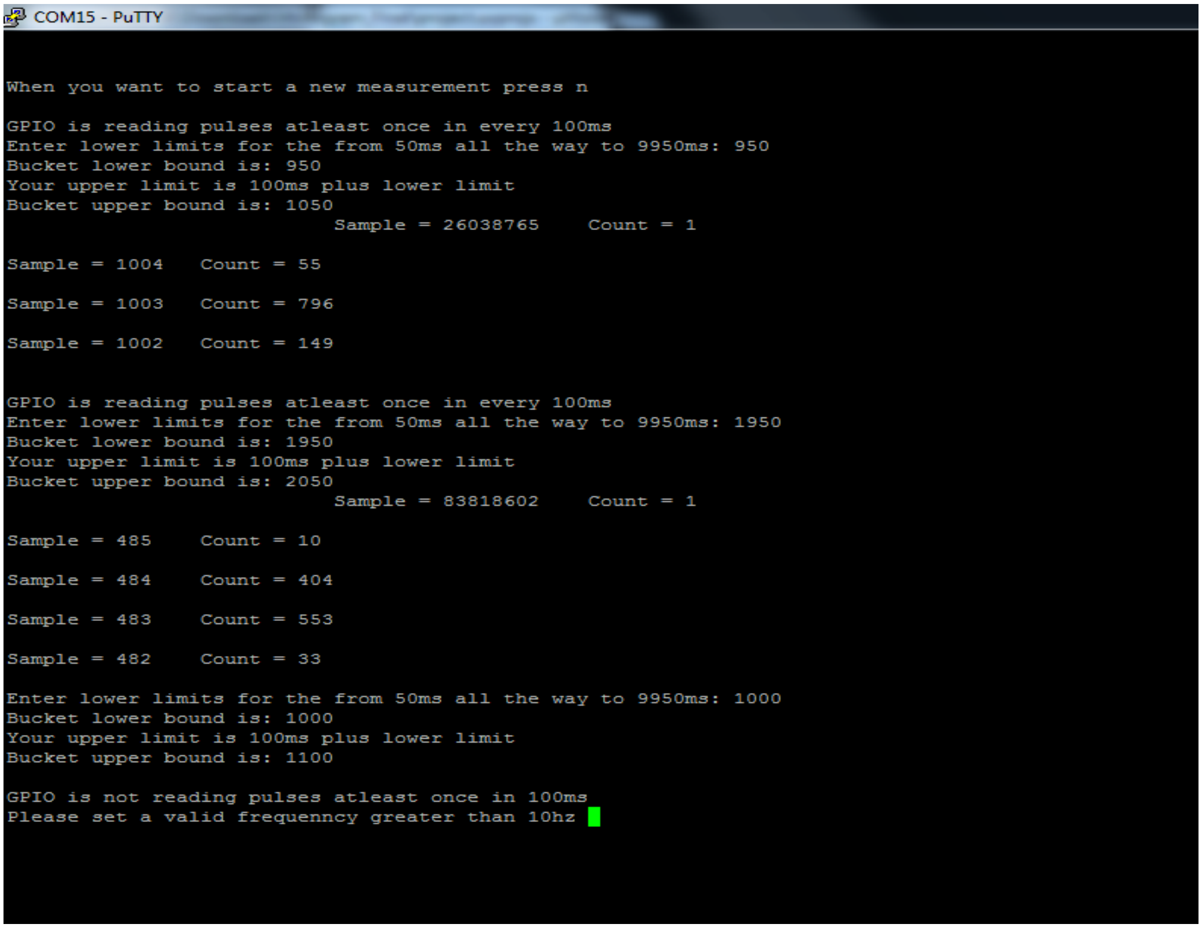
Post Function is created to test the program on the fly.

If the value of the square wave pulse measured across the input capture PortA is between 10Hz and 10Khz the Post Function is deemed as success. Else program returns to seek new measurement and start again. This is tested by giving a series of inputs frequencies ranging from sub 10hz to more than 10khz.

Check if an invalid set of lower bound is set or not. If set then ask the customer to check and restart the program.

The RED Led is ON until program is started by the user with a valid frequency. While computing the histogram in a valid range of the frequency the GREEN Led is set ON.

Project Results:



```
When you want to start a new measurement press n

GPIO is reading pulses atleast once in every 100ms
Enter lower limits for the from 50ms all the way to 9950ms: 950
Bucket lower bound is: 950
Your upper limit is 100ms plus lower limit
Bucket upper bound is: 1050
                           Sample = 26038765    Count = 1

Sample = 1004   Count = 55

Sample = 1003   Count = 796

Sample = 1002   Count = 149


GPIO is reading pulses atleast once in every 100ms
Enter lower limits for the from 50ms all the way to 9950ms: 1950
Bucket lower bound is: 1950
Your upper limit is 100ms plus lower limit
Bucket upper bound is: 2050
                           Sample = 83818602    Count = 1

Sample = 485    Count = 10

Sample = 484    Count = 404

Sample = 483    Count = 553

Sample = 482    Count = 33

Enter lower limits for the from 50ms all the way to 9950ms: 1000
Bucket lower bound is: 1000
Your upper limit is 100ms plus lower limit
Bucket upper bound is: 1100

GPIO is not reading pulses atleast once in 100ms
Please set a valid frequenncy greater than 10hz
```

Figure3: User Interface for servo motor

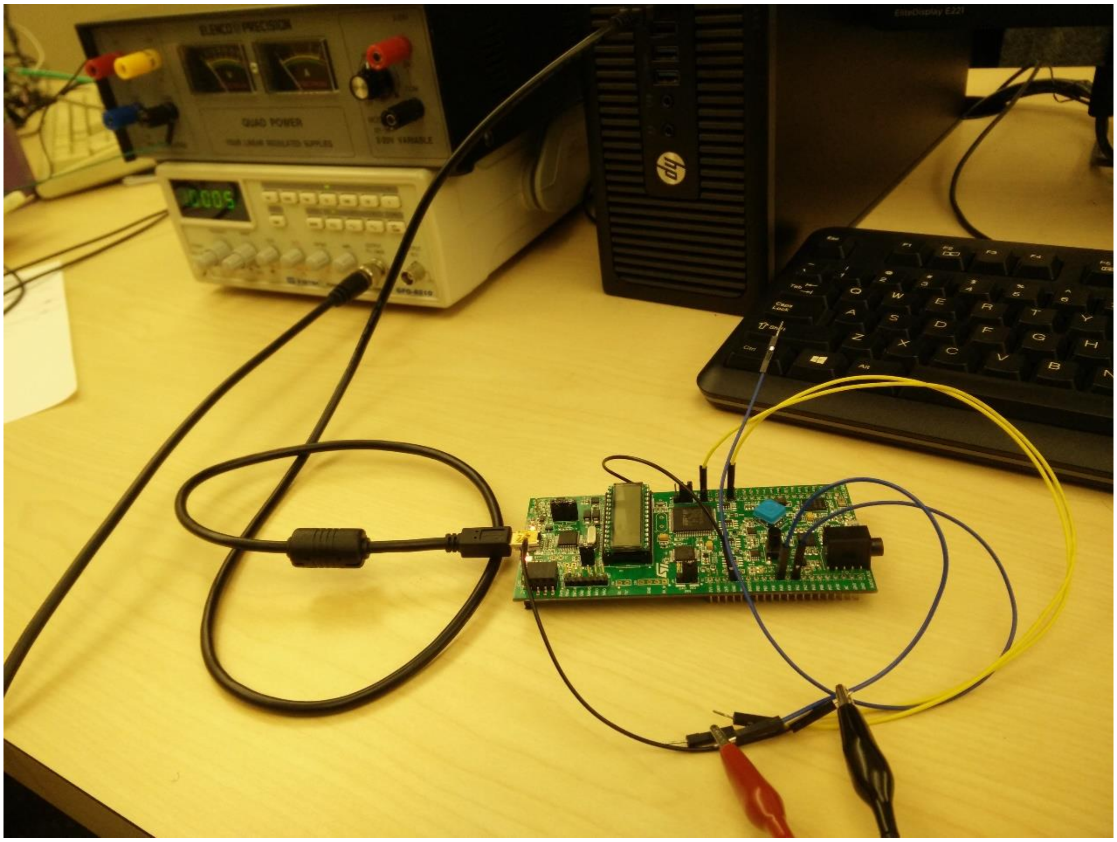


Figure4: Connecting STM32

Lessons Learned:

- The hardest part was tweaking the registers. Especially for the Input capture.
- Multiple iteration and debugging skills was improved.
- To set a value in a register sometimes bit OR operation alone would not be of suffice. Precise setting of the register is required using shifting, bit AND etc.,
- Bare metal programming and driver development for input capture and understanding how UART and LED software were used.
- Computing the clock pre-scalar.
- Working with Polling of the inputs rather than the interrupts.
- Modular coding approach.

Submission:

The project for collecting the histogram was thoroughly tested and the output were verified successfully for the given constraints. The project also introduced successfully various aspects of Bare metal embedded systems programming, functional design, coding practices, debugging and planning.