1.0 The System Architecture and Design Features

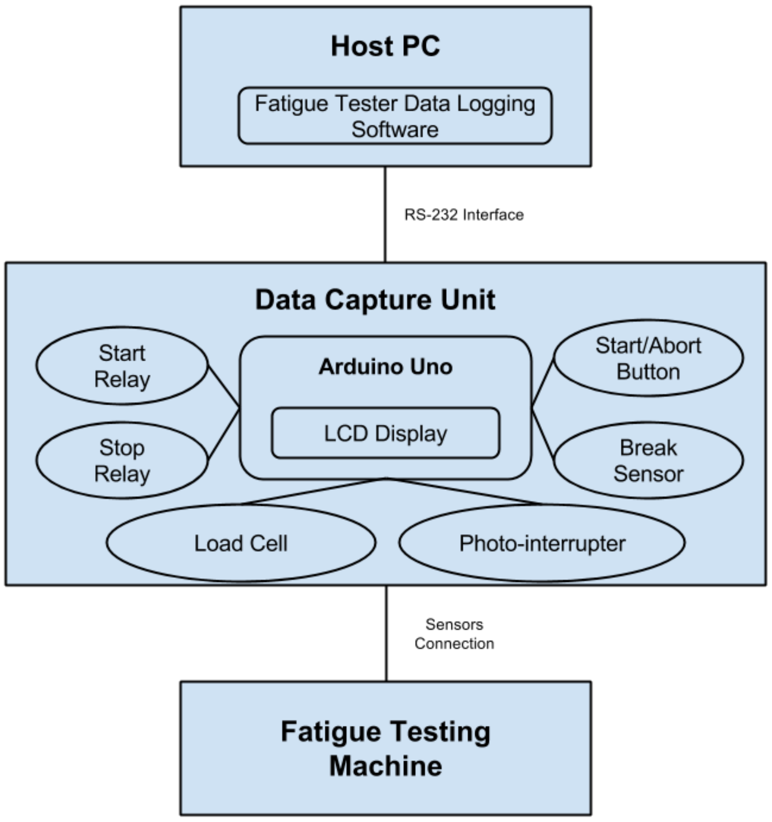


Figure 1. System Architecture

The overall guiding design philosophy behind the Data Capture and Logging System Architecture is to have a clean design that presents the system user with a simple, intuitive User Interface with fully integrated functionality synchronized across three domains/components. It must only allow for simple user and system inputs/outputs whilst hiding and automating complex system interactions.

Figure 1 shows a flow chart of the components that make up the Data Capture and Logging system. The components are as follows:

1. ***The Fatigue Tester Data Logging Software*** **-** This custom made application resides on a desktop computer and communicates with the next component (the Data Capture Unit) via a RS-232 (Serial) connection. It functions as:

   a) A semi-automatic data logger,

   b) A post processor of the data,

   c) A remote Controller of the Data Capture Unit.

   It allows the user to view live data in a table format or as a graph. Furthermore, it facilitates the exporting of the data:

   d) As a CSV file, an Excel compatible format,

   e) As an SVG image file or

   f) As a PDF file.

2. ***The Data Capture Unit (DCU)*** **-** This custom made hardware consist of several components which enables the:

   a) Capturing of data from the Fatigue Testing Machine.

   b) Unit to function as a Digital counter and timer. The following data is locally displayed by the counter feature:

      i. Total cycle Count,

      ii. Motor speed in hertz (Hz),

      iii. Input load in Newtons (N),

      iv. Test complete time,

      v. Important changes in system state.

   c) Remote displaying of the above data via the Data Logging Software.

   d) Local and remote control of the Fatigue testing machine.

3. ***The Fatigue Testing Machine*** **-** This unit is the source of all test data.

1.1 The Fatigue Tester Data Logging Software

The Fatigue Tester Data Logging Software is a custom built desktop application. This application was written using open source tools and libraries. There are a total of 2,154 lines of code in this application. The development tools used to create this application are Processing (both the programming language and Integrated Development Environment); G4P GUI Builder (http://www.lagers.org.uk/g4p/) was used to create the user interfaces and gwoptics_p5lib (http://www.gwoptics.org/processing/gwoptics_p5lib/) was used to create the 2D plots for the graphs. The final code was converted to Java and compiled to run on the Java virtual machine. This makes the code for the application portable across the Windows, Linux and Mac operating systems. Table 1 below shows the minimum system requirements for the application on windows.

Processing is a programming language based on Java developed at MIT. “Today, there are tens of thousands of students, artists, designers, researchers, and hobbyists who use Processing for learning, prototyping, and production” (https://processing.org/). Appendix _ has a copy of the full Java source code that includes all open source libraries used.

Table 1. Minimum system requirements

Operating System	Windows 7 32bit
Processor	Intel Celeron 1.2GHz
Ram	1Gb
Java	Version 7 update 40 or higher
Disk space	1 GB
Drivers	Arduino IDE 1.2 or above

1.11 Main GUI

The following is a detail description of the features and parts of the Fatigue Tester Data Logging Software as referenced in figure 2 below:

1) ***Digital Counter Stats panel*** - This panel is a remote display of the information which is displayed locally on the digital counter which is part of the Data capture unit. It shows the:

a) Motor speed in hertz (Hz),

b) Total Cycle Count,

c) Time elapsed,

d) Specimen Load (See calibration and testing section for details).

2) ***Digital Counter Log panel*** **-** This panel is the main user interactive area of the program. It consists of several sub-panels and a log view area which allows the user to interact with the program by:

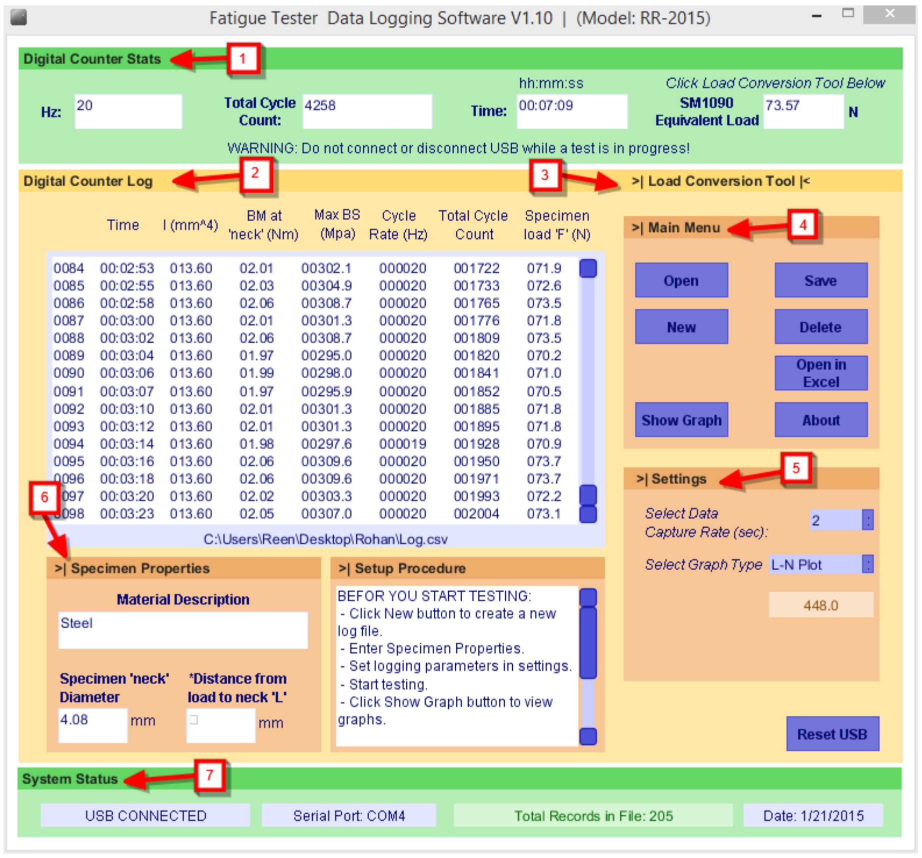
a) Viewing data in a table format,

b) Open, close, save or delete data log files,

c) Switch to the graph viewing interface,

d) Opening the log file directly in Excel for formatting, printing or any related processing activity.

3) ***Load Conversion Tool panel*** **-** Figure 3 show the load conversion tool in detail. The purpose of this tool is to allow the user to calculate the equivalent load on the SM1090 Fatigue Tester for a given input load on the RR-2015 Data Capture Unit. This can be used for calibration purposes and as an aid in determining the correct load input necessary for regular testing operations.



Fatigue Tester Data Logging Software V1.10 | (Model: RR-2015)

Digital Counter Stats

Hz: 20 | Total Cycle Count: 4258 | Time (hh:mm:ss): 00:07:09 | *Click Load Conversion Tool Below* SM1090 Equivalent Load: 73.57 N

WARNING: Do not connect or disconnect USB while a test is in progress!

Digital Counter Log

	Time	I (mm^4)	BM at 'neck' (Nm)	Max BS (Mpa)	Cycle Rate (Hz)	Total Cycle Count	Specimen load 'F' (N)
0084	00:02:53	013.60	02.01	00302.1	000020	001722	071.9
0085	00:02:55	013.60	02.03	00304.9	000020	001733	072.6
0086	00:02:58	013.60	02.06	00308.7	000020	001765	073.5
0087	00:03:00	013.60	02.01	00301.3	000020	001776	071.8
0088	00:03:02	013.60	02.06	00308.7	000020	001809	073.5
0089	00:03:04	013.60	01.97	00295.0	000020	001820	070.2
0090	00:03:06	013.60	01.99	00298.0	000020	001841	071.0
0091	00:03:07	013.60	01.97	00295.9	000020	001852	070.5
0092	00:03:10	013.60	02.01	00301.3	000020	001885	071.8
0093	00:03:12	013.60	02.01	00301.3	000020	001895	071.8
0094	00:03:14	013.60	01.98	00297.6	000019	001928	070.9
0095	00:03:16	013.60	02.06	00309.6	000020	001950	073.7
0096	00:03:18	013.60	02.06	00309.6	000020	001971	073.7
0097	00:03:20	013.60	02.02	00303.3	000020	001993	072.2
0098	00:03:23	013.60	02.05	00307.0	000020	002004	073.1

C:\Users\Reen\Desktop\Rohan\Log.csv

>| Specimen Properties

Material Description: Steel

Specimen 'neck' Diameter: 4.08 mm | 'Distance from load to neck 'L'': mm

>| Setup Procedure

BEFOR YOU START TESTING:
- Click New button to create a new log file.
- Enter Specimen Properties.
- Set logging parameters in settings.
- Start testing.
- Click Show Graph button to view graphs.

>| Load Conversion Tool |<

>| Main Menu

>| Settings

Select Data Capture Rate (sec): 2

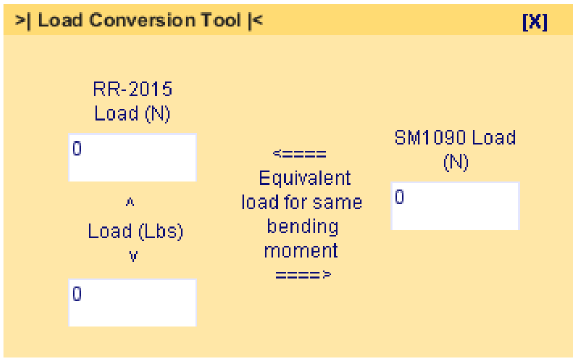
Select Graph Type L-N Plot

448.0

System Status

USB CONNECTED | Serial Port: COM4 | Total Records in File: 205 | Date: 1/21/2015

Figure 2. Main Software GUI



>| Load Conversion Tool |< [X]

RR-2015 Load (N): 0

^ Load (Lbs) v: 0

<==== Equivalent load for same bending moment ====>

SM1090 Load (N): 0

Figure 3. Load Conversion Tool

4) ***Main Menu panel*** - This panel contains the main menu functions that the user would need to interact with the program.

5) ***Settings panel*** - This panel provides the interface which allows the user to set

the program's automatic data capture rate in seconds and select the graph type to plot. User can select a minimum capture rate of two (2) seconds and a maximum of sixty (60) seconds in steps of two (2) seconds. The systems has a rate precision of plus or minus one (1) second. At the moment only the LN plot graph type is implemented, but provision has been made for an S-N curve feature to be implemented if necessary. The user can also select None as the graph type to manually update the data log.

6) ***Specimen Properties panel*** - This panel allows the user to enter the properties of the specimen being currently tested.

7) ***System Status panel*** - This panel displays the current status of the program. It indicates whether there is a valid USB connection to the Data Capture Unit and the serial port that is being used for communication. The user can use the Reset USB button provided to reestablish a broken connection. It also shows the total records in the log table as well as the current date as additional information.

1.12 Graph GUI

Figure 4 shows a screen shot of the Graph program interface. This interface allows the user to view a live plot of the data whiles a test is in progress or a plot of data previously recorded and saved. It consists of the following sections:

1) ***The General graph plot area*** - This is the area which displays the graph.

2) ***The Zoom Tool panel*** - This panel allows the user to zoom in and expand the plot points displayed to view small details on the graph. The user can select the following zoom options:

a) *None* - This option displays the all the data at once.

b) *Beginning* - This option displays the first half of the data set to the user.

c) *Graph End* - This option displays the second half of the data set to the user.

After selecting one of the above options the user can then use the slider on the tool to increase or decrease the range of data to be viewed.

3) ***The Graph menu panel*** - This panel contains the main menu functions which allows the user to:

a) Save the graph as a PDF or SVG file,

b) Open a saved log file to graph its contents,

c) Navigate back to the Main GUI.

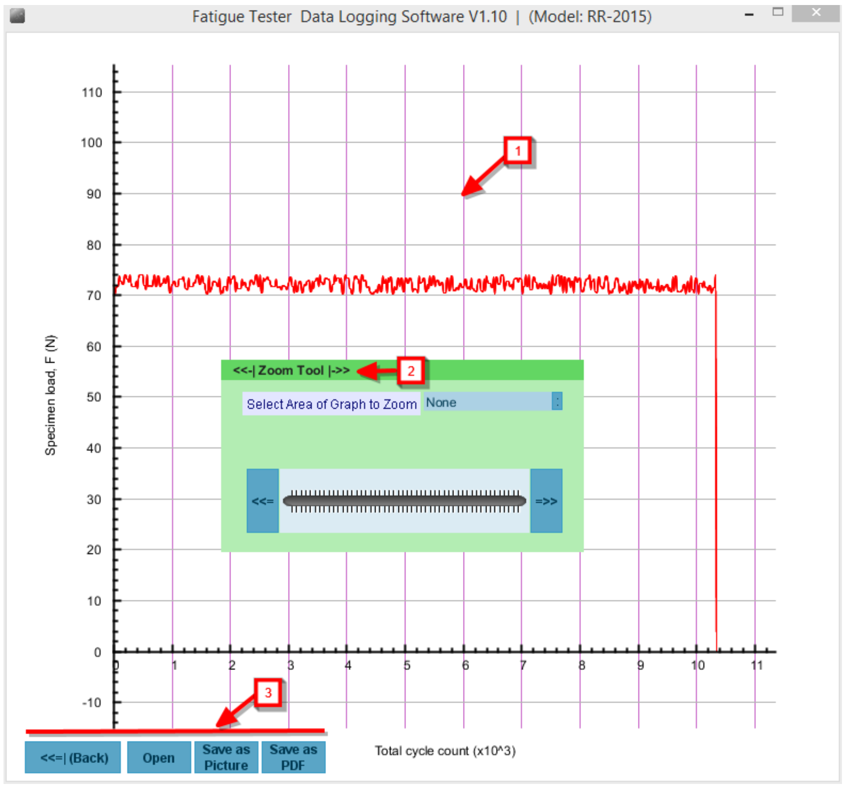


Figure 4. Graph GUI

1.13 The Control Panel

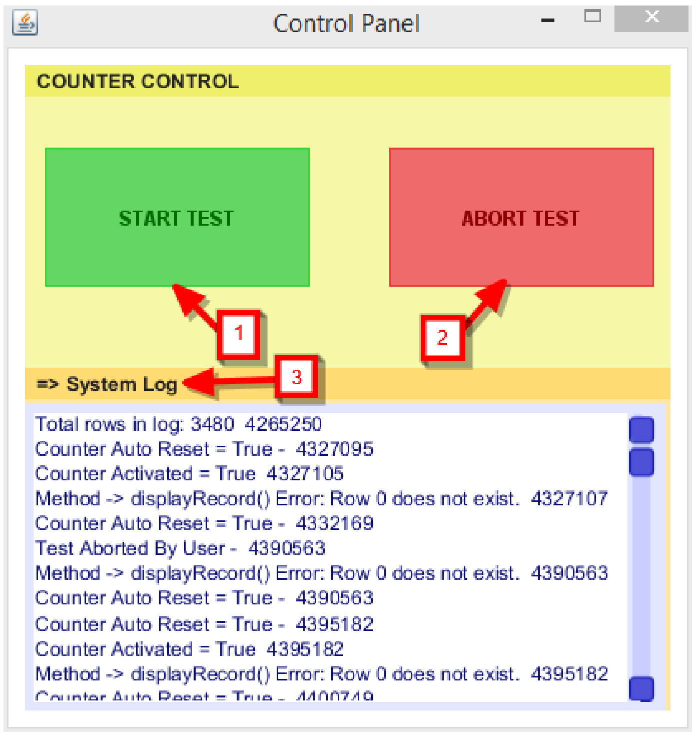
The Control Panel is the interface which allows the user to remotely control the Data Capture Unit. Figure 5 shows a screen shot of the window. Its exist as a separate window in the Main application and has the following features:

1) ***Start Test Button*** - This button allows for the remote starting of the fatigue tester via the computer.

2) ***Abort Test Button*** - This button allows the user to remotely abort a test in progress.

3) ***System Log*** - This provides information as to the state of the Data capture unit as well as that of the Data logging software. This information is useful for debugging operational errors that may present themselves during the use of the system.

4) ***Reset Button*** - This button allows the user to remotely reset the Data Capture Unit in case of operational errors.



```
Total rows in log: 3480  4265250
Counter Auto Reset = True -  4327095
Counter Activated = True  4327105
Method -> displayRecord() Error: Row 0 does not exist.  4327107
Counter Auto Reset = True -  4332169
Test Aborted By User -  4390563
Method -> displayRecord() Error: Row 0 does not exist.  4390563
Counter Auto Reset = True -  4390563
Counter Auto Reset = True -  4395182
Counter Activated = True  4395182
Method -> displayRecord() Error: Row 0 does not exist.  4395182
Counter Auto Reset = True -  4400749
```

Figure 5. The Control Panel

1.14 Internal Program Logic and Control

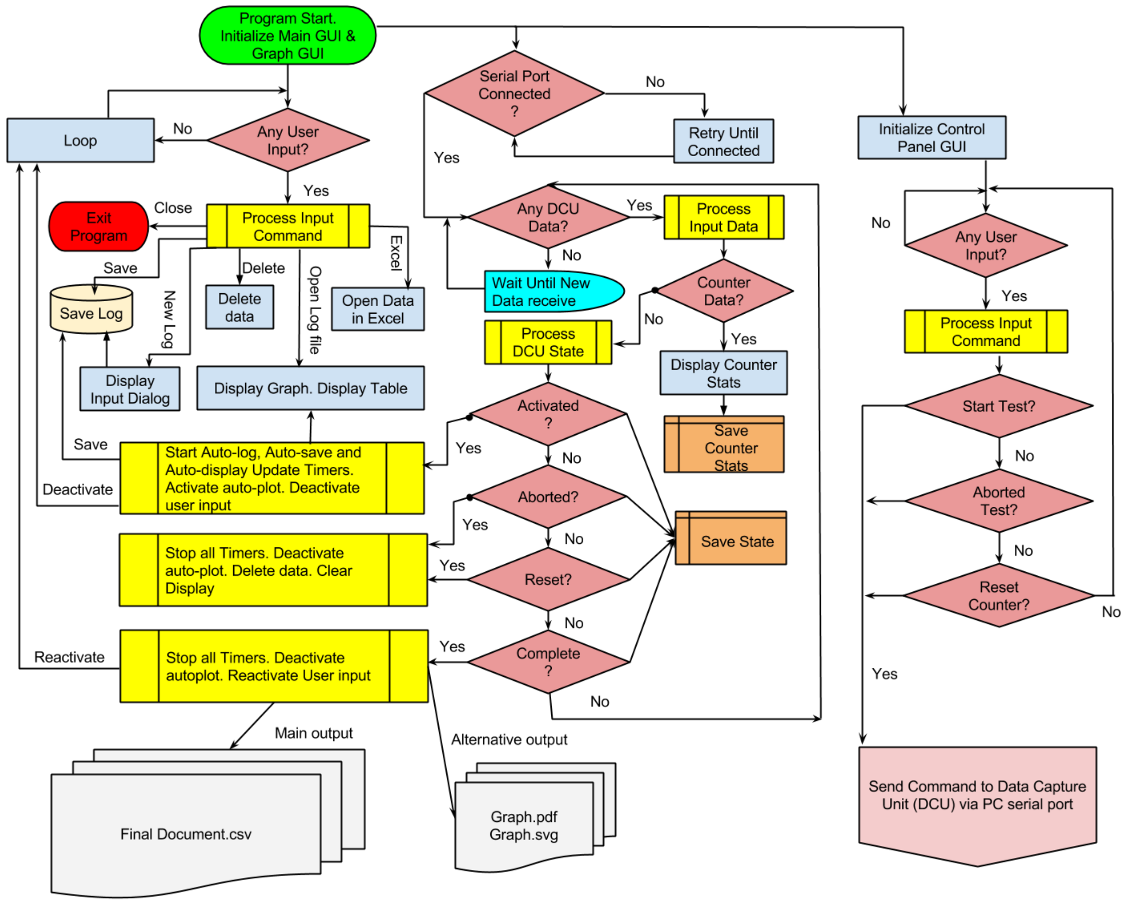


Figure 6. Application Flow Chart

Figure 6 shows a flow chart that is a simplified representation of how the application’s internal logic and control was designed. The application is designed as a multi-threaded program. When the user starts the program its various properties and graphic user interfaces (GUI) are initialized and a serial connection is established between the application and the Data capture unit. If no connection is made, the application continues to try until such connection is established. Concurrent with the above process, the application also waits for the user to press a menu button or make changes to settings. The user inputs/selections are then processed and the required action/s are performed by the application.

Concurrently the application also reads any serial data received and process it then performs the required actions automatically. If the user clicked a button on the Control panel, the application sends the required command via the serial port to the Arduino Uno in the DCU. The Uno process the command, performs the actions, then send a status/state message back to the application. The application process the resulting status message, then performs the required actions automatically for the duration of a test. At the completion of a test the application outputs a standard formatted log.csv file. The standard format is shown in Table 2 below.

Table 2. Standard Data output format

Time	**Specimen Properties**			**Calculated Data**			**RR-2015**		
Time	**Material Description**	**Specimen 'neck' Diameter**	**Distance from load to neck 'L' (mm)**	**Second Moment of Area (mm^4)**	**Bending Moment at 'neck'(mm^4)**	**Maximum Bending Stress (Mpa)**	**Cycle Rate (HZ)**	**Total cycle Count**	**Specimen load 'F' (N)**

1.15 Application Usage Procedure

Before the user setup the Fatigue testing machine to begin testing they must follow the following procedure to setup the Data logging application:

- Lunch the Fatigue Tester Data Logging Software from the desktop folder where it saved.
- Click the New button on the Main menu panel to create a new log file then choose the directory location on the computer where the file will be save from

the file dialog. Give the file a unique name then click Ok. You can also open an existing file and overwrite its contents.

- Enter the Specimen Properties in the properties panel. These properties are used to calculate addition data about the test specimen, so please ensure that the information is correctly entered.
- Set logging parameters in settings. Determine the data capture rate and the graph type you want to work with then select the appropriate option from the lists provided.
- Ensure that the USB cable is connected to the computer and the Data Capture Unit. Check the System status panel to ensure that a valid connection exist. If it indicates that the USB is disconnected click the Reset USB button and follow the instructions provided. Proceed to the next step upon conformation of a positive status.
- Setup the Fatigue Testing machine. See Data Capture Unit setup procedure for directions.
- Begin testing. Click the Start Test button on the Control panel or green button on the Data Capture Unit.
- Click the Show Graph button on the Main menu panel to view graph.

After testing complete:

- Click the Save button to Save the log file.
- Click Open in Excel button to view log file in excel to format for printing.
- Click Save as picture or Save as PDF button to export the graph as a PDF or SVG file. Figures 7, 8 & 9 shows sample outputs from the program.
- Restart the above steps from the beginning for next test.

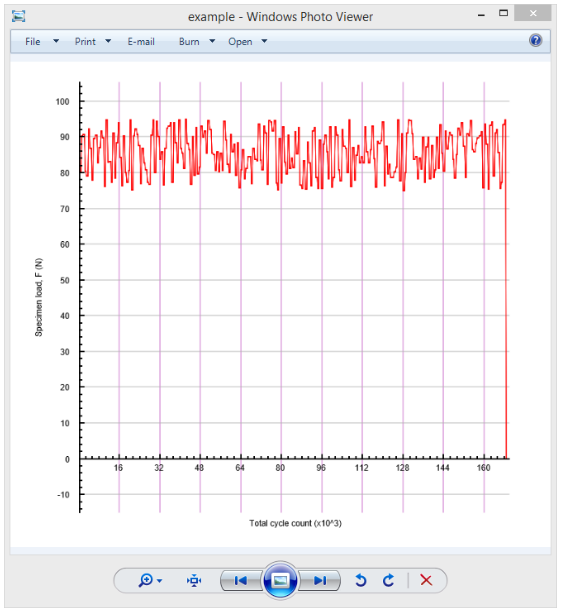


Figure 7. An example of a picture output

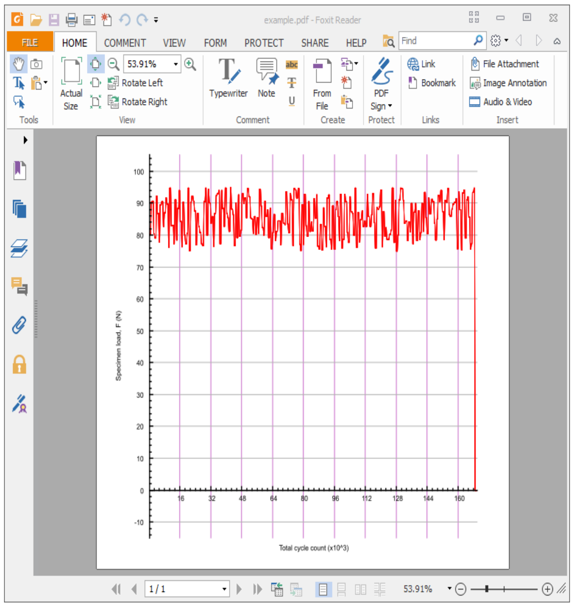
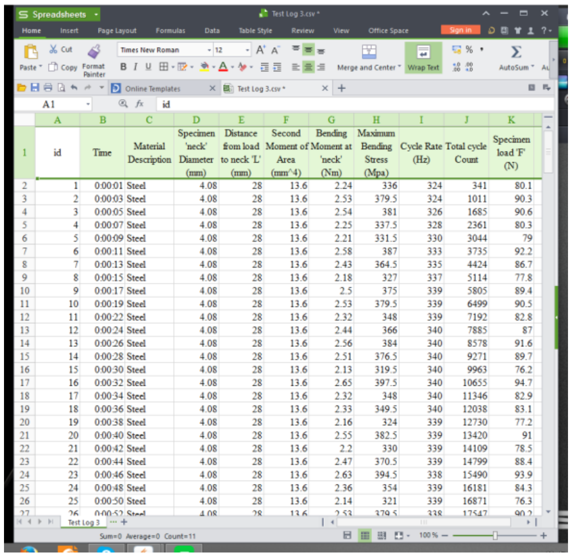


Figure 8. An example of a PDF output



A1 id

	A	B	C	D	E	F	G	H	I	J	K
1	id	Time	Material Description	Specimen 'neck' Diameter (mm)	Distance from load to neck 'L' (mm)	Second Moment of Area (mm^4)	Bending Moment at 'neck' (Nm)	Maximum Bending Stress (Mpa)	Cycle Rate (Hz)	Total cycle Count	Specimen load 'F' (N)
2	1	0:00:01	Steel	4.08	28	13.6	2.24	336	324	341	80.1
3	2	0:00:03	Steel	4.08	28	13.6	2.53	379.5	324	1011	90.3
4	3	0:00:05	Steel	4.08	28	13.6	2.54	381	326	1685	90.6
5	4	0:00:07	Steel	4.08	28	13.6	2.25	337.5	328	2361	80.3
6	5	0:00:09	Steel	4.08	28	13.6	2.21	331.5	330	3044	79
7	6	0:00:11	Steel	4.08	28	13.6	2.58	387	333	3735	92.2
8	7	0:00:13	Steel	4.08	28	13.6	2.43	364.5	335	4424	86.7
9	8	0:00:15	Steel	4.08	28	13.6	2.18	327	337	5114	77.8
10	9	0:00:17	Steel	4.08	28	13.6	2.5	375	339	5805	89.4
11	10	0:00:19	Steel	4.08	28	13.6	2.53	379.5	339	6499	90.5
12	11	0:00:22	Steel	4.08	28	13.6	2.32	348	339	7192	82.8
13	12	0:00:24	Steel	4.08	28	13.6	2.44	366	340	7885	87
14	13	0:00:26	Steel	4.08	28	13.6	2.56	384	340	8578	91.6
15	14	0:00:28	Steel	4.08	28	13.6	2.51	376.5	340	9271	89.7
16	15	0:00:30	Steel	4.08	28	13.6	2.13	319.5	340	9963	76.2
17	16	0:00:32	Steel	4.08	28	13.6	2.65	397.5	340	10655	94.7
18	17	0:00:34	Steel	4.08	28	13.6	2.32	348	340	11346	82.9
19	18	0:00:36	Steel	4.08	28	13.6	2.33	349.5	340	12038	83.1
20	19	0:00:38	Steel	4.08	28	13.6	2.16	324	339	12730	77.2
21	20	0:00:40	Steel	4.08	28	13.6	2.55	382.5	339	13420	91
22	21	0:00:42	Steel	4.08	28	13.6	2.2	330	339	14109	78.5
23	22	0:00:44	Steel	4.08	28	13.6	2.47	370.5	339	14799	88.4
24	23	0:00:46	Steel	4.08	28	13.6	2.63	394.5	338	15490	93.9
25	24	0:00:48	Steel	4.08	28	13.6	2.36	354	339	16181	84.3
26	25	0:00:50	Steel	4.08	28	13.6	2.14	321	339	16871	76.3
27	26	0:00:52	Steel	4.08	28	13.6	2.53	379.5	338	17547	90.2

Figure 9. An example of a Spreadsheet/Excel output

1.2 The Data Capture Unit

This is a custom built hardware consisting of several components as shown in figure 10, 11 & 12 and detailed below. At the heart of the Unit is an Arduino Uno, a popular prototyping platform. The Uno takes the ATmega328 Micro-controller and its related properties and exposed them in an easy to use manner which allows for the design and rapid prototyping of Mechatronics projects as well as Instrumentation and Controls projects. Table 3 below shows a summery of the Arduino Uno specifications.

Table 3. Summery specifications

Microcontroller	ATmega328
Operating Voltage	5V
Digital I/O Pins	14
DC Current per I/O Pin	40 mA
Flash Memory	32 KB (ATmega328)
SRAM	1 KB (ATmega328)
Clock Speed	16 MHz

(Source: http://arduino.cc/en/Main/ArduinoBoardUno)

1.21 Components and Design

The Data Capture Unit is designed around the following components as shown in figures 10, 11 & 12:

1) ***LCD Counter Screen*** - This is a 16 x 2 character LCD. It shows the:

e) Motor speed in hertz (Hz),

f) Total Cycle Count,

g) Time elapsed (When test completes),

h) Specimen Load.

2) ***LED Indicator*** - This indicates when the counter is activated.

3) ***Arduino Uno*** - This is the Microcontroller board which manages the other components of the Data Capture Unit. It receives input data from the Load Cell sensor, the Break Sensor, the Photo-interrupter Sensor and the Start/Abort button processes it, then outputs the results to the LCD Counter Display and to the serial port.

4) ***Potentiometer*** - This is use to control the brightness of the LCD.

5) ***Start/Abort Button*** - This is a momentary push button switch and it is the only user input into the system. A single press activates the counter and starts a test. If the counter is activated and the button is pressed again while a test is in progress the system will change from activate to abort mode and stop the test. This will also trigger the data logging program on the PC to auto delete any log data that was previously captured and saved.

6) ***Photo-Interrupter Sensor*** - This sensor is connected to one of the Arduino Uno external interrupts and it controls the counters used to calculate the motor speed and the Total Cycle Count of the test.

7) ***Break Sensor*** - This is a momentary long hinge lever switch. It is connected to the other Arduino Uno external interrupt. When a change in the state of the switch is detected by the Uno, the system will immediately change state from Activated to Test Complete, stop the motor and lock the Total Cycle counter.

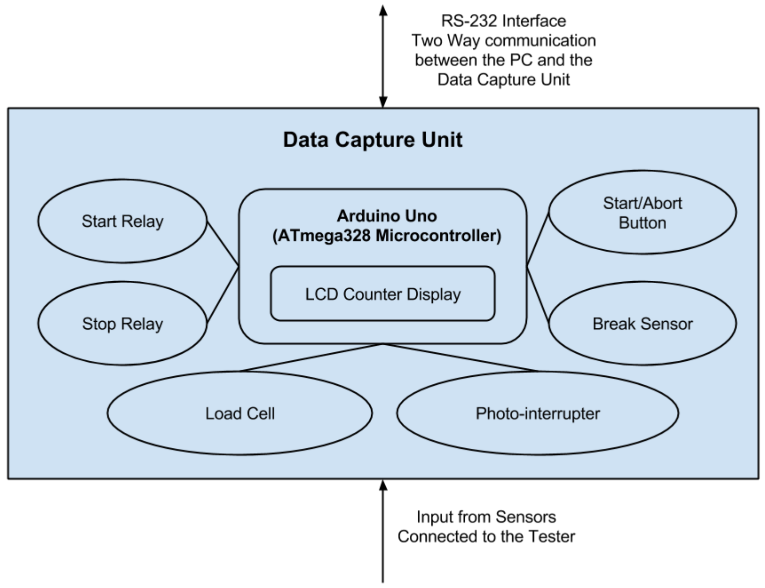

RS-232 Interface
Two Way communication between the PC and the Data Capture Unit
Data Capture Unit
Start Relay
Arduino Uno (ATmega328 Microcontroller)
Start/Abort Button
LCD Counter Display
Stop Relay
Break Sensor
Load Cell
Photo-interrupter
Input from Sensors Connected to the Tester

Figure 10. Control Unit Architecture

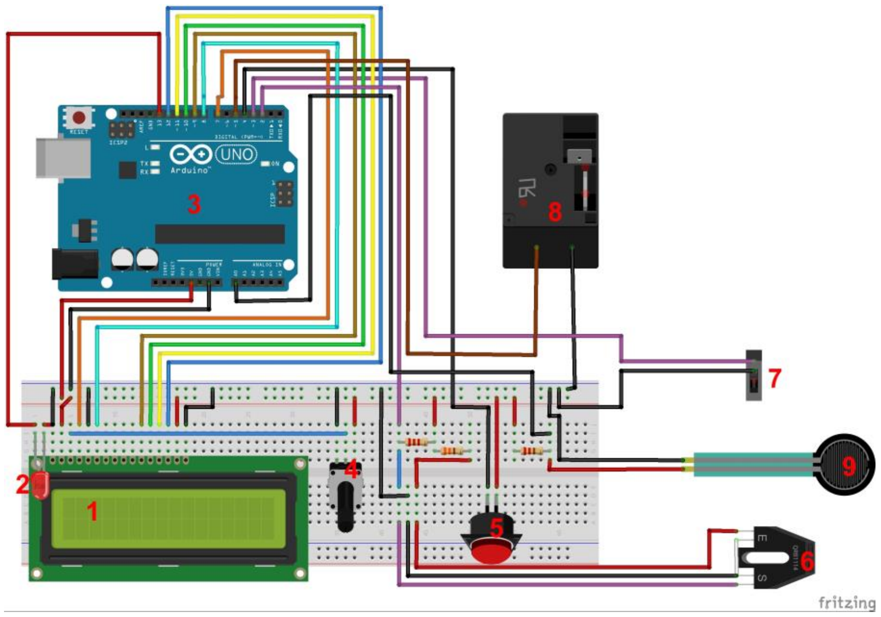


Figure 11. Data Capture Unit wiring diagram

8) ***Start/Stop Relay*** - This relay is interfaced into the motor circuit of the fatigue tester. Its is used to control the starting and stopping of the motor on the machine.

9) ***Load Cell Sensor*** - The image in the figure 11 is a place holder for the load cell sensor. Figure 12 show the complete load cell circuitry. This sensor is used to capture the load hanging from the specimen at the beginning of a test and any variation in the load during the test. Table 4 show a summary of its specifications.

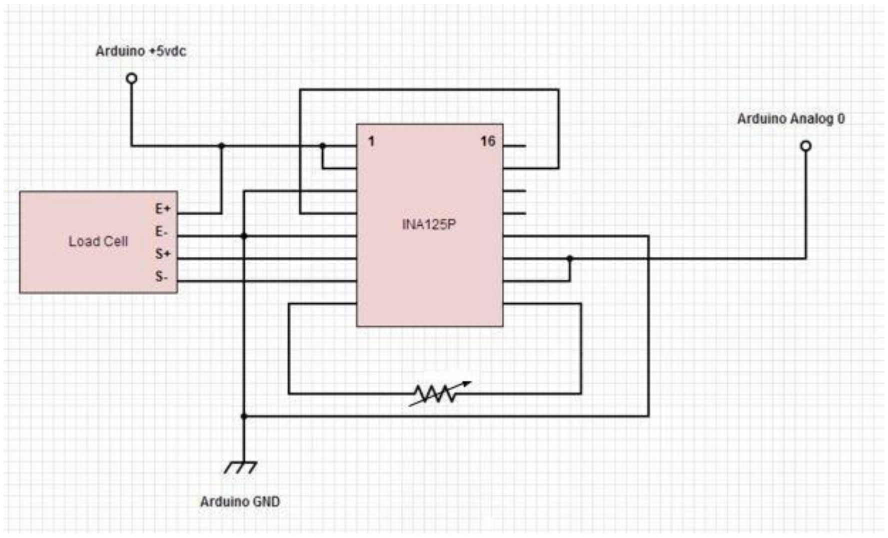


Figure 12. Load Cell Wiring Diagram

(Source:http://www.instructables.com/id/Arduino-Load-Cell-Scale/)

Table 4. Summary of the Load Cell specifications

Rated Load	5kg - (50N)
Safe Overload	150% - (123N)
Ultimate Overload	200% - (147N)
Operating Temperature Range	-10 deg to 40 deg

(source:http://www.amazon.com/gp/product/B00900PALA/ref=oh_aui_detailpage_o07_s00?ie=UTF8&psc=1)

1.22 Internal Program Logic and Control

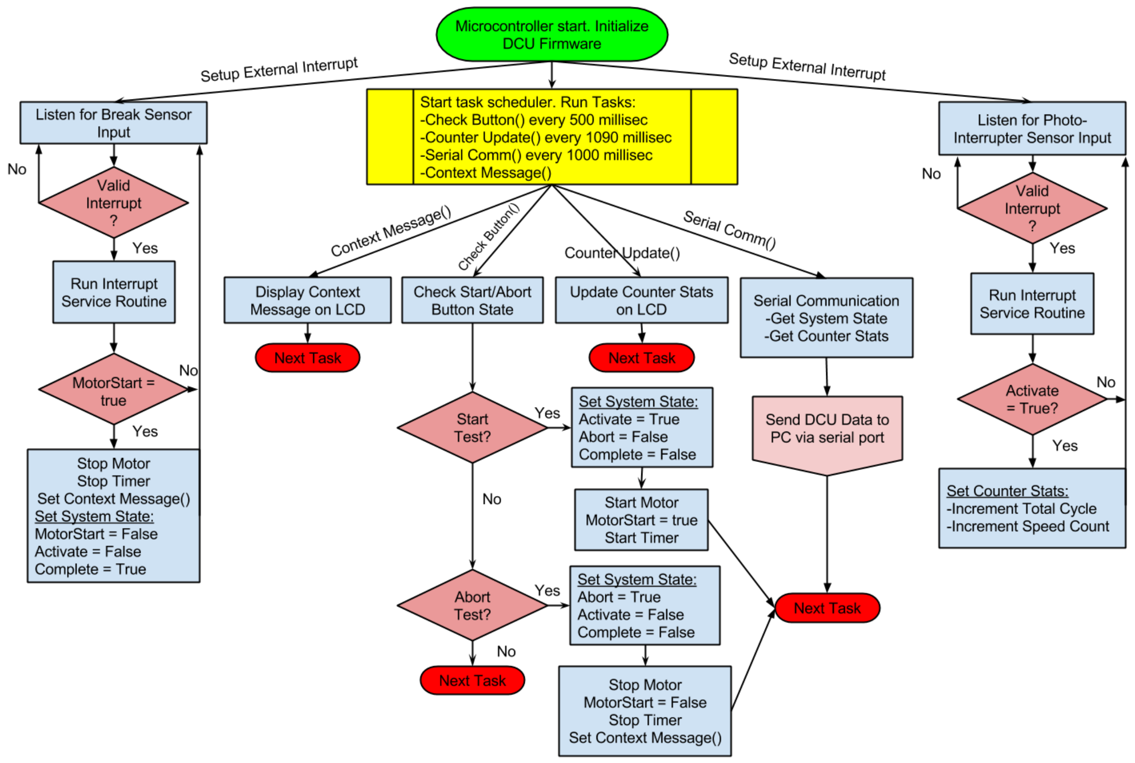


Figure 13. The Data Capture Unit (DCU) program flow chart

The firmware code that implements the Data Capture Unit's programming logic was written to enable a real time system response. It was written in the Arduino IDE using open source libraries. The current implementation have a total of 527 lines of code This is needed because the entire Data Capture and Logging System Architecture interactions are time dependent. The system needs to be able to:

a) Respond quickly to a user input.

b) Process the data output to the LCD and Serial port at fixed time intervals.

c) Respond immediately to every single rotation of the motor output shaft to count the test cycle and measure motor speed.

d) Respond immediately to a complete break in the test specimen.

e) Keep track of the time elapsed during the test.

f) Measure the load applied to the specimen and keep track of any variations during testing.

To achieve this level of real time response, a mixture of interrupt driven routines and time scheduled tasks were used. The tasks were manually scheduled as a part of the system programming and the scheduler implements a round robin algorithm to manage the scheduled tasks at run time. The interrupt services routines handlers ensure that time critical data generating events, such as those generated by the break sensor and the photo-interrupter, are processed immediately when they occur. Figure 13 show a simplified flow diagram detailing the general programming logic. For a more detailed understanding of the process see Appendix_ for the full source code.

1.23 Hardware Usage Procedure

Before the user setup the Fatigue testing machine to begin testing they must follow the following procedure:

- ➢ Setup the Fatigue Data Logging Software.
- ➢ Setup the specimen in the Fatigue testing machine.
- ➢ Apply the required loads.
- ➢ Ensure that all safety covers are in place then plug in the power cable for the machine.
- ➢ Ensure that the machine is clear of any impediments that can prevent the rotation of the motor, then turn on the Data Capture Unit.Begin testing by either pressing the Green Start/Abort button on the DCU or the Start Test button on the Data Logging Software Control Panel. The system will now take over and automate the entire test.

After test Completes:

- Remove the safety cover from the machine.
- Remove the loads used in the test.
- Remove the broken specimen.
- Clean work area.
- Repeat the above steps for a new test.

1.4 Poka Yoke

Despite the system simple design and limited room for user interaction, as users use the system they are bound to make errors that can affect the overall system operation. Part of the system design was to predict common user errors and implement error correction/prevention measures in the system to minimize their impact. The following outlines the precautions taken in each of the three components of the system in this regard:

1) ***The Fatigue Tester Data Logging Software*** **-** The following are inherent in the program design to minimize errors:

a) The user is only allowed to change setting, enter specimen properties or use most Menu functions when a test is not in progress. During a test all user input/output are blocked except those that are need to change data views or save the graph.

b) The user is only allowed to view the log file in the Data Logger. They are not allowed to make changes.

c) The program is design to automate the configuration of the graph and the initial formatting of the output Log is fixed. The user can only change this formatting by importing the log in a spreadsheet application like Excel. Please note, if the user delete any of the columns in the file it would render the log file useless and it would not be properly displayed when reopened in the Data Logger.

d) The program is designed to automate the data logging process unless the user select a manual update process. In that case the program presents the user with a manual update button which the user can use to update the log.

e) An error log was built in the system to capture errors produced by the system due to user error or random system error. These are errors were not predicted and designed for however, there is a built-in error handling routine in each method/function that will ensure that these errors do not stop the operation of the program. Only critical system errors outside the control of the program will impact the program operation.

2) ***The Data Capture Unit*** - The Data Capture Unit was designed as a “black box” with no visible way for the user to change settings or system control variables. The only interaction the user have is to turn the system on and off and to start/Abort a test. This ensures that the system always works the way it was designed to work. To change any of the DCU configuration the system will need to be reprogrammed.