Loughborough University

Task 3 Testing Procedure

Joshua Smith B221322, Igor Spirin B020777
10/3/2017

Contents

General Testing Strategy: 2
Manual 2
Automatic 2
Static 2
Visual inspection 2
Linting 2
Dynamic 2
White Box test with breakpoints 2
Black Box self-test using success/failure of LED indicators 2
Manual Tests 2
Self-Tests 2
Simulated user input 2
Black Box 2
White Box 4
Results of tests 5

General Testing Strategy

	Manual	Automatic
Static	Visual inspection	Linting
Dynamic	White Box test with breakpoints	Black Box self-test using success/failure of LED indicators

Table 1: The testing approach combined elements from manual, automatic, static and dynamic testing approaches

Manual Tests

The following manual tests were performed:

- Pressing the use input button to see if the LED comes on
- Visual inspection the code in order to detect errors

Self-Tests

Linting: *A flag is raised in response to incorrect or suspicious constructs*

Load test vector into data_store: *If no data is loaded, no output can be produced*

Compare with output_log to pre-calculated expected values: *This will be further discussed later in this paper*

Simulated user input

The general methodology was to carry out a black box test followed by a white box test if it was deemed appropriate. Errors create by user inputs were deemed to be unlikely and inconsequential seeing as it would be kept to a minimum throughout the operation of the device.

Black Box

A known data set was given to the program to process. A convolution, mean and variance operations were performed on the data set in order to produce a known output that could then be checked against the output of the code.

The test would be considered successful if the code output matched the known output values.

An important note, however, is that the values used in the code are truncated, thus there may be minor cases of information loss.

The values were computed in Excel and are as follows:

i	Amplitude	Filtered data	Mean	Variance
1	0			
2	49			
3	98			
4	147			
5	195			
6	243			
7	290	2		
8	337	0		
9	383	2.5		
10	428	0.5		
11	471	3	1.6	1.34
12	514	1.5	1.5	1.3
13	556	4	2.3	1.46
14	596	1.5	2.1	1.54
15	634	3.5	2.7	1.06
16	672	3	2.7	1.06
17	707	5	3.4	1.34
18	741	1.5	2.9	1.74
19	773	6.5	3.9	2.94
20	803	2.5	3.7	3.26
21	831	5.5	4.2	3.56
22	858	4	4	3.4
23	882	5.5	4.8	1.96
24	904	3.5	4.2	1.36
25	924	6.5	5	1.2
26	942	4.5	4.8	1.16
27	957	6	5.2	1.16
28	970	5	5.1	1.14
29	981	6.5	5.7	0.66
30	989	5.5	5.5	0.5
31	995	5.5	5.7	0.26
32	999	6.5	5.8	0.36
33	1000	5.5	5.9	0.24
34	999	5.5	5.7	0.16
35	995	7	6	0.4
36	989	5	5.9	0.54
37	981	7	6	0.7
38	970	5.5	6	0.7
39	957	5.5	6	0.7
40	942	6.5	5.9	0.54
41	924	5.5	6	0.4
42	904	5.5	5.7	0.16
43	882	6.5	5.9	0.24

44	858	5	5.8	0.36
45	831	6	5.7	0.26
46	803	4.5	5.5	0.5
47	773	6.5	5.7	0.66
48	741	3.5	5.1	1.14
49	707	5.5	5.2	1.16
50	672	4	4.8	1.16
51	634	5.5	5	1.2
52	596	2.5	4.2	1.36
53	556	6.5	4.8	1.96
54	514	1.5	4	3.4
55	471	5	4.2	3.56
56	428	3	3.7	3.26
57	383	3.5	3.9	2.94
58	337	1.5	2.9	1.74
59	290	4	3.4	1.34
60	243	1.5	2.7	1.06
61	195	3	2.7	1.06
62	147	0.5	2.1	1.54
63	98	2.5	2.3	1.46
64	49	0	1.5	1.3

Table 2: Table of pre-calculated values

Also LEDs were used as indicators to show that both the tasks are running and after completing all the vectors the results were successful.

Led #	**State**
LED9	Success
LED10	Failure

Table 3: LED state table

In the event that the black box fails, white box test procedure would be initiated with the above parameters. However the code would be run in individual blocks rather than as a whole.

The user input button was out of scope of the tests. Therefore during testing it should not be used.

White Box

White Box test involves iterating through specific areas of code in order to perform a deeper analysis of the code. It allows detection of more sophisticated errors that may either be hard to detect by running the whole code.

The methodology used in relation to White Box testing was similar to what was used for black box testing in a sense that is was performed by using the same data set with a known output from convolution performed by the code.

Results of tests

The following points were the main takeout from the test procedures.

- In general avoiding minor errors such as mismatching parenthesis was handled very well by the linter
- Visual inspection was useful in detecting wrong variables such that the results did not result in failure however the variables use to produce the were not accurate
- Black Box testing was simple to perform and was a clear indicator of fatal flaws in the code
- Some issues were contained within specific functional blocks. Thus Black Box testing was ineffective in locating the specific location of a defect. However White Box testing allowed for easier detection of defects through inspection of individual software components