Test Report for Knaves NES

Niko Savas Joe Crozier Sam Nalwa

1300247 1311502 1332792

November 28, 2015

Software Engineering 3XA3, Lab #3

Contents

1 Non-Functional Tests

1.1 Test 1 - CPU Speed

Summary: This test is executed by checking the frequency and cycles per second of the emulated 6502 CPU. This test will give us a more accurate idea on how close our cpu emulation is to the actual 6502 CPU, and is one that must be executed as an emulator relies heavily on the CPU being able to keep up so that there is no delay within the system causing lag later on when the graphics need to be rendered. This test is also required to make sure that the emulated process is not running faster than required, since that would also have a negative impact once the entire emulator is constructed including the graphics rendering module.

Process: This test will be done within the code by using the inbuilt system timer. The time taken by each instruction set will be measured, and then the frequency and cycles per second of the processor will be calculated. The test will then be repeated over and over again until the desired emulated processor speed is achieved.

Result: This test is due to be completed later on in the duration of this project, and thus there are no results for this test as of yet.

1.2 Test 2 - Usability Test

Summary: This test is done to ensure that the software is easy to use. The test requires at least one test subject who is presently unfamiliar with the software, but is capable of basic BASH commands.

Process: The test subject is instructed to enter the command knavesnes –help which echo a short help document into the console. The test subject is then instructed to use this information to successfully run a ROM file in the current directory. The test is deemed successful if the subject is able to execute the ROM without any further assistance.

Initial State: knaves-nes folder open in terminal and rom file stored within folder

Input: User enters knavesnes -help and finds out how to open the .nes file in the emulator, and follows the instructions to successfully open the file

Expected Results : The user is able to successfully open the .nes file without any issues.

Test Results : Passed; Actual output matched expected output

2 System Test Cases

2.1 Test 1 - LDA Instruction Test

Initial State: A terminal window is opened in the folder which contains the knavesnes executable and LDA Instruction test rom file.

Input: ldatest.nes with the instruction included in appendix [1].

Expected Result: The outputted log file with the memory state outlined in appendix [2].

Test Result: Passed; Actual output matched expected output

2.2 Test 2 STA Instruction Test

Initial State: A terminal window is opened in the folder which contains the knavesnes executable and STA Instruction test rom file.

Input: statest.nes with the instruction included in appendix [3].

Expected Result: The outputted log file has the memory state outlined

in appendix [4].

Test Result: Passed; Actual output matched expected output

2.3 Test 3 ADC Instruction Test

Initial State: A terminal window is opened in the folder which contains the knavesnes executable and ADC Instruction test rom file.

Input: adctest.nes with the instruction included in appendix [5].

Expected Result: The outputted log file is in agreement with the desired memory state outlined in appendix [6].

Test Result: Passed; Actual output matched expected output

2.4 Test 4 TAX Instruction Test

Initial State: A terminal window is opened in the folder which contains the knavesnes executable and TAX Instruction test rom file.

Input: taxtest.nes with the instruction included in appendix [7].

Expected Result: The outputted log file is in agreement with the memory state outlined in appendix [8].

Test Result: Passed; Actual output matched expected output

2.5 Test 5 CMP Instruction Test

Initial State: A terminal window is opened in the folder which contains the knavesnes executable and CMP Instruction test rom file.

Input: cmptest.nes with the instruction included in appendix [9].

Expected Result: The outputted log file is in agreement with the memory state outlined in appendix [10].

Test Result: Passed; Actual output matched expected output

2.6 Test 6 BNE Instruction Test

Initial State: A terminal window is opened in the folder which contains the knavesnes executable and BNE Instruction test rom file.

Input: bnetest.nes with the instruction included in appendix [11].

Expected Result: The outputted log file is in agreement with the memory state outlined in appendix [12].

Test Result: Passed; Actual output matched expected output

2.7 Test 7 Loading a Non-Existent ROM file

Initial State: A terminal window is opened in the folder, which contains the knavesnes executable and does not contain a file named xxyyzz.nes.

Input: xxyyzz.nes as a non-existent file.

Expected Result: Program terminates and tells user to re-check the file for its existence.

Test Result: Passed; Actual output matched expected output

2.8 Test 8 Loading a Corrupted ROM file

Initial State: A terminal window is opened in the folder, which contains the knavesnes executable and a corrupt (originally a word doc) file named mario.nes.

Input: mario.nes as a corrupt rom file.

Expected Result: Program terminates and tells user to re-check the file as it is corrupt/unreadable.

Test Result: Passed; Actual output matched expected output

3 Automated Testing

Automated testing can be done using a bash script, which runs through a list of .nes ROM files, and for each file runs knavesnes, takes a screenshot of the screen, and goes onto the next one. These screenshots can be quickly scrolled through to make sure all the games have executed properly. This list can be as big as a 1000 games, and the test can be automated to run over and over again to make sure it always produces the correct results. Automated testing is part of our project, but is one that will be completed later on in the project and thus results for automated testing are not ones that will be included as part of this report.

4 Traceability to Requirements Document

All test cases outlined in this document follow the requirements set out for this project in the requirements document. All use cases as outlined in Section 6 of the requirements document have adequately been tested as a part of this test report, and we believe that all the test results, which use modules outlined in the use cases, have been successful.

5 Traceability to Modules

The cartridge, CPU, and memory modules have all been tested by means of our test cases. In order to run each and every one of our tests all three modules were used, and therefore we can conclude that all modules are working well in individually and in conjunction to achieve the results that we had planned for.

6 Code Coverage Metrics

Code coverage metrics have not been utilized for this project yet as they are part of the long-term project and not within the scope of what we expect to complete by the end of this course.

Appendices

Appendix 1

```
LDA #$c0
```

Appendix 2

```
A = $c0
X = $00
Y = $00

All mem = $00
```

Appendix 3

```
LDA #$10
STA $01
LDA #$20
STA $02
LDA #$c0
STA $c2
LDA #$fe
STA $00
LDA #$19
STA $00
```

Appendix 4

```
A = $19 X = $00 Y = $00

NV-BDIZC
00110000
```

Appendix 5

```
LDA #$10
STA $01
ADC $01
LDA #$01
STA $01
ADC $12

LDA #$03
STA $02
ADC $23
LDA #$05
STA $03
ADC $34
LDA #$ff
STA $03
ADC $34
```

Appendix 6

```
A=$c2 X=$00 Y=$00

NV-BDIZC
10110000

0000: 00 01 03 c2 00 00 00 00 00 00 00 00 00 00 00 00
0010: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00a0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
00b0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00c0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00d0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00e0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00f0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
...
```

Appendix 7

```
LDA #$01
TAX
LDA #$ff
TAX
LDA #$50
TAX
LDA #$30
TAX
LDA #$21
TAX
```

Appendix 8

```
A=$21 X=$21 Y=$00
NV-BDIZC
00110000

All mem = $00
```

Appendix 9

```
DA #$13
STA #$00
CMP #$13
```

Appendix 10

```
A=$13 X=$00 Y=$00
NV-BDIZC
00110011

0000: 13 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0010: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00a0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00b0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00c0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00d0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00e0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00f0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Appendix 11

```
  LDA #$02
  STA $01
add:
  ADC $01
  CMP #$06
  BNE add
  LDA #$32
  BRK
```

Appendix 12

```
A=$32 X=$00 Y=$00
NV-BDIZC
```

```
00110001

0000: 00 02 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0010: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00a0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00b0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00c0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00d0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00e0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00f0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Revision History

Revision	Date	Author(s)	Description
1.0	Nov 27th, 2015	Sam Nalwa	created