# Design Document for Knaves NES

Niko Savas 1300247

Joe Crozier 1311502

Sam Nalwa 1332792

November 6, 2015

Software Engineering 3XA3, Lab #3

# Contents

# 1 Introduction

KnavesNES is a software project which aims to emulate an MOS Technology 6502 CPU, as found in the popular Nintendo Entertainment System. The emulated CPU is able to perform all operations performed by the physical CPU and reads and manipulates from its own space of virtual memory. KnavesNES is programmed entirely in C++ using SDL and Visual Studio 2015. The choice to use C++ was that of optimization, as C++ is an exceptionally fast systems programming language. KnavesNES employs several object oriented programming principles, namely inheritance, modularity, and single class responsibility.

This document aims to explain and deconstruct the various modules used by KnavesNES. While SRS was used to identify what was to be done by the software, the design document will instead explain how the requirements be accomplished.

Included in this document is a module hierarchy of the behavior hiding process, a traceability matrix between modules and requirements, and a definition and explanation of each module.

# 2 Anticipated and Unlikely Changes

## 2.1 Anticipated Changes

**AC1** - The specific hardware on which KnavesNES is running.

**AC2** - The length and content of the passed ROM file.

**AC3** - The intermittent and final memory states.

## 2.2 Unlikely Changes

**UC1** - The passed ROM files will be encoded as machine code instructions.

**UC2** - The instruction set of the CPU will be identical to that of the 6502.

**UC3** - Memory will be outputted as ASCII encoded .log files.

**UC4** - The size of the emulated memory will be that of the NES

**UC5** - KnavesNES will be manipulated solely through the command line.

# 3 Module Heirarchy

Because the Knaves NES emulator is simply emulating the CPU of an NES, it doesn't require significant module depth. In the larger scope of an NES emulator, the modules included in the CPU emulator would be lower level modules to the general emulator. For this reason, Knaves NES heirarchy is relatively flat, including only one Level 1 module.

**M1:** Command Line Interfacing Module

**M2:** CPU Module

**M3:** Cartridge Module

**M4:** Memory Module

| Level 1 | Level 2 |
|---|---|
| Command Line Interfacing Module | CPU Module |
| - | Cartridge Module |
| - | Memory Module |

Table 1: Module Heirarchy for Knaves NES

# 4 Connection Between Requirements and Design

| **AC** | **Modules** |
|---|---|
| AC1 | M1, M2 |
| AC2 | M2, M3 |
| AC3 | M2, M4 |

Table 2: Connections between Requirements and Design for Knaves NES

# 5 Module Decomposition

## 5.1 Command Line Interfacing Module

**Secrets:** The instances of the other modules, the CPU loop.

**Services:** Controls the running of the Knaves NES. This is the entry point from the command line which will interface with the other modules in Knaves NES.

**Implemented By:** Knaves NES

## 5.2 CPU Module

**Secrets:** CPU Registers getters/setters/values, Instruction Functions **Services:** Controls the execution of instructions. Reads an instruction from memory using the program counter register

**Implemented By:** Knaves NES

## 5.3 Cartridge Module

**Secrets:** -

**Services:** Reads instructions from a ROM file, converts them to chars, then writes them to the necessary locations in memory.

**Implemented By:** Knaves NES

## 5.4 Memory Module

**Secrets:** Memory values

**Services:** Reads and writes values to and from memory.

**Implemented By:** Knaves NES

# 6 Traceability Matrix

| Label | Description |
|---|---|
| R1 | Knaves NES must be able to run standard NES ROMs (.nes files) |
| R2 | Knaves NES CPU must be able to execute opcodes from the 6502 MOS CPU architecture |
| R3 | Knaves NES CPU must be able to manage 2kB of RAM |
| R4 | Knaves NES will log its memory state into a .log file in the users working directory at a specified interval. |

Table 3: Original Requirements for Knaves NES (Included in Requirements document, inserted here for reference.)

| Req. | Modules |
|---|---|
| R1 | M1, M2, M3, M4 |
| R2 | M2, M4 |
| R3 | M2, M4 |
| R4 | M4 |

Table 4: Traceability Matrix for Knaves NES

# 7 Use Heirarchy Between Modules

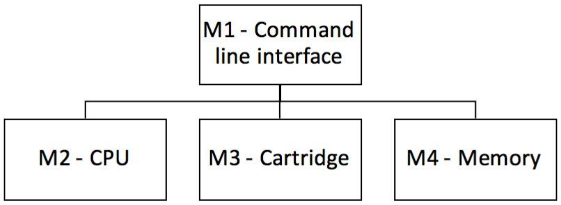
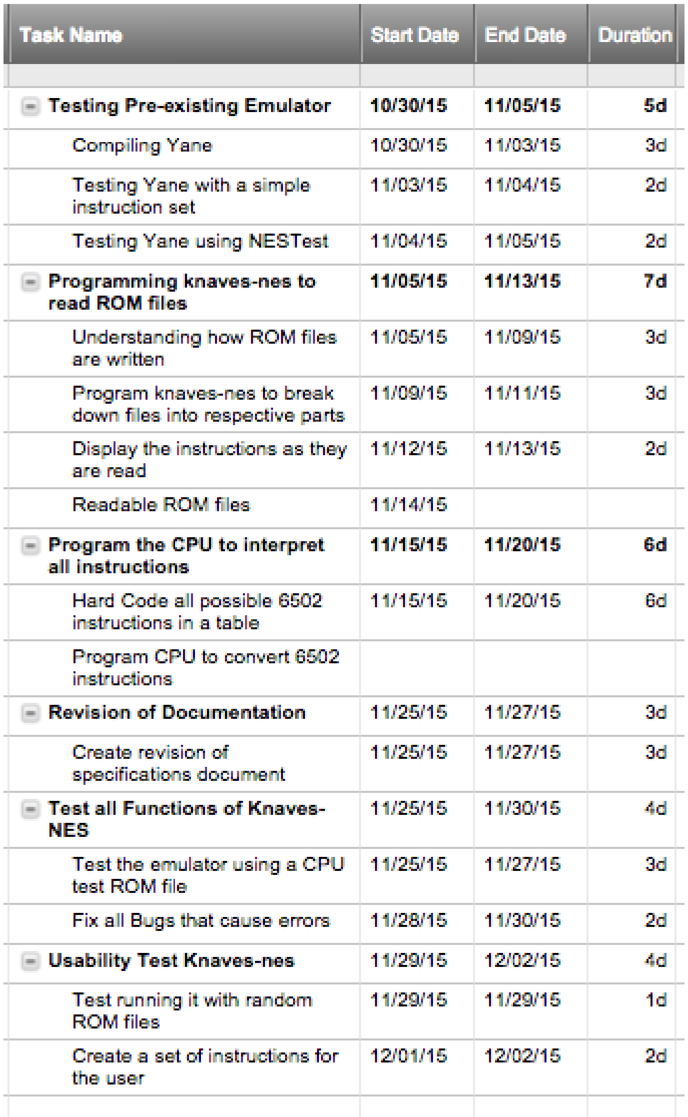


Figure 1: Use Heirarchy Among Modules

# 8 Project Timeline



| Task Name | Start Date | End Date | Duration |
|---|---|---|---|
| **Testing Pre-existing Emulator** | **10/30/15** | **11/05/15** | **5d** |
| Compiling Yane | 10/30/15 | 11/03/15 | 3d |
| Testing Yane with a simple instruction set | 11/03/15 | 11/04/15 | 2d |
| Testing Yane using NESTest | 11/04/15 | 11/05/15 | 2d |
| **Programming knaves-nes to read ROM files** | **11/05/15** | **11/13/15** | **7d** |
| Understanding how ROM files are written | 11/05/15 | 11/09/15 | 3d |
| Program knaves-nes to break down files into respective parts | 11/09/15 | 11/11/15 | 3d |
| Display the instructions as they are read | 11/12/15 | 11/13/15 | 2d |
| Readable ROM files | 11/14/15 | | |
| **Program the CPU to interpret all instructions** | **11/15/15** | **11/20/15** | **6d** |
| Hard Code all possible 6502 instructions in a table | 11/15/15 | 11/20/15 | 6d |
| Program CPU to convert 6502 instructions | | | |
| **Revision of Documentation** | 11/25/15 | 11/27/15 | 3d |
| Create revision of specifications document | 11/25/15 | 11/27/15 | 3d |
| **Test all Functions of Knaves-NES** | 11/25/15 | 11/30/15 | 4d |
| Test the emulator using a CPU test ROM file | 11/25/15 | 11/27/15 | 3d |
| Fix all Bugs that cause errors | 11/28/15 | 11/30/15 | 2d |
| **Usability Test Knaves-nes** | 11/29/15 | 12/02/15 | 4d |
| Test running it with random ROM files | 11/29/15 | 11/29/15 | 1d |
| Create a set of instructions for the user | 12/01/15 | 12/02/15 | 2d |

Figure 2: Timeline for Knaves NES

# 9 Pert Chart

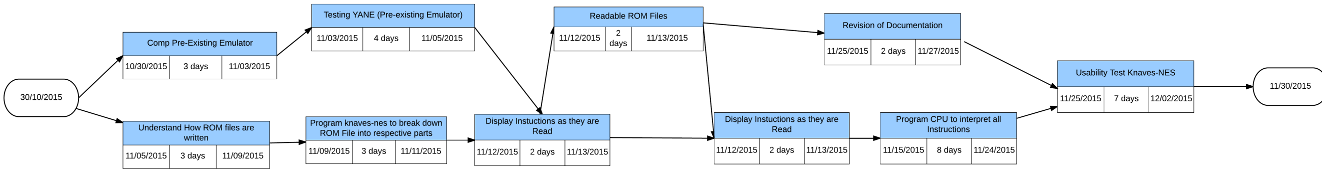

30/10/2015
Comp Pre-Existing Emulator
10/30/2015
3 days
11/03/2015
Testing YANE (Pre-existing Emulator)
11/03/2015
4 days
11/05/2015
Understand How ROM files are written
11/05/2015
3 days
11/09/2015
Program knaves-nes to break down ROM File into respective parts
11/09/2015
3 days
11/11/2015
Display Instuctions as they are Read
11/12/2015
2 days
11/13/2015
Readable ROM Files
11/12/2015
2 days
11/13/2015
Display Instuctions as they are Read
11/12/2015
2 days
11/13/2015
Revision of Documentation
11/25/2015
2 days
11/27/2015
Program CPU to interpret all Instructions
11/15/2015
8 days
11/24/2015
Usability Test Knaves-NES
11/25/2015
7 days
12/02/2015
11/30/2015

Figure 3: Pert Chart for Knaves NES (pert.png in this folder)

# 10 Gantt Chart

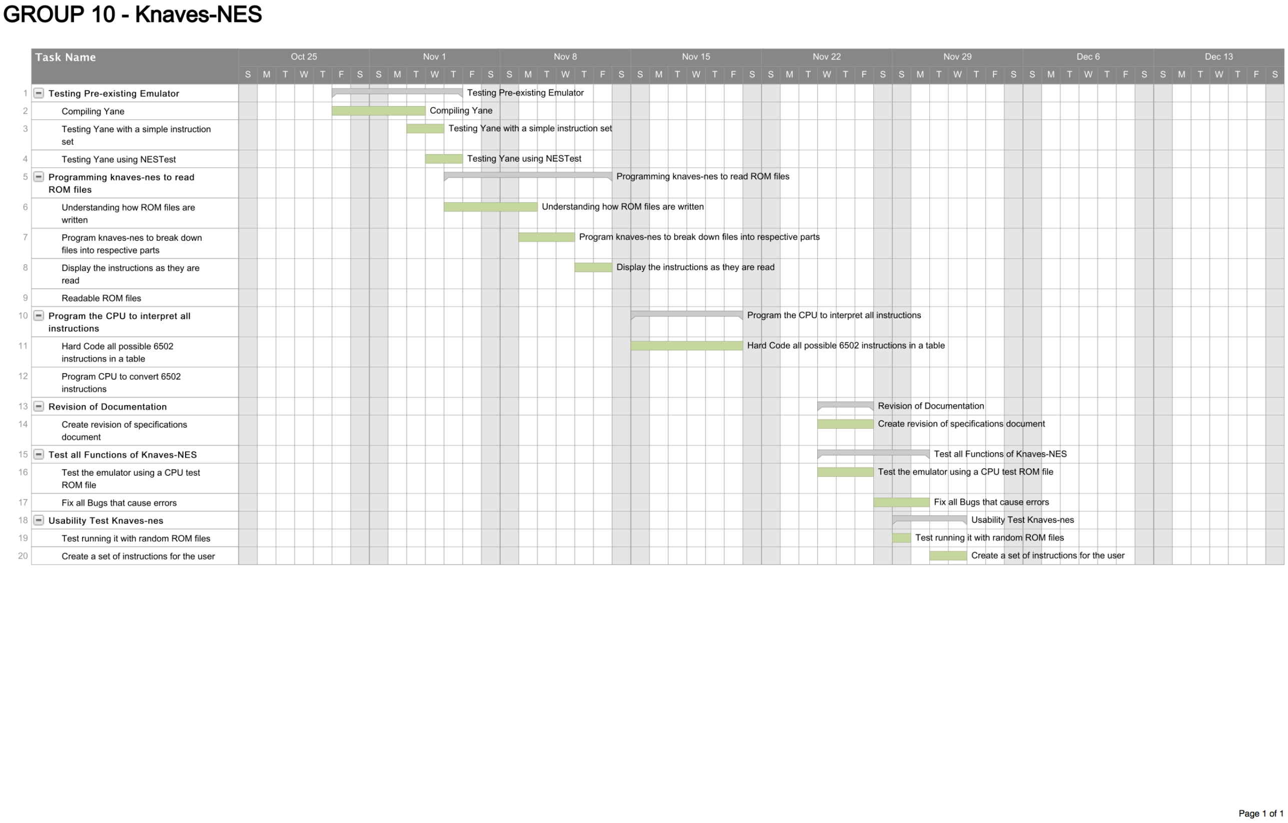


Figure 4: Gantt Chart for Knaves NES (gantt.png in this folder)

# Revision History

| Revision | Date | Author(s) | Description |
|---|---|---|---|
| 1.0 | Nov 5, 2015 | Niko Savas | created |