ASCII Dungeon Run

Custom Project Final Report

Winter 2018

Edward Zabrensky

Table of Contents

[**Introduction**](#_n9o44neiquk5) **2**

[**Hardware**](#_vqmbsuo5p330) **3**

[Parts List](#_xtqh2tzibtrr) 3

[Pinout](#_8yptvhjzojq8) 3

[**Software**](#_w9543qe124on) **3**

[**Complexities**](#_sn48u4uktu3c) **13**

[Completed Complexities:](#_tp9dnsse4u9p) 13

[Incomplete complexities:](#_99jbzhpy2eks) 13

[**Youtube Link**](#_u9cxbem9510r) **13**

[**Known Bugs and Shortcomings**](#_grlzpb6vy2cq) **13**

[**Future work**](#_qbv6f31drpex) **13**

**List of Files 14**

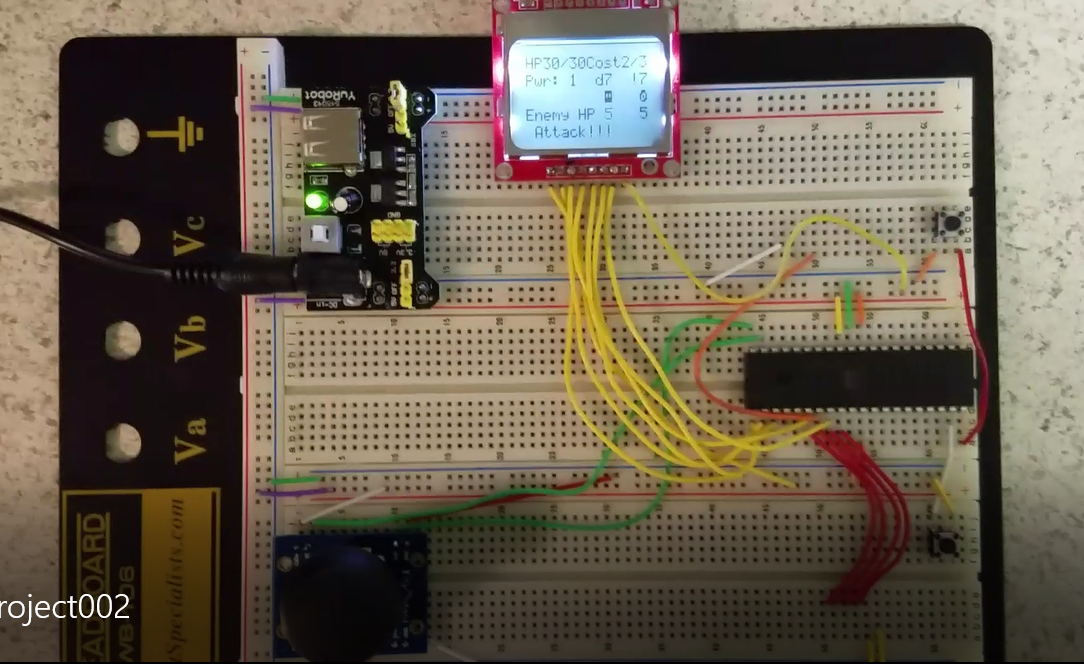
# Introduction

The goal of the game is to defeat as many bosses as possible.. Each boss has different health points and attack power. A player defeats a boss when the boss reaches zero health points. The player begins with a collection of 12cards, consisting of 5 attack, 5 defense, and 2 powerup cards (a, d, p). The player and the boss alternate turns to attack each other.

A player may play a attack card to attack the boss. This brings the bosses health down a certain amount of health points depending on how much damage the attack card does. A player may use defense cards to block damage that the boss will deal to them that turn. A player may also use power up cards to boost the amount that their attack cards do.

Each card has a cost of 1. A player can only play up to 3 cost per turn. For example, if the play has a hand of “a(1), d(2), p(3), a(4), p(2)”, the player could only play the a combination of 3 cards in the hand like a(1), d(2), p(2).

Every time a card is used it is moved to the discard pile, when the player runs out of cards the discard pile becomes the draw pile.



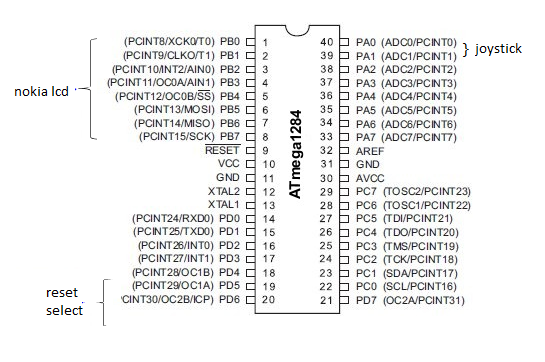
# Hardware

## Parts List

The hardware that was used in this design is listed below. The equipment that was not taught in this course has been bolded.

* ATMega1284p microcontroller
* Buttons
* **Joystick**
* **Nokia 5110 LCD**

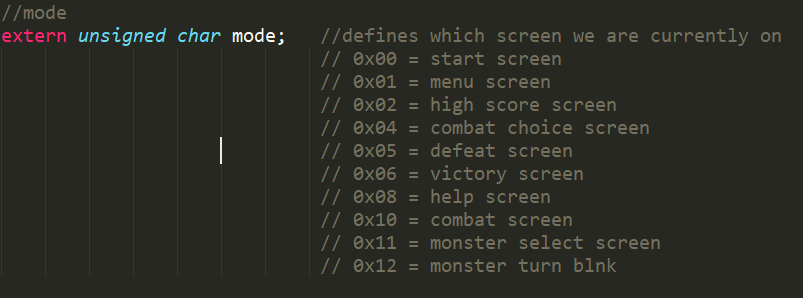
## Pinout



# Software

High level state machine Part 1:

For combat - the control will make a blinking bool 1. Then the combat FSM will start counting its blink counter. When the control reads that the counter reaches a certain value it will update the HP, cost, power, enemy, or enemyHP appropriately.



HLSM Part2: 

# 



# 





There are wait States between stateslike start-menu, menu-combatchoice, menu-help, .. etc. To take care of long button presses, but not included to save space.

# Control Part 2







# Complexities

## Completed Complexities:

* Integrating and calibrating the joystick
* Using EEPROM to save the high score, HP, score, deck
* Creating custom characters for bosses on the LCD screen
* Using nokia 5110 LCD to display game

# Youtube Link

<https://youtu.be/bRe-RUD5FYA>

# Known Bugs and Shortcomings

* A lot of the SMs with 2 states could be generalized to one state machine that indexes an array of the desired output
* The menu and combat choice SM could be reduced to 2 states like the victory screen
* The joystick could be calibrated better sometimes you have to go all the way right or all the way left to register a left or right movement

# Future work

I am going to generalize a lot of these SM with 2 states to one SM that indexes what to output with mode. I am going to make the combat choice and victory screen 2 states and include it in the general 2 state output synchsm. I am also adding a feature to view the current deck that the player has. I am also going to calibrate the joystick more appropriately so that the joystick does not have to move all the way right or all the way left to register the direction.

List of Files:

Nokia Screen library: <https://github.com/LittleBuster/avr-nokia5110>

ADC\_read function & timer.h - from lab

[Combat.h](https://drive.google.com/open?id=1BEJSU3EDBWczruNcWzafgeRQWMS4_5SD) - controls the filling of the combat buffer, used when combat is active

[combatChoice.h](https://drive.google.com/open?id=1jGklrHTxQcxNRCb6NDkPCYPGN52UISd6) - controls combat choice screen + animation of arrow

[Common.h](https://drive.google.com/open?id=1S6K1EIvOPSDVrG4DvjyDGEhKRK7e7xfh) - common variables between SMs

[Defeat.h](https://drive.google.com/open?id=1y2eTsTHcJo-DpqFB3cyHpxizFz0aMwu0) - displays defeat screen also displays new high score if a high score is amde

[helpScreen.h](https://drive.google.com/open?id=1tmkpRtjzKFmtTK-z1e9fD2xR7WAjh-cG) - displays help Screen + help screen animation scrolling

[highScore.h](https://drive.google.com/open?id=16K-gKVJYA09mULNZoC8UPcYR32Of8oBK) - fiulls high score depending on high score

[Main.c](https://drive.google.com/open?id=1wYGuDt1Ak-phALxy-e3dUS6Etw3wrN4U) - runs task struct with all the tasks

[menuScreen.h](https://drive.google.com/open?id=1ibehdHKkZUrO5IvE3oHwdVVTYjCT6DRR) - fills menuScreen buffer + animations

[Nokia5110.h](https://drive.google.com/open?id=1KW4Ni_VKjgSOYXipdWJG9j3LpH4CxrNh) - original library + modified variables of where the nokia\_lcd goes

[Nokia5110.c](https://drive.google.com/open?id=19iGF-O73nmw_i7j9A5xZui96VV5Yc08Q) - original library + modified functions for my code

[Nokia5110\_chars.h](https://drive.google.com/open?id=1GqVvyaddaMFmW5MjXDCrzItMcf28m82l) -contains custom characters of bosses and other ascii chars

[Outputlcd](https://drive.google.com/open?id=1ruiFY1z2Mmz7DV8nnydYmjgJaJH07z7b) - outputs the chosen lcd buffer depending on mode variable

[selectControl.h](https://drive.google.com/open?id=1zfxh2TKGrG3U2uogB0eXL1QKJmH4_GsB) - control that modifies mode and choice so certain only certain buffers are filled

[startScreen.h](https://drive.google.com/open?id=1qiTeQmjiu7Hzk_fTgcXEnkcJhLNSjT4L) - fills start screen display buffer w/ animation

[Timer.h](https://drive.google.com/open?id=1SDy-IIWr0bmSOGyzuqy6auCfP5zzgNGV) -timer file from lab

[Victory.h](https://drive.google.com/open?id=105YHfA61usJMF_pG-iaLxCHfw-29FzMQ) - fills victory display buffer w/animation