Progromming the Nintendo Entertoinment System

Peter J Hopkins
Major: CIT
Concentration: Web Design
Year: Senior
CIS 386

Overview of NES

- Released in Japan in 1983 and the U.S. in 1985
- One of the most popular video game systems of the late 1980's and early 1990's
- Popular series:
  - Super Mario Bros.
  - The Legend of Zelda
  - Metroid
- Great way to start learning assembly programming!

Technical Specs.

- 8 bit processor—MOS 6502 family & PPU
- 1.79 MHz
- 16 bit address bus
- 2Kb of onboard work Ram & 2Kb graphics
- Resolution: 256 x 240 @ 60 fps
- 64 colors—25 on screen at once
- 64 sprites on screen at once—max 8 per line
- Sound: 2 square, 1 triangle, white noise, PCM

NES Assembly

- Assembly is faster than C
- 55 opcodes
- Only addition, subtraction, and bit shifting
- Accumulator, X & Y, status register

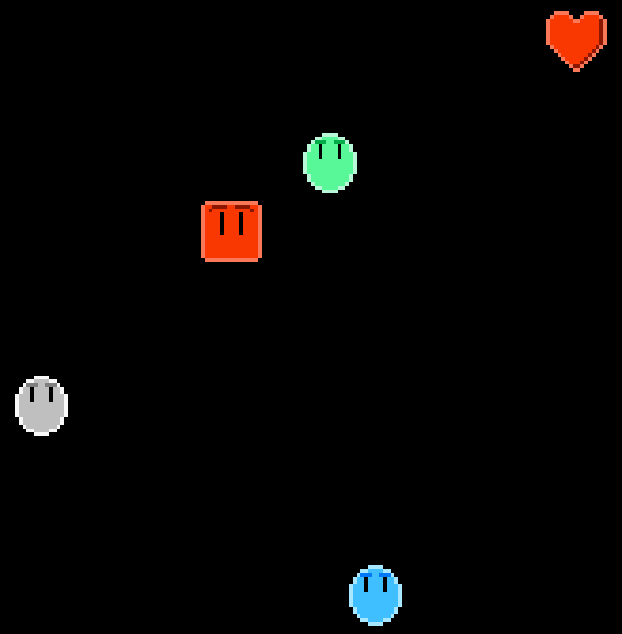
```
249 Enemy2Controller:       ;enemy 2 chases the nearest player
250     LDA enemy2Timer
251     AND #%00111111
252     BEQ Enemy2ControllerContinue    ;the following lines are necessary for the skip becau
253     JMP Enemy2ControllerDone
254 Enemy2ControllerContinue:
255     LDA #$00                ;reset the enemy2 controller to prepare it for a new directio
256     STA enemy2Cont
257     LDX #$00                ;prepare to loop through both players to find distance
258     FindPlayerDistance:     ;find the vertical and horizontal distance to each player
259         LDA enemy2Ver
260         SEC
261         SBC player1Ver, X
262         STA player1VDistance, X
263         BCS FindPlayerHorizontal    ;if the player's position is higher than the enemy's
```

V-Blank and NMI Interrupt

- v-blank: the time between picture frames
- PPU updates must be made during v-blank
- NMI interrupt occurs at v-blank
- Main program loop and NMI must be in sync!

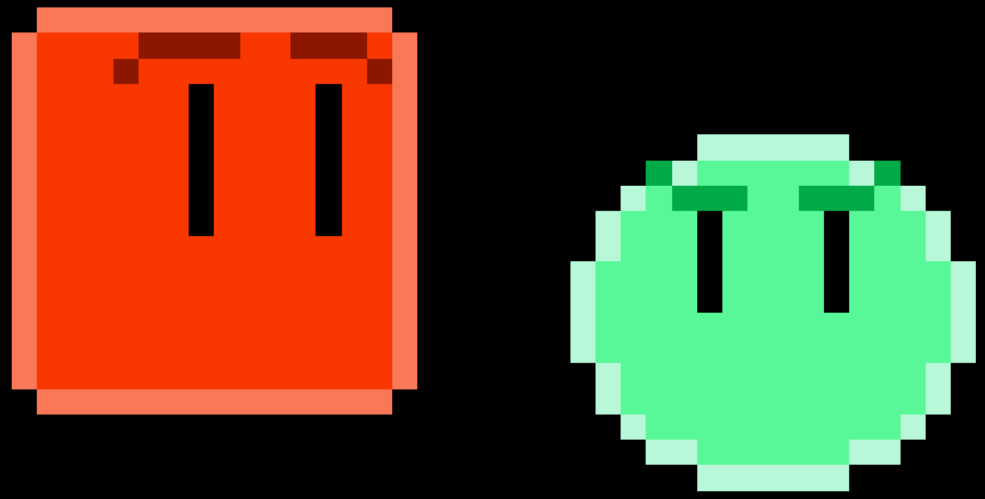
My Gome

- Developed organically
- Inspired by Pac-Man
- Basic ‘avoid enemies, collect items’ game



Collision Detection

- Grid based collision
  - Easy to program: BEQ & BNE
- Hitbox based collision
  - More accurate: BCC & BCS



Rondom Numbers

- No computer can generate truly random Numbers!
- Pseudo-random number sequences
- Loop through first 256 memory addresses
  - Add them together
  - Store in a new address
  - New random number every frame

Enemy AI

- 3 different enemy behavior ‘personalities’
- Chasing behavior is complicated
  - Pick nearest player
  - Follow that player
  - Simple distance formula
  - BCC & BCS for determining direction of movement

Future Work

- Finish graphics
- Design background maze
- Wall collision detection
- Sound effects and music
- Item collection and scoring

Development Tools

- Notepad++
- NESASM3
- NESten emulator for testing code
- Tile Layer Pro for graphics design
- EverDrive-N8 flash cart for testing on real NES hardware