KWALA ELECTRONICS

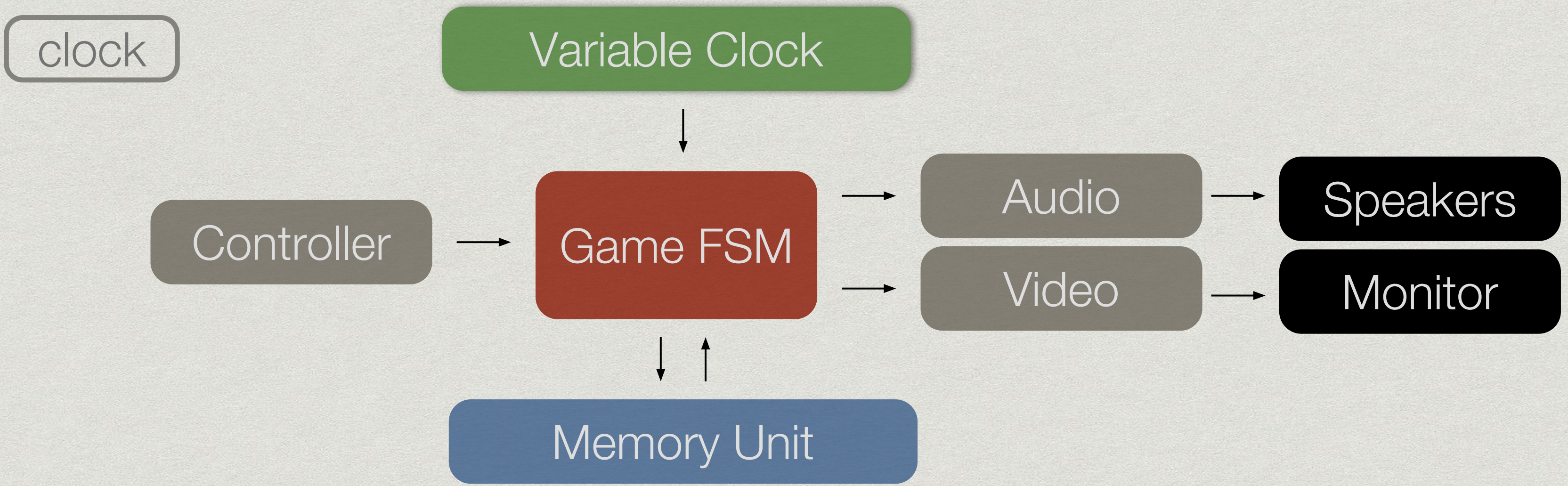
KENNETH COLLINS & WALAA ALKHANAIZI

Tiger Electronics

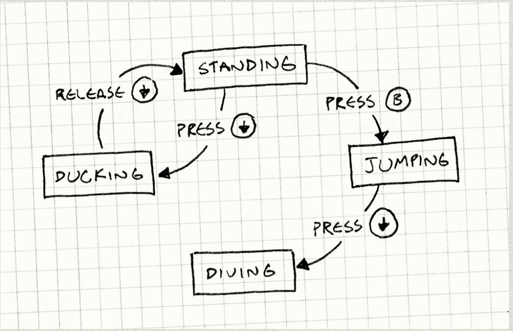
Hardware

- Nexys 4 FPGA
- External Memory (USB)
- USB Classic SNES Controller (Maybe)
- Monitor
- Speakers

Block Diagram

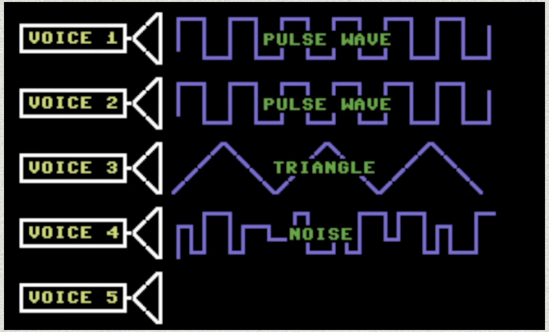


The Games: FSMs



- All games are finite state machines
- Explore whether we can use a generic FSM to represent/implement multiple games at once
- Receives input from controller and variable clock, and sends output to video, audio, and memory units
- **Challenges**:
    - Detecting collisions

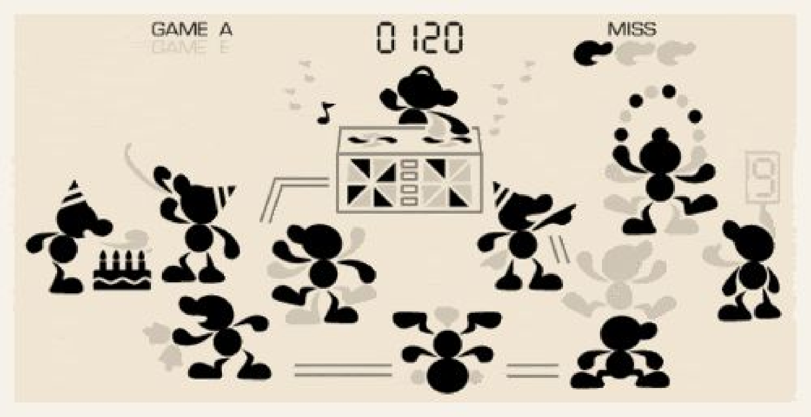
Audio Control



- Store music in memory.
- Music playback syncs with variable game clock
- Maybe add sound effects for dodging, or getting hit
- **Challenges**:
  - Choose between flash memory, RAM, ROM
  - How will it be stored in memory
  - Music generation

Video Unit

- Responsible for generating the graphics of the characters, projectiles, obstacles
- Everything has predefined potential positions on the screen
- Background image is fixed
- **Challenge(s):**
  - Best way to create/store the background?
  - Storing/drawing the sprites at each position?
  - Randomization?



Game Controller

- Use buttons on FPGA to represent the user input

- If everything works out, try using the USB Nintendo NES controller and serially read the controller input.

- **Challenges**:
  - Properly reading the usb data from the controller

Memory Unit
(for highscores)

- Enable high-score saving with a name option
- Use a flashdrive to save the data to so that it is not wiped by the FPGA at reset
- If all works out, maybe add a progress saving procedure (basically just needs to load score and variable clk speed)
- **Challenges:**
    - Storing this data directly on the FPGA
    - Recording capability?

Accelerating Clock

- As the game progresses, the difficulty is increased by speeding up the clock
- This can affect the scrolling speed and enemy/object generation
- **Challenges:**
  - Is the player’s input clocked by the original clock or the variable clock?

Timeframe

11/5 - 11/11

- Implement variable clock, and start testing
- Choose a first game to implement, start implementing?

11/12 - 11/18

- Implement controller unit
- Implementation of specific game FSM
- Start incorporating graphics, and testing

11/19 - 11/25

- Audio integration
    - Ideally get one game running with video, audio, and controls.
- Start memory unit

11/26 - 12/2

- More work on memory- saving scores
- Investigate a general FSM module
- Implement a few different games
- Debugging

12/3 - 12/9

- Debugging
- If going well, try using the USB Nintendo NES controller

Questions?