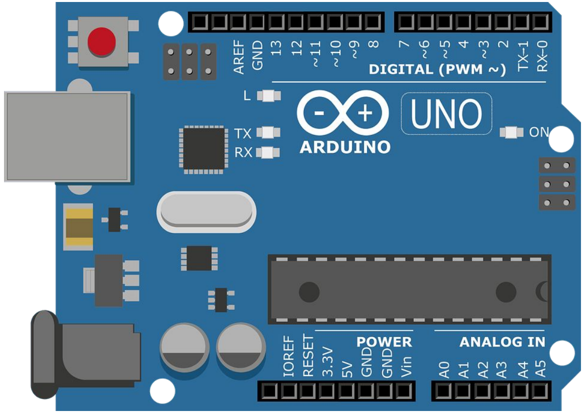
UAEC Arduino Pong

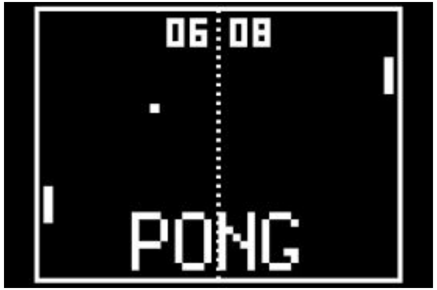
Overview

- Review hardware used
- Review software used
- Show working demo of game
- Explain Game making process
- Application of process to pong

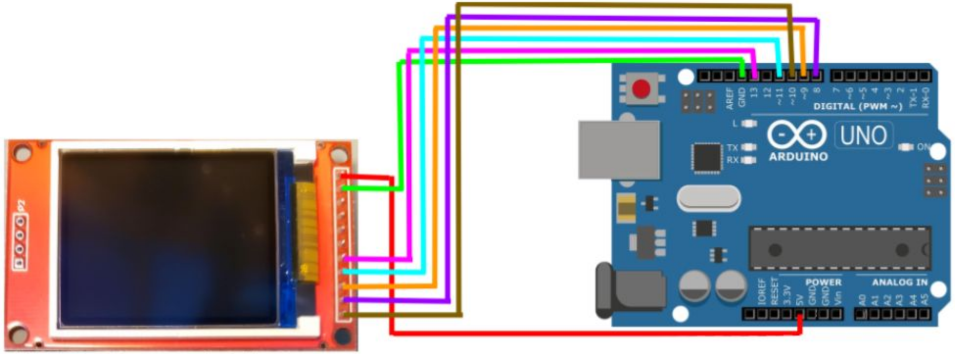
The Arduino Platform

- This week will be taking what we know and recreating a popular video game we all know!
- We will being using the OLED display, Joystick and UNO device to recreate….



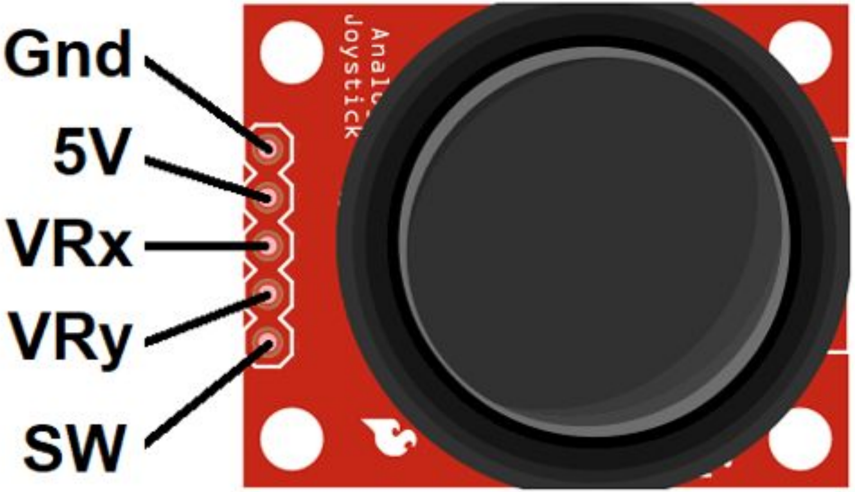


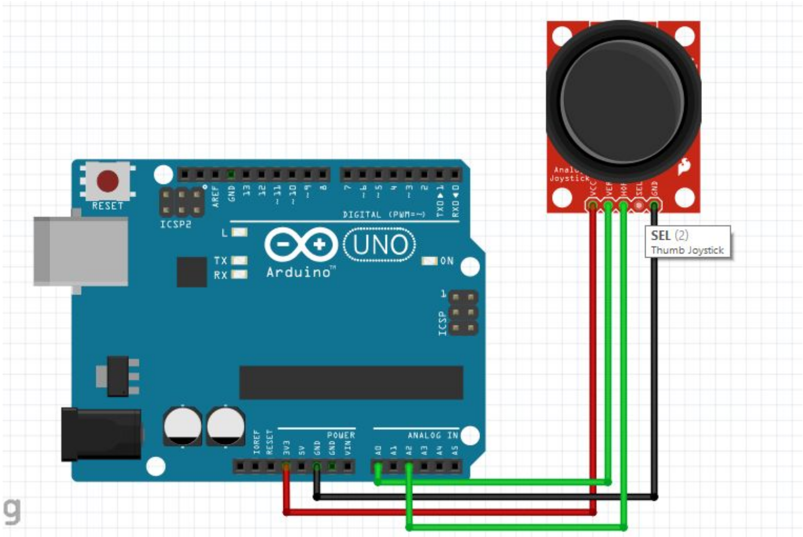
Hardware - TFT LCD



TFT LCD		Arduino
VCC	--	5V
GND	--	GND
CLK/SCK	--	13
SDA/MOSI	--	11
RS/DC	--	9
RST	--	8
CS	--	10

Hardware - Joystick



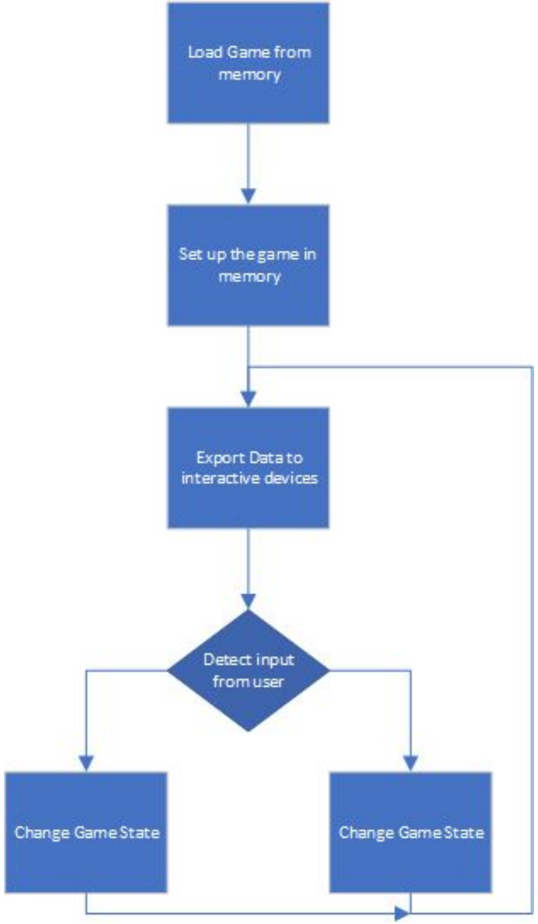


Demo of the game

Game Making Process

Game Making Process

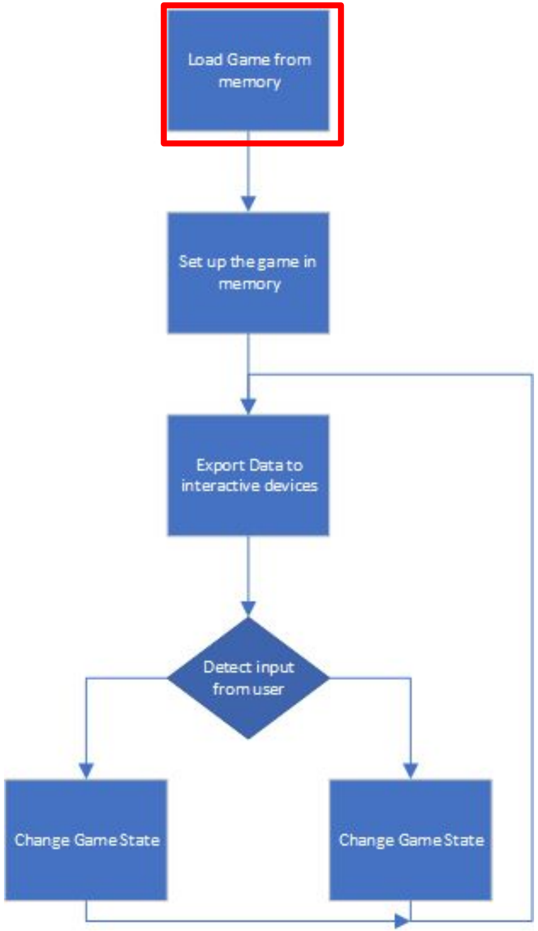
1. Load the game from memory
2. Set up and prepare the game in memory
3. Export data to human interactive devices/sensors
4. Detect input from user
   a. Input has been detected
      i. Change Game State
   b. Input has not been detected
      i. Change Game State
5. Go to 3



How does this process was used to create Arduino Pong

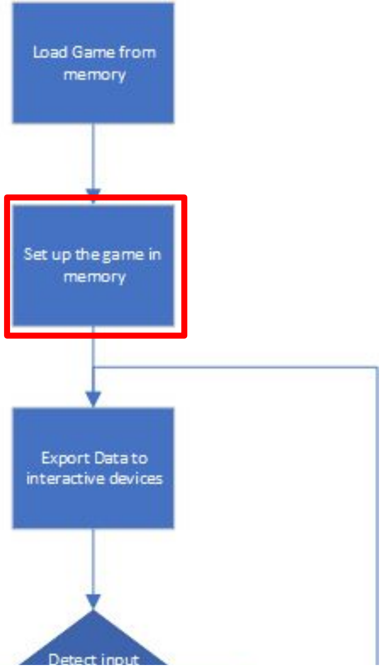
Load the game into memory

- Lines 1-47 all set up various variables
- Major things to set up
    - Pins numbers for the the screen
    - Pins numbers for the Joystick
    - Create variables for the ball
    - Create variables for the paddle
    - Define colors
    - Define physic constants
    - Define spawn areas



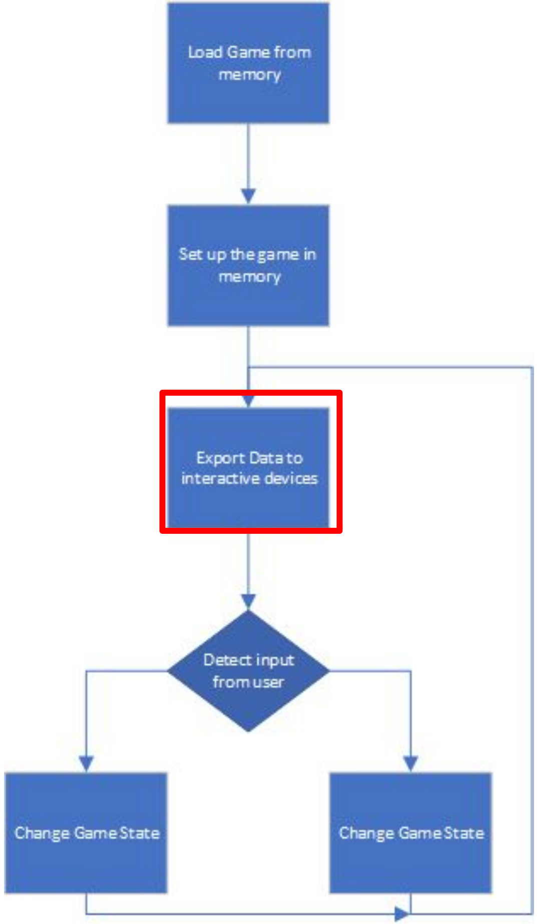
Set up and Prepare the Game in Memory

- Lines 49-93 all set up various devices
- Major things to set up
  - Initialize the screen
  - Initialize the Joystick
  - Spawn the ball
  - Span the paddles
  - Define spawn areas



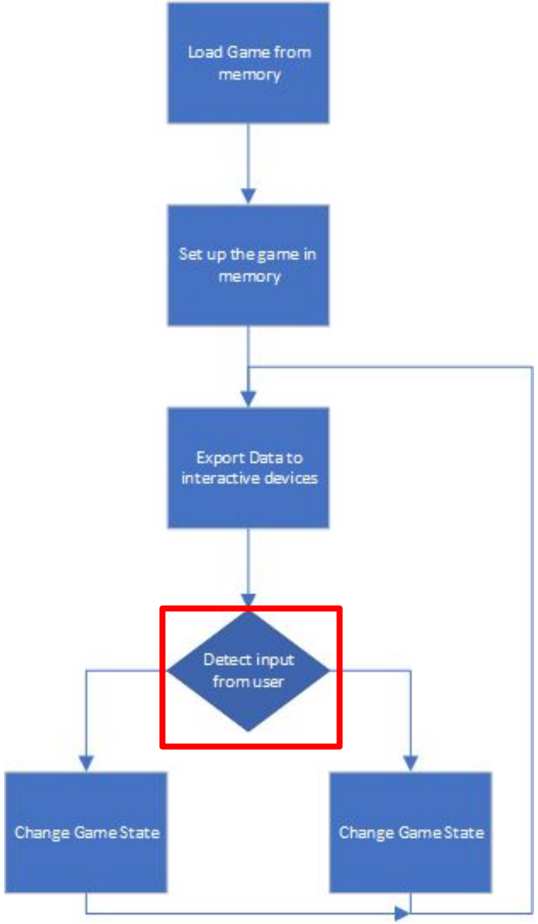
Export Data to interactive devices

- Draw splash screen
- Draw ball
- Draw paddles



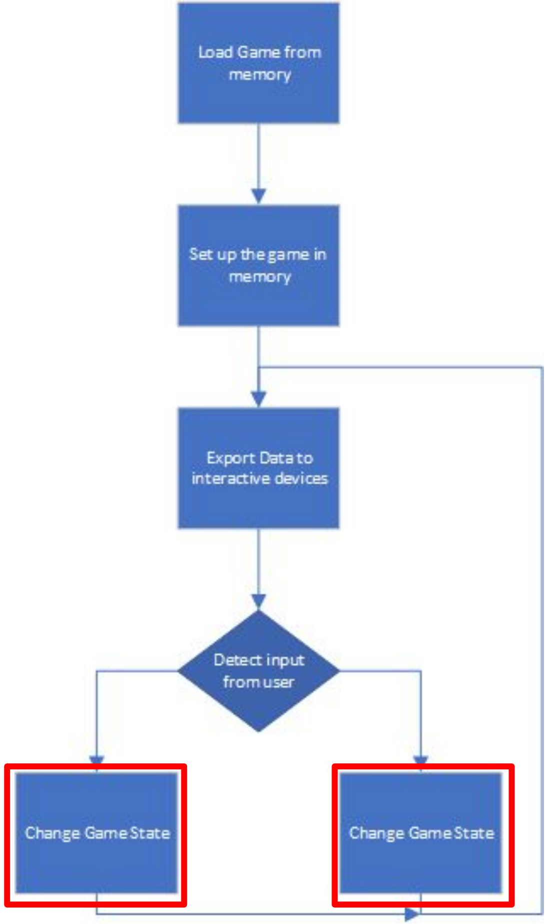
Detect user input

- Read analog pins to determine paddle location



Change the Game State

- Calculate if the user moved the paddle to hit the ball
    - If true, calculate angle of incidence and update ball variable
    - If false, score++ for the person who scored



Tips for Making Games

Tips for Making Games

- Make code modular
  - Allows you to reuse code easily
- Add frequent if statements to bypass large blocks of code
  - Keeps game refresh rate high
- Keep the game simple
  - Arduino is limited on cpu and memory
- Limit portions of the screen that need to be redrawn each frame
  - Update small portions of the screen rather than blacking out the screen and redrawing everthing

Where to find code example code

- UAEC Arduino pong can be found on github
    - https://github.com/asaunier555/UAEC_Arduino_Pong