Circle Pong

A CS2710 Final Project

Scott MacIntosh
Allan Stewart
Dec 16th 2017

Gameplay Description

Much like the classic game pong, the player hits a moving ball with a paddle to prevent it from leaving the screen. However, this game has one player, and the paddle can wrap around the entire square borders. It's a one player endurance game, where you get a point every time you make a save and keep the ball on the screen. A detailed demo of the game can be viewed here, and the full documentation and source code is available on the GitHub repository.

Design & Implementation

Components:

- A start/reset button is available to begin the game, restart the game when the game has finished, or start a new game while the game is still running.
- A sound mode button changes between different sound volumes and modes. The sound modes that are cycled through are:
  - Normal (All sounds at normal volume.)
  - Low Volume (All sounds, reduced volume.)
  - No Music (All sounds except waiting music, at reduced volume.)
  - Only Classic Tones (Only pong-like tones from passive buzzer.)
  - No Sounds
- A passive buzzer and an MP3 player play sound effects. The MP3 player plays longer sound effects when time delays are not a problem, due to the long (in ms) delays inherent to the serial mp3 interface, while the passive buzzer plays short tones based on the original Pong game for sound effects that require as minimal of a delay as possible, such as when the ball hits the paddle or is missed.
- A joystick is used to control the paddle's position, setting the paddle position to the joystick's angle. Whenever the joystick is in a neutral position (determined by a square margin on both the x & y axis), the most recent angle it formed is used. As well, the joystick switch (push-button) can restart the game, but only when a game is not currently active.
- An LCD display tells the player their highscore and their current score while playing, or prompts them to press the start button to begin when they're not.
- The 8x8 LED grid displays the current state of the game. While the game is being played, it displays the paddle's position and the ball's current position. While there is no active game, the paddle still functions, but the ball moves randomly around the screen.

Arduinos:

This implementation uses two Arduino Megas. One Arduino runs the actual game, while the other focuses on the comparatively simple yet all consuming task of managing the 8x8 LED display.

By implementing the display on a second Arduino, a more consistent lighting can be achieved, as displaying a paddle and ball requires rapidly alternating between 3 different configurations; the horizontal paddle component, the vertical paddle component, and the ball. Whenever the data pin turns high, an interrupt is triggered and the display Arduino reads in the position of the paddle (expressed as a binary number 0-27, requiring 5 pins) and the x and y coordinates of the ball (each expressed as a binary number 0-8, requiring 3 pins each). If debugging is enabled within game.ino, the serial monitor can emulate the display Arduino.

The game Arduino handles driving the game, receiving input from the user, as well as outputting to the LCD display, the passive buzzer, the MP3 player and the display Arduino. When the game is running, it will move the ball in one of 8 directions. When the ball hits the border, it will check if it's adjacent to the paddle. If so, the ball will move in a randomized direction away from where it hit and the user scores a point. If not the game is lost. If the game is not on one of the quieter settings, and a new high score was set, the game plays a congratulatory wav file via the MP3 player, otherwise the lose wav file is played. The music that plays when no game is running is Tranceincidental by cvessey on SoundCloud.

Arduino setup instructions

Parts Required

1. 2 Arduino Megas
2. 1 8x8 LED Grid
3. 1 16x2 LCD Display
4. 1 Joystick
5. 2 Pushbuttons
6. 1 Passive Buzzer
7. 1 MP3 Player with SD card support

Display Arduino

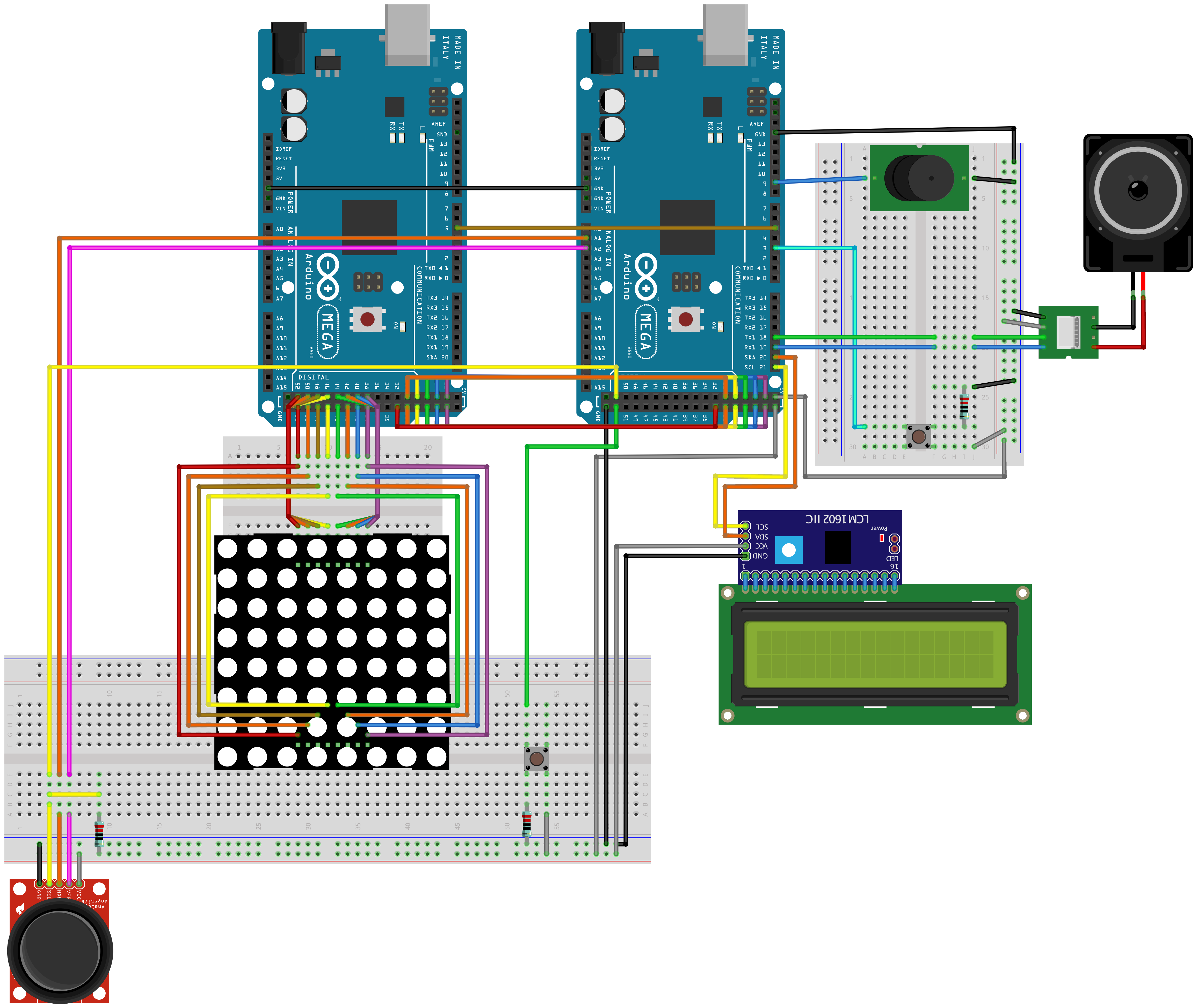
- Pins to game Arduino: 5 (interrupt pin), 22 - 31, 33
- Pins to 8x8: 40 - 53

Game Arduino

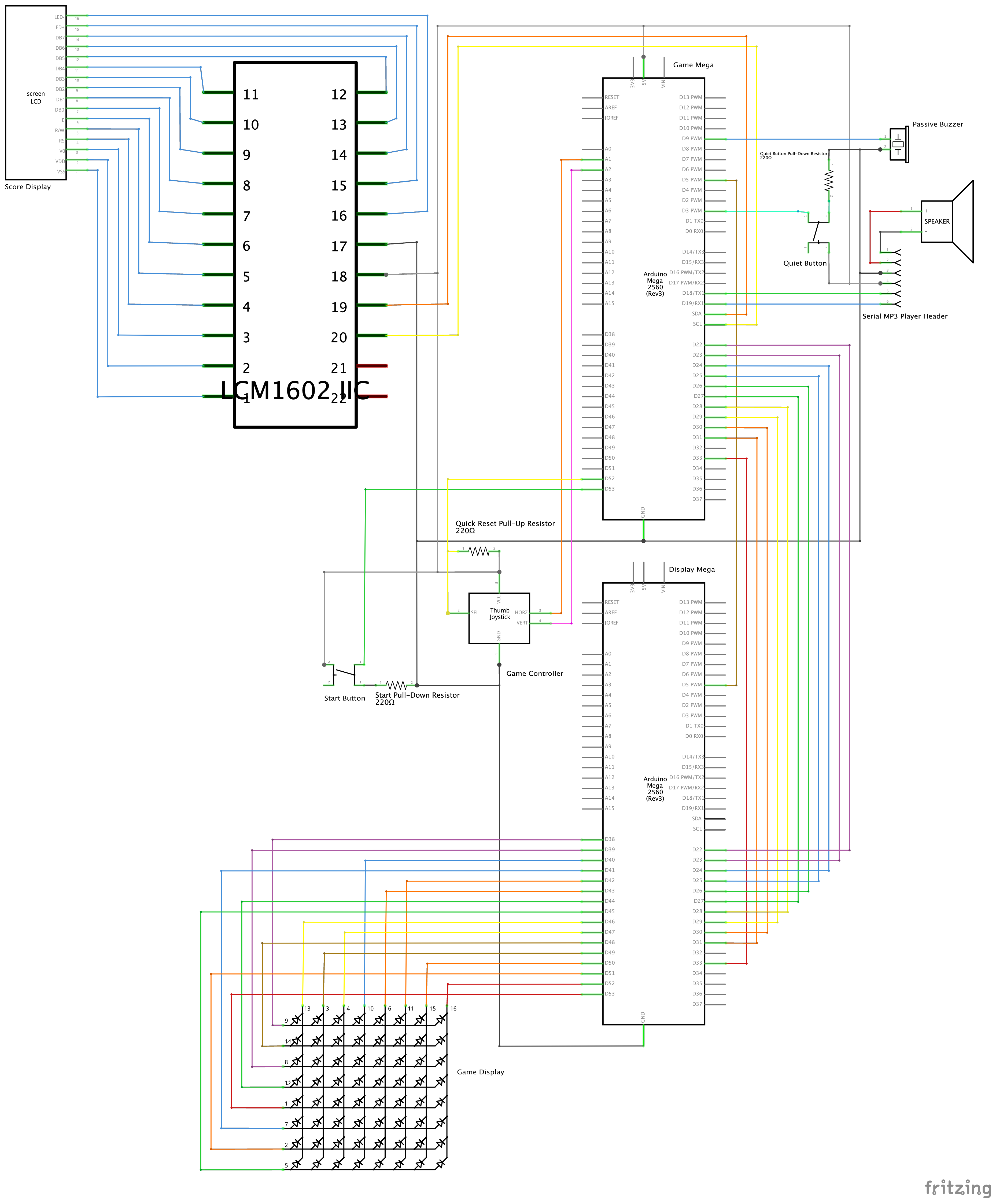
- Pins to display Arduino: 5 (interrupt pin), 22 - 31, 33
- Pins to reset button: 53
- Pins to LCD: 20, 21 (SDA, SLC)
- Pins to joystick: A1, A2, 52
- Pins to MP3 player: 18, 19 (Serial1)
- Pins to Buzzer: 9

Build instructions

- Download the source code from the GitHub repository.
- Copy folders in /audio/ to an SD card to use with mp3 player.
- Wire the components according to the diagrams.
- Compile and send display.ino to the display Arduino.
- Compile and send game.ino to the game Arduino.



Arduino MEGA
DIGITAL
COMMUNICATION
ANALOG IN
POWER
LCM1602 IIC
SCL
SDA
VCC
GND
fritzing


Score Display
screen LCD
LCM1602 IIC
Game Mega
Arduino Mega 2560 (Rev3)
Passive Buzzer
Quiet Button Pull-Down Resistor 220Ω
SPEAKER
Quiet Button
Serial MP3 Player Header
Quick Reset Pull-Up Resistor 220Ω
Display Mega
Arduino Mega 2560 (Rev3)
Thumb Joystick
Game Controller
Start Button
Start Pull-Down Resistor 220Ω
Game Display
fritzing