Light Arcade

Scott James
floridaconsulting@gmail.com
Github.com/scottjames

Game Idea

- Build a simple/fun 2-player game
  Entry for Maker Faire Orlando
- Short/Fast games, quick turn around
  No player lines, easy to observe
- No skill: Just hit buttons fast
- Countdown clock to add pressure to players
  10 seconds per game.

Approach – Hardware Build

- Use new methods and materials
  not PVC and plastic boxes
- Used 4" ECT, MDF custom milled
- Paint with "hammered" for industrial look.
- High impact display - stand out in a crowd
  not just another video game
- Portable (able to take to Maker Faire)

Approach – Electronics Build

- Simple, DIY electronics
- No custom fabricated PCB
- Use Vero Strip-Board
  Easier for people to copy/hack their own.
- High impact display - stand out in a crowd
- Use LED lights, not classic “video” game.
- Portable

Design

- Use Arduino, much as possible
- Raspberry Pi for multi-media display
- Data feed to Pi is *receive only* (simpler)

Implementation

- Platform: CPU board
- Arduino pro mini (ATMega328, no USB)
- Raspberry Pi, HDMI output to monitor
- 3-to-5 volt level shifter
- Data flows from Arduino -> level shift -> Pi

Inputs

- Inputs - buttons
- Always did software debounce.
  This time use hardware debouncing.
- Research: 10 msec debounce period.
- Utility buttons: start game, and ?
- Multiple headers/pins to hook on and test inputs/outputs.

Display

- NeoPixel display LED strips
- Install LED inside buttons for feedback
- VGA screen, Raspberry Pi

Arduino software

- NeoPixel Library (adafruit.com)
- Serial 115200 baud -> Pi data feed.
- Serial text commands, easier debug/reading in terminal.
- Event loop to drive game
- State machine:
  DEMO -> PLAY -> FINISH -> DEMO
- Modular approach, Arduino-friendly, simple - Buttons - Game - Display - Serial

Arduino software

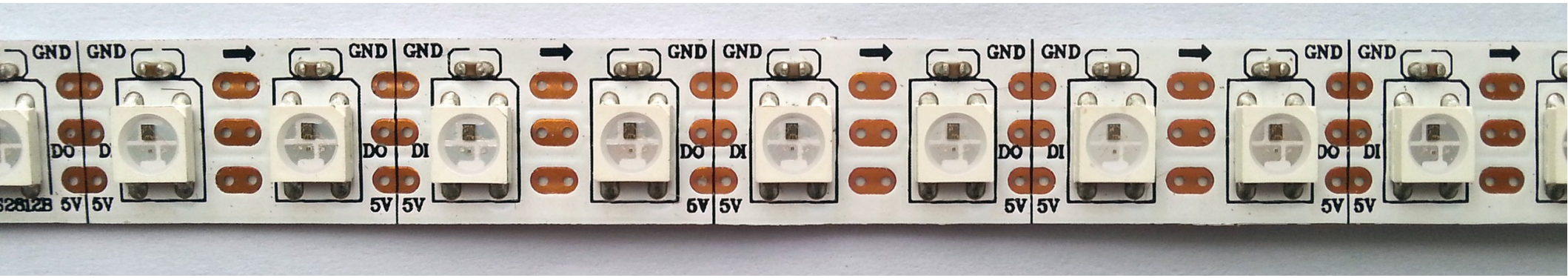
- NeoPixel display LED strips
- Install White LED inside buttons for feedback
- VGA screen, Raspberry Pi

Neo Pixels

- RGB LED (Worldsemi WS2812B chip)
- Individually addressable
- 24-bit color, 255 levels of brightness.
- Great for basic animation, color lighting.
- Not great for POV, video playback (too slow), newer WS2801/WS2822 can do 2-wire, 25MHz.
- 30 LED/Meter, 5 Meter roll: ~$30 (AliExpress)

Neo Pixels

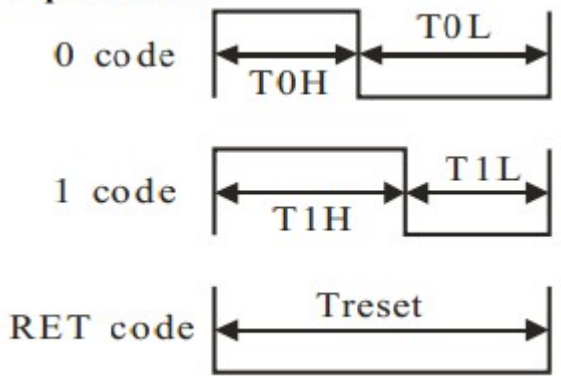
- Serial Data, like a bucket brigade.



Neo Pixels

- Not I2C,TWI or SPI – proprietary timing :(
  Use bit-bang to program.
- 1-bit cycle is about 1.2 microsecond.
- ~50+ usec frame (10 is ok)

Sequence chart:

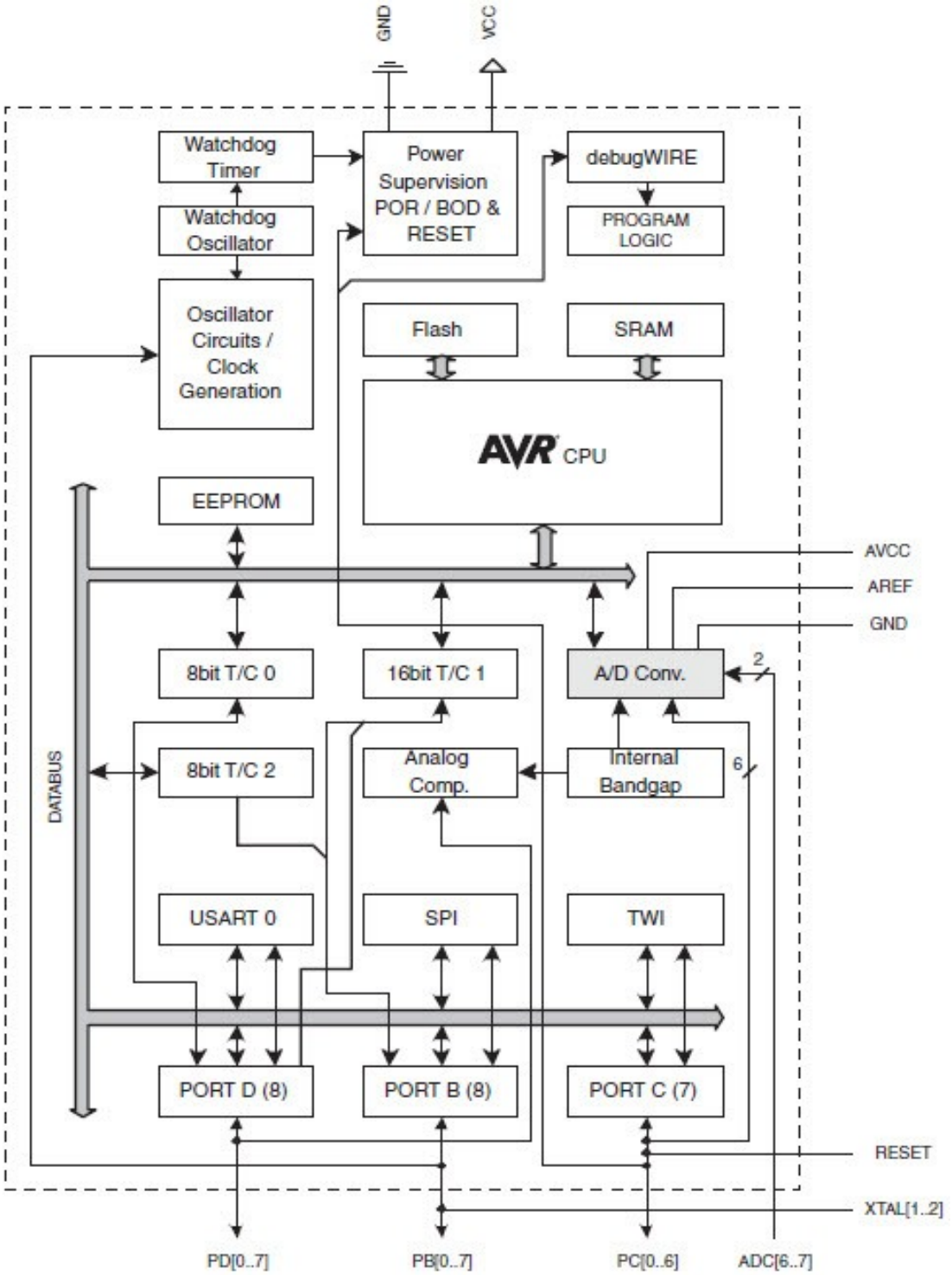


T0H	0 code ,high voltage time	0.35us	±150ns
T1H	1 code ,high voltage time	0.7us	±150ns
T0L	0 code , low voltage time	0.8us	±150ns
T1L	1 code ,low voltage time	0.6us	±150ns
RES	low voltage time	Above 50μs	

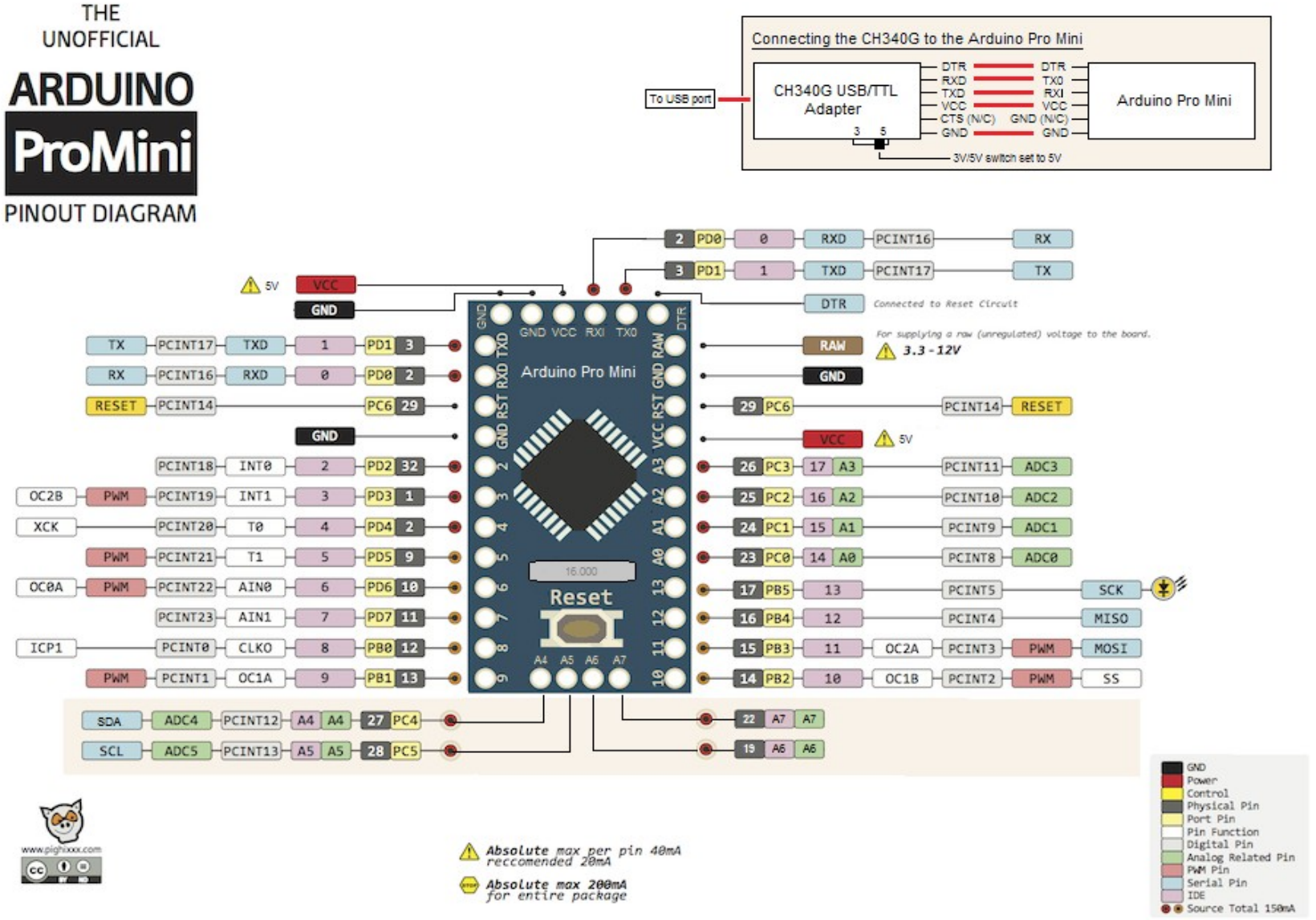
[Heart of Arduino]

Atmel ATMega328 Features...

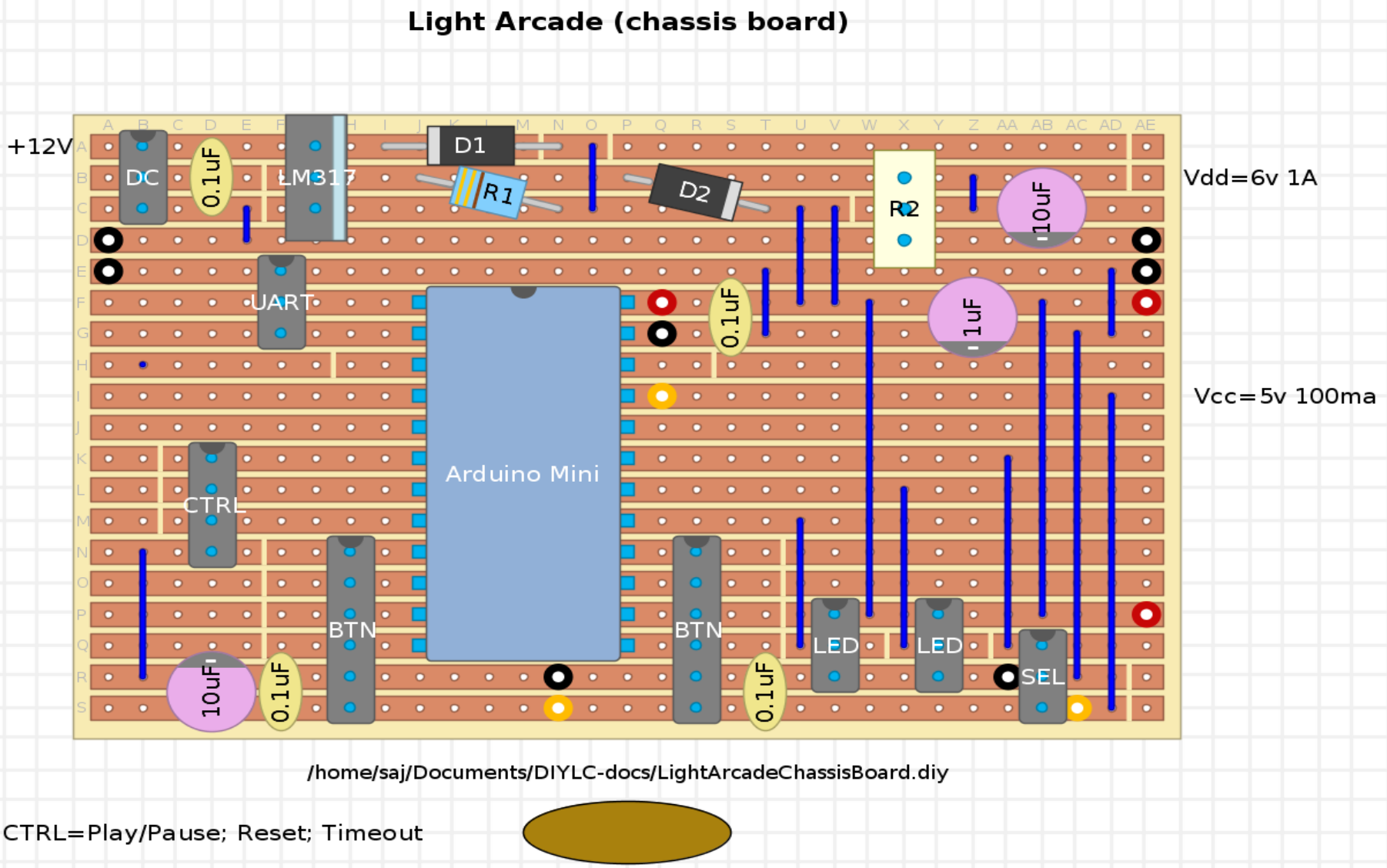
AVR risc cpu
UART (Serial I/O)
Gen. Purp. I/O
Flash (non volatile 32k)
Ram (8k)
Timer (3 total)
Analog → Digital



Arduino Pro Mini



Main Board


Light Arcade (chassis board)
+12V
DC
0.1uF
LM317
D1
R1
D2
R2
10uF
Vdd=6v 1A
UART
0.1uF
1uF
Arduino Mini
Vcc=5v 100ma
CTRL
BTN
BTN
LED
LED
SEL
10uF
0.1uF
0.1uF
/home/saj/Documents/DIYLC-docs/LightArcadeChassisBoard.diy
CTRL=Play/Pause; Reset; Timeout

Arduino: Pin Mapping

```
// ===  PIN CONFIGURATION ====================

#define PIN_P1B1   6   // btn 1
#define PIN_P1L1   7   // led 1
#define PIN_P1B2   8   // btn 2
#define PIN_P1L2   9   // led 2

#define PIN_P2B1  13   // btn 1
#define PIN_P2L1  12   // led 1
#define PIN_P2B2  11   // btn 2
#define PIN_P2L2  10   // led 2

#define PIN_NEWGAME 2    // util pin 2
#define PIN_PAUSE   3    // util pin 3

#define PIN_SELECT_POT  A2  // voltage divider with Potentiometer to select game (someday)

#define PIN_STRIP0  A0   // strip0 / player 1
#define PIN_STRIP1  A1   // strip1 / player 2


// IMPORTANT: To reduce NeoPixel burnout risk, add 1000 uF capacitor across
// pixel power leads, add 300 - 500 Ohm resistor on first pixel's data input
// and minimize distance between Arduino and first pixel.  Avoid connecting
// on a live circuit...if you must, connect GND first.
```

LightArcade Program

- Modular Design:
  - LightArcadeSerial – main module
  - LAButton – button handler
  - LAGame – game “model”, “controller”
  - LAGraphics – LED display “view”
  - LASerial – Serial interface, text reports

- Module common “methods”:
  - InitXXX()
  - updateXXX()

Arduino: Globals

```
// ===  MODULE GLOBALS ==============================

// store pin configuration in arrays for iteration later on
int  pinLed[4]    = {
     PIN_P1L1, PIN_P1L2,  // Player 0 (Left/Blue)
     PIN_P2L1, PIN_P2L2   // Player 1 (Right/Red)
};
int  pinButton[6] = {
     PIN_P1B1, PIN_P1B2, PIN_P2B1, PIN_P2B2, // P0, P1
     PIN_NEWGAME, PIN_PAUSE
};

int  pinUtil[2] = { PIN_NEWGAME, PIN_PAUSE }; // utility inputs

// ~48" strip is 39 LEDs
#define NUM_STRIP_ELEMENTS 39

Adafruit_NeoPixel strip[2] = {
  Adafruit_NeoPixel(NUM_STRIP_ELEMENTS, PIN_STRIP0, NEO_GRB + NEO_KHZ800),
  Adafruit_NeoPixel(NUM_STRIP_ELEMENTS, PIN_STRIP1, NEO_GRB + NEO_KHZ800)
};

long now; // current millis, global clock

uint16_t x; // global loop counter for sequencing of lights, cycles 0 - 0xFFFF
```

Arduino: Setup()

```
void setup() {

  delay(2000); // delay at startup to power up

  // open fast serial port
  Serial.begin(115200);
  while (!Serial)
    ; // wait for serial port to connect. Needed for Leonardo only

  initButtons();

  // Some example procedures showing how to display to the pixels:
  Serial.println("# BEGIN SETUP");

  // setup strips and button LEDs; do lamp test.
  initGraphics();

  // setup new game
  initGameModel();

  x = 0;
  Serial.println("# END SETUP");
}
```

Arduino: Loop()

```
/**
 * Arduino loops with short period.
 * Only a couple calls are blocking (and they fix the clock)
 * Service Inputs, Game model, output to LED and Serial
 */
void loop() {

  now = millis();

  // service input buttons, debouncing and toggle delay values.
  updateButtons(x);

  // update game, calc button rate from toggle delay, calc player scores
  updateGameModel(x);

  // update display on led strips with current player scores
  updateGraphics(x);

  // send report to Raspberry Pi with: clockTick, button status, game state, etc.
  updateSerial(x);

  x++;  // this global 16 bit unsigned integer will overflow (recycle) at 0xFFFF

//  delay(5); // 5 msec sleep/tick interval
  delay(1); // 1 msec sleep/tick interval

}
```

Read Button

- Buttons are active low (gnd == 0 == pressed)

```
// read 6 buttons, debounce, load into bitmask
// 76543210 = mask
//     1100 = player 0/1
//     1010 = button 0/1
//   32     = util aka Pi gpio 2,3
int readButtonsDebounce() {

  buttonCurrMask = 0;
  for (int b = 0; b < 6; b++) {
    buttonCurrMask |= ( (digitalRead(pinButton[b]) ? 0 : 1) << b ); // invert, LOW=1
  }
}
```

Button Logic (Toggle)

- Bit mask to read each button
- Detect on/off state comparing each loop
- Enforce toggle in bitmask

```
unsigned char  toggleMask = 0x05;
// toggle bit mask for both pairs of buttons !MUST INIT 0x0101

// states for player0 (toggle with player0 ANDMASK = 0011 = 0x03)
// if btn0 on and toggleMask 0001, then toggleMask = 0010
// if btn1 on and toggleMask 0010, then toggleMask = 0001
// states for player1 (toggle with player1 ANDMASK = 1100 = 0x0c)
// if btn2 on and toggleMask 0100, then toggleMask = 1000
// if btn3 on and toggleMask 1000, then toggleMask = 0100
// XOR with toggleBit of 0x03 (btn0,1) or 0x06 (btn2,3)
int tryToggleButton(int btn, uint16_t x) {
  if ( (toggleMask & (0x01 << btn)) ) {
    // toggle the mask and wait for the other button
    // NOTE: btn & 0x02 is 0 for btn0,1 and is 1 for btn 2,3
    uint16_t player = (btn & 0x02) >> 1;
    uint16_t toggleBit = ( player ? 0x0c : 0x03 ); // player 1 ANDMASK=0x0c
(0b1100), player 0 ANDMASK=0x03 (0b0011)
    toggleMask ^= toggleBit; // toggle both bits in mask with XOR
    toggleDelay[player] = (now - toggleTime[player]);
    // scale down delay, limit to MAX_DELAY
    toggleDelay[player] = ( toggleDelay[player] > MAX_DELAY) ? MAX_DELAY :
toggleDelay[player];
    toggleTime[player] = now;
    //sendToggleReport(x);  // DEBUG: send toggle report for testing...
    return true;
  }
  else {
    return false;
  }
}
```

Arduino Tour...

LightArcadeSerial

Pi Software

- Need multi-media audio/video display
- Easy to program/modify
- Options?
  - Node.js
  - JohnnyFive robot api.
  - Python,
  - Pygame - perfect.

Pygame (needs)

- Slide Show – show pictures of build
- Read serial data from Arduino
- Event loop
- Handle “reports” (no ACK required)
- Demo: slides
- Play: scores, countdown
- Winner: show winner team

Pygame State Machine

- Demo: show slides
- Play: scores, countdown clock
- Winner: show winner team

Pygame Programming

- LASerial – read text 'report' data stream
- Photo – display: Slides, Clock/Score, Winner
- Sound – play 'start' and 'winner' sound
- Tour pygame code...

Python Tour...

LightArcade.py

IRL: Button problems

- Button debounce too slow
  tempo was important
- Worked ok...some kids 'ran the table' as a result of learning tempo.
- Met game programmer at Maker Faire Orlando "10 msec is glacial"

IRL: Data timing problems

- To fix “score lag” last minute change...
- Push toggleReport() on each button press.
- toggleReport() overloaded python Serial buffer.
- “finish” and sound had delay lag (out of sync).
- Had to turn off Screen/Pi in Orlando :-(

It worked: fun time

Saturday: 10k people
Sunday: 4k people
Monday: lost voice

IRL: Connector problems

- Pin Connector with RTV was flexible acted as strain relief.
- Added Sugru (looks great!)
- Stiff connector had no strain relief, acted as fulcrum, pulled wire.
- Solder joint broke (left blue LED strip)

LA++: next generation

Evolve the Game

LA++: design ideas

- offload LED strip display to dedicated Arduino
- New buttons, better LEDs for feedback
- Offload sound/clock (remove Pi dependency)
- Web cam with Pi / Open CV to capture player image at win/loss, like “Photo-Finish”
- Wi-Fi Portal / NoCatNet / Access Point

LA++: implementation ideas

- I2C instead of Serial text; bus pirate
- Serial used to flash Arduino, pain to disconnect.
- I2C MP3 for sound? no Pi required.
- I2C large 7-segment countdown.

I2C easy: using Wire.h in Arduino.

Arduino Tour... I2C...

Join us!

SuncoastMakers.org

- Hack Light Arcade, or make your own

- Other Ideas coming soon...
- Video Projection Mapping
- Computer Vision (OpenCV)
- Cicada POV (Stationary Fast LED Strips)