Arduino 101 & 102
Workbook & Reference Material

Earl Girbovan
Seth Neumann

Contents

Blink an LED

```
/* Blink

This program blinks an LED on and off
In the TOOLS BOARD tab in the Arduino IDE, ensure 'Arduino Nano w/ATmega328' is selected

*/

int LED2 = 2;  //D2

void setup()
{
  pinMode( LED2, OUTPUT );

}

void loop()
{
  digitalWrite( LED2, HIGH );
  delay( 1000);
  digitalWrite( LED2, LOW );
  delay ( 1000 );

}
```

Fire Effect Code

```
/* LED Fire Effect
   Uses two amber and one red LED to simulate fire
   From the instructables.com web site, slightly modified
*/

int ledPin1 =  9;
int ledPin2 = 10;
int ledPin3 = 11;

void setup()
{
pinMode(ledPin1, OUTPUT);
pinMode(ledPin2, OUTPUT);
pinMode(ledPin3, OUTPUT);
}

void loop()
{
analogWrite(ledPin1, random(135, 255));
analogWrite(ledPin2, random(135, 255));
analogWrite(ledPin3, random(135, 255));
delay(random(1, 100));
}
```

While Loop: Turning the fire off and on

```
/* Intermittent LED Fire Effect
  Uses two amber and one red LED to simulate fire
  Included a WHILE loop, so that the effect turns on and off.
*/

int ledPin1 =  9;
int ledPin2 = 10;
int ledPin3 = 11;
int OnTime;
int OffTime;
int i;

void setup()
{
        pinMode(ledPin1, OUTPUT);
        pinMode(ledPin2, OUTPUT);
        pinMode(ledPin3, OUTPUT);
}

void loop()
{
        OnTime = random( 1000, 5000 );
        while( OnTime > 0 )
                {
                analogWrite(ledPin1, random(120)+135);
                analogWrite(ledPin2, random(120)+135);
                analogWrite(ledPin3, random(120)+135);
                i = random(1, 100);
                delay( i );
                OnTime = OnTime - i;
                }

         // off time, ensure the LEDs are turned off
         analogWrite(ledPin1, 255 );
         analogWrite(ledPin2, 255 );
         analogWrite(ledPin3, 255 );
         OffTime = random ( 1000, 3000 );
         delay ( OffTime );
}
```

Servo Sweep

```
/*
Servo Sweep

This program slowly sweeps a servo back and forth from 0 to a presettable maximum number
of degrees.
*/

#include <Servo.h>

Servo Servo1;

int pos;
int MaxPos = 60;
int DelayTime = 90;

void setup()
{
 //tell the Arduino & the servo driver that the servo is connected to pin 7
 Servo1.attach(7);
}

void loop()
{
 //sweep the servo back and forth
 for( pos = 1; pos <=MaxPos; pos++ ) //sweep from 0 to MaxPos in 1 degree steps
 {
  Servo1.write(pos);
  delay(DelayTime);
 }
 delay( 1000 );

 //now sweep it back
 for( pos = MaxPos; pos >=1; pos--)  //sweep from MaxPos to 0 in 1 degree steps
 {
  Servo1.write(pos);
  delay(DelayTime);
 }
 delay( 1000 );
}
```

Grade Crossing Simulator

```
/*
Grade Crossing

Ties together the blinking lights and servomotor sweep,
along with reading an input, to drive a grade crossing.
The servo simulates a crossing arm.

The sequence is a bit stylized to keep the code manageable
for the clinic.
*/

#include <Servo.h>
Servo Servo1;

int pos;
int MaxPos   = 90;   // max position in degrees
int ServoPos;        // where the servo is in it's sweep
int StepTime = 90;   // time between steps
int ServoTime;       // time since last step

const int TriggerPin = 12;  //Digitial input D12
int Trigger;

int BlinkRate = 1000;  //rate at which the LED blinks
int LEDTime;
int Delta = 90;    // number of msec for each loop iteration
int LEDPin = 2;
boolean LEDState;

void setup()
{
  pinMode( TriggerPin, INPUT_PULLUP);
  //tell the Arduino & the servo driver that the servo is connected to pin 7
  Servo1.attach(7);
  Servo1.write( 0 );   // ensure the crossing gate is raised
  ServoPos = 0;
  LEDState = false;    // LED starts off
  pinMode ( LEDPin, OUTPUT );
  digitalWrite ( LEDPin, HIGH );
  } // end of setup
```

```
void loop()
{
 //poll, looking for a LOW to start the sequence
 do
 {
  Trigger = digitalRead( TriggerPin );
 }
 while( Trigger == HIGH );

 //start the sequence
 ServoTime  = StepTime;   // initialize the crossing arm timer
 LEDTime    = 0;          // go into the routine with the LED off, so it will turn on
 LEDState   = false;
 ServoPos   = 0;         // crossing arm raised position
 do
  {
  // see if we need to change the stat of the LED
  LEDTime = LEDTime - Delta;
  if( LEDTime <= 0 )
   {
    LEDState = !LEDState;  //LED to other state; on or off
    if (LEDState)
    {
     digitalWrite ( LEDPin, LOW );
    }
    else
    {
     digitalWrite ( LEDPin, HIGH );
    }
    LEDTime = BlinkRate;  //load up the next blink cycle
   }

  //see if we need to change the state of the crossing arm
  ServoTime = ServoTime - Delta;
  if( ServoTime <= 0 )
   {
    //if arm not at down position, move it
    if( ServoPos < MaxPos )
     {
      ServoPos++;
      Servo1.write( ServoPos );
      // and reset the timer for the next step
      ServoTime = StepTime;
     }
```

```
    }
  delay ( Delta );
  Trigger = digitalRead( TriggerPin );
  }
 while ( Trigger == LOW );

 //switch has been released, turn off the light and raise the arm
 digitalWrite( LEDPin, HIGH );

 for( int pos = ServoPos; pos >=1; pos--)
 {
  Servo1.write(pos);
  delay( StepTime );
 }


} // end of loop
```

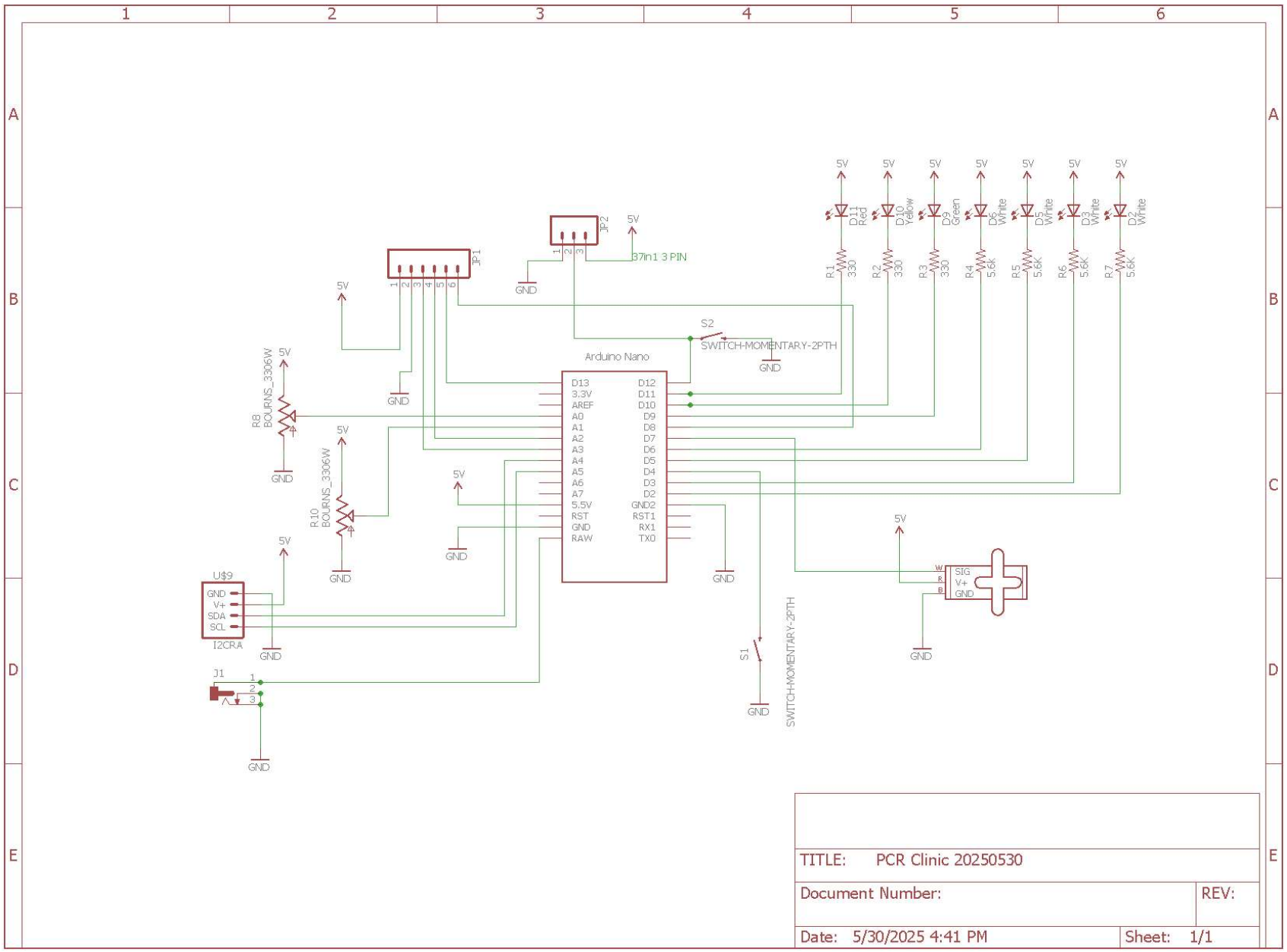
Servo with Travel Limits Set by Pots

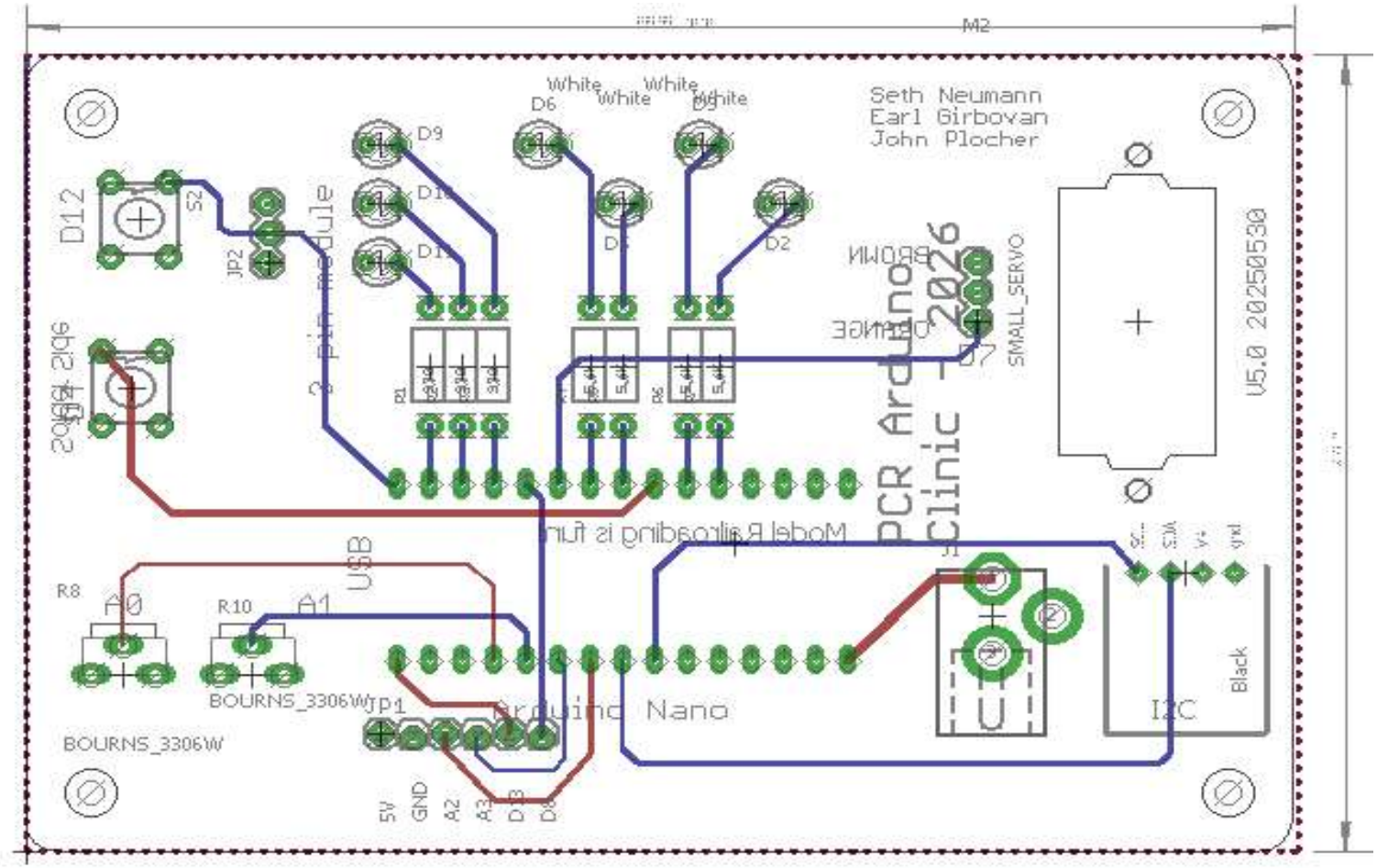
```
#include <Servo.h> // servo Sweep is by BARRAGAN <http://barraganstudio.com> This example code is in the public domain.
Servo myservo;     // create servo object to control a servo
                   // twelve servo objects can be created on most boards

int pos;

//const int LED9 = 9; PWM functionality diabled by using servo library
//const int LED10 = 10; PWM functionality diabled by using servo library

int pot0val = 0; // value for pot read
int pot1val = 0; // value for pot read
int lowpos = 1;  // set it in the range of servo
int highpos = 127;// set it in the range of servo

void setup() {

myservo.attach(7);  // attaches the servo on pin 7 to the servo object

//byte i;
}

void loop() {

 pot0val = analogRead (0); // first pot
 lowpos = pot0val/6; //scale to 1-~180, this is actually 170 degrees
 pot1val = analogRead (1); // second pot
 highpos = pot1val/6; //scale to 1-~180
 {for ( pos=lowpos; pos<=highpos; pos++){

   myservo.write(pos);              // tell servo to go to POSition
   delay(20);                       // waits 15ms for the servo to reach the position
  }
 delay(500); // time to recyle
  }

 // from Banzi example 6B
}
```

```
// PCR demo sketch implementing a voltmeter
// Lawrence Crowl, 18 April 2026


// The voltmeteter displays voltage via a servo arm.

#include <Servo.h>
Servo servo_1;


// The board pinout

const int led_white_4 =  2; // output digital only
const int led_white_3 =  3; // output digital or analog/pwm
const int gnd_button  =  4; // input digital pullup, active low
const int led_white_2 =  5; // output digital or analog/pwm
const int led_white_1 =  6; // output digital or analog/pwm
const int servo_ctl   =  7; // output pwm
const int screw_6     =  8; // in/out digital only
const int led_green   =  9; // output digital or analog/pwm
const int led_yellow  = 10; // output digital or analog/pwm
const int led_red     = 11; // output digital or analog/pwm
const int module_2    = 12; // input digital pullup, active low
const int screw_5     = 13; // output digital only
const int pot_1             = 0; // input analog
const int pot_2             = 1; // input analog
const int screw_4     = 2; // input analog ? or in/out digital ?
const int screw_3     = 3; // input analog ? or in/out digital ?

// All LEDs are off at 255 and on full at 0.


// We use led_green to show that power is on.
// We use led_yellow to show that voltage is being measured.
// We use led_red to show that the indicator arm is being moved.

// We need to slow down the voltmeter so that humans can see it.
int measure_time = 300; // milliseconds
int display_time = 300; // milliseconds
int repeat_time = 300; // milliseconds

int measure()
{
```

```
  digitalWrite( led_yellow, LOW ); // turn led on
  int volt;
  if ( digitalRead( gnd_button ) ) {
    volt = analogRead( pot_1 );
  }
  else {
    volt = analogRead( screw_4 );
  }
  // Convert measured voltage to pointer arm position.
  int arm = volt / 6;
  delay( measure_time );
  digitalWrite( led_yellow, HIGH ); // turn led off
  return arm;
}


int last_arm = 0; // Keep track of last pointer arm position.

void display( int arm )
{
  digitalWrite( led_red, LOW ); // turn led on
  servo_1.write( arm );
  last_arm = arm;
  delay( display_time );
  digitalWrite( led_red, HIGH ); // turn led off
}


void setup()
{
  pinMode( gnd_button, INPUT_PULLUP ); // input digital

  // Not moving the pointer arm.
  pinMode( led_red, OUTPUT ); // output digital
  digitalWrite( led_red, HIGH ); // turn led off

  // Not measuring the voltage.
  pinMode( led_yellow, OUTPUT ); // output digital
  digitalWrite( led_yellow, HIGH ); // turn led off

  // Power is on!
  pinMode( led_green, OUTPUT ); // output digital
  digitalWrite( led_green, LOW ); // turn led on
```

```
  // Initialize the servo.
  servo_1.attach( servo_ctl );

  // Make the first measurement and display it.
  display( measure() );
}


void loop()
{
  delay( repeat_time );
  int next_arm = measure();
  // Avoid potential shuddering in the servo
  // by not commanding an empty move.
  if ( next_arm != last_arm ) {
    display( next_arm );
  }
}
```


1
2
3
4
5
6
A
B
C
D
E
JP2
37in1 3 PIN
JP1
Arduino Nano
D13
3.3V
AREF
A0
A1
A2
A3
A4
A5
A6
A7
5.5V
RST
GND
RAW
D12
D11
D10
D9
D8
D7
D6
D5
D4
D3
D2
GND2
RST1
RX1
TX0
S2
SWITCH-MOMENTARY-2PTH
S1
SWITCH-MOMENTARY-2PTH
R8
BOURNS_3306W
R10
BOURNS_3306W
R1
330
R2
330
R3
330
R4
5.6k
R5
5.6k
R6
5.6k
R7
5.6k
D11
Red
D10
Yellow
D9
Green
D6
White
D5
White
D3
White
D2
White
U$9
GND
V+
SDA
SCL
I2CRA
J1
SIG
V+
GND
5V
GND
TITLE: PCR Clinic 20250530
Document Number:
REV:
Date: 5/30/2025 4:41 PM
Sheet: 1/1



10% Discount Coupon on all Nano and related products through May 30! Use code PCR2026N. 1 use per customer!