**Arduino - based Automatic Block Signalling (on a budget!)**

This small project describes an automatic block signalling system controlling 4 signals (one pair of signals for the down track and one pair for the up track) using one “Arduino Nano” (< £9 including terminals) – and 8 infra-red sensors in the track (~ 60p each).

A close up of a map

Description automatically generatedOur layout has twin tracks with a station at one end, and a level crossing near the other end, and both tracks are fitted with two 3 aspect signals with sensors placed immediately after each signal. The other 2 sensors per track are placed just before and just after the fiddle yards for each circuit – see the simplified schematic below. The outer loop direction is clockwise, the inner loop anticlockwise.

The Arduino sketch is shown below – split into 3 columns to get it on two sides of A4. If you would like an electronic copy, please e-mail me at [sdwadsworth@yahoo.co.uk](mailto:sdwadsworth@yahoo.co.uk)

// Automatic Signal system for 2 loops each with 2 signals and 4 sensors

// Designed for an Arduino Nano with 8 analogue inputs and 12 digital outputs

// Each sensor is connected to an analogue input pin, and outputs are connected to the 12 digital output pins.

// This sketch is set up for common anode connection to the +5Volt supply via a 1K resistor so the LED is switched on when the output is LOW

// Signals 1 and 2 are controlled by Sensors 1 to 4; Signals 3 and 4 are controlled by Sensors 5 to 8

// With thanks to Jimmy of the DIY and Digital Railroad YouTube channel for initial ideas, and a "State Machine” Arduino Tutorial explaining how to handle more than one loop at a time.

// Set-up Pins for sensors and signals

int Sensor1 = A0;

int Sensor2 = A1;

int Sensor3 = A2;

int Sensor4 = A3;

int Sensor5 = A4;

int Sensor6 = A5;

int Sensor7 = A6;

int Sensor8 = A7;

int RED1 = 2;

int YELLOW1 = 3;

int GREEN1 = 4;

int RED2 = 5;

int YELLOW2 = 6;

int GREEN2 = 7;

int RED3 = 8;

int YELLOW3 = 9;

int GREEN3 = 10;

int RED4 = 11;

int YELLOW4 = 12;

int GREEN4 = 13;

// Define input and output pin types, and set up the default signals for each loop - Signal 1 Yellow and Signal 2 Green etc.

void setup() {

Serial.begin(9600);

pinMode(Sensor1, INPUT);

pinMode(Sensor2, INPUT);

pinMode(Sensor3, INPUT);

pinMode(Sensor4, INPUT);

pinMode(Sensor5, INPUT);

pinMode(Sensor6, INPUT);

pinMode(Sensor7, INPUT);

pinMode(Sensor8, INPUT);

pinMode(RED1, OUTPUT);

pinMode(RED2, OUTPUT);

pinMode(RED3, OUTPUT);

pinMode(RED4, OUTPUT);

pinMode(GREEN1, OUTPUT);

pinMode(GREEN2, OUTPUT);

pinMode(GREEN3, OUTPUT);

pinMode(GREEN4, OUTPUT);

pinMode(YELLOW1, OUTPUT);

pinMode(YELLOW2, OUTPUT);

pinMode(YELLOW3, OUTPUT);

pinMode(YELLOW4, OUTPUT);

}

// Setup variables for each track loop

enum SIGNALSTATES1

{

ST\_GY1,

ST\_RG1,

ST\_RR1,

ST\_YR1,

};

enum SIGNALSTATES2

{

ST\_GY2,

ST\_RG2,

ST\_RR2,

ST\_YR2,

};

// Initial states for each track loop

SIGNALSTATES1 signalState1 = ST\_GY1;

SIGNALSTATES2 signalState2 = ST\_GY2;

// Main Loop

void loop() {

int valA1 = analogRead(Sensor1);

int valA2 = analogRead(Sensor2);

int valA3 = analogRead(Sensor3);

int valA4 = analogRead(Sensor4);

Serial.println(valA1);

Serial.println(valA2);

Serial.println(valA3);

Serial.println(valA4);

int valA5 = analogRead(Sensor5);

int valA6 = analogRead(Sensor6);

int valA7 = analogRead(Sensor7);

int valA8 = analogRead(Sensor8);

Serial.println(valA5);

Serial.println(valA6);

Serial.println(valA7);

Serial.println(valA8);

switch (signalState1) // First loop of 2 signals and 4 sensors

{

case ST\_GY1:

signalgy1(valA1, valA2, valA3, valA4);

break;

case ST\_RG1:

signalrg1(valA1, valA2, valA3, valA4);

break;

case ST\_RR1:

signalrr1(valA1, valA2, valA3, valA4);

break;

case ST\_YR1:

signalyr1(valA1, valA2, valA3, valA4);

break;

}

switch (signalState2) // Second loop of 2 signals and 4 sensors

{

case ST\_GY2:

signalgy2(valA5, valA6, valA7, valA8);

break;

case ST\_RG2:

signalrg2(valA5, valA6, valA7, valA8);

break;

case ST\_RR2:

signalrr2(valA5, valA6, valA7, valA8);

break;

case ST\_YR2:

signalyr2(valA5, valA6, valA7, valA8);

break;

}

}

// End of main loop

// What to do with the results for Signals 1 and 2

void signalgy1(int valA1, int valA2, int valA3, int valA4) {

digitalWrite(GREEN1, LOW);

digitalWrite(RED1, HIGH);

digitalWrite(YELLOW1, HIGH);

digitalWrite(GREEN2, HIGH);

digitalWrite(RED2, HIGH);

digitalWrite(YELLOW2, LOW);

if (valA1 < 500 && valA2 > 500) {

signalState1 = ST\_RG1;

}

else if (valA1 > 500 && valA2 < 500) {

signalState1 = ST\_RR1;

}

}

void signalrg1(int valA1, int valA2, int valA3, int valA4) {

digitalWrite(GREEN1, HIGH);

digitalWrite(RED1, LOW);

digitalWrite(YELLOW1, HIGH);

digitalWrite(GREEN2, LOW);

digitalWrite(RED2, HIGH);

digitalWrite(YELLOW2, HIGH);

if (valA2 < 500) {

signalState1 = ST\_RR1;

}

}

void signalrr1(int valA1, int valA2, int valA3, int valA4) {

digitalWrite(GREEN1, HIGH);

digitalWrite(RED1, LOW);

digitalWrite(YELLOW1, HIGH);

digitalWrite(GREEN2, HIGH);

digitalWrite(RED2, LOW);

digitalWrite(YELLOW2, HIGH);

if (valA1 > 500 && valA2 > 500 && valA3 < 500) {

signalState1 = ST\_YR1;

}

}

void signalyr1(int valA1, int valA2, int valA3, int valA4) {

digitalWrite(GREEN1, HIGH);

digitalWrite(RED1, HIGH);

digitalWrite(YELLOW1, LOW);

digitalWrite(GREEN2, HIGH);

digitalWrite(RED2, LOW);

digitalWrite(YELLOW2, HIGH);

if (valA4 < 500) {

signalState1 = ST\_GY1;

}

}

// Do the same for Signals 3 and 4

void signalgy2(int valA5, int valA6, int valA7, int valA8) {

digitalWrite(GREEN3, LOW);

digitalWrite(RED3, HIGH);

digitalWrite(YELLOW3, HIGH);

digitalWrite(GREEN4, HIGH);

digitalWrite(RED4, HIGH);

digitalWrite(YELLOW4, LOW);

if (valA5 < 500 && valA6 > 500) {

signalState2 = ST\_RG2;

}

else if (valA5 > 500 && valA6 < 500) {

signalState2 = ST\_RR2;

}

}

void signalrg2(int valA5, int valA6, int valA7, int valA8) {

digitalWrite(GREEN3, HIGH);

digitalWrite(YELLOW3, HIGH);

digitalWrite(RED3, LOW);

digitalWrite(YELLOW4, HIGH);

digitalWrite(GREEN4, LOW);

digitalWrite(RED4, HIGH);

if (valA6 < 500) {

signalState2 = ST\_RR2;

}

}

void signalrr2(int valA5, int valA6, int valA7, int valA8) {

digitalWrite(GREEN3, HIGH);

digitalWrite(YELLOW3, HIGH);

digitalWrite(RED3, LOW);

digitalWrite(YELLOW4, HIGH);

digitalWrite(GREEN4, HIGH);

digitalWrite(RED4, LOW);

if (valA5 > 500 && valA6 > 500 && valA7 < 500) {

signalState2 = ST\_YR2;

}

}

void signalyr2(int valA5, int valA6, int valA7, int valA8) {

digitalWrite(GREEN3, HIGH);

digitalWrite(YELLOW3, LOW);

digitalWrite(RED3, HIGH);

digitalWrite(YELLOW4, HIGH);

digitalWrite(GREEN4, HIGH);

digitalWrite(RED4, LOW);

if (valA8 < 500) {

signalState2 = ST\_GY2;

}

}

The Arduino Nano with terminal block

A circuit board

Description automatically generated

IR sensor module with 5mm Tx/Rx diodes.

(For n-gauge replace with 3mm diodes)

A circuit board

Description automatically generated