VIKING

ARD-PLC-4CH
PLC Shield, 4 Channel
Wiring & Application Guide

www.vikingmachinery.co.nz

Version 1.1 – April 2020

0.0 Safety Statement

All machinery, especially CNC or automated machinery, has inherent dangers and risks. It is the responsibility of the system designer to ensure that any systems built using any Viking Machinery Ltd. products are safe for use. Any technical information is provided as a reference only, and does not constitute a recommendation as to the fitness of use in any particular application.

Viking Machinery Ltd. strongly urges customers to seek expert advice when dealing with potentially dangerous electrical voltages and sources of mechanical energy. Information contained in this document does not constitute a substitute for expert advice.

This device is not intended to switch low voltage (known as "Mains voltage"). If you need to switch a high power load, use this device to switch a contactor or relay of appropriate rating.

Under no circumstances should this product ever be used in a safety critical application.

1.0 Product Specifications

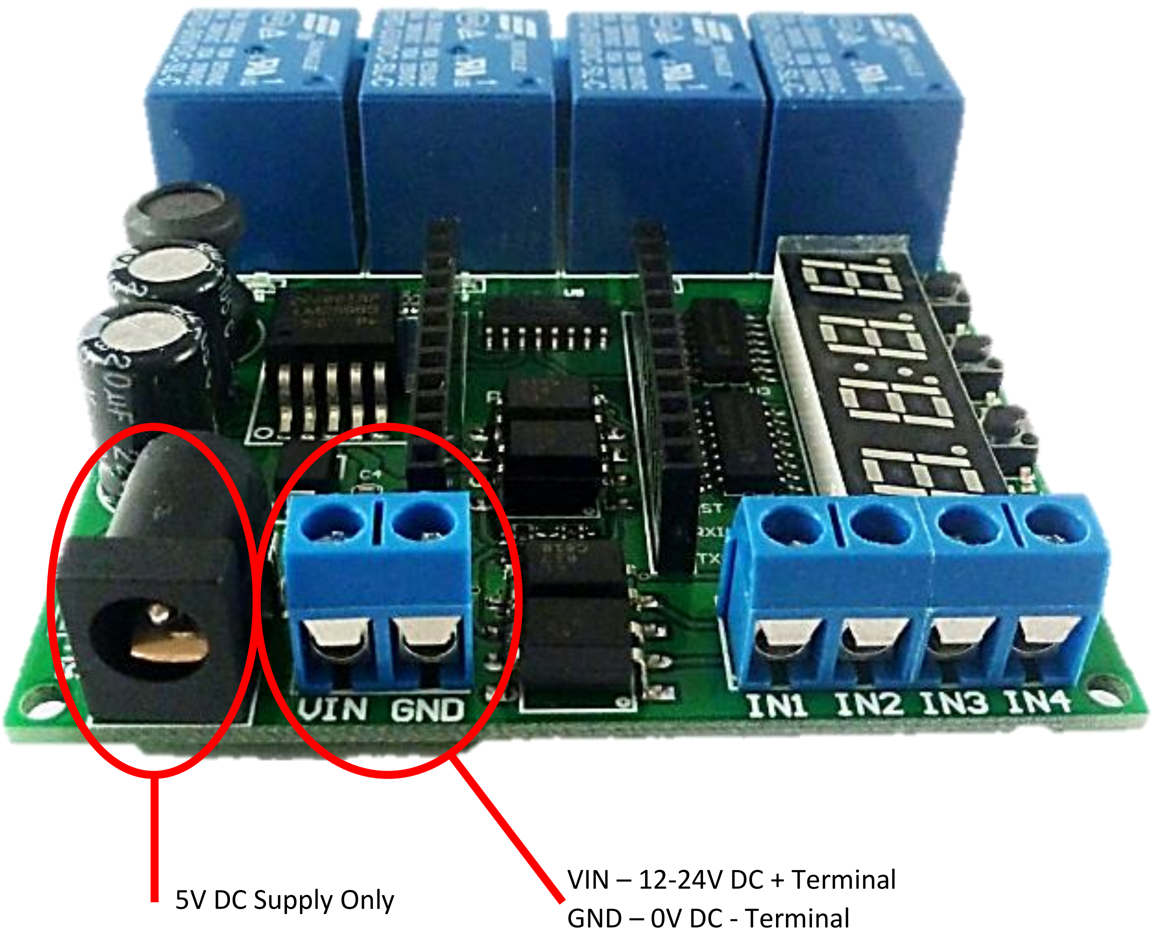
- Supply Voltage – 5-24VDC
- Operating Current – 15mA – 140mA
- Voltage Ripple – 10% peak to peak maximum
- Number of Relay Channels – 4
- Number of External Inputs – 4
- Number of Onboard Push Button Inputs – 4
- Display – 4 digit 7 segment display
- Processor – Supplied with Arduino Pro Micro, 5V, 16MHz
- Size – 80mm x 66mm x 19mm

2.0 Scope of Document

This document is designed to give an overview of the wiring options for the ARD-PLC-4CH Arduino PLC shield. Wiring examples are given for typical examples that a user may encounter. These are by no means exhaustive, but are a good starting point for beginners. At the end of the manual a demonstration program is given for programming the device.

3.0 Power Connection

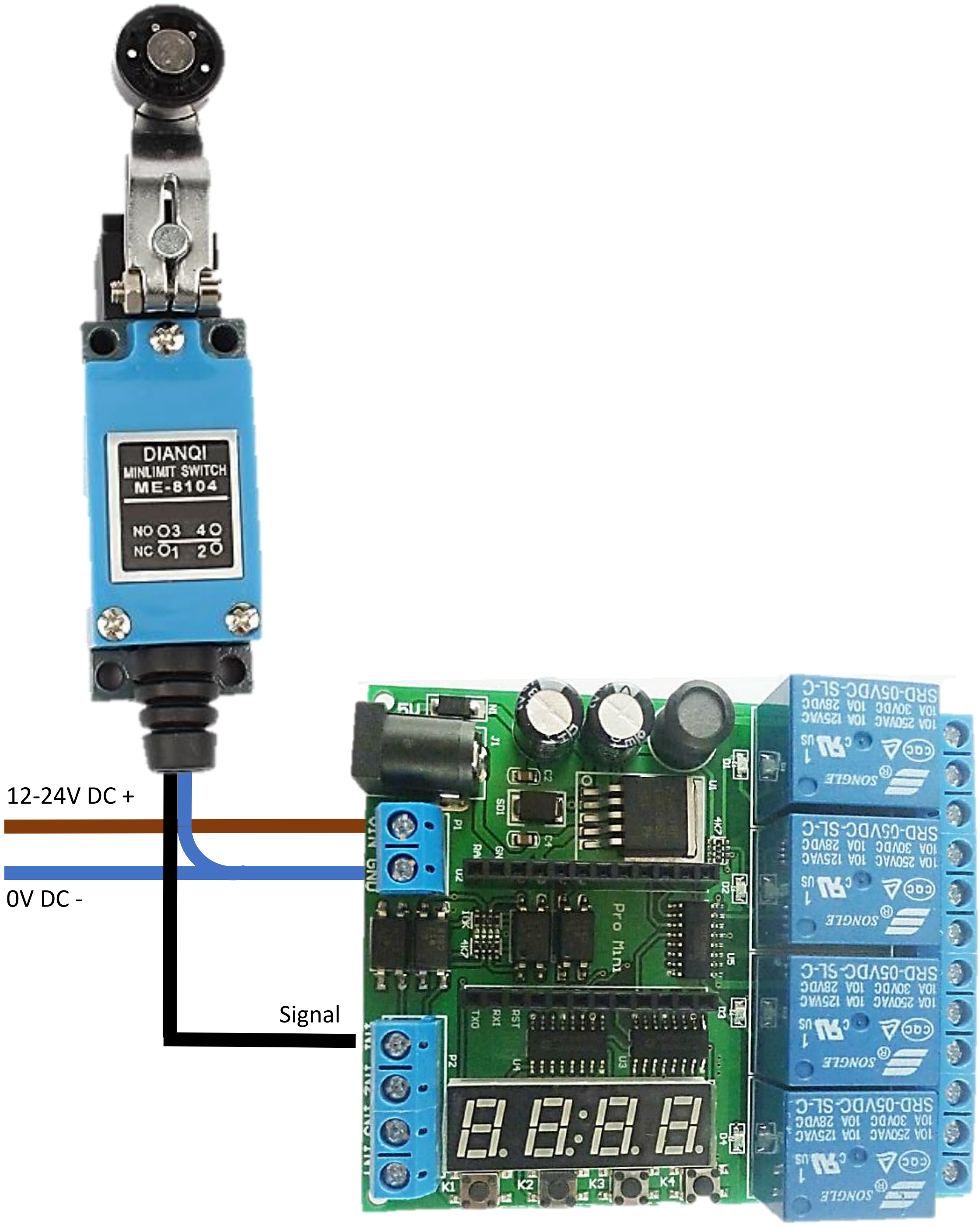
Your PLC shield has two power options. It can be powered with 5V DC via the 4.8mm barrel jack, or else with 12-24V DC via the "VIN" & "GND" terminals. Do not apply more than 5V DC to the barrel jack as this will damage the unit.



4.0 Wiring to a Switch

The input terminals of the board need to be switched to 0V for the shield to register an input. We recommend that a beginner wired their switch to the NORMALLY OPEN (NO) contact set of their switch. This is the "failsafe" wiring configuration for this shield and so is least likely to give unexpected behaviour on start up.

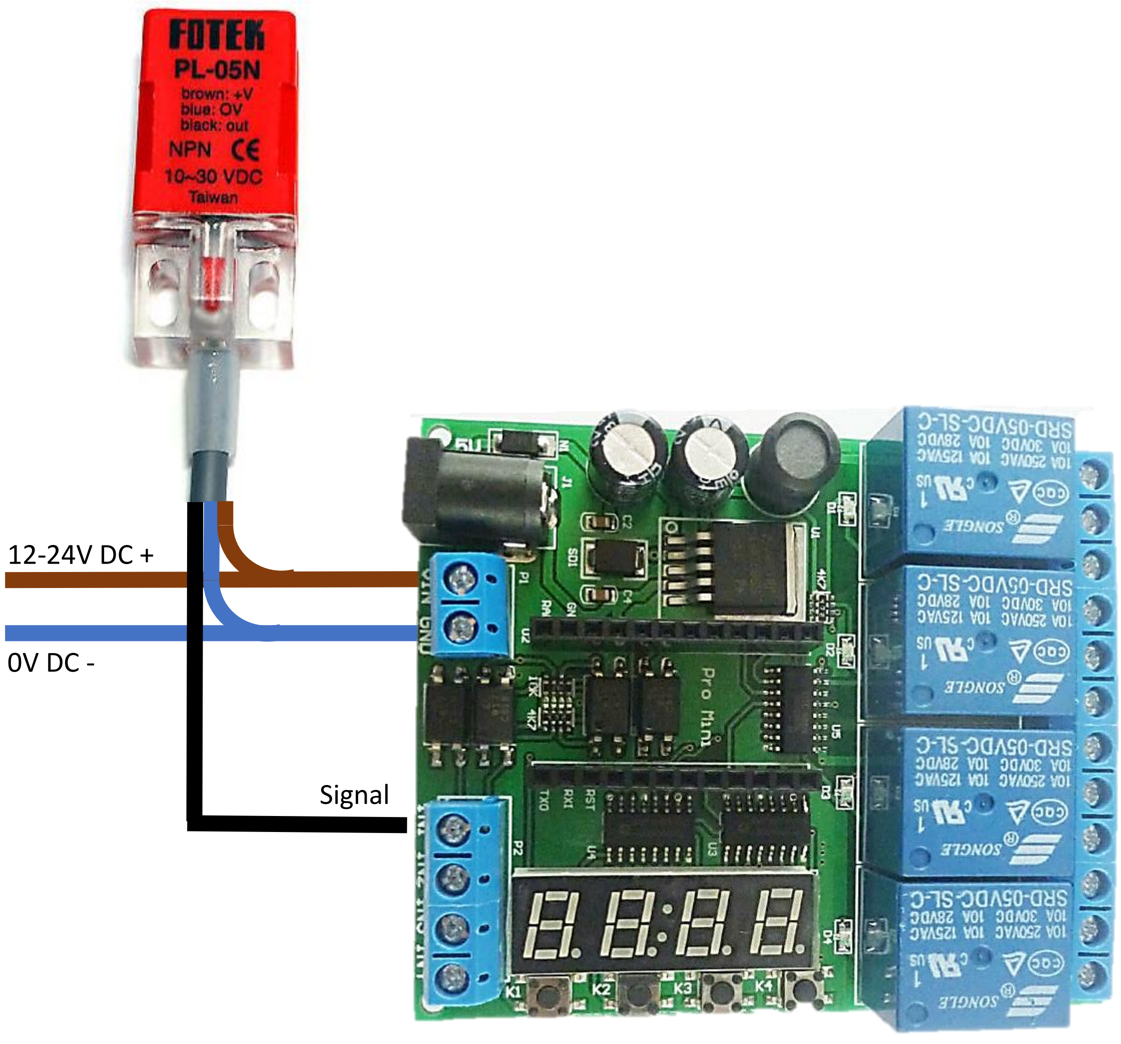
The PLC shield supports up to four proximity sensor inputs (marked IN1 – IN4 on the board), so the wiring below can be repeated for each input number.



5.0 Wiring to a Proximity Sensor

Wiring to the NPN proximity sensors that Viking Machinery sells is easily accomplished in the manner shown below. The PLC shield will run on 12-24V DC, which is the same range as the proximity sensors, so wiring is very straight forward. These sensors can be powered from the same source as the shield, and then the black signal line is used to switch the input terminal to ground.

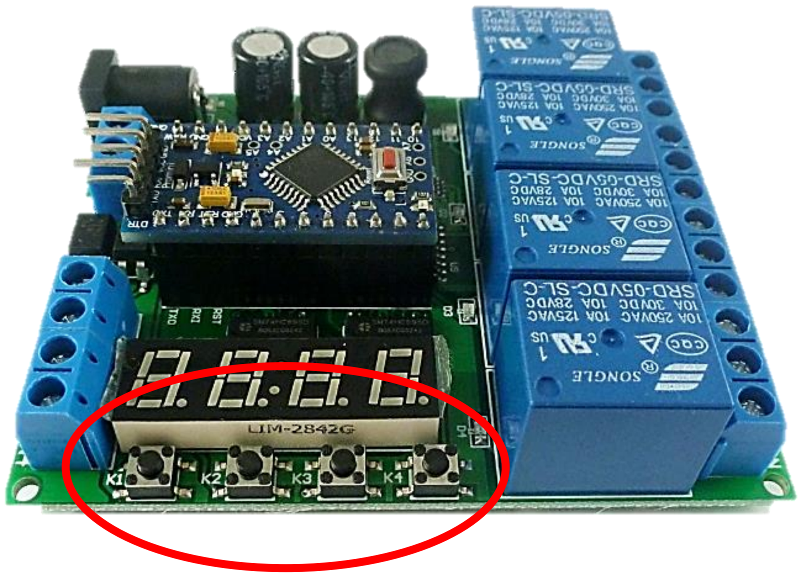
The PLC shield supports up to four proximity sensor inputs (marked IN1 – IN4 on the board), so the wiring below can be repeated for each input number.



6.0 Onboard Input Buttons

The board contains four individual input buttons. These can be used to change settings on the board, start or stop a program, or in any other way you choose to program.

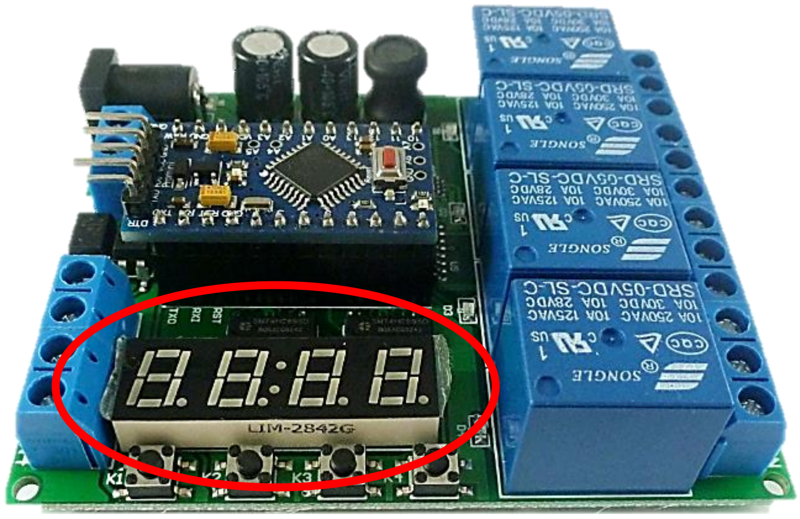
These inputs are numbered 1-4, from left to right as shown. They are marked K1-K4 (Key one – key four) on the PCB.



7.0 Onboard Segment Display

The board contains four individual seven segment display digits. These can be used to display any number and many letter characters from the program.

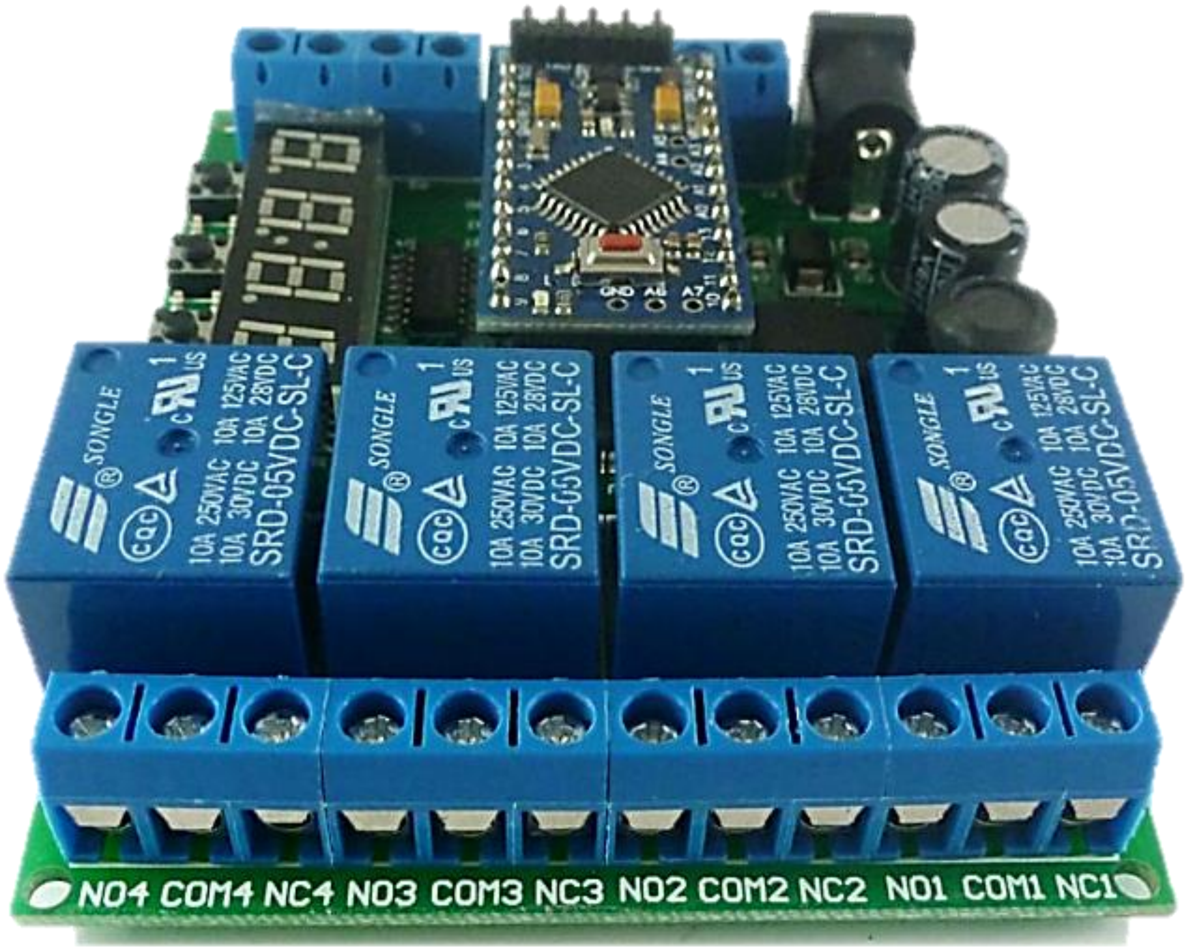
These digits are numbered 1-4, from left to right as shown.



8.0 Outputs

The PLC shield has four relay outputs. These are numbered from 1 to 4 from the right to left below.

Each relay output has three terminals; Common (COM), Normally Open (NO), and Normally Closed (NC). When the relay operates, the connection between COM & NC will break, and the connection between COM & NO will make.



9.0 Demonstraition Program

Below is a sample code that maps and makes use of all inputs, button inputs, outputs and segment display outputs. This is likely a good starting point for your own project and can be downloaded (along with the flexi timer 2 library that you will need) from our GitHub page. You can also just copy and paste from here! If you do get stuck, we offer programming as a service and can build and program for you!

```
// PLC Shield Demo Code

// Written by; Euan Stewart & James Hussey

// Viking Machinery Ltd.

// Version 1.1

// 15/11/2017

// www.vikingmachinery.co.nz

// Original code for Pro-Mini

// Up-rev for Pro Micro 26/02/2019

// Viking Machinery ONLY USES 5V, 16MHz Pro Micros!


//------------------------------ Pin Names etc. --------------------------------------------------------------------//


#include <FlexiTimer2.h>  // Install this library if you have not already. We use it to read the push
buttons real-time(ish)


#define uchar unsigned char

#define uint  unsigned int


int display_dat;


const int led = 13;     //Onboard LED


const int latch = 8;    //"Latch" pin of Digital Tube Module

const int clock = 9;    //"Clock" pin of Digital Tube Module

const int data = 7;     //"Data" pin of Digital Tube Module
const int K1 = 2;       //These are the tactile switch pins

const int K2 = 3;

const int K3 = 4;

const int K4 = 5;


const int INPUT1 = A1;  //These are the external input pins (driven through screw terminals)

const int INPUT2 = A0;

const int INPUT3 = A3;

const int INPUT4 = A2;


const int R1 = 10;      //These are the relay outputs

const int R2 = 16;      // (Pin 11 on Pro Mini)

const int R3 = 14;      // (Pin 12 on Pro Mini)

const int R4 = 6;


int disp1 = 0;          //These are the values for each digit of the display, automagically updated

int disp2 = 0;

int disp3 = 0;

int disp4 = 0;


/*------------------------------------------------------------------------------------------------------------------

  --------------------------------------- 7 Segment Display Subroutine -----------------------------------------------

  ---------------------------------- Do not modify unless you know what you are doing ----------------------------
------

  -------------------------------------------------------------------------------------------------------------------*/


//      NO.:0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22 23 24 25 26 27 28

// Character :0,1,2,3,4,5,6,7,8,9,A, b, C, c, d, E, F, H, h, L, n, N, o, P, r, t, U, -,
uchar  TUBE_SEG[] = {0xc0, 0xf9, 0xa4, 0xb0, 0x99, 0x92, 0x82, 0xf8, 0x80, 0x90, 0x88, 0x83, 0xc6,
0xa7, 0xa1, 0x86, 0x8e, 0x89, 0x8b, 0xc7, 0xab, 0xc8, 0xa3, 0x8c, 0xaf, 0x87, 0xc1, 0xbf, 0xff};
//Common anode Digital Tube Character Gallery


uchar TUBE_NUM[8] = {0xfe, 0xfd, 0xfb, 0xf7, 0xef, 0xdf, 0xbf, 0x7f}; //Tube bit number


uchar dat_buf[8];


uchar dat;      //dat : Data to be displayed
uchar com_num;  //com_num :  Digital Tube Common


void TubeDisplayOneBit()
{
  uchar tube_dat;
  uchar bit_num;
  tube_dat = TUBE_SEG[dat];
  bit_num = ~TUBE_NUM[com_num];


  digitalWrite(latch, LOW);                //Brings Pin Low to signal start of data transmission


  shiftOut(data, clock, MSBFIRST, bit_num); // Data transmission
  shiftOut(data, clock, MSBFIRST, tube_dat);


  digitalWrite(latch, HIGH);               //Brings latch pin High to signal end of data transmission
}


uchar OneSecondCnt;


void UpdateDisplay(void)
{
  if (com_num < 3) com_num ++; else com_num = 0;
  dat = dat_buf[com_num];
  //Only updates 1 character of the display every cycle
  TubeDisplayOneBit();


  dat_buf[0] = disp1;
  dat_buf[1] = disp2;
  dat_buf[2] = disp3;
  dat_buf[3] = disp4;
}


/*------------------------------------------------------------------------------------------------------------------
  --------------------------------------- End of 7 Segment Display Subroutine ------------------------------------------
-
  ------------------------------------------------------------------------------------------------------------------*/




/*-------------------------------------------------------------------------------------
  ------------------------------------- Setup Loop -------------------------------------
  ------------------------------ Runs once only on Startup -------------------------------
  -------------------------------------------------------------------------------------*/


void setup() {


  // Open serial port, 9600 baud rate
  Serial.begin(9600);


  // LED pin set as output
  pinMode(led, OUTPUT);


  // 7 Segment display pins set as outputs
  pinMode(latch, OUTPUT);
  pinMode(clock, OUTPUT);
  pinMode(data, OUTPUT);

  // Tactile switch pins set as inputs, pullup resistors activated
  pinMode(K1, INPUT_PULLUP);
  pinMode(K2, INPUT_PULLUP);
  pinMode(K3, INPUT_PULLUP);
  pinMode(K4, INPUT_PULLUP);

  // External input pins set as inputs, pullup resistors activated
  pinMode(INPUT1, INPUT_PULLUP);
  pinMode(INPUT2, INPUT_PULLUP);
  pinMode(INPUT3, INPUT_PULLUP);
  pinMode(INPUT4, INPUT_PULLUP);

  // Relay pins are set as outputs
  pinMode(R1, OUTPUT);
  pinMode(R2, OUTPUT);
  pinMode(R3, OUTPUT);
  pinMode(R4, OUTPUT);

  // Sets up FlexiTimer function with every 5ms as the check interval
  FlexiTimer2::set(5, 1.0 / 1000, UpdateDisplay);
  // Starts the FlexiTimer function
  FlexiTimer2::start();

  // Sets the relay outputs to Low
  digitalWrite(R1, LOW);
  digitalWrite(R2, LOW);
  digitalWrite(R3, LOW);
  digitalWrite(R4, LOW);
```

```
  //Sets a "Welcome Message" of ---- on the 7 segment display
  disp1 = 27;
  disp2 = 27;
  disp3 = 27;
  disp4 = 27;

  // Sends a welcome message on serial
  Serial.print("Welcome to the PLC shield demo");
  Serial.println("");

  //Flashes the onboard LED twice to say hello
  digitalWrite(led, LOW);
  delay(800);
  digitalWrite(led, HIGH);
  delay(800);
  digitalWrite(led, LOW);
  delay(800);
  digitalWrite(led, HIGH);
  delay(800);
  digitalWrite(led, LOW);
}

/*------------------------------------------------------------------------------------------
  ------------------------------ Main Program Void Loop ----------------------------------
  ------------------------------- Loops Infinately ---------------------------------------
  ------------------------------------------------------------------------------------*/



void loop() {
 while (1)
 {
  //----------------------Read Inputs and Switch Relays Accordingly---------------------------


  if (digitalRead(INPUT1) == LOW) // Read Input 1. If activated, Activate Relay 1.
  {
   digitalWrite(R1, HIGH);
   Serial.print("Input One Active");
   Serial.println("");
  }
  else if (digitalRead(INPUT1) == HIGH) // Read Input 1. If deactivated, Deactivate Relay 1.
  {
   digitalWrite(R1, LOW);
  }



  if (digitalRead(INPUT2) == LOW) // Read Input 2. If activated, Activate Relay 2.
  {
   digitalWrite(R2, HIGH);
   Serial.print("Input Two Active");
   Serial.println("");
  }
  else if (digitalRead(INPUT2) == HIGH) // Read Input 2. If deactivated, Deactivate Relay 2.
  {
   digitalWrite(R2, LOW);
  }



  if (digitalRead(INPUT3) == LOW) // Read Input 3. If activated, Activate Relay 3.
  {
   digitalWrite(R3, HIGH);
```

```
    Serial.print("Input Three Active");
    Serial.println("");
  }
  else if (digitalRead(INPUT3) == HIGH) // Read Input 3. If deactivated, Deactivate Relay 3.
  {
    digitalWrite(R3, LOW);
  }



  if (digitalRead(INPUT4) == LOW) // Read Input 4. If activated, Activate Relay 4.
  {
    digitalWrite(R4, HIGH);
    Serial.print("Input Four Active");
    Serial.println("");
  }
  else if (digitalRead(INPUT4) == HIGH) // Read Input 4. If deactivated, Deactivate Relay 4.
  {
    digitalWrite(R4, LOW);
  }

  //----------------Read push buttons and increment 7 segment display counter by +1 with rollover
after 9 for each key press ---------------

  if    (digitalRead(K1) == LOW) {
    disp1++;
    if (disp1 > 9) {
      disp1 = 0;
    } delay (250);
  }
  else if (digitalRead(K2) == LOW) {
    disp2++;
```

```
     if (disp2 > 9) {
       disp2 = 0;
     } delay (250);
   }
   else if (digitalRead(K3) == LOW) {
     disp3++;
     if (disp3 > 9) {
       disp3 = 0;
     } delay (250);
   }
   else if (digitalRead(K4) == LOW) {
     disp4++;
     if (disp4 > 9) {
       disp4 = 0;
     } delay (250);
   }
  }
}

//------------------------------------ ...and that's all folks! -------------------------
//---------------------------------------------------------------------------------------
```

7.0 Reference Links

Viking Machinery – Home Page

www.vikingmachinery.co.nz

Viking Machinery – TradeMe Store

https://www.trademe.co.nz/Members/Listings.aspx?member=4906214

Viking Machinery – Email

vikingmachinerynz@gmail.com

Viking Machinery – Social Media

https://www.instagram.com/james_viking_machinery/

https://www.cgtrader.com/viking-nz

https://www.youtube.com/channel/UCgnl_7dUO9MeNOyI_jWO5QQ

https://www.thingiverse.com/VikingNZ/about

https://grabcad.com/james.hussey-3

Viking Machinery – Git Hub

https://github.com/Viking-Machinery