/\* ====== IR\_Remote\_Basics\_V3 ===== by Tony Goodhew - 22 September 2014

FunDuino/Arduino UNO R3 - All available from HobbyComponents.com

Arduino compatible Multi Function experimenter shield (HCARDU0085)

IR Remote control - NEC codes - Hobby Components (HCIRRC0001)

IR receiver on pin 2 - 1838B, (HCSENS0014) - X front faces the buttons in U4 - tight fit!

!!!!!!!!!!!! DO NOT USE SFH506-38 as socket does not match its pins !!!!!!!!!!!!!!!!!

This program illustrates input via an InfraRed Receiver from IR Remote control

and output via the 7-segment display and individual LEDs.

Some code based on HOBBY COMPONENTS documentation.

You may copy, alter and reuse this code in any way you like, but please leave

reference to HobbyComponents.com and the author, Tony Goodhew, in your comments if you redistribute this code.

This software may not be used for the purpose of promoting or selling products

that directly compete with Hobby Components Ltd's own range of products.

THIS SOFTWARE IS PROVIDED "AS IS". HOBBY COMPONENTS AND THE AUTHOR, Tony Goodhew, MAKE NO WARRANTIES, WHETHER

EXPRESS, IMPLIED OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF

MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ACCURACY OR LACK OF NEGLIGENCE.

HOBBY COMPONENTS, OR Tony Goodhew, SHALL NOT, IN ANY CIRCUMSTANCES, BE LIABLE FOR ANY DAMAGES,

INCLUDING, BUT NOT LIMITED TO, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES FOR ANY

REASON WHATSOEVER.

\*/

// Pat McMahon 25/11/2020

//A035-MFS3

// This mfs sketch uses the 38Kz IR receiver,NEC remote (mini remote), It displays on the 7 seg display and also go to Tools>Serial Monitor> see buttons pushed on Screen.

// I modified the code from the Car mp3 ie CH-,CH,CH- etc to 1,2,3 etc like on the NEC Mini remote with the up and down arrows.

#include <IRremote.h> // https://github.com/shirriff/Arduino-IRremote

int RECV\_PIN = 2;

IRrecv irrecv(RECV\_PIN);

decode\_results results;

// Define shift register pins used for seven segment display

#define LATCH 4

#define CLK 7

#define DATA 8

long wait = 1000; // Display time in millis

int Number = 8888; // Initial 7 segment LED check value -

byte x = 0; // LED code

// LED block byte codes for symbols 0 to 9

// { 0 1 2 3 4 5 6 7 8 9 }

const byte DigitCode[] = {192, 249, 164, 176, 153, 146, 130, 248, 128, 152};

// Byte pointers to select blocks 0 to 3 - 0 is Most Significant Block

const byte Block[] = {241, 242, 244, 248};

//LED Pins

const byte LEDs[] = {13, 12, 11, 10}; // Pin # of the single LEDs

void setup()

{

 Serial.begin(9600);

 irrecv.enableIRIn(); // Start the receiver

 // Set pin modes for shift register control

 pinMode(LATCH,OUTPUT);

 pinMode(CLK,OUTPUT);

 pinMode(DATA,OUTPUT);

 show(); // 7-seg LED test "8888"

 blank(); // Clear 7-seg LEDs

 // Set up single LEDs

 for (int i = 0; i <= 3; i++) {pinMode(LEDs[i], OUTPUT);}

 x=0;

 LEDset(); // Clear LEDs on Pins 10-13

}

void loop() // ===================== Main Loop ==============================

{

 if (irrecv.decode(&results))

 {

 //Scorpio NEC Mini with Arrows & Digits 0-9

 if((results.value) == 0xFF9867){Number = 0; show(); Serial.println(" - 0 -");}

 if((results.value) == 0xFFA25D){Number = 1; show(); Serial.println(" - 1 -");}

 if((results.value) == 0xFF629D){Number = 2; show(); Serial.println(" - 2 -");}

 if((results.value) == 0xFFE21D){Number = 3; show(); Serial.println(" - 3 -");}

 if((results.value) == 0xFF22DD){Number = 4; show(); Serial.println(" - 4 -");}

 if((results.value) == 0xFF02FD){Number = 5; show(); Serial.println(" - 5 -");}

 if((results.value) == 0xFFC23D){Number = 6; show(); Serial.println(" - 6 -");}

 if((results.value) == 0xFFE01F){Number = 7; show(); Serial.println(" - 7 -");}

 if((results.value) == 0xFFA857){Number = 8; show(); Serial.println(" - 8 -");}

 if((results.value) == 0xFF906F){Number = 9; show(); Serial.println(" - 9 -");}

 if((results.value) == 0xFF6897){Number = 100; show(); Serial.println("- Asterisk Key- ");}

 if((results.value) == 0xFFB04F){Number = 200; show(); Serial.println(" - Hash Key -");}

 if((results.value) == 0xFF4AB5){x=1; LEDupdate(); Serial.println(" - Down Arrow - ");}

 if((results.value) == 0xFF38C7){x=2; LEDupdate(); Serial.println(" - OK Button -");}

 if((results.value) == 0xFF18E7){x=3; LEDupdate(); Serial.println(" - Up Arrow - ");}

 if((results.value) == 0xFF10EF){x=4; LEDupdate(); Serial.println(" - Left Arrow - ");}

 if((results.value) == 0xFF5AA5){x=5; LEDupdate(); Serial.println(" - Right Arrow - ");}

 //if((results.value) == 0xFFC23D){x=6; LEDupdate(); Serial.println(" - >|| - Play/Pause - ");}

 //if((results.value) == 0xFFE01F){x=7; LEDupdate(); Serial.println(" - VOL- - ");}

 //if((results.value) == 0xFFA857){x=8; LEDupdate(); Serial.println(" - VOL+ - ");}

 //if((results.value) == 0xFF906F){x=9; LEDupdate(); Serial.println(" - EQ -");}

 irrecv.resume(); // Receive the next value

 }

}

// Write a decimal number between 0 and 9999 to the 7-Segment display without leading zeros

void WriteNumber()

{

 if (Number > 999) {Send(0, Number / 1000);} // MSB - Blank leading zero

 if (Number > 99) {Send(1, (Number / 100) % 10);}

 if (Number > 9) {Send(2, (Number / 10) % 10);}

 Send(3, Number % 10); // LSB - remains lit unless cleared with blank()

}

void Send(byte BlockIndex, byte Value)

// Send a symbol between 0 and 9 to one of the 4 LED blocks of the display

{

 digitalWrite(LATCH, LOW);

 shiftOut(DATA, CLK, MSBFIRST, DigitCode[Value]);

 shiftOut(DATA, CLK, MSBFIRST, Block[BlockIndex] );

 digitalWrite(LATCH, HIGH);

}

void blank()

// Clears LS block of 7-segment display

{

 digitalWrite(LATCH, LOW);

 shiftOut(DATA, CLK, MSBFIRST, 255); // 255 = all segments OFF

 shiftOut(DATA, CLK, MSBFIRST, 3 );

 digitalWrite(LATCH, HIGH);

}

void show()

// Number to 7-segment display

{

 long timenow = millis();

 while (millis() < timenow + wait){WriteNumber();} // Display Number - very fast loop

 blank();

}

void LEDset()

// Display to single LEDs

{

 digitalWrite(LEDs[0], !(x & 1)); // 0 = ON, 1 = OFF

 digitalWrite(LEDs[1], !(x & 2));

 digitalWrite(LEDs[2], !(x & 4));

 digitalWrite(LEDs[3], !(x & 8));

}

void LEDupdate()

// Display x in binary on single LEDs

{

 special();

 LEDset();

 delay(wait);

 x=0; // All 4 LEDs OFF

 LEDset();

 blank();

}

void special()

// Write "L" to 7-segment display LSB to indicate data is in single LEDs in binary

{

 digitalWrite(LATCH, LOW);

 shiftOut(DATA, CLK, MSBFIRST, 199); // 199 = "L"

 shiftOut(DATA, CLK, MSBFIRST, Block[3]);

 digitalWrite(LATCH, HIGH);

}