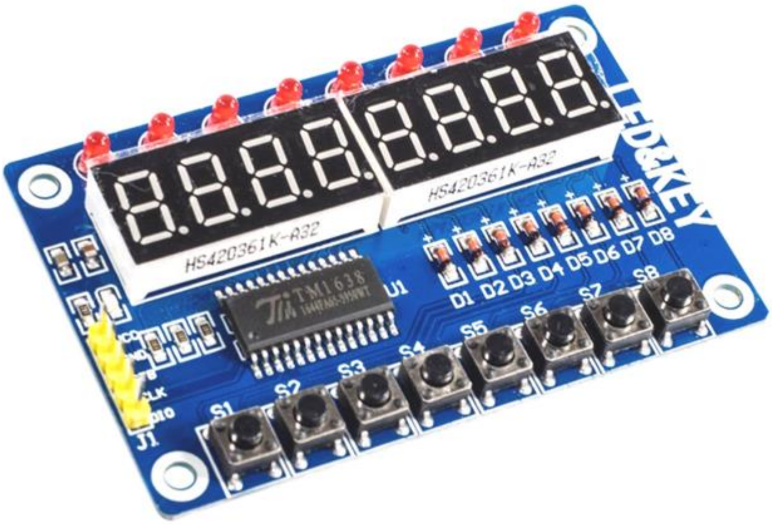
Handson Technology

User Guide

TM1638 7 Segment Display Keypad & LED Module

This 8 digit seven segment display module uses a TM6138 controller allowing full control of the display using just 3 digital pins on a micro-controller. In addition to the seven segment display there are 8 individually controllable 3mm LEDs and a keypad with 8 push buttons arranged in a single row. These can also be controlled through the TM6138 IC and so require no extra digital pins. A standard 5 pin header provides easy interface to the module from microcontroller.



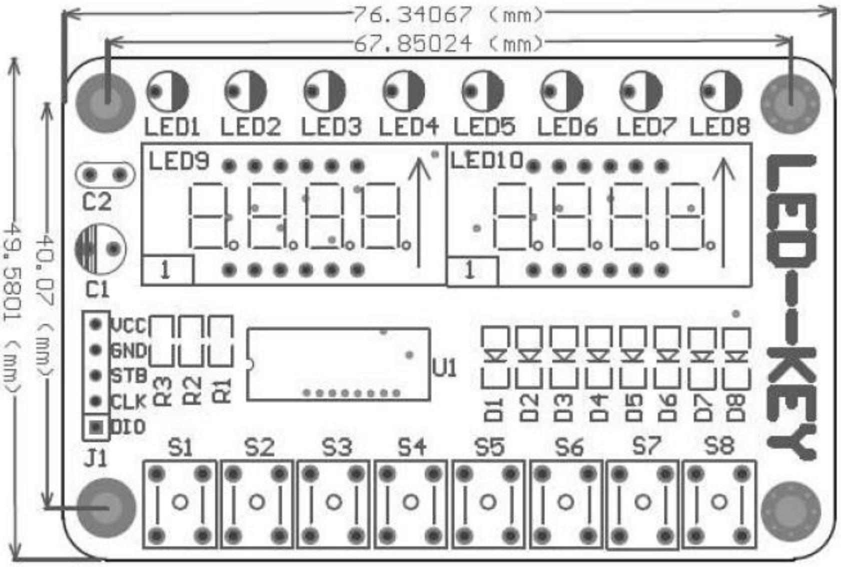
SKU: MDU1093

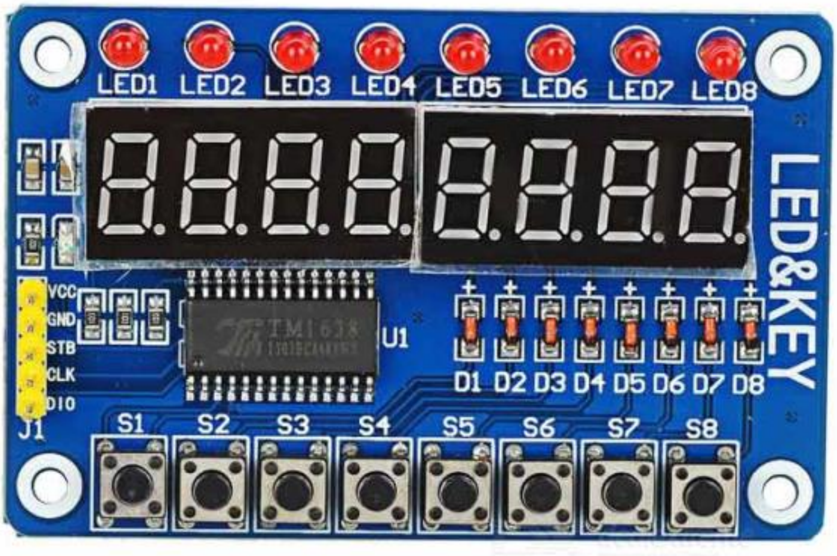
Brief Data:

- Display Type: 8-Digit 7-Segment.
- LED: 8 Red Color 3mm.
- Key: 8 Tactile Switches.
- Supply Voltage: 5V
- Controller: TM1638.
- Brightness Control on Segment.
- Interface: 3 Lines Serial (CLK, STB, DIO).
- Dimensions: 76mm x 50mm.

Mechanical Dimension:

Unit: mm

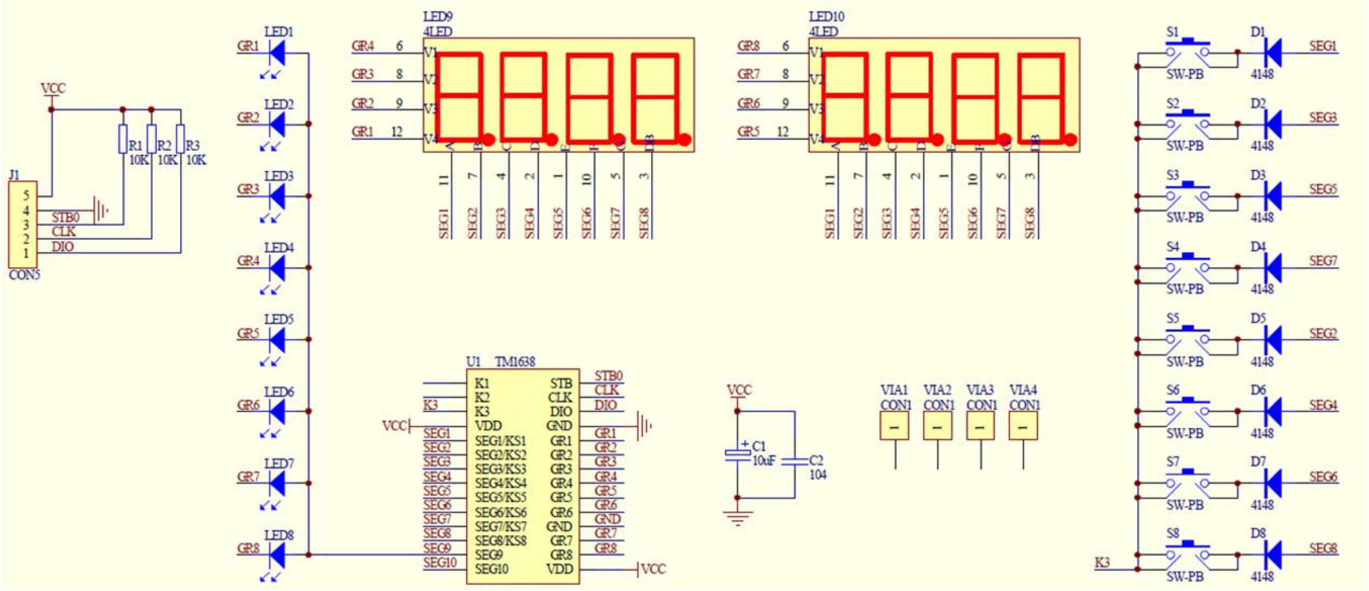




Pin Assignment:

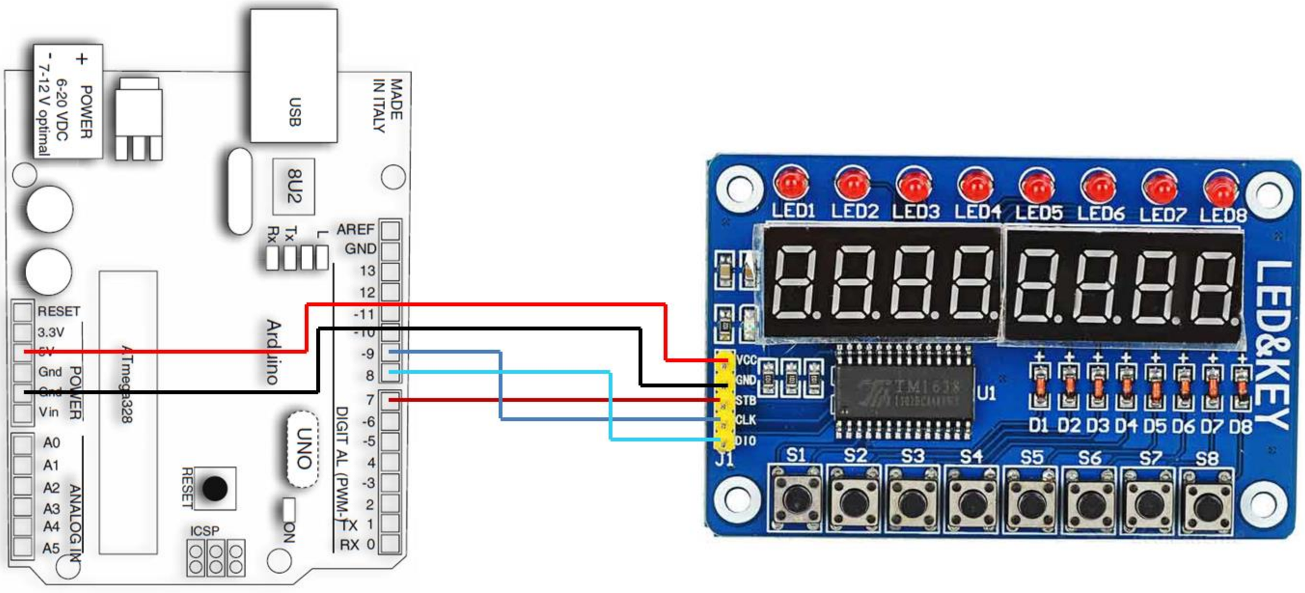
Pin Name	Function
VCC	5~Vdc
GND	Ground
STB	Chip Select Input.
CLK	Clock Input
DIO	Data Input/output

Schematic Diagram:



Application Example with Arduino:

Connect the TM1638 module to Arduino Uno as the below schematic:



Arduino Sketch:

Copy the following Arduino Sketch into Arduino IDE and upload to Arduino Uno board:

```
/*==========================================================================================
//  Author      : Handson Technology
//  Project     : Arduino Uno with TM1638 Display+Keypad Module
//  Description : Display message on 8-digit 7-Segments, switch On LED with each key
press.
//  Source-Code : TM1638_2.ino
// =========================================================================================
*/
```

```
const int strobe = 7;
const int clock = 9;
const int data = 8;

void sendCommand(uint8_t value)
{
  digitalWrite(strobe, LOW);
  shiftOut(data, clock, LSBFIRST, value);
  digitalWrite(strobe, HIGH);
}

void reset()
{
  sendCommand(0x40); // set auto increment mode
  digitalWrite(strobe, LOW);
  shiftOut(data, clock, LSBFIRST, 0xc0);   // set starting address to 0
  for(uint8_t i = 0; i < 16; i++)
  {
    shiftOut(data, clock, LSBFIRST, 0x00);
  }
  digitalWrite(strobe, HIGH);
}

void setup()
{
  pinMode(strobe, OUTPUT);
  pinMode(clock, OUTPUT);
  pinMode(data, OUTPUT);

  sendCommand(0x8f);  // activate
  reset();
}


#define COUNTING_MODE 0
#define SCROLL_MODE 1
#define BUTTON_MODE 2

void loop()
{
  static uint8_t mode = COUNTING_MODE;

  switch(mode)
  {
  case COUNTING_MODE:
    mode += counting();
    break;
  case SCROLL_MODE:
    mode += scroll();
    break;
  case BUTTON_MODE:
    buttons();
    break;
  }

  delay(200);
}

bool counting()
{
                       /*0*/ /*1*/ /*2*/ /*3*/ /*4*/ /*5*/ /*6*/ /*7*/ /*8*/ /*9*/
  uint8_t digits[] = { 0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d, 0x7d, 0x07, 0x7f, 0x6f };

  static uint8_t digit = 0;
  sendCommand(0x40);
  digitalWrite(strobe, LOW);
  shiftOut(data, clock, LSBFIRST, 0xc0);
  for(uint8_t position = 0; position < 8; position++)
  {
    shiftOut(data, clock, LSBFIRST, digits[digit]);
    shiftOut(data, clock, LSBFIRST, 0x00);
  }

  digitalWrite(strobe, HIGH);

  digit = ++digit % 10;
  return digit == 0;
}

bool scroll()
{
  uint8_t scrollText[] =
  {
    /* */ /* */ /* */ /* */ /* */ /* */ /* */ /* */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    /*H*/ /*E*/ /*L*/ /*L*/ /*O*/ /*.*/ /*.*/ /*.*/
    0x76, 0x79, 0x38, 0x38, 0x3f, 0x80, 0x80, 0x80,
    /* */ /* */ /* */ /* */ /* */ /* */ /* */ /* */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    /*H*/ /*E*/ /*L*/ /*L*/ /*O*/ /*.*/ /*.*/ /*.*/
    0x76, 0x79, 0x38, 0x38, 0x3f, 0x80, 0x80, 0x80,
  };

  static uint8_t index = 0;
  uint8_t scrollLength = sizeof(scrollText);

  sendCommand(0x40);
  digitalWrite(strobe, LOW);
  shiftOut(data, clock, LSBFIRST, 0xc0);

  for(int i = 0; i < 8; i++)
  {
    uint8_t c = scrollText[(index + i) % scrollLength];

    shiftOut(data, clock, LSBFIRST, c);
    shiftOut(data, clock, LSBFIRST, c != 0 ? 1 : 0);
  }

  digitalWrite(strobe, HIGH);

  index = ++index % (scrollLength << 1);

  return index == 0;
}

void buttons()
{
  uint8_t promptText[] =
  {
    /*P*/ /*r*/ /*E*/ /*S*/ /*S*/ /* */ /* */ /* */
    0x73, 0x50, 0x79, 0x6d, 0x6d, 0x00, 0x00, 0x00,
    /* */ /* */ /* */ /* */ /* */ /* */ /* */ /* */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    /*b*/ /*u*/ /*t*/ /*t*/ /*o*/ /*n*/ /*S*/ /* */
    0x7c, 0x1c, 0x78, 0x78, 0x5c, 0x54, 0x6d, 0x00,
    /* */ /* */ /* */ /* */ /* */ /* */ /* */ /* */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  };
  static uint8_t block = 0;

  uint8_t textStartPos = (block / 4) << 3;
  for(uint8_t position = 0; position < 8; position++)
  {
    sendCommand(0x44);
    digitalWrite(strobe, LOW);
    shiftOut(data, clock, LSBFIRST, 0xC0 + (position << 1));
    shiftOut(data, clock, LSBFIRST, promptText[textStartPos + position]);
    digitalWrite(strobe, HIGH);
  }

  block = (block + 1) % 16;

  uint8_t buttons = readButtons();

  for(uint8_t position = 0; position < 8; position++)
  {
    uint8_t mask = 0x1 << position;

    setLed(buttons & mask ? 1 : 0, position);
  }
}

uint8_t readButtons(void)
{
  uint8_t buttons = 0;
  digitalWrite(strobe, LOW);
  shiftOut(data, clock, LSBFIRST, 0x42);

  pinMode(data, INPUT);

  for (uint8_t i = 0; i < 4; i++)
  {
    uint8_t v = shiftIn(data, clock, LSBFIRST) << i;
    buttons |= v;
  }

  pinMode(data, OUTPUT);
  digitalWrite(strobe, HIGH);
  return buttons;
}

void setLed(uint8_t value, uint8_t position)
{
  pinMode(data, OUTPUT);

  sendCommand(0x44);
  digitalWrite(strobe, LOW);
  shiftOut(data, clock, LSBFIRST, 0xC1 + (position << 1));
  shiftOut(data, clock, LSBFIRST, value);
  digitalWrite(strobe, HIGH);
}
```

Once successful uploaded, observe the message display on the 7-Segment display and the LEDs. Try pressing any one of the key switch and observe the correspond LED being lighted.

Alternatively, you can you the ready available arduino library for TM1638 to simplify the driving of this display+key board. Download the library from the below link and extract it to Arduino IDE Library folder.

https://github.com/rjbatista/tm1638-library

Try to upload some examples on this library and observe the display.

HandsOn Technology provides a multimedia and interactive platform for everyone interested in electronics. From beginner to diehard, from student to lecturer. Information, education, inspiration and entertainment. Analog and digital, practical and theoretical; software and hardware.

open source hardware

Hands*On* Technology support Open Source Hardware (OSHW) Development Platform.

Learn : Design : Share

www.handsontec.com

The Face behind our product quality…

In a world of constant change and continuous technological development, a new or replacement product is never far away – and they all need to be tested.

Many vendors simply import and sell wihtout checks and this cannot be the ultimate interests of anyone, particularly the customer. Every part sell on Handsotec is fully tested. So when buying from Handsontec products range, you can be confident you're getting outstanding quality and value.

We keep adding the new parts so that you can get rolling on your next project.

Breakout Boards & Modules

Connectors

Electro-Mechanical Parts

Engineering Material

Mechanical Hardware

P

Electronics Components

Power Supply

Arduino Board & Shield

Tools & Accessory