Extended I/O with 8 Digit LED Display

Contents

[Bill of Materials 2](#_Toc451529265)

[Schematic 4](#_Toc451529266)

[Assembly 5](#_Toc451529267)

[1 – Resistors 5](#_Toc451529268)

[2 – Capacitors 5](#_Toc451529269)

[3 – Jumper and IO Headers 6](#_Toc451529270)

[3 – LED Display and LEDS 7](#_Toc451529271)

[3 – Transistors 8](#_Toc451529272)

[4 – IC 8](#_Toc451529273)

[9 – Arduino Connectors 8](#_Toc451529274)

[Configuration 9](#_Toc451529275)

[Jumpers 9](#_Toc451529276)

[Testing 10](#_Toc451529277)

[Sample Code 10](#_Toc451529278)

[M5451LED Library 15](#_Toc451529279)

[Programming 16](#_Toc451529280)

[I/O information 16](#_Toc451529281)

[Pin Outs 17](#_Toc451529282)

[Exrentded I/O 17](#_Toc451529283)

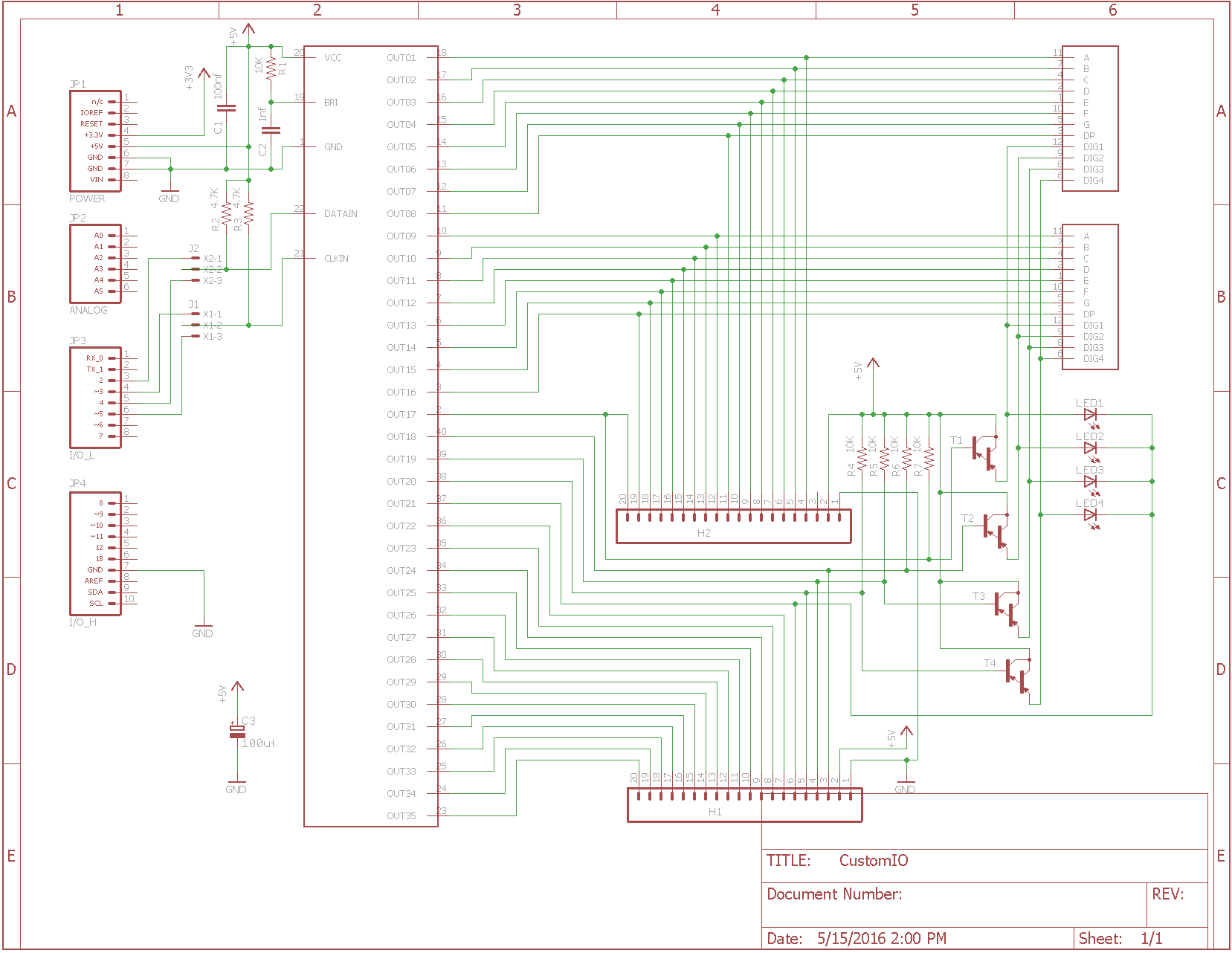
[Hardware Extension 17](#_Toc451529284)

[M5451 18](#_Toc451529285)

[7 Segment Display 18](#_Toc451529286)

## Bill of Materials

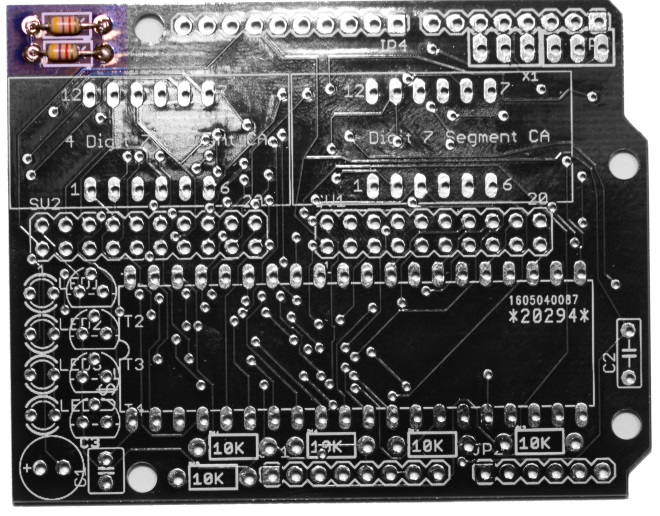
|  |  |  |  |
| --- | --- | --- | --- |
| C1 | 100nf (0.1) capacitor  (104 marking) | 1 |  |
| C2 | 1nf (0.001 capacitor  (102 marking) | 1 |  |
| C3 | 100uf Electrolytic capacitor | 1 |  |
| JP1,JP3 | 8 Position 8 x 1 .1 (2.54mm) Tall Header Arduino Connector | 2 | 2144631 |
| JP2 | 6 Position 6 x 1 .1 (2.54mm) Tall Header Arduino Connector | 1 | 2144614 |
| JP4 | 10 Position 10 x 1 .1 (2.54mm) Tall Header Arduino Connector | 1 | 2177627 |
| LED1 – LED4 | 3MM LED | 4 |  |
| R8,R9 | 4.7K 1/8 W  (picture for reference only)  (Yellow – Violet – Red) 4 band  (Yellow Violet Black Brown) 5 Band | 2 | Image result for 4.7k 1/8 w |
| R10 – R14 | 10K 1/8 W  (picture for reference only)  (Brown-Black-Orange) 4 Band  (Brown-Black-Black-Red)5 Band | 5 | Image result for 4.7k 1/8 w |
| SV1-SV2 | 20 Position, 10x 2 Header | 2 | Molex 10-89-7061 |
| T1 – T4 | 2N2907 PNP Transistor | 4 | Image result for 2n2907A |
| U1 | M5451 PDIP 40pin Display Controller | 1 |  |
| U2 -U3 | LDS-3461B 4 Digit 7 Segment Display | 2 |  |
| X1,X2 | 3 Position x 1 Header | 2 | TSW-103-07-L-S Samtec Inc. | SAM1031-03-ND DigiKey Electronics |
| Jumpers | 2 Position x 1 row .1 (2.54mm) Shunt | 2 | 2018503 |
| PCB |  | 1 |  |

Schematic

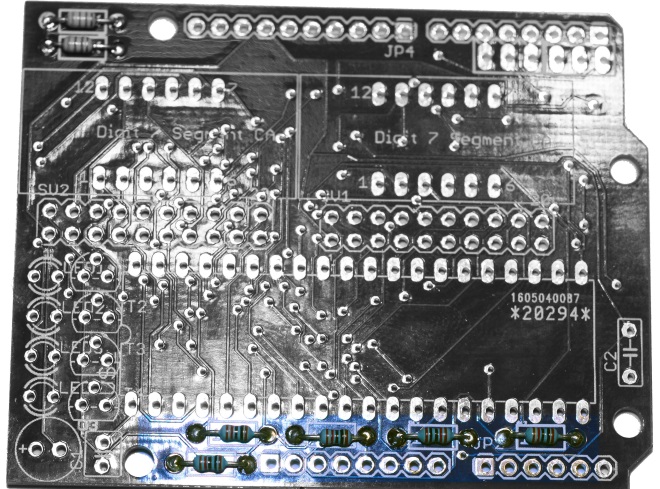
## Assembly

### 1 – Resistors

Solder in place the 4.7K resistors.

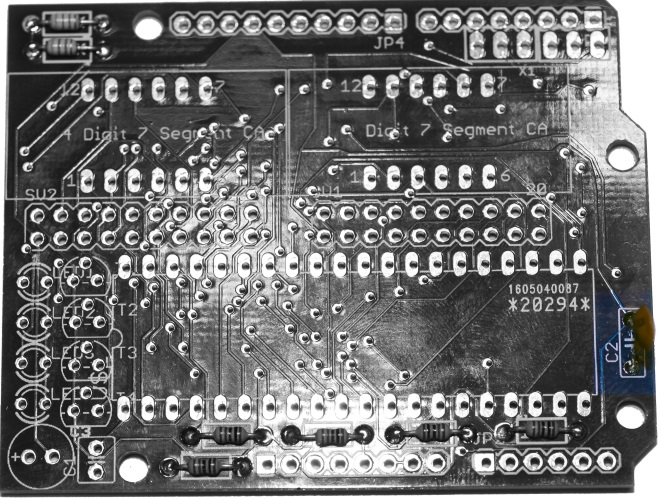
****

Solder in place the 10K resistors.

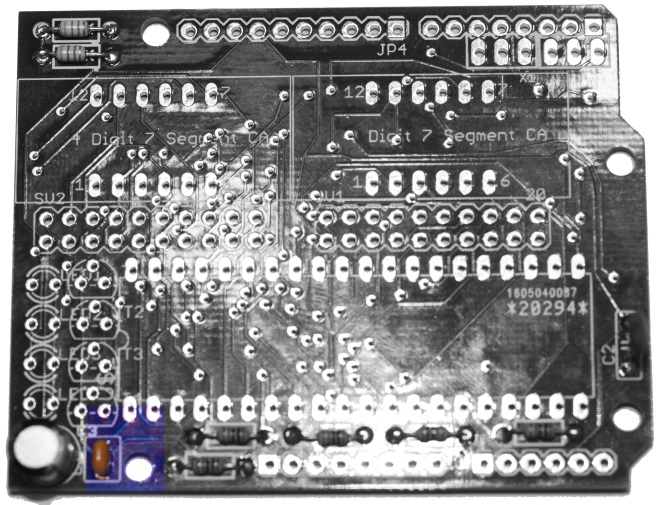


### 2 – Capacitors

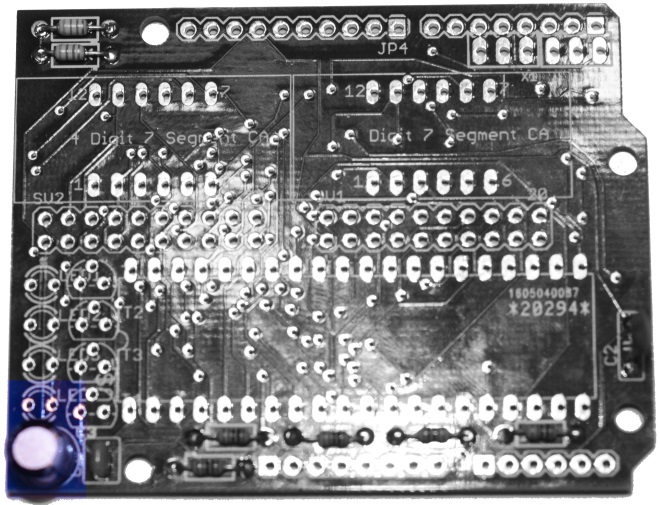
Solder in place the 100nf (.001f) Cermic Disc Capacitor.

****

Solder in place the 0.1f Capicator

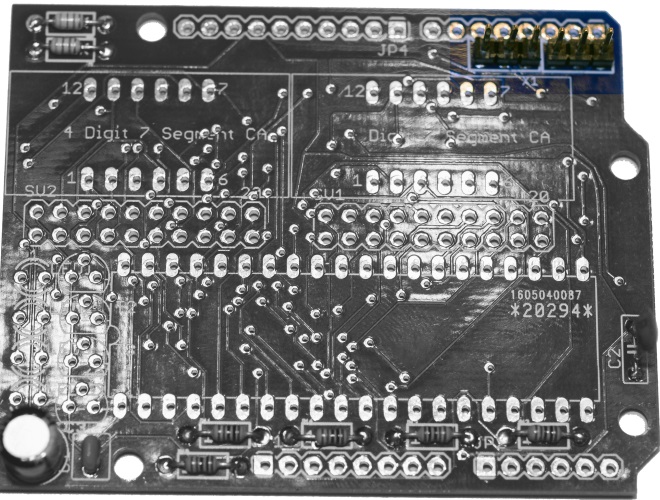


Solder in place the 100f 6.3V Electrolytic Capacitor. **It is important to note the polarity.** The Capacitor is marked for the **Negative lead.**

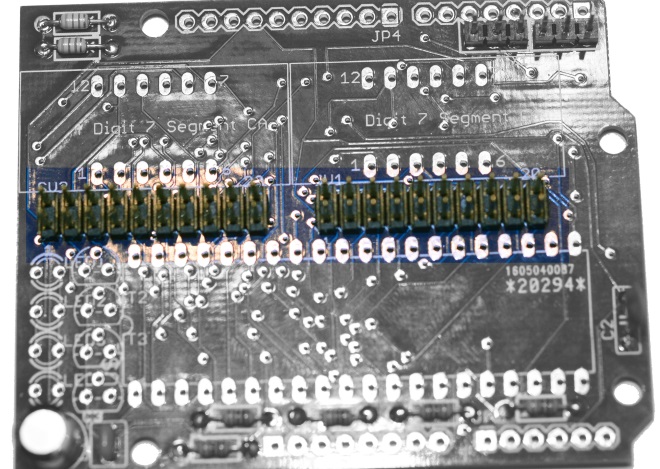
****

### 3 – Jumper and IO Headers

Solder in the 3 position by 1 row headers.

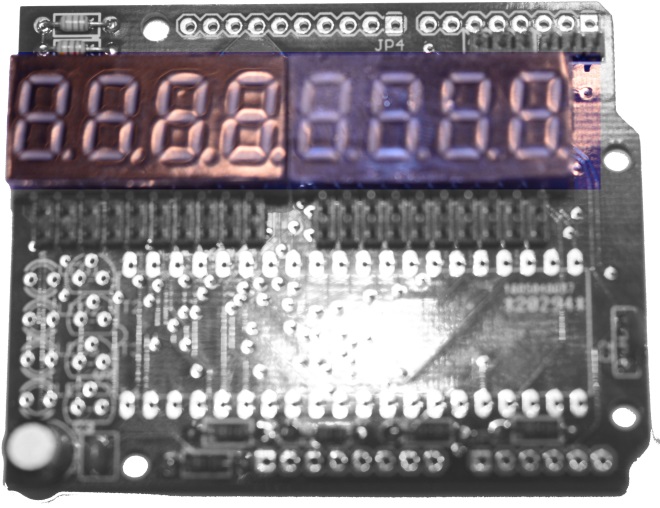


Solder in the 20 position (10 x2) Headers.

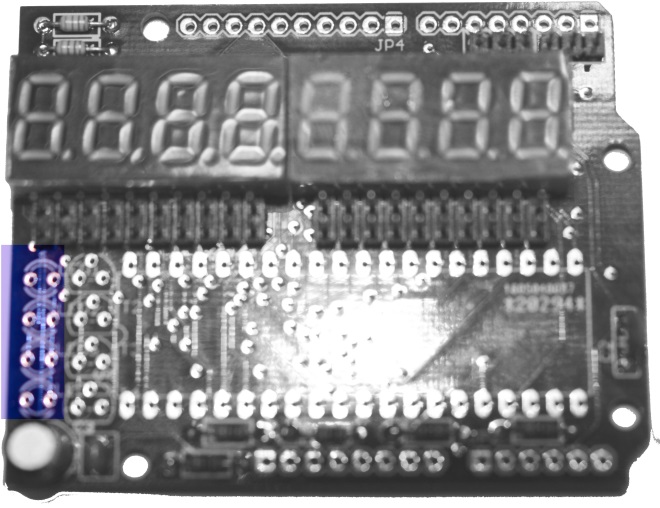


### 3 – LED Display and LEDS

Solder in the LED 7 Segment Displays

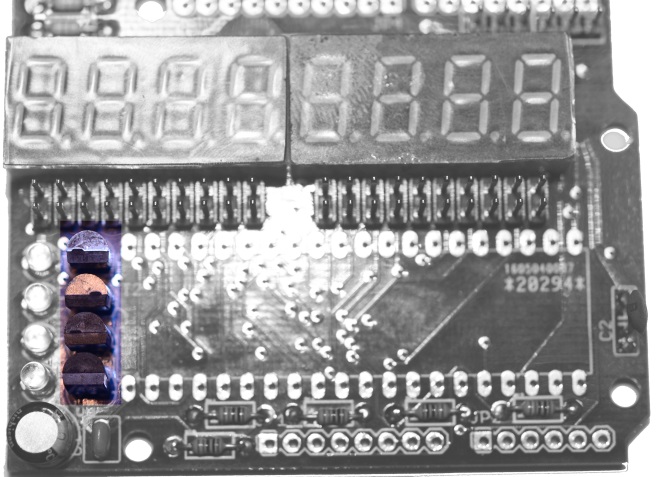


Solder in the LEDS. **It is important to note the polarity. The Flat side is towards the middle of the board or the long lead is towards the edge of the board.**



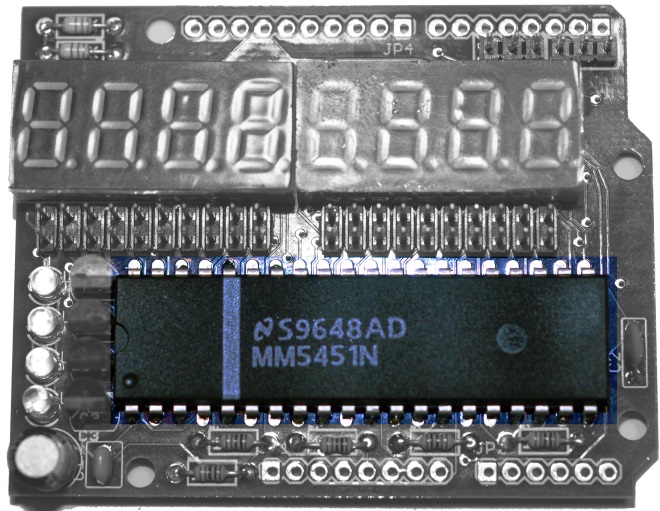
### 3 – Transistors

Solder in the Transistors.



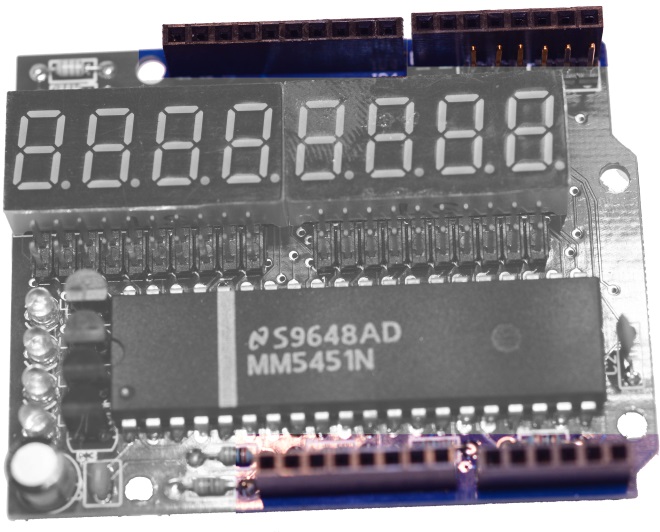
### 4 – IC

Solder in the IC. **Pin 1 is towards the Transistors. (notch/dimple) Be careful as the pins are delicate.**



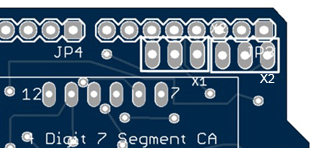
### 9 – Arduino Connectors

Solder in the Arduino Headers.



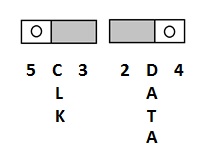
## Configuration

### Jumpers



|  |  |  |
| --- | --- | --- |
| X1 – M5451 Clock to Arduino Data Pin Selection | Arduino Data Pin 3 |  |
| X1 – M5451 Clock to Arduino Data Pin Selection | Arduino Data Pin 5 |  |
| X2 – M5451 Data to Arduino Data Pin Selection | Arduino Data Pin 4 |  |
| X2 – M5451 Data to Arduino Data Pin Selection | Arduino Data Pin 2 |  |

**Note – For running the Sample Programs, set the Jumpers for Arduino Pins 2 and 3.**

****

## Testing

### Sample Code

A Library is available on . It includes two example programs.

The First Example demonstrates working with the shield to update the display within the Main loop on an Arduino Sketch.

#include <M5451LED.h>

//clock pin is jumper selectable between 3 or 5

int myClockPin = 3; // Arduino pin that goes to M5451 clock

//data pin is jumper selectable between 2 or 4

int mySerDataPin = 2; // Arduino pin that goes to M5451 data

M5451 board(myClockPin,mySerDataPin);

void UpdateDisplay()

{

unsigned long int j;

for(byte dig = 0; dig < 4; dig++)

{

j = 0x00010000;

j = j << dig; //set the set of digits

j = j | ((unsigned long int)board.digitCodes[dig+4]) << 8; //shift digit segments into position

j = j | board.digitCodes[dig]; //or in other digit segments

board.setiopins(j,board.ExtendedIOPins); //load the shift register

//delay(10);

}

}

void loop() // run over and over again

{

//Sample code

//negative number check //or read a value, temp, sensor etc. upyo you

board.setNumber(102499,2);

for(int c = 0; c < 4000; c++)

{

UpdateDisplay();

}

board.BlankDisplay();

board.setNumber(-2499,2);

for(int c = 0; c < 4000; c++)

{

UpdateDisplay();

}

board.BlankDisplay();

board.setNumber(100000,2);

for(int c = 0; c < 4000; c++)

{

UpdateDisplay();

}

board.BlankDisplay();

byte dp = 0;

// count up to 499

// deminstrates changing the displayed value which is

// then displayed by the Interrupt service routine

for (unsigned long int cnt=0;cnt < 500;cnt++)

{

board.ExtendedIOPins = board.ExtendedIOPins << 1;

if ((board.ExtendedIOPins == 0x0000)||(board.ExtendedIOPins == 0x8000))

{

board.ExtendedIOPins = 0x0001;

}

board.setNumber(cnt,0); //cnt, decimal places

for(int c = 0; c < 250; c++)

{

UpdateDisplay();

}

}

//End Sample Code

//more code goes here...

}

//runs a circle around each digit (segments a,b,c,d,e,f)

//an extended IO pin is held constant while each segment is is toggled

//deminstrates segments and extended I/O updated at different timings

//the timer interrupt must be disabled while running loops

void littlechaser()

{

unsigned long int val;

board.ExtendedIOPins = 0x0001;

for(byte y = 0; y <15; y++) // outer loop is when the extended IO is updated

{

val = 0x000F0101; //All Digits ON, start with segment a

for(byte segs = 0; segs < 6; segs ++) //inner loop is when the segments are updated

{

board.setiopins(val,board.ExtendedIOPins);

val = val<<1L;

val = val & 0x0000FFFF; //keep segments

val = val | 0x000F0000; //all digits on

delay(75);

}

board.ExtendedIOPins = board.ExtendedIOPins << 1;

}

}

//runs a circle around all the digits

//extended IO pins are changed with each segment

//the timer interrupt must be disabled while running chaser

void bigchaser()

{

static const unsigned int ExtIOPattern[] = { 0xAAAA, 0x5555 };

static const unsigned int long bigchaserMap[] = {

0x00010001, // 000 00000000 00000001 00000000 00000001 // digit 1 segment a

0x00010020, // 000 00000000 00000001 00000000 00100000 // digit 1 segment f

0x00010010, // 000 00000000 00000001 00000000 00010000 // digit 1 segment e

0x00010008, // 000 00000000 00000001 00000000 00001000 // digit 1 segment d

0x00020008, // 000 00000000 00000010 00000000 00001000 // digit 2 segment d

0x00040008, // 000 00000000 00000100 00000000 00001000 // digit 3 segment d

0x00080008, // 000 00000000 00001000 00000000 00001000 // digit 4 segment d

0x00010800, // 000 00000000 00000001 00001000 00000000 // digit 5 segment d

0x00020800, // 000 00000000 00000010 00001000 00000000 // digit 6 segment d

0x00040800, // 000 00000000 00000100 00001000 00000000 // digit 7 segment d

0x00080800, // 000 00000000 00001000 00001000 00000000 // digit 8 segment d

0x00080400, // 000 00000000 00001000 00000100 00000000 // digit 8 segment c

0x00080200, // 000 00000000 00001000 00000010 00000000 // digit 8 segment b

0x00080100, // 000 00000000 00001000 00000001 00000000 // digit 8 segment a

0x00040100, // 000 00000000 00000100 00000001 00000000 // digit 7 segment a

0x00020100, // 000 00000000 00000010 00000001 00000000 // dugit 6 segment a

0x00010100, // 000 00000000 00000001 00000001 00000000 // digit 5 segment a

0x00080001, // 000 00000000 00001000 00000000 00000001 // digit 4 segment a

0x00040001, // 000 00000000 00000100 00000000 00000001 // dugit 3 segment a

0x00020001};// 000 00000000 00000010 00000000 00000001 // digit 2 segment a

byte co, c = 0;

for (co=0;co < 6;co++)

{

for(c = 0; c< 20;c++)

{

board.setiopins(bigchaserMap[c],ExtIOPattern[c & 0x01]);

delay(75);

}

}

for (co=0;co < 6;co++)

{

for(c = 20; c>0;c--)

{

board.setiopins(bigchaserMap[c-1],ExtIOPattern[(c-1) & 0x01]);

delay(75);

}

}

}

void chkLEDS()

{

unsigned long int j;

for (byte c = 0; c < 5; c++)

{

j = 0x00110000;

board.setiopins(j,0);

delay(75);

j = 0x00120000;

board.setiopins(j,0);

delay(75);

j = 0x00140000;

board.setiopins(j,0);

delay(75);

j = 0x00180000;

board.setiopins(j,0);

delay(75);

}

}

void setup() // run once, when the sketch starts

{

littlechaser();

bigchaser();

chkLEDS();

}

The second example demonstrates using an Interrupt Service Routine to update the display.

#include <TimerOne.h>

#include <M5451LED.h>

//clock pin is jumper selectable between 3 or 5

int myClockPin = 3; // Arduino pin that goes to M5451 clock

//data pin is jumper selectable between 2 or 4

int mySerDataPin = 2; // Arduino pin that goes to M5451 data

M5451 board(myClockPin,mySerDataPin);

// Every 20mS the values in the DigitCodes array and

// The values in the ExtendedIPPins are written out

void timerIsr()

{

unsigned long int j;

for(byte dig = 0; dig < 4; dig++)

{

j = 0x00010000;

j = j << dig; //set the set of digits

j = j | ((unsigned long int)board.digitCodes[dig+4]) << 8; //shift digit segments into position

j = j | board.digitCodes[dig]; //or in other digit segments

board.setiopins(j,board.ExtendedIOPins); //load the shift register

//delay(10);

}

}

void loop() // run over and over again

{

//Sample code

//negative number check

board.setNumber(102499,2);

delay(750);

board.setNumber(-2499,2);

delay(750);

board.setNumber(100000,2);

delay(750);

board.BlankDisplay();

delay(1);

byte dp = 0;

// count up to 499

// deminstrates changing the displayed value which is

// then displayed by the Interrupt service routine

for (unsigned long int cnt=0;cnt < 500;cnt++)

{

board.ExtendedIOPins = board.ExtendedIOPins << 1;

if ((board.ExtendedIOPins == 0x0000)||(board.ExtendedIOPins == 0x8000))

{

board.ExtendedIOPins = 0x0001;

}

board.setNumber(cnt,0); //cnt, decimal places

delay(150);

}

//End Sample Code

//more code goes here...

}

//runs a circle around each digit (segments a,b,c,d,e,f)

//an extended IO pin is held constant while each segment is is toggled

//deminstrates segments and extended I/O updated at different timings

void littlechaser()

{

unsigned long int val;

board.ExtendedIOPins = 0x0001;

for(byte y = 0; y <15; y++) // outer loop is when the extended IO is updated

{

val = 0x000F0101; //All Digits ON, start with segment a

for(byte segs = 0; segs < 6; segs ++) //inner loop is when the segments are updated

{

board.setiopins(val,board.ExtendedIOPins);

val = val<<1L;

val = val & 0x0000FFFF; //keep segments

val = val | 0x000F0000; //all digits on

delay(75);

}

board.ExtendedIOPins = board.ExtendedIOPins << 1;

}

}

//runs a circle around all the digits

//extended IO pins are changed with each segment

void bigchaser()

{

static const unsigned int ExtIOPattern[] = { 0xAAAA, 0x5555 };

static const unsigned int long bigchaserMap[] = {

0x00010001, // 000 00000000 00000001 00000000 00000001 // digit 1 segment a

0x00010020, // 000 00000000 00000001 00000000 00100000 // digit 1 segment f

0x00010010, // 000 00000000 00000001 00000000 00010000 // digit 1 segment e

0x00010008, // 000 00000000 00000001 00000000 00001000 // digit 1 segment d

0x00020008, // 000 00000000 00000010 00000000 00001000 // digit 2 segment d

0x00040008, // 000 00000000 00000100 00000000 00001000 // digit 3 segment d

0x00080008, // 000 00000000 00001000 00000000 00001000 // digit 4 segment d

0x00010800, // 000 00000000 00000001 00001000 00000000 // digit 5 segment d

0x00020800, // 000 00000000 00000010 00001000 00000000 // digit 6 segment d

0x00040800, // 000 00000000 00000100 00001000 00000000 // digit 7 segment d

0x00080800, // 000 00000000 00001000 00001000 00000000 // digit 8 segment d

0x00080400, // 000 00000000 00001000 00000100 00000000 // digit 8 segment c

0x00080200, // 000 00000000 00001000 00000010 00000000 // digit 8 segment b

0x00080100, // 000 00000000 00001000 00000001 00000000 // digit 8 segment a

0x00040100, // 000 00000000 00000100 00000001 00000000 // digit 7 segment a

0x00020100, // 000 00000000 00000010 00000001 00000000 // dugit 6 segment a

0x00010100, // 000 00000000 00000001 00000001 00000000 // digit 5 segment a

0x00080001, // 000 00000000 00001000 00000000 00000001 // digit 4 segment a

0x00040001, // 000 00000000 00000100 00000000 00000001 // dugit 3 segment a

0x00020001};// 000 00000000 00000010 00000000 00000001 // digit 2 segment a

byte co, c = 0;

for (co=0;co < 6;co++)

{

for(c = 0; c< 20;c++)

{

board.setiopins(bigchaserMap[c],ExtIOPattern[c & 0x01]);

delay(75);

}

}

for (co=0;co < 6;co++)

{

for(c = 20; c>0;c--)

{

board.setiopins(bigchaserMap[c-1],ExtIOPattern[(c-1) & 0x01]);

delay(75);

}

}

}

void chkLEDS()

{

unsigned long int j;

for (byte c = 0; c < 5; c++)

{

j = 0x00110000;

board.setiopins(j,0);

delay(75);

j = 0x00120000;

board.setiopins(j,0);

delay(75);

j = 0x00140000;

board.setiopins(j,0);

delay(75);

j = 0x00180000;

board.setiopins(j,0);

delay(75);

}

}

void setup() // run once, when the sketch starts

{

littlechaser();

bigchaser();

chkLEDS();

Timer1.initialize(300); //play with this value for more or less updates

Timer1.attachInterrupt( timerIsr );

}

Video’s of the running examples can be found at

<https://youtu.be/1ersXG6CpGg>

and

<https://youtu.be/Mj3757OHWhI>

### M5451LED Library

The library serves two main purposes.

* It contains the Code to shift out the data to the M5451 Chip.
* It contains the Code to convert a given number into the data needed to display the correct segments on the display.

## Programming

### I/O information

The M5451 has 35 I/O Lines, within the sample program they are mapped into an unsigned Long int and an unsigned int. This allows separate control of the display and the extended I/O.

Unsigned Long int

0000 0000 0000 0000 0000 0000 0000 0000

|||| |||| |||| |||| |||| |||| |||| ||||

|||| |||| |||| |||| |||| |||| |||| |||--> Segment A 1 Output 1(pin 18)

|||| |||| |||| |||| |||| |||| |||| ||---> Segment B 1 Output 2(pin 17)

|||| |||| |||| |||| |||| |||| |||| |----> Segment C 1 Output 3(pin 16)

|||| |||| |||| |||| |||| |||| |||| -----> Segment D 1 Output 4(pin 15)

|||| |||| |||| |||| |||| |||| |||-------> Segment E 1 Output 5(pin 14)

|||| |||| |||| |||| |||| |||| ||--------> Segment F 1 Output 6(pin 13)

|||| |||| |||| |||| |||| |||| |---------> Segment G 1 Output 7(pin 12)

|||| |||| |||| |||| |||| |||| ----------> DP 1 Output 8(pin 11)

|||| |||| |||| |||| |||| |||------------> Segment A 2 Output 9(pin 10)

|||| |||| |||| |||| |||| ||-------------> Segment B 2 Output 10(pin 9)

|||| |||| |||| |||| |||| |--------------> Segment C 2 Output 11(pin 8)

|||| |||| |||| |||| |||| ---------------> Segment D 2 Output 12(pin 7)

|||| |||| |||| |||| |||-----------------> Segment E 2 Output 13(pin 6)

|||| |||| |||| |||| ||------------------> Segment F 2 Output 14(pin 5)

|||| |||| |||| |||| |-------------------> Segment G 2 Output 15(pin 4)

|||| |||| |||| |||| --------------------> DP 2 Output 16(pin 3)

|||| |||| |||| |||----------------------> Digit 1 Output 17 (pin 2)

|||| |||| |||| ||-----------------------> Digit 2 Output 18 (pin 40)

|||| |||| |||| |------------------------> Digit 3 Output 19 (pin 39)

|||| |||| |||| -------------------------> Digit 4 Output 20 (pin 38)

|||| |||| |||---------------------------> LED Enable Out 21 (pin 37)

------------------------- --------------> Not Used

Unsigned int

0000 0000 0000 0000

|||| |||| |||| ||||

|||| |||| |||| |||---> Output 22 (pin 36)

|||| |||| |||| ||----> Output 23 (pin 35)

|||| |||| |||| |-----> Output 24 (pin 34)

|||| |||| ||||-------> Output 25 (pin 33)

|||| |||| |||--------> Output 26 (pin 32)

|||| |||| ||---------> Output 27 (pin 31)

|||| |||| |----------> Output 28 (pin 30)

|||| |||| -----------> Output 29 (pin 29)

|||| |||-------------> Output 30 (pin 28)

|||| ||--------------> Output 31 (pin 27)

|||| |---------------> Output 32 (pin 26)

|||| ----------------> Output 33 (pin 25)

|||------------------> Output 34 (pin 24)

||-------------------> Output 35 (pin 23)

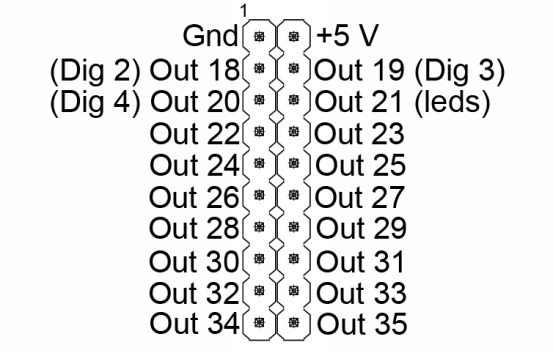
---------------------> Not Used

## Pin Outs

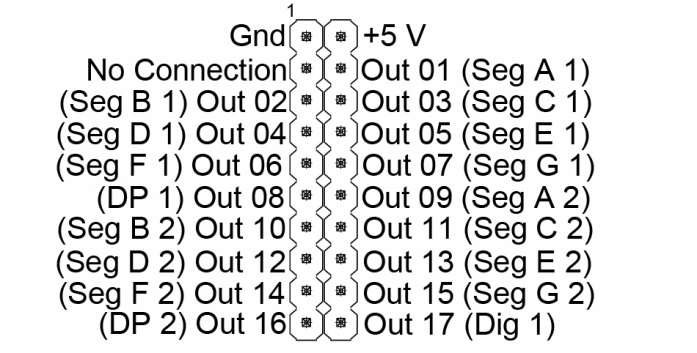
### Exrentded I/O

Headers on the shield make available all of the I/O lines from the M5451. Lines which are used on the shield are identified in parentheses following the M5451 output name. (Note. This is the output, not the M5451 actual Pin)

Header One

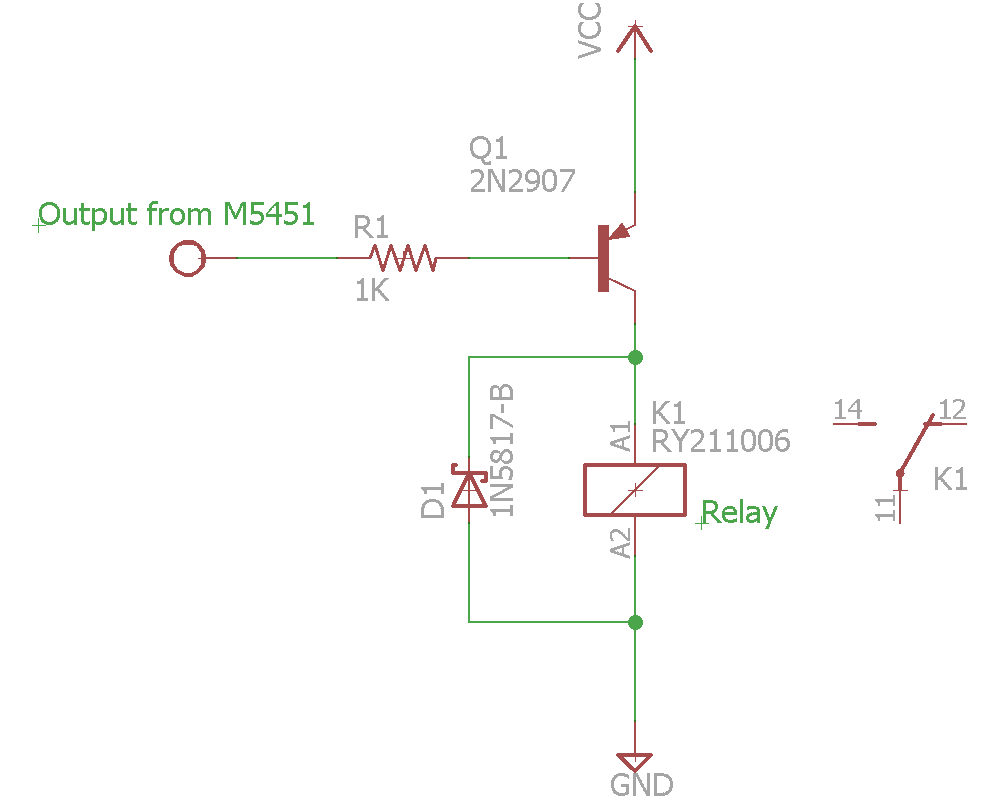


Header Two

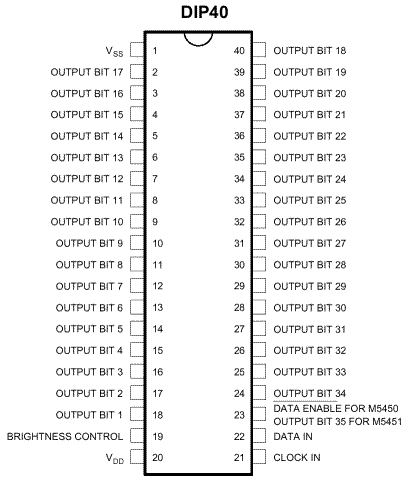


### Hardware Extension

Outputs from the M5451 will sink about 25mA, so transistors should be used for external control of loads such as relays or motors.



### M5451



### 7 Segment Display

Common Anode

