uProc Clock
Assembly Manual

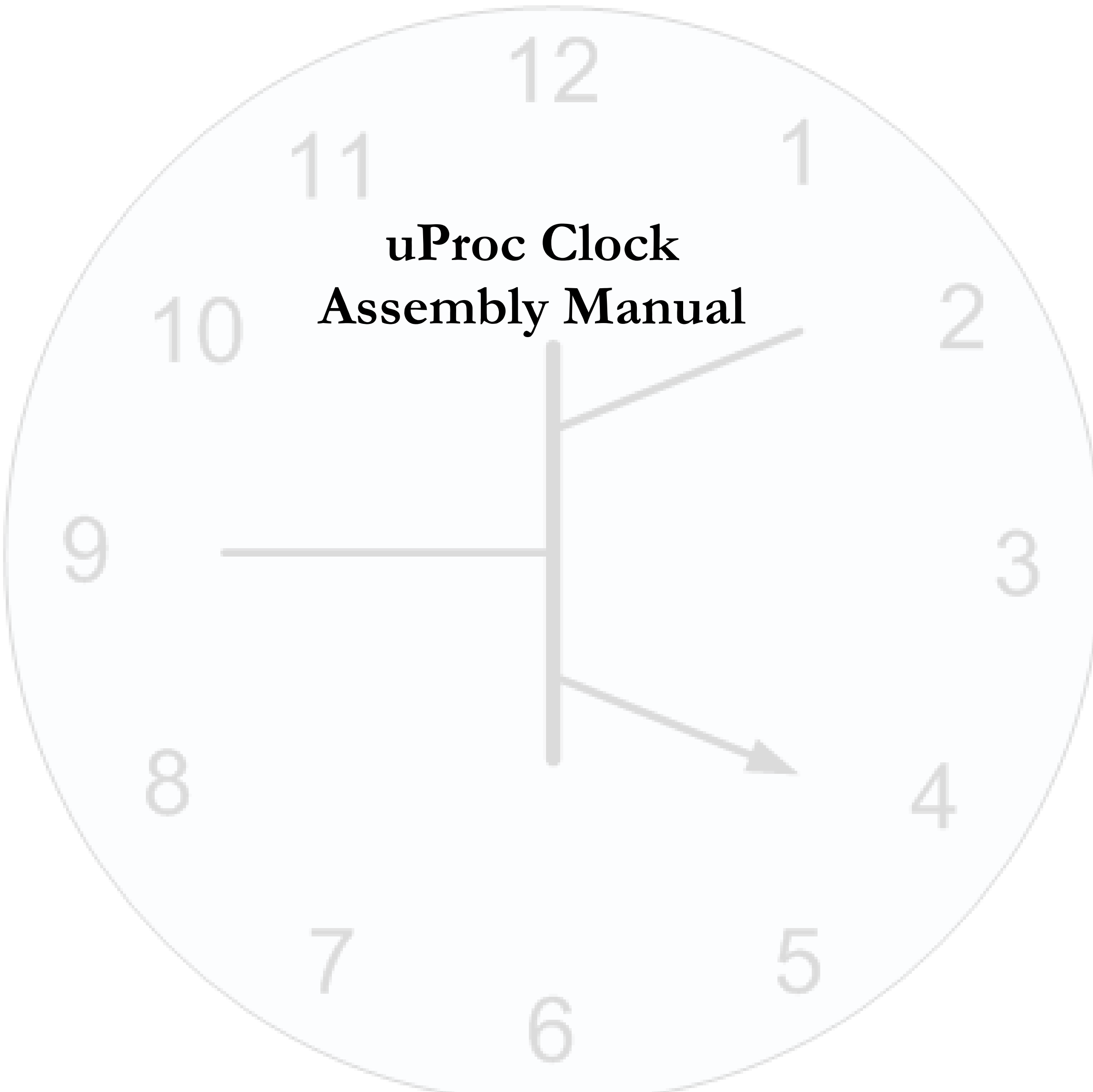


Table of Contents

Document version 1.01 for use with PC board version 2

Introduction

Thanks for buying this kit. I hope you enjoy building it and learn a thing or two. The clock is only in existence to show the difference in complexity and size between it and the other clocks in the series. This clock would not exist without the Transistor Clock and the TTL Clock to contrast against.

This kit is aimed at a somewhat experienced hobbyist; the most challenging part will be soldering the SMT part. All you experienced folks can turn right now to the assembly section and go for it (watch out for the 7-segment LED placement, otherwise, this kit really is as simple as it seems).

If you are not skilled in soldering parts onto a PC board, you should read the soldering section and also search the web to learn about soldering, there are many good sites that teach soldering.

You need to make 261 good solder joints to have the clock work.

If you are not familiar with the components, look carefully at the pictures, but frankly, this kit was designed for a somewhat knowledgeable electronic hobbyist. Be certain of a part before you solder it.

Unpacking/Parts List

Gently unpack the contents of the box. Check off the items below as you unpack. This will also serve as a chance to become familiar with the parts.

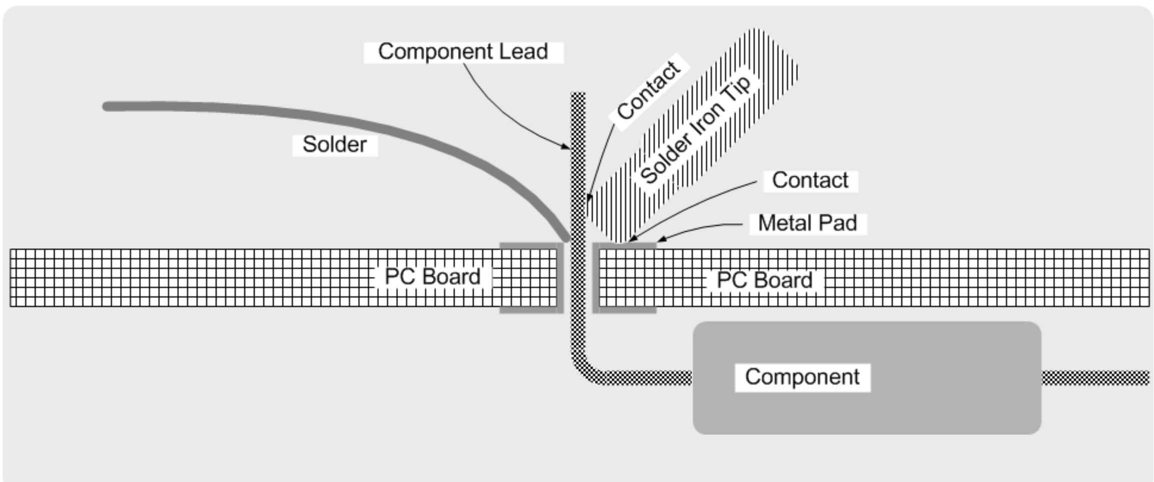
Item	Desc	Qty	Unpacked
	This Manual	1	✔
	0.22 uF Capacitor	2	
	10 uF Capacitor	1	
	0.01 uF Capacitor	5	
	Large Diode	1	
	Red LED	4	
	LDS8164 7 segment display	6	
	header, wire capture	1	
	22K Ohm Resistor	1	
	33K Ohm Resistor	1	
	push switch	3	
	LM7805	1	
	PIC16F84A uProcessor	1	
	DS32KHz TCXO	1	
	SAA1064 Display Driver	3	
	4-40 nut	1	
	4-40 bolt	1	
	18 pin socket	1	
	solder	1	
	Wall transformer	1	
	PC Board	1	

Soldering

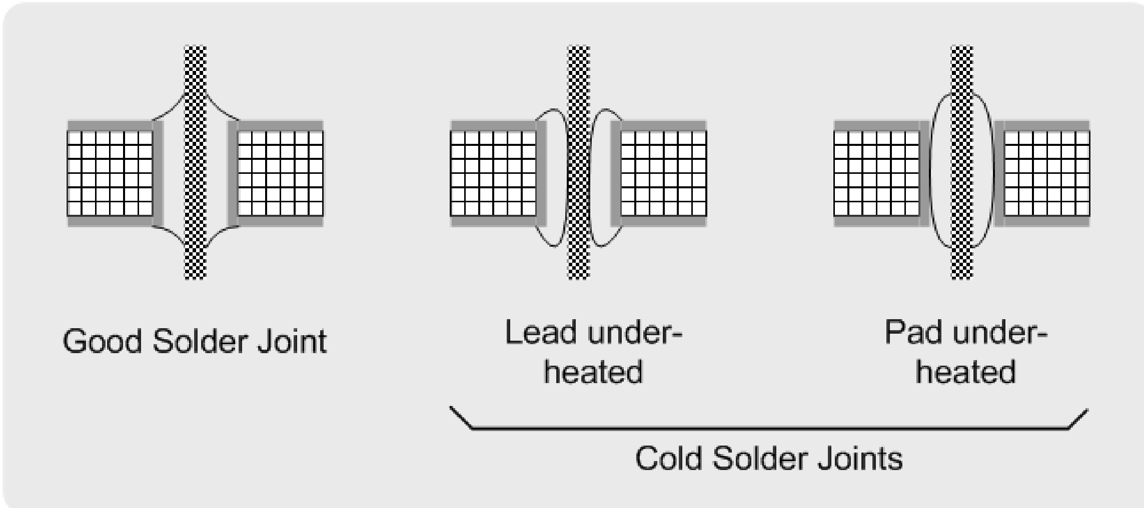
This manual can't teach the art of soldering, but here are the basics. Remember, you need to make over 260 **GOOD** solder joints; each bad joint will be an adventure in troubleshooting.

Wet the tip of the iron with a bit of solder and wipe off the excess solder on the wet sponge occasionally, or when you notice the joint is not heating properly. The small amount of solder left on the tip helps conduct heat to the lead and the pad.

Insert the component leads and press the component flat against the PC Board. Slightly spread the leads to hold it in place when you flip the board over to expose the back side with the component lead facing up.



- Make good contact between the iron, the lead, and the pad on the board so the lead and the pad both heat up enough to melt the solder.
- It should take from 0.5 to 1 second for the joint to become hot enough to melt the solder.
- Don't overheat the joint, as soon as the solder melts and wicks into the joint remove the iron and hold still for a few seconds until the joint freezes.



A good solder joint will form a shiny curved surface bonding the lead and the pad on the PC board. If the lead wasn't heated enough to melt the solder, the solder will wick in to the pad, but will not adhere to the lead. You may notice a dark line around the lead where the solder dives down through the hole.

A similar bad joint forms when the pad wasn't heated enough to melt the solder. A cold solder joint can often be fixed by reheating the joint; sometime a little more solder will be needed.

More detailed instructions can be found on the internet with a little searching on the topic of soldering.

After inspecting the solder joint, clip off the excess lead using the diagonal cutter. Cut at the top of the solder joint; don't dig into the solder joint.

Optional Clean-up

The clear sticky rosin left around the solder site can be cleaned up with alcohol and a toothbrush, but be sure to let the board dry before expecting the board to operate properly. You won't hurt the circuit by powering it up, but it won't count right until the board has completely dried. Water soluble rosin-cored solder is available at hobby stores. As for me, I don't bother to clean up the board.

Assembly Instructions

At the highest level, the assembly instructions are;

Place each component in its identified location with its leads through the proper holes, solder the leads into place leaving the component snugly against the PC board, and clip the component leads flush with the top of the solder joint. Repeat for all 31 components.

OK, that's a bit too short. After the following general information, there are detailed instructions.

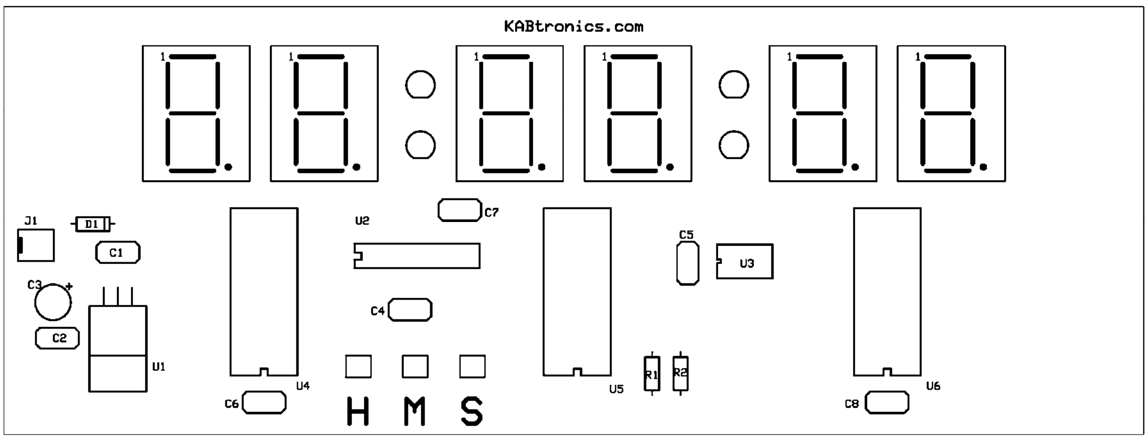
You will need the following tools to build your clock.

- Soldering Iron meant for electrical work, and a very small tip for SMT soldering
- Small Diagonal Cutter
- Very small flat-blade screwdriver to tighten the wire terminals
- Optional: A multi-meter to check for shorts and power supply operation

Component Side

One side of the PC board has white paint markings, shapes, and reference designators. That marked side is the component side, upon which all the components will be placed.

The following diagram is a copy of the white markings on the board, which is known as the silkscreen. Following that is the Bill of Materials, knows as the BOM.



So, to build this clock you need to;

- pick a location on the board,
- read the reference designator there, either from the board or the silkscreen drawing,
- use the BOM to cross index that reference to the part,
- find that part,
- place it in the holes oriented correctly,
- solder it in place, and
- clip off the leads

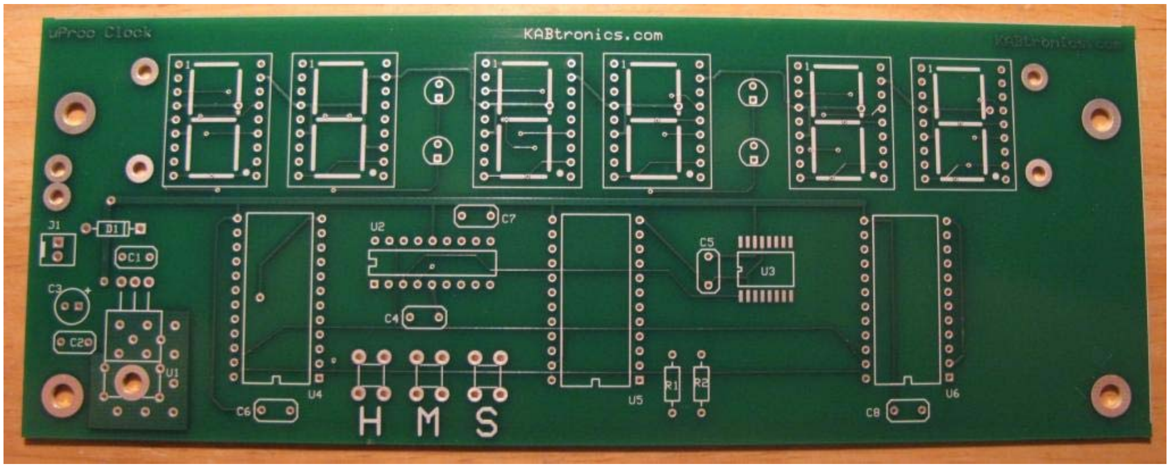
  Repeat for all components.

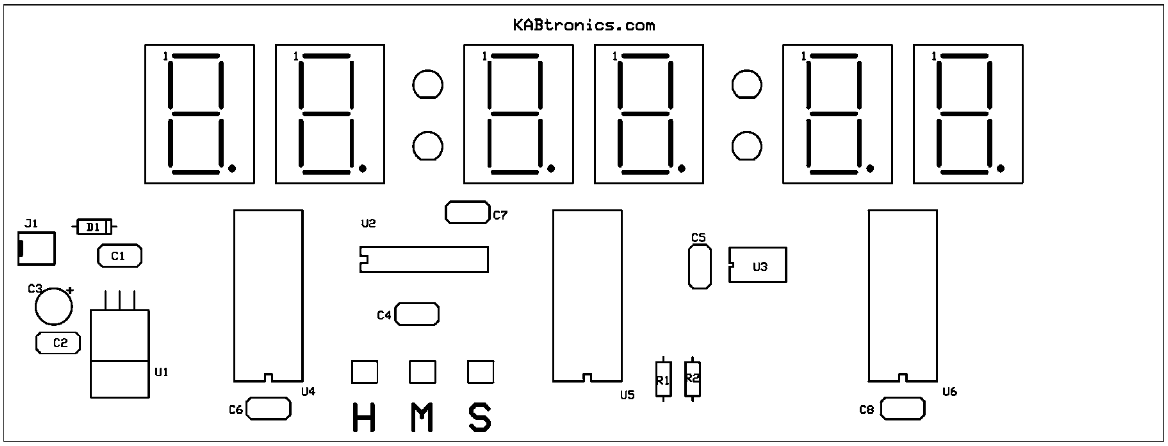
Following the BOM, there is a suggested order of assembly. If you want to build in your own order, you may want to read through the following pages to pick up some hints on mounting odd components.

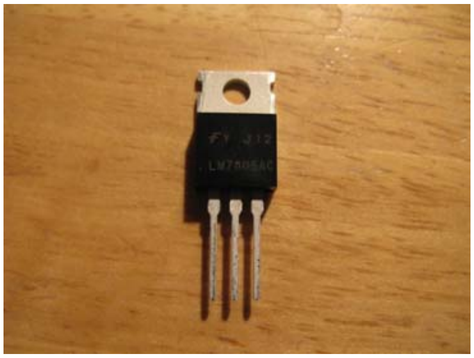
Bill Of Materials

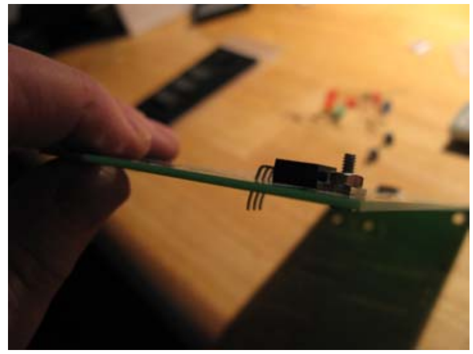
Ref	Desc
C1	0.22 uF Capacitor
C2	0.22 uF Capacitor
C3	10 uF Capacitor
C4	0.01 uF Capacitor
C5	0.01 uF Capacitor
C6	0.01 uF Capacitor
C7	0.01 uF Capacitor
C8	0.01 uF Capacitor
D1	Large Diode
D2	Red LED
D3	Red LED
D4	Red LED
D5	Red LED
DIS1	LDS8164 7 segment display
DIS2	LDS8164 7 segment display
DIS3	LDS8164 7 segment display
DIS4	LDS8164 7 segment display
DIS5	LDS8164 7 segment display
DIS6	LDS8164 7 segment display
J1	header, wire capture
R1	22K Ohm Resistor
R2	33K Ohm Resistor
SW1	push switch
SW2	push switch
SW3	push switch
U1	LM7805
U2	PIC16F84A uProcessor
U3	DS32KHz TCXO
U4	SAA1064 Display Driver
U5	SAA1064 Display Driver
U6	SAA1064 Display Driver

Assembly

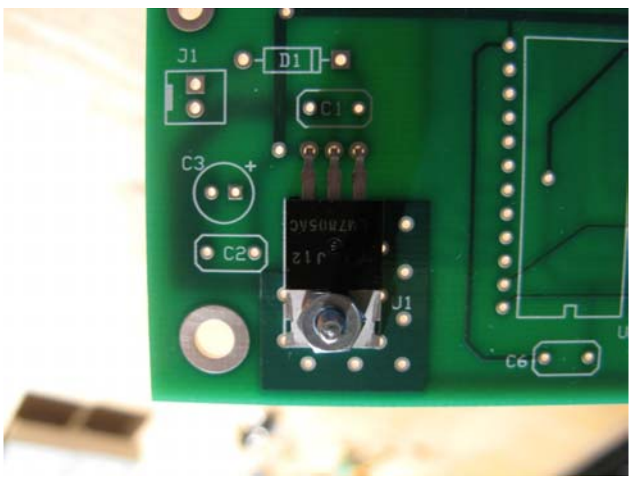

KABtronics.com
uProc Clock
J1
D1
C1
C2
C3
C4
C5
C6
C7
C8
U1
U2
U3
U4
U5
U6
R1
R2
H
M
S


KABtronics.com
J1
D1
C1
C3
C2
U1
U2
C7
C4
C6
U4
H
M
S
U5
R1
R2
C5
U3
U6
C8



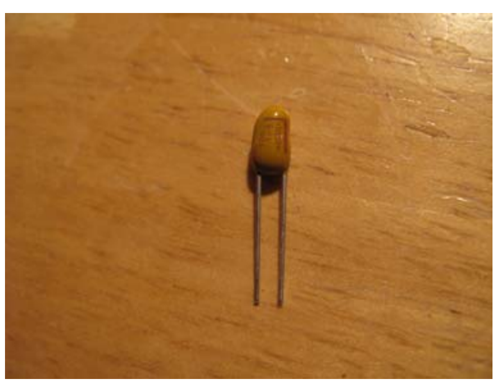


Find U1 – the regulator packaged in a TO-220 package –carefully bend the leads and place the part in position. Before soldering, use the nut and bolt to fasten U1. The point is to hold it snugly against the board for heat transfer, don’t over torque it. Solder and clip the leads.



C3 is a tantalum capacitor and needs to be placed with the positive lead in the hold near the + sign. The long dark line indicates the + side of the cap. Solder and clip the leads.

Negative side



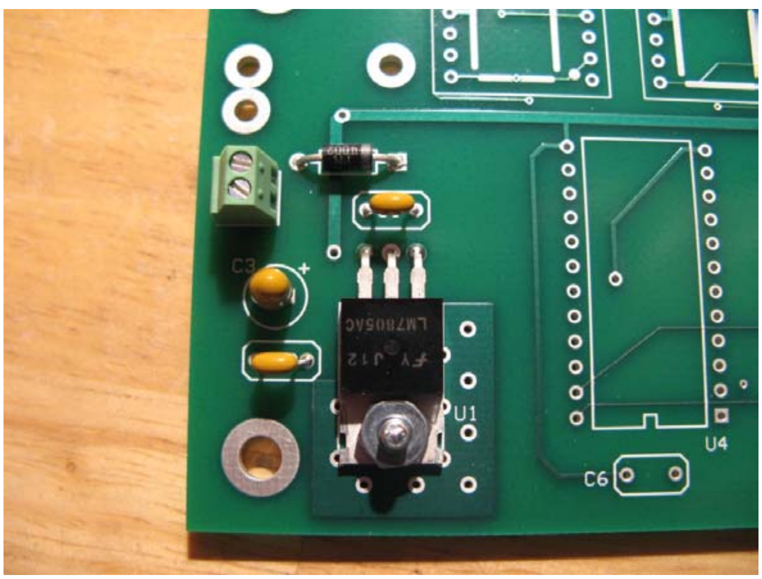
Positive side

Clip off the plastic nubs on the header J1 and place it on the board. Be sure the wire openings are facing towards the edge of the board. Solder it.

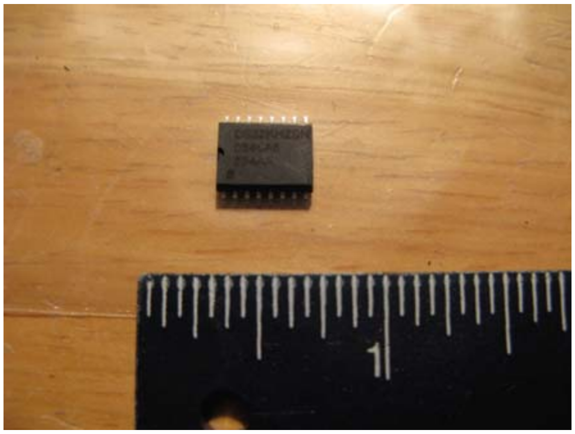
Find C1 and C2, they are the larger two of the ceramic capacitors. Place them, solder and clip.

Place the diode with the line on the part matching the line on the board. Solder and clip.

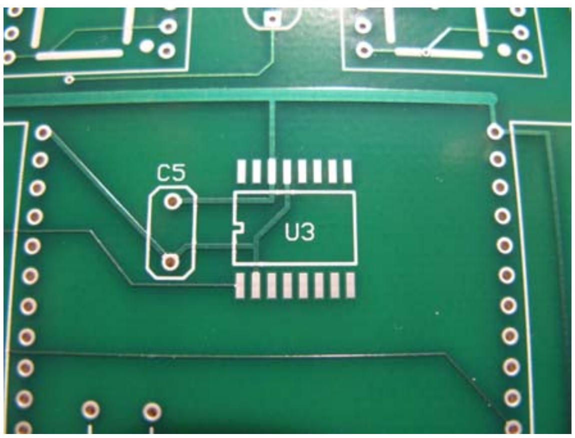
Your board should look like this picture;



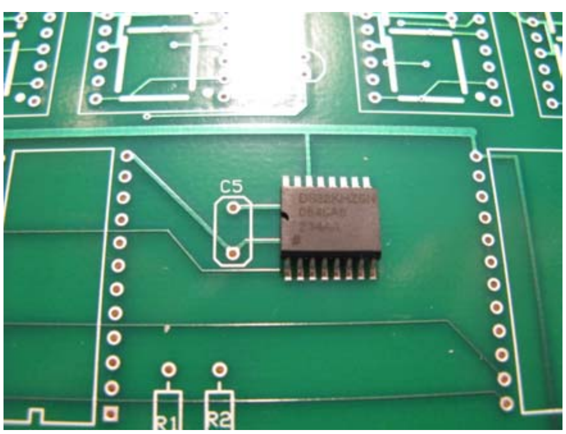
Next find U3 the Surface Mount component.



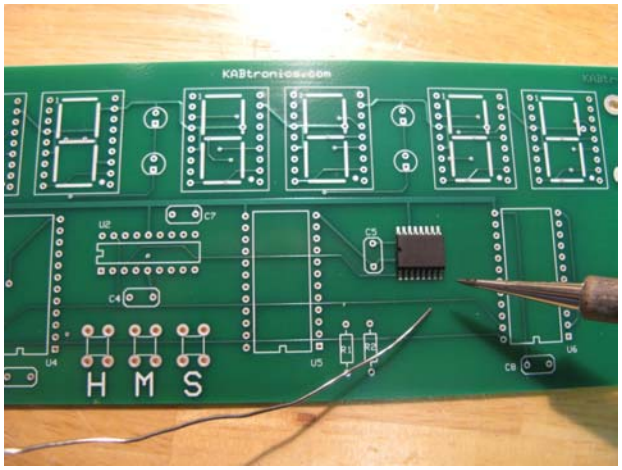
Notice that only four of the pins are actually used. You might still want to solder the other pins but be sure those four are properly soldered.



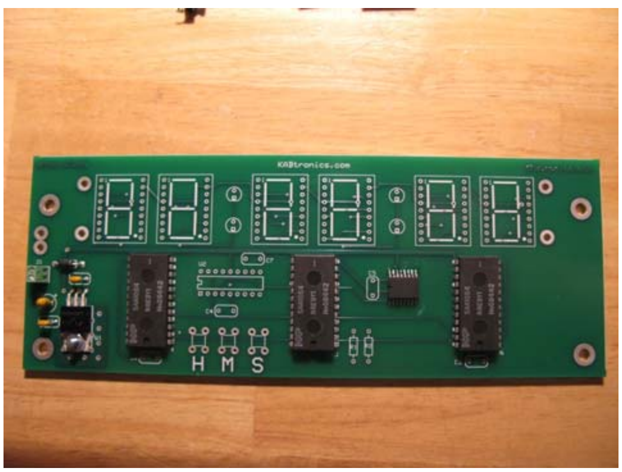
Carefully, without bending any of the leads… place it on the board.



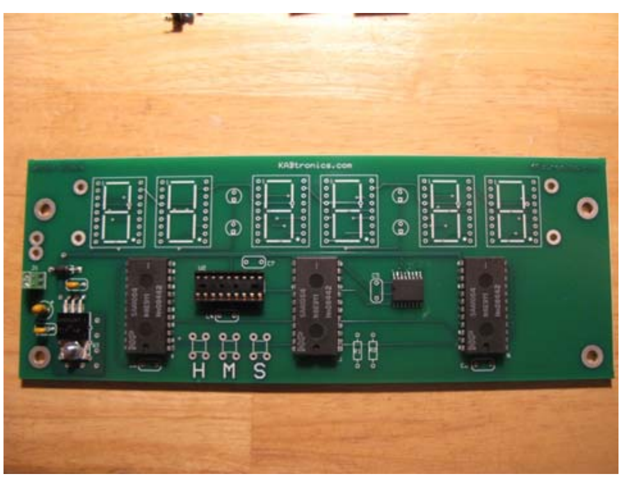
Using a very fine tip soldering iron, solder it down. I found I could place a small amount of solder on the tip and touch the pin and pad simultaneously causing the solder to wick into the joint. Don't use too much solder, it is easy to cause a solder bridge, shorting out two pins.



Next place U4, U5, U6. You may have to squeeze the pins together a bit to have them fit.



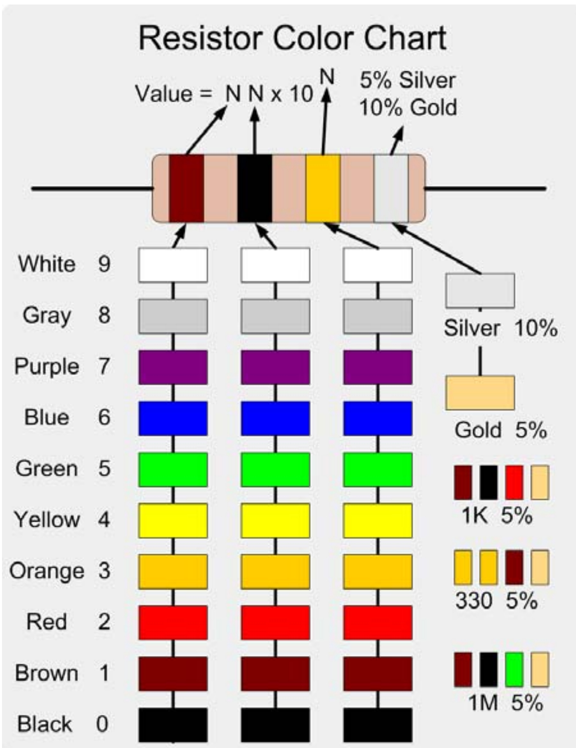
Solder the 18 pin socket for U2, be sure to orient it correctly.

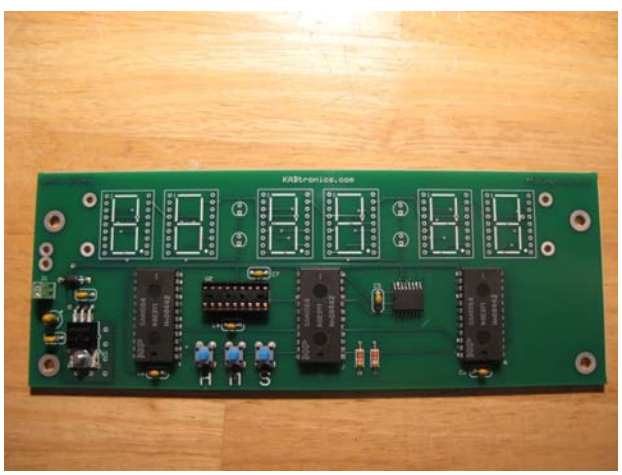


Solder in the three switches SW1, SW2, SW3.

Next mount R1 (22K) and R2 (33K), the following chart shows how resistors are labeled, if printed on a color printer.

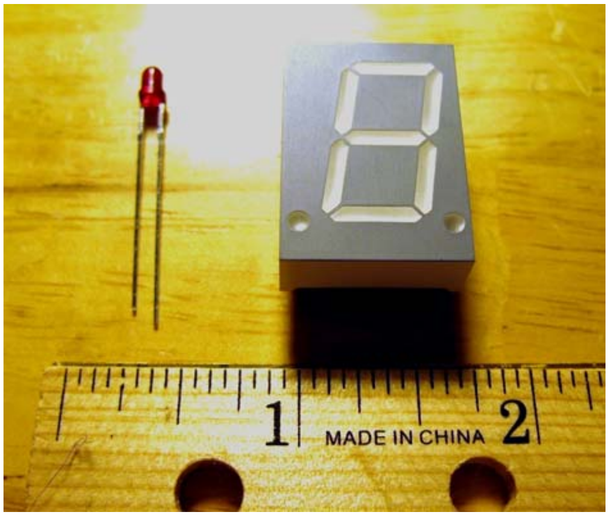
22K is red-red-orange and 33K is orange-orange-orange.

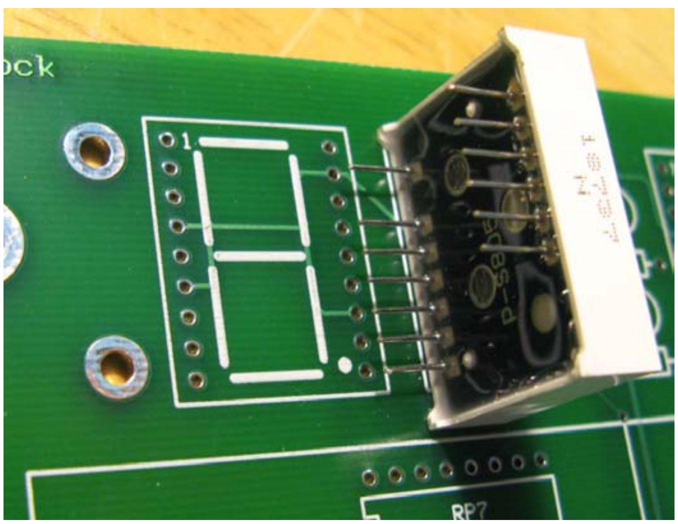




Also mount the remaining five 0.01 uF ceramic capacitors C4, C5, C6, C7, and C8.

Find the LEDs; D2, D3,D4, D5 and the displays; DIS1, DIS2, DIS3, DIS4, DIS5, DIS6





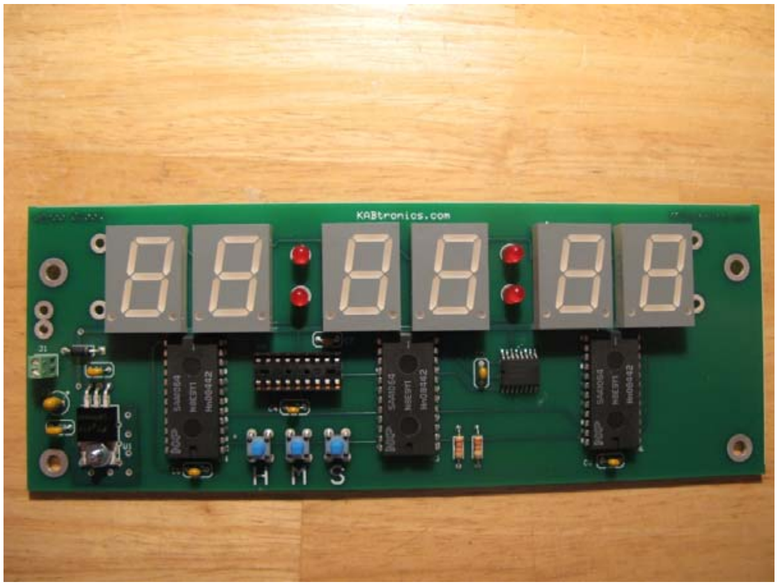
Mount the Displays.

There is a danger of incorrectly mounting these 7 segment displays.

Note how pin one is marked by the numeral 1 in the upper left. The white dot in the lower right of the silkscreen for the display is the DP - not a pin indicator.

The 7-segment LED displays are missing 5 pins; 1,8,9,16,18. This allows for the part to fit into the socket two ways, the wrong way and the right way. The right way is decimal points on the lower edge and the part closer to the bottom edge. Pin one hole on the board will not have a lead in it when the part is mounted correctly.

Next solder in the four single LEDs, the short pin goes into the lower hole, there may also be a flat side on the shorter pin side.



Your wall transformer may come with prepared wires, if not, prepare the wall transformer wires as shown below.

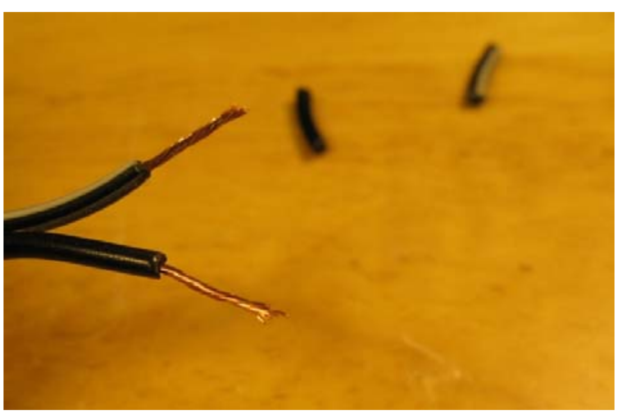
Find the wall transformer.

Cut off the connector at the end of the wire.

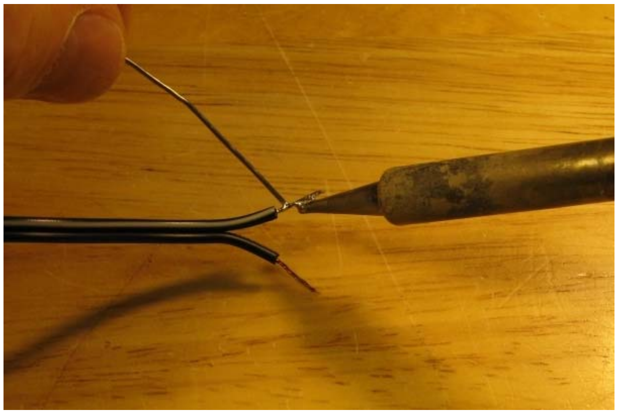
Separate the wires for an inch.

Strip off ½ inch of the insulation.

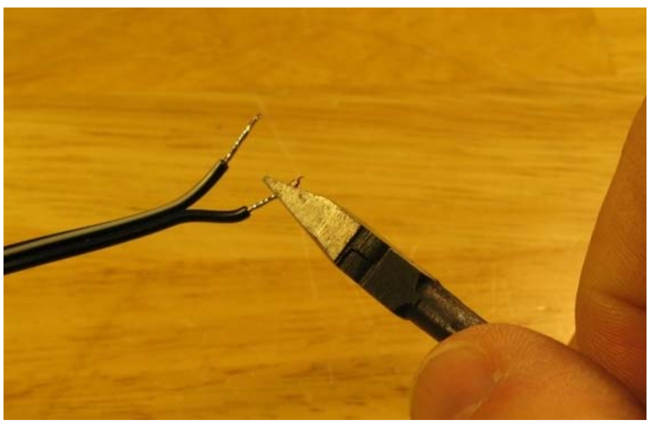
Twist the wire strands together



Use soldering iron to tin (coat with solder) the wire ends.

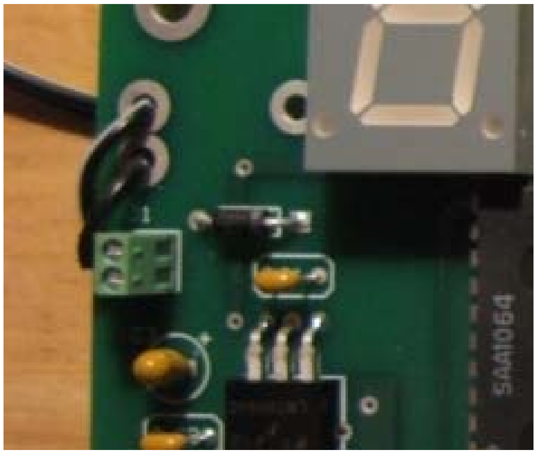


Clip off excess leaving ¼ inch



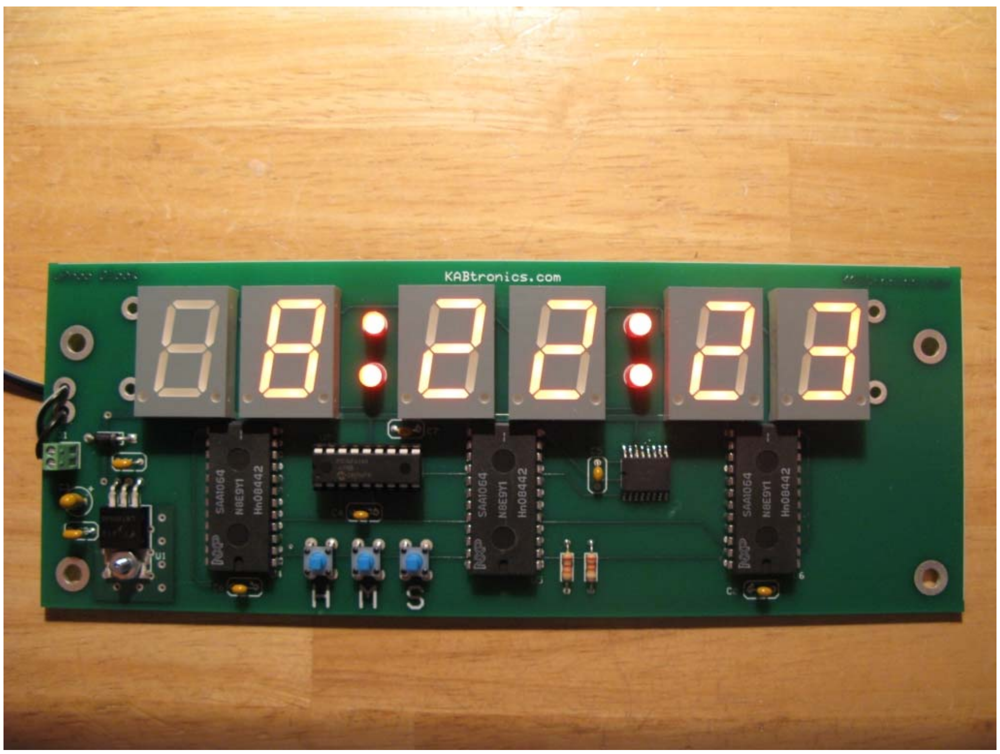
Finished wire ends.

Thread the wires through the two holes and have the one with the white strip connect to the top hole.



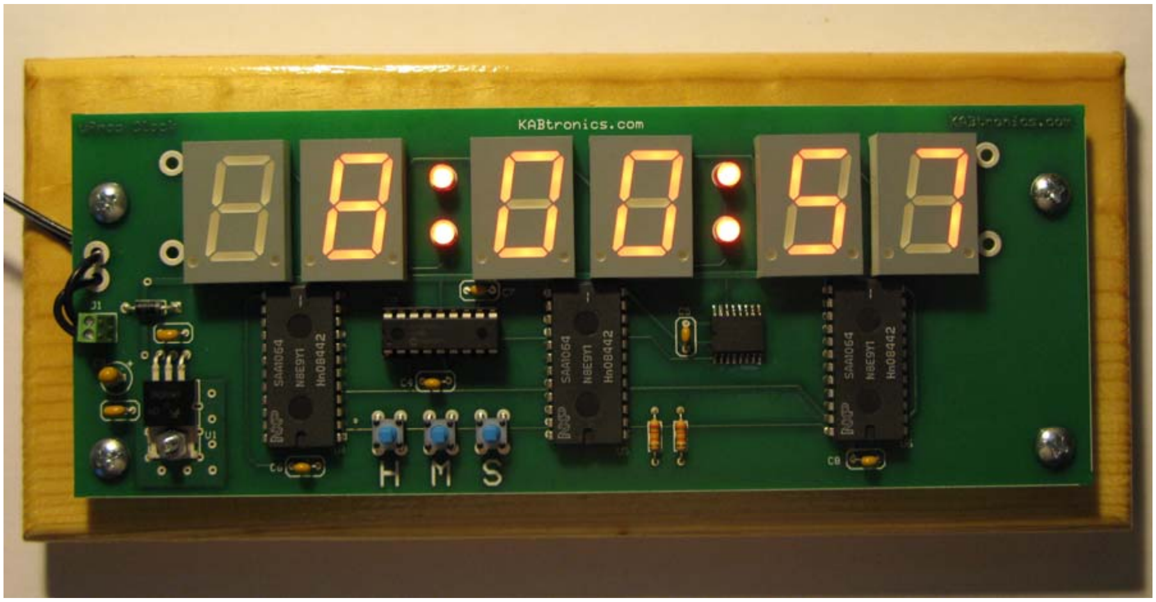
Insert the PIC IC U2 into the socket.

At this point you can plug in the wall transformer. You should see the display come up at 12:00:00. The time should be settable with using the H and M buttons. The S button resets the seconds to the nearest minute.



If you wish, you can mount it to a plaque.

You are done.



Operating Instructions

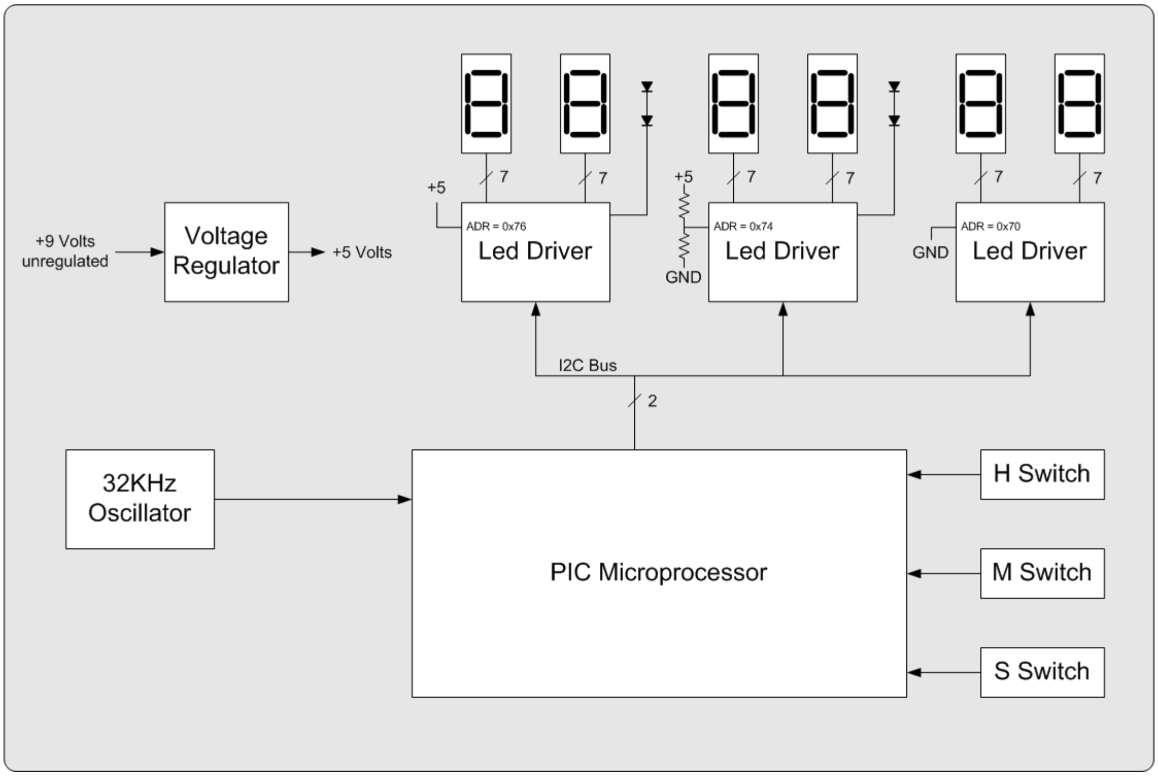
Upon powering up, the display flashes to show power loss.

Press H to advance the hours to the current time.

Press M to advance the minutes.

Pressing the S button will advance or back-up the clock to the nearest minute.

Theory of Operation



Power Supply:

The wall module delivers about 9 to 10 volts to the linear voltage regulator which produces 5 volts powering the rest of the circuitry. The clock draws about 300 milliamps so the regulator dissipates about 1.5 watts. The copper around the regulator dissipates the heat.

Timebase:

The clock depends upon a 32.768 KHz temperature compensated quartz crystal oscillator. The 32.768 KHz is divided in the PIC by 256 producing a 128 Hz clock interrupt.

Displays:

The 7 segment LED displays and the single LEDs are driven by serial bus controlled current drivers. The I2C bus is common to the three drivers. The address pin of each driver is set at a different voltage to set the address of each driver shown in the block diagram.

Time Setting Switches:

Each switch grounds a pin of the PIC, the pins are set to use the internal pull-ups and debounce the signal in software.

Software:

The 128 Hz interrupt counts and upon reaching 128, increments a seconds counter, which will in turn increment the minutes and hours counters. The current time is stored in global registers visible to the mainline program.

The switches are read at each interrupt. The debounced switch settings are stored in global variables and are visible to the mainline.

The mainline runs through the set up code and falls into a perpetual loop the acts when the time changes or a switch is pressed.

The code for the clock is shown below.

```
// PIC clock
//
// PIC16F628A
//
// Pin connections
//
// RA0 (17) out  SDA
// RA1 (18) out SCL
// RA2 (1)  out testp1
// RA3 (2)  out testp2
// RA4 (3)  in 32KHz Clock
// RA5 (4)
// RA6 (15)
// RA7 (16)

// RB0 (6)
// RB1 (7)  in pushbutton H
// RB2 (8)  in pushbutton M
// RB3 (9)  in pushbutton S
// RB4 (10)
// RB5 (11)
// RB6 (12)
// RB7 (13)

// data direction registers  1=input
// trisa = 0xf0
// trisb = 0xff

// 32.768KHz on TMR0 input
```

```
//


unsigned char cnt_p125;  // prescaler
unsigned char cnt_seconds;
unsigned char cnt_minutes;
unsigned char cnt_hours;
unsigned int cnt_days;
unsigned char sw1; // 1 when H pressed
unsigned char sw2; // 1 when M pressed
unsigned char sw3; // 1 when S pressed
signed char sw1cnt; // debounce counter
signed char sw2cnt; // debounce counter
signed char sw3cnt; // debounce counter
unsigned char runtime; // 1 = run  0 = stop
unsigned char powerup; // 0 = not set yet
unsigned char flash; // 0=don't flash, 1=low 2=normal

//
// 128Hz interupt
//
void interrupt( void )
{
// inc time
   cnt_p125++;
// toggle debug bit A.2
if (cnt_p125==0)
   set_bit( porta, 2 );
if (cnt_p125==64)
          clear_bit( porta, 2 );
// operate power fail flash
if (powerup==0)
{
   if (cnt_p125==0)
   {
      flash=1;
   }
   else if (cnt_p125==64)
   {
      flash=2;
   }
}
if (cnt_p125>127)
{
   cnt_p125=0;
  if(runtime>0)
  {
```

```
   cnt_seconds++;
   if (cnt_seconds>59)
   {
          cnt_seconds=0;
          cnt_minutes++;
          if (cnt_minutes>59)
          {
             cnt_minutes=0;
             cnt_hours++;
             if (cnt_hours>11)
             {
                    cnt_hours=0;
                    if (cnt_days>0)
                    {
                       cnt_days--;
                    }
             }
          }
   }
  }
}

// switches
if (portb.1==1)
{
   if(sw1==0)
   {
          if(sw1cnt<3)
          {
             sw1cnt++;
          }
          else
          {
             if(sw1==0)
                sw1=1;
          }
   }
}
else
{
   if(sw1==1)
   {
          if(sw1cnt>0)
          {
             sw1cnt--;
          }
```

```
            else
            {
               if(sw1==1)
                  sw1=0;
            }
   }
}

// switches
if (portb.2==1)
{
   if(sw2==0)
   {
            if(sw2cnt<3)
            {
               sw2cnt++;
            }
            else
            {
               if(sw2==0)
                  sw2=1;
            }
   }
}
else
{
   if(sw2==1)
   {
            if(sw2cnt>0)
            {
               sw2cnt--;
            }
            else
            {
               if(sw2==1)
                  sw2=0;
            }
   }
}

// switches
if (portb.3==1)
{
   if(sw3==0)
   {
            if(sw3cnt<3)
```

```
            {
               sw3cnt++;
            }
            else
            {
               if(sw3==0)
                  sw3=1;
            }
   }
}
else
{
   if(sw3==1)
   {
            if(sw3cnt>0)
            {
               sw3cnt--;
            }
            else
            {
               if(sw3==1)
                  sw3=0;
            }
   }
}
//clear_bit( portb, 1 );
            clear_bit( intcon, T0IF );  //clear TMR0 overflow flag
}


// i2c connection routines, have these next 5 call bit and delay routines

void sda_high(void)
{
   set_bit( porta, 0 );
}


void sda_low(void)
{
   clear_bit( porta, 0 );
}


void scl_high(void)
{
```

```
    set_bit( porta, 1 );
}


void scl_low(void)
{
    clear_bit( porta, 1 );
}

void i2c_delay()
{
            delay_us(20);
}

// i2c routines

void i2c_start(void)
{
    scl_high();
    sda_high();
    i2c_delay();
    sda_low();
    i2c_delay();
    scl_low();
    i2c_delay();
}

void i2c_stop(void)
{
    sda_low();
    scl_low();
    i2c_delay();
    scl_high();
    i2c_delay();
    sda_high();
    i2c_delay();
}


void i2c_bitout(unsigned char b)
{
    // set data bit
    if (b!=0)
    {
                sda_high();
            }
```

```
    else
    {
            sda_low();
        }
    // pulse clock
    i2c_delay();
    scl_high();
    i2c_delay();
    scl_low();
    i2c_delay();
}

void i2c_ack()
{
    i2c_delay();
    scl_high();
    i2c_delay();
    scl_low();
    i2c_delay();
}

void i2c_byteout(unsigned char b)
{
    i2c_bitout(b&0x80);
    i2c_bitout(b&0x40);
    i2c_bitout(b&0x20);
    i2c_bitout(b&0x10);
    i2c_bitout(b&0x08);
    i2c_bitout(b&0x04);
    i2c_bitout(b&0x02);
    i2c_bitout(b&0x01);
    i2c_ack();                      // ack cycle
}


unsigned char get7seg(char i)
{
unsigned char rv;
switch (i&0x0f)
    {
        case 0:
            rv=0xfa;
            break;
        case 1:
            rv=0x60;
            break;
```

```
        case 2:
            rv=0xdc;
            break;
        case 3:
            rv=0xf4;
            break;
        case 4:
            rv=0x66;
            break;
        case 5:
            rv=0xb6;
            break;
        case 6:
            rv=0xbe;
            break;
        case 7:
            rv=0xe0;
            break;
        case 8:
            rv=0xfe;
            break;
        case 9:
            rv=0xf6;
            break;
        case 10:
            rv=0xee;
            break;
        case 11:
            rv=0x3e;
            break;
        case 12:
            rv=0x9a;
            break;
        case 13:
            rv=0xf8;
            break;
        case 14:
            rv=0x9e;
            break;
        case 15:  // blank
            rv=0x00;
            break;
    }
  return(rv);
}
```

```
// send time digits
void showtime()
{
   i2c_start();
   i2c_byteout(0x70); // address
   i2c_byteout(0x00); // instruction
   if (flash==2)
      i2c_byteout(0x16); // control - flash low
   else
      i2c_byteout(0x36); // control
   i2c_byteout(get7seg(cnt_seconds%10)); // secs
   i2c_byteout(get7seg(cnt_seconds/10)); // 10secs
   i2c_stop();
   i2c_start();
   i2c_byteout(0x74); // address
   i2c_byteout(0x00); // instructionl
   if (flash==2)
      i2c_byteout(0x16); // control - flash low
   else
      i2c_byteout(0x36); // control
   i2c_byteout(get7seg(cnt_minutes%10)+1); // minutes
   i2c_byteout(get7seg(cnt_minutes/10)); // 10minutes
   i2c_stop();
   i2c_start();
   i2c_byteout(0x76); // address
   i2c_byteout(0x00); // instruction
   if (flash==2)
      i2c_byteout(0x16); // control - flash low
   else
      i2c_byteout(0x36); // control
   i2c_byteout(get7seg((cnt_hours+1)%10)+1); // hours
   i2c_byteout(get7seg(cnt_hours<9?15:1)); // blank zerohours
   i2c_stop();
}

//
// mainline loop
//
void main()
{
int seconds_shown;
unsigned char hour;
unsigned char tenshour;
unsigned int day;
```

```
// set port directions
trisa = 0xf0;       // porta direction
trisb = 0xff;       // portb direction
cmcon = 0x07;   // turn off comparitors
clear_bit( option_reg, 7 ); // turn on portb pullups

// clear time and state variables
seconds_shown=0;
cnt_p125=0;
cnt_seconds=0;
cnt_minutes=0;
cnt_hours=11;
sw1=1;
sw2=1;
sw3=1;
sw1cnt=0;
sw2cnt=0;
sw3cnt=0;

runtime=0;
powerup=0;
flash=2; // dim display

// enable interrupts
set_bit( intcon, T0IE ); //enable TMR0 overflow bit
set_bit( intcon, GIE ); // start interrupts

showtime();
while( 1 )
{
// if hours switch pressed
   if (sw1==0)
   {
// stop time
               runtime=0;
// mark setting of time to stop display flashing
               powerup=1;
//inc hours
               cnt_hours++;
               if (cnt_hours>11)
                      cnt_hours=0;
// display time
               showtime();
// delay short time .2S?
               delay_ms(200);
   }
```

```
// if minutes switch pressed
   else if (sw2==0)
   {
// stop time
                    runtime=0;
// mark setting of time to stop display flashing
                    powerup=1;
//inc hours
                    cnt_minutes++;
                    if (cnt_minutes>59)
                              cnt_minutes=0;
// display time
                    showtime();
// delay short time .2S?
                    delay_ms(200);
   }
// if seconds switch pressed
   else if (sw3==0)
   {
// stop time
                    runtime=0;
// zero seconds , round to closest minute
                    if (cnt_seconds<30)
                    {
                              cnt_seconds=0;
                    }
                    else
                    {
                       cnt_seconds=0;
                       cnt_minutes++;
                       if(cnt_minutes>59)
                       {
                                        cnt_minutes=0;
                                        cnt_hours++;
                                        if(cnt_hours>11)
                                           cnt_hours=0;
                       }
                    }
          }
//if no switch pressed
          else
          {
// start time if stoped
                    if (runtime==0)
                              runtime=1;
// show time if seconds don't match
```

```
		else if ((seconds_shown!=cnt_seconds) | | (flash!=0))
		{
			seconds_shown=cnt_seconds;
			showtime();
			flash=0;
		}
	}
} //endless while loop
} // program end
```

In Case of Difficulty

Work down the table, check for the conditions on the left, don't progress down until fault condition is fixed.

Condition	Action
Using voltmeter, check for about 10 volts across J1, positive on upper pin.	Disconnect wall transformer leads, plug in wall transformer and check for voltage. If you don't see it, your power is out or the wall transformer is bad.
Check for about 10 volts on both sides of D1	Missing on both sides: you have a short between ground and power somewhere on your board, find it. Missing on side connected to U1: check if diode D1 is backwards. If your voltmeter can check diodes, test if your diode is faulty and open.
Check for 5 volts on pin 3 of U1	About 0 volts: you have a short between ground and power somewhere on your board, find it. Over 5.5 volts: Your voltage regulator is bad, along with U2 and U3 which may have been damaged by over voltage. Look for misplaced parts around the regulator and bad solder joints.
5 volt power is good but no displays	The displays are turned on by serial signals from U2, U2 depend upon a 32KHz square wave clock from U3, so check the placement and soldering of U3, and check the placement of U2 in the socket.
One or more digits are blank	Check if the IC driving that display U4, U5, U6 is placed properly and soldered well. Use the schematic and notice that the driver ICs are close to the Displays.
Some segments of a digit are light wrong or blank.	Shorted traces from U4, U5, U6 to the DIS1-6 will cause bad segments
A time setting switch does not work.	Each of these switches ground one pin of U2, with the power removed, you should be able to test for zero ohms for each switch pressed. Use the schematic and silkscreen to find the right pins.

This list is my guess at the cause of any problem in decreasing order of probability;

- Wrong component loaded
- Component loaded backwards
- Bad solder joint causing open or short to adjacent pad
- A clipped-off lead has stuck to the back of the board and is shorting out the circuit
- The board is resting on something conductive shorting out the circuit
- A component is bad because;
    - it was over heated when soldered (medium chance)
    - came that way with the kit (low chance)
    - was damaged by static electricity when handled (unlikely)
- The PC board is damaged or was fabricated incorrectly (very unlikely)

So take a close look at the back of the board looking for bad solder joints and clipped leads in the area of the problem. Check for proper components and orientations. If you haven't found the problem, after thoroughly thinking through the problem and checking everything you can think of, feel free to email customerservice@transistorclock.com. I can't fix the problem for you, but I can work with you to find the problem and supply a spare part or two if needed to get you running.

Specifications

PC board Size: 3 inches wide by 8 inches high

(Allow ¼ inch behind board and ¾ inch above when loaded for parts clearance)

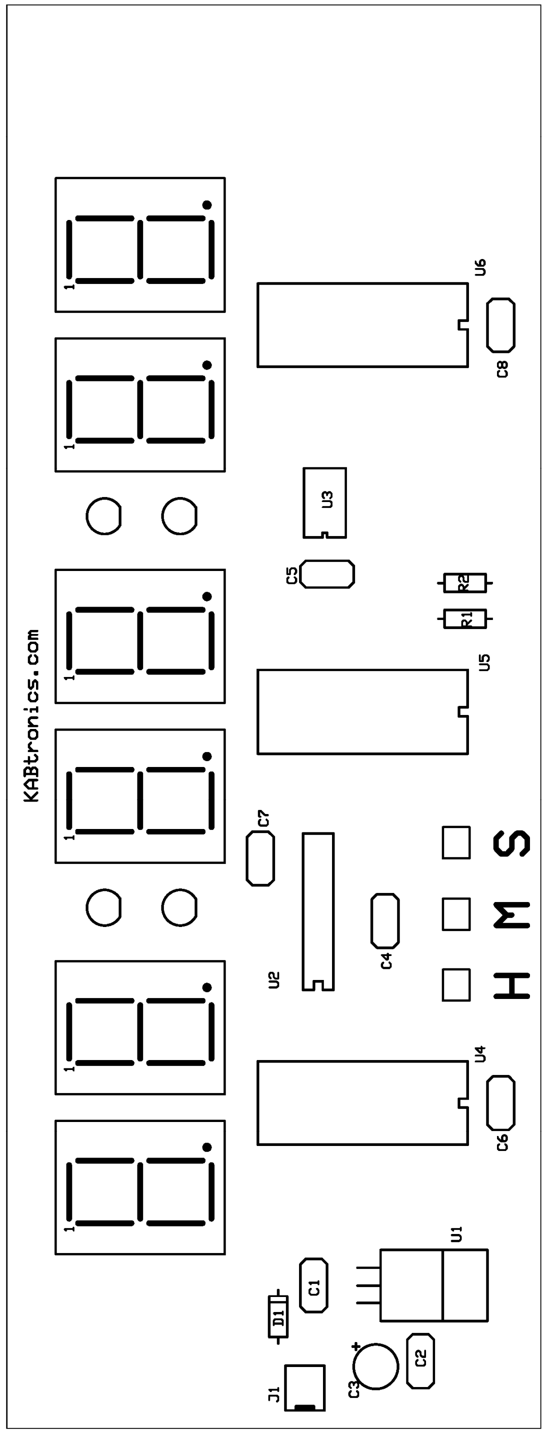
Weight: About 3.5 oz (add 9 oz for wall transformer)

Power consumption: About 3 watts, (0.3 amps @ 10 volts DC)

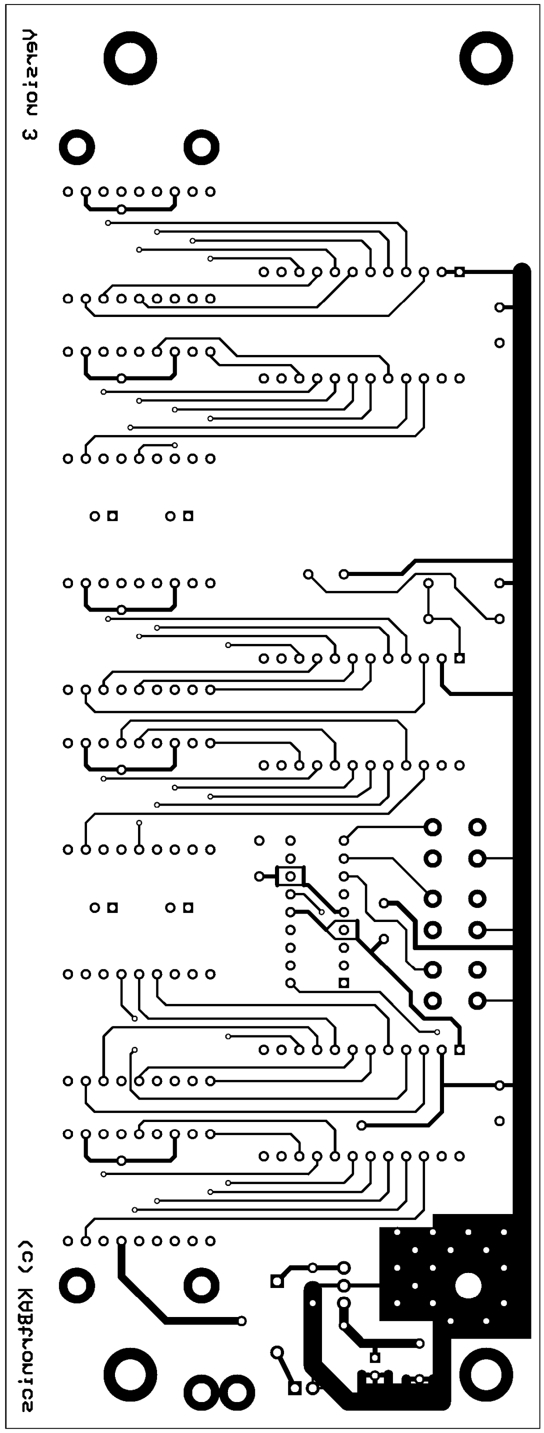
Temperature limits: Designed for room temperature operation, 60-80 °F

Warrantee: There is no warrantee of any kind. KABtronics wants you to succeed and be happy with your clock, so don't hesitate to email customerservice@transistorclock.com with questions if you are having difficulty.

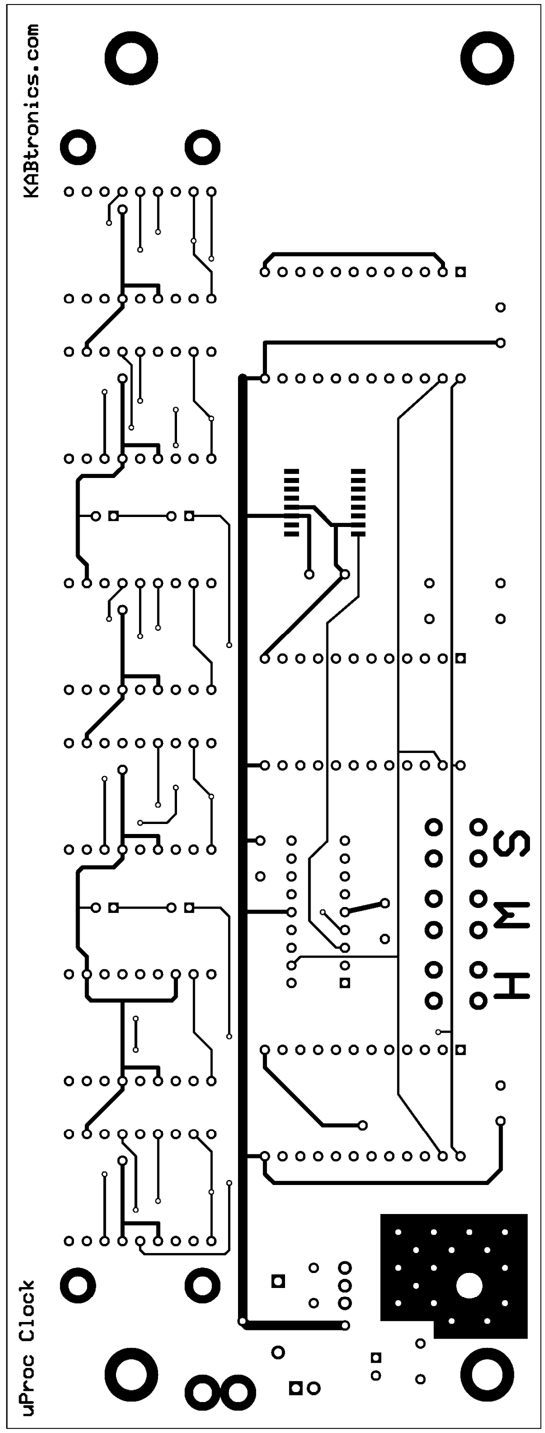
Circuit Board Views



Silkscreen

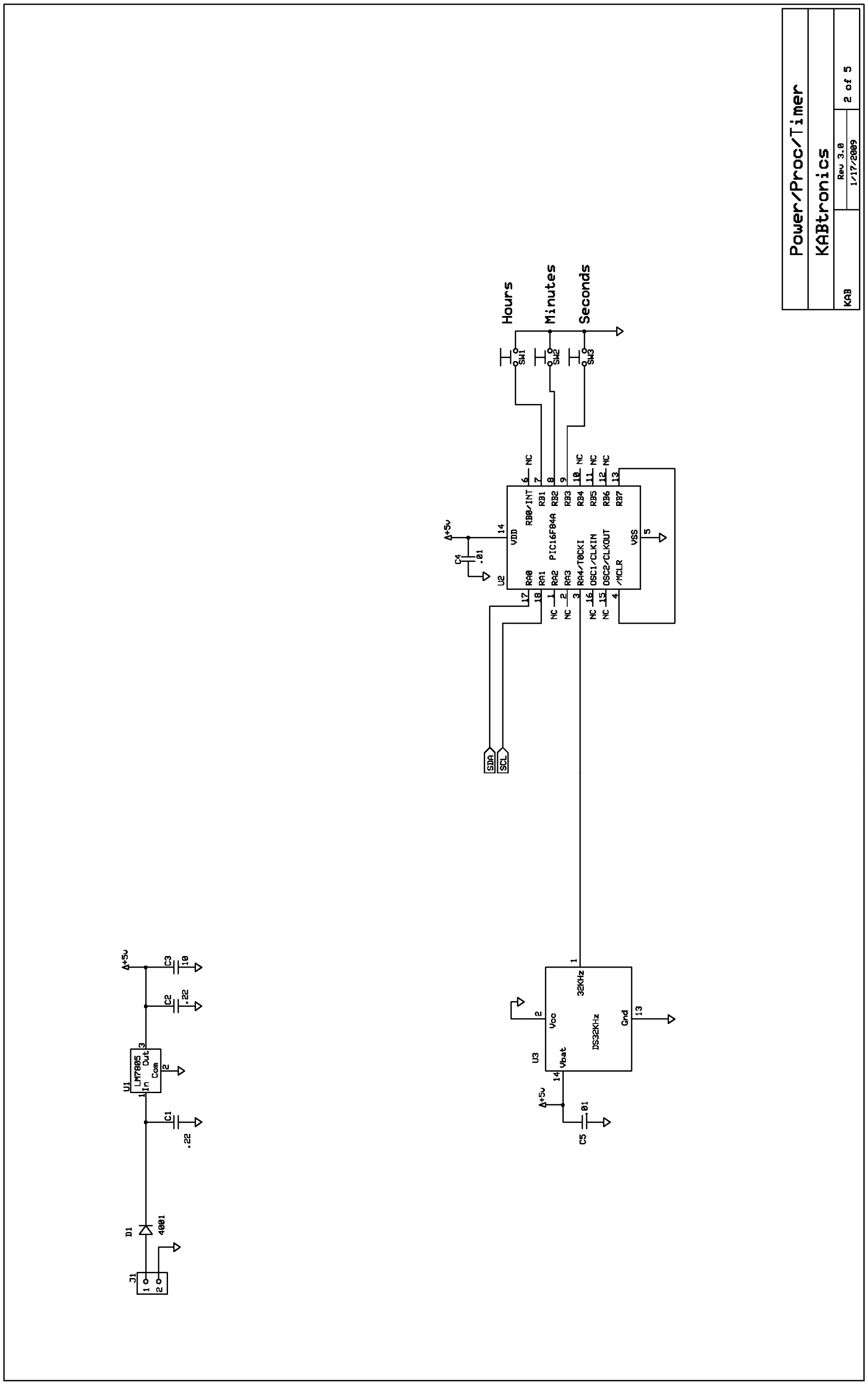


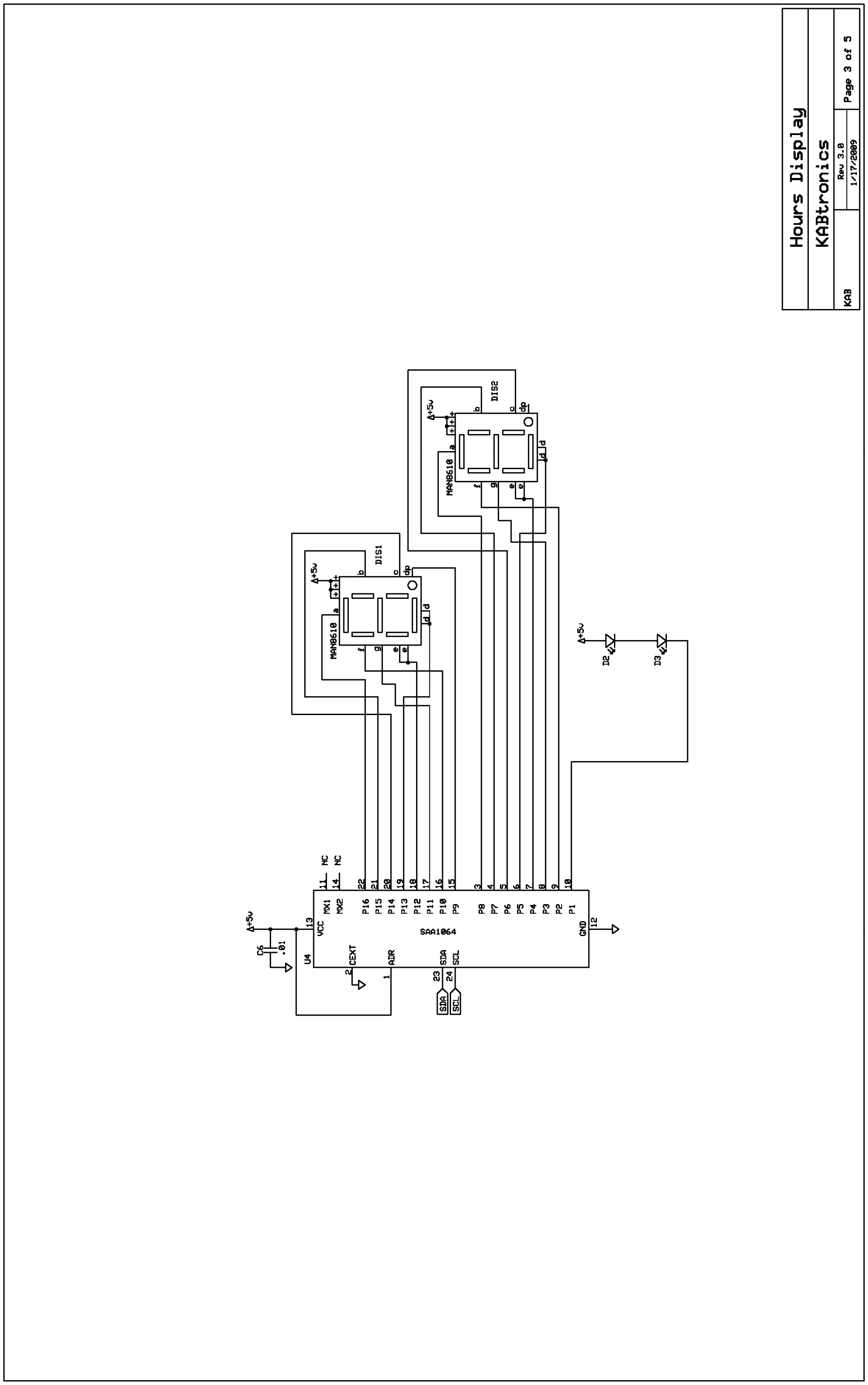
Bottom Copper

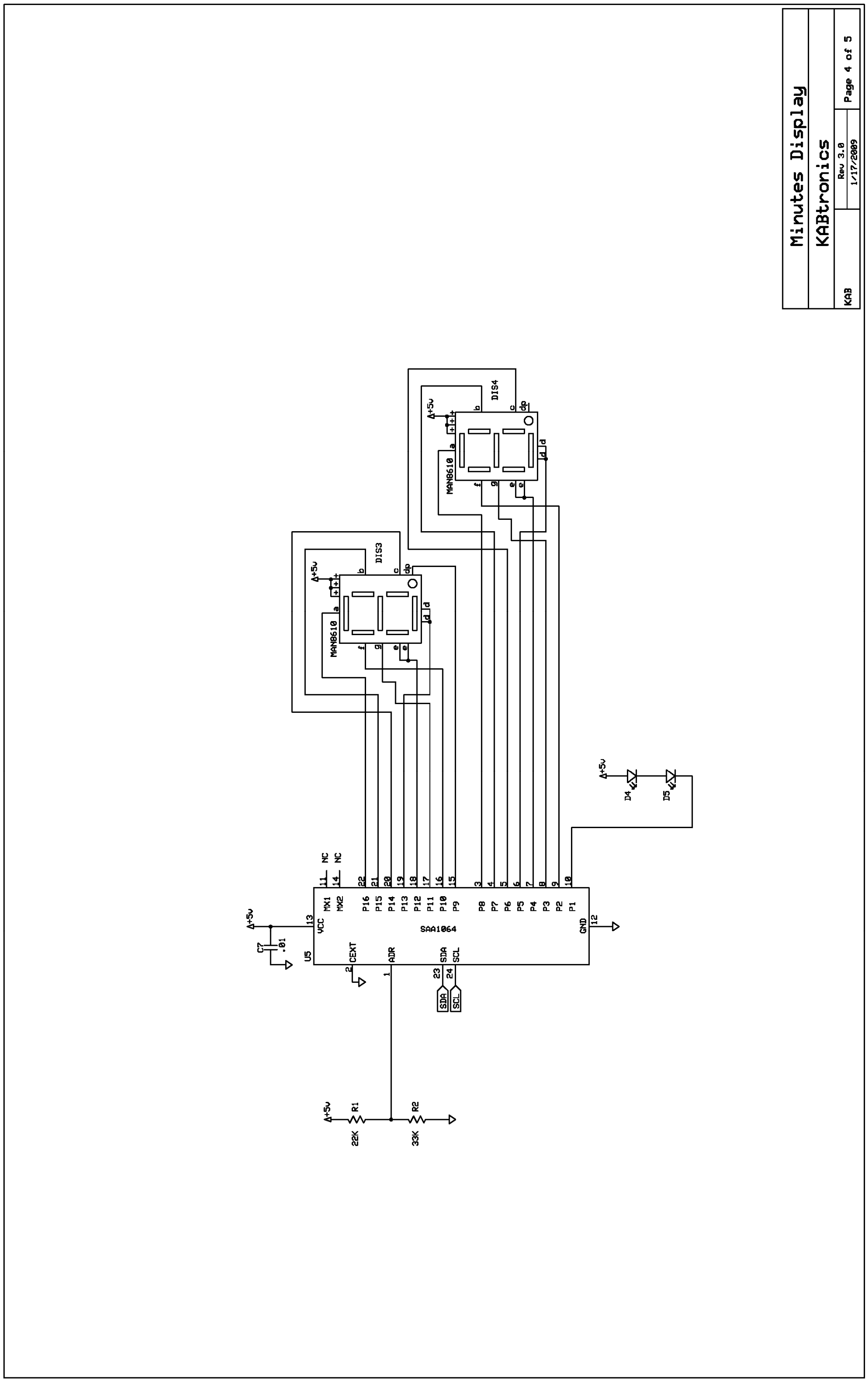


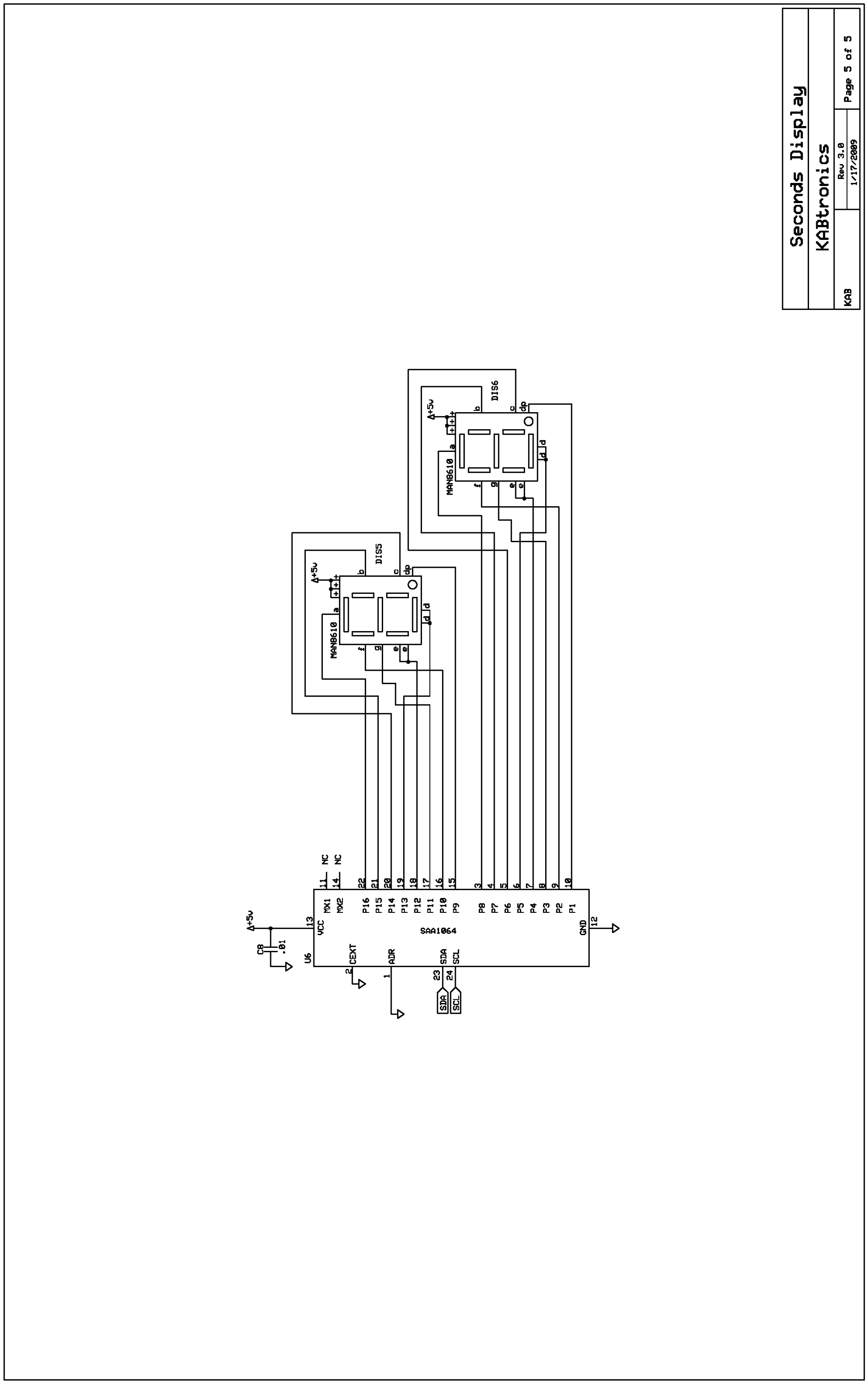
Top Copper

Schematic


Power/Proc/Timer
KABtronics
Rev 3.0
1/17/2009
2 of 5
KAB
Hours
Minutes
Seconds
PIC16F84A
DS32KHz
LM7805
SDA
SCL


Hours Display
KABtronics
KAB
Rev 3.0
1/17/2009
Page 3 of 5
DIS1
DIS2
MAN8610
MAN8610
SAA1064
U4
C6
.01
D2
D3
SDA
SCL
+5v
NC
NC


Minutes Display
KABtronics
Rev 3.0
1/17/2009
Page 4 of 5
KAB
DIS3
DIS4
MAN8610
MAN8610
SAA1064
U5
C7
.01
R1
22K
R2
33K
D4
D5
SDA
SCL
+5v


Seconds Display
KABtronics
Rev 3.0
1/17/2009
Page 5 of 5
KAB
DIS5
DIS6
MAN8610
SAA1064
U6
C8
.01
+5v
VCC
GND
CEXT
ADR
SDA
SCL
MX1
MX2
NC