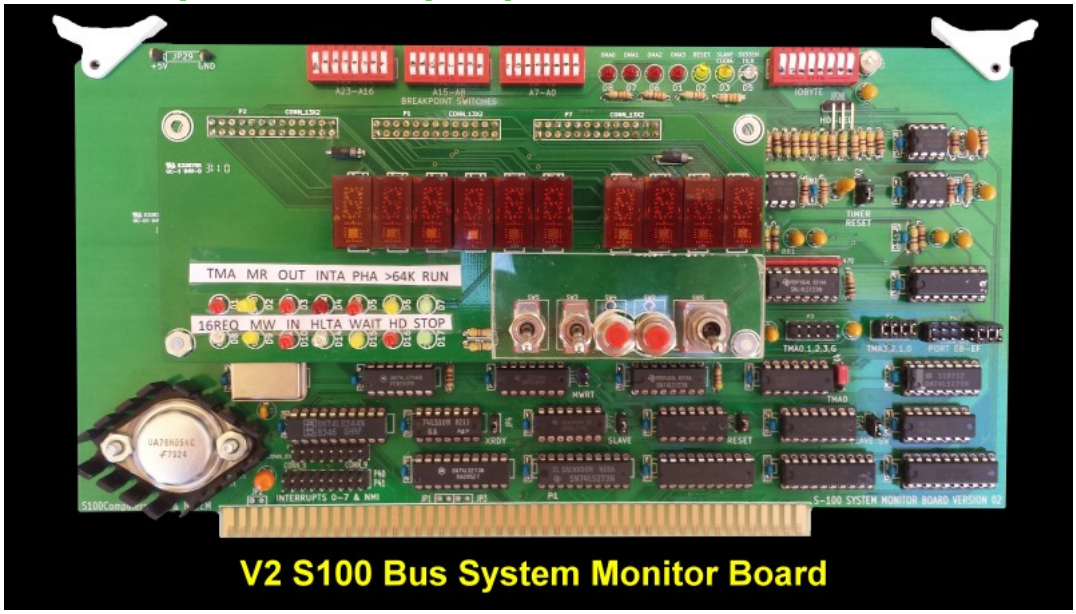
S100 Computers
A Web Site For S-100 Bus Computer Owners

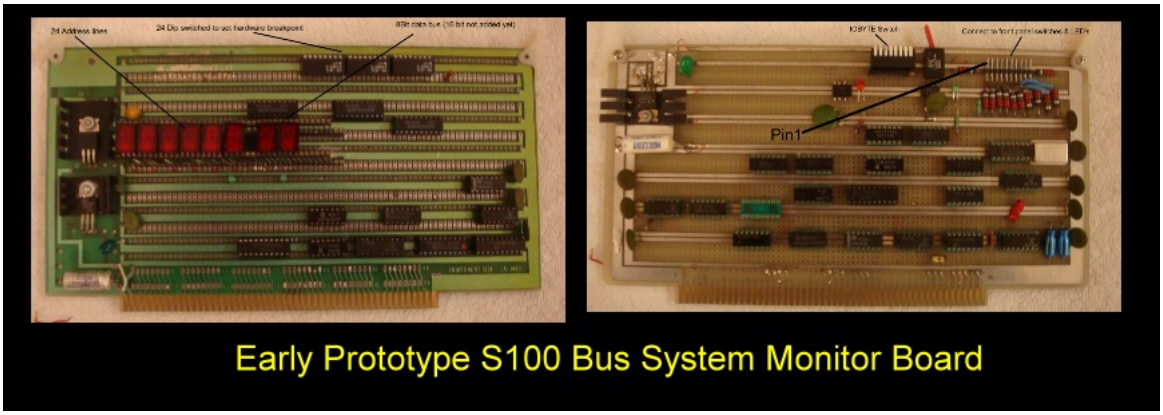
A S-100 Bus System Monitor (SMB) Board



V2 S100 Bus System Monitor Board

Most S-100 users that like to work with hardware have some kind of front panel or hardware monitor for their system. It allows you to see what was "going on". Often in a very general sense but in the case of systems that can "single step" and/or display the bus memory and data lines they can be very useful indeed for debugging.

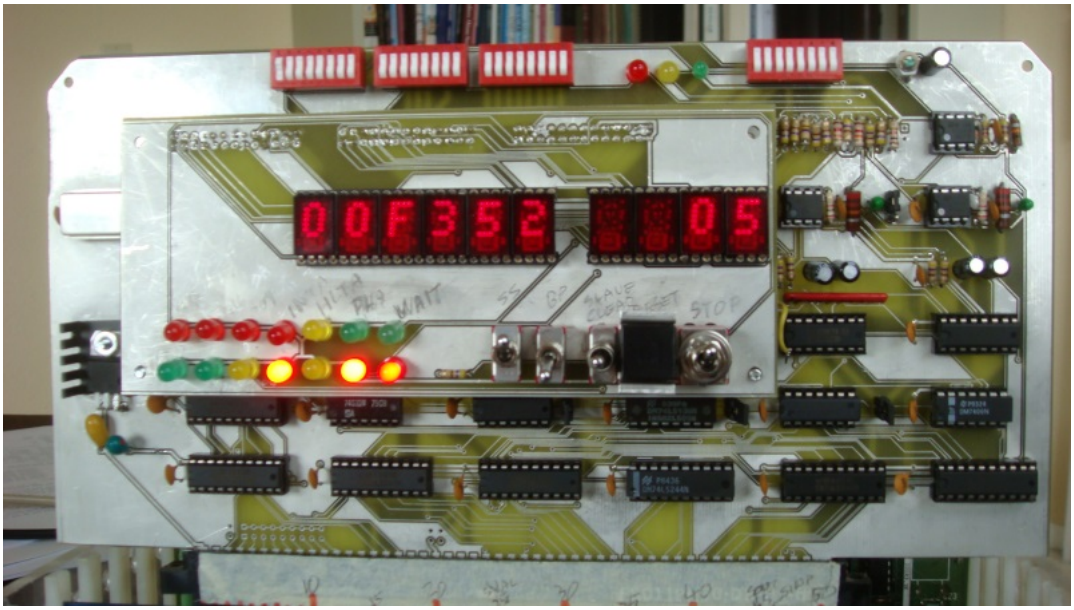
The gold standard was the IMSAI front panel. Beautifully done but unfortunately setup for old S-100 systems with little ability to give meaningful information with high speed/extended addressing for later S-100 systems. In my opinion the Intersystem's front panel was perhaps the best of that era. However in the end, and for a number of reasons I decide to build my own board. Initially it was in fact two boards with all chips point to point connected with soldered wires. The project evolved. I kept adding more and more functions. The Breakpoint and Single Step circuitry come directly from Sol Libes & Mark Garetz's book "Interfacing to the S-100/IEEE 696 Microcomputers". A must have book BTW, for anybody doing S-100 Computers hardware. Here is a picture of the early pair of SMB prototype board pair:-



Early Prototype S100 Bus System Monitor Board

The board utilized HP 5082-7340 Hex display LED's to show the 24 bit data address and 8 bit data being read by the CPU (a Z80) in real time. It had an "IOBYTE port" (see below) for BIOS configuration and a 24 bit hardware breakpoint switch. Because the board was point to point wired and soldered it utilized two S-100 boards and connected to the front of my computer box with LED's and switches via a ribbon cable and connector

As I utilized the board over the years I really appreciated its capability to tell me at all times where and what the (master) CPU was doing. Nevertheless as time went by I started to think of further additions and improvements to the board. The board described below is the result of these considerations. Here is a picture of the board. Let me briefly outline first the components of the board.



The SMB Board
This S-100 SMB board has the following functions:-

Important S-100 Bus signals. The following S-100 signals are brought out as LED's:-

Memory Read, Memory Write, Port In, Port Out, Interrupt Acknowledge, Processor Halt, Phantom
RAM Addresses Greater than 64K, 16 Bit Data Request, Temporary Master 1 in Control.

These signals can be brought out to the front of the computer box via a ribbon cable connector or displayed on a daughter board attached to the S-100 card (see below). Here is a picture of the front of my system displaying the above indicator LED's.

Please see here for a description of a description of these important S-100 signals. By looking at these LED's you can immediately see if an S-100 signal is "stuck" high or low. Active high S-100 lines are red LED's, active low lines are green. Because many of the signals emit only a narrow pulse that would not be seen by the eye (for example INTA), the board uses a number of 74LS123 "single shots" to stretch the light pulse time so they are visible.

HEX LED display of all Data and Address lines. At all times the address of the master (or slave) CPU's program counter is displayed as a Hex LED display. If an 8 bit CPU is controlling the bus the current 8 bit instruction it is reading is displayed. If a 16 bit CPU is currently the master the 16 bit bidirectional data bus is displayed. Note because most 16 bit CPU's pipeline their instructions the data in and address bus may be slightly ahead of the actual instruction the CPU is executing.

Hardware Breakpoint Switches. The board has a row of 20 small dip switches and circuitry such that if the CPU ever goes to a specified address it will stop. From there it can then be single stepped forward one CPU instruction at a time displaying the current 24 bit address and 8/16 bit data in being read on the bus. This is very useful for both hardware and software debugging. Unlike software debuggers it is memory location and CPU type independent. It will work anywhere in the 24 bit S-100 memory address space with a master or any slave CPU.

Stop and Single Step Circuitry. This circuitry allows one to stop the current running master CPU at any time and restart from that location again or single step the CPU forward one instruction at a time.

Hardware Reset and Slave Clear Circuitry. In the past I have had problems with the length of time a low going reset signal was presented on the bus. For example the SD Systems 8024 Video board will not reset its onboard CRT controller in time if the signal is too short. It is also important to have a single sharp square wave reset signal for CPU's like the 80286. Finally many older S100 CPU boards do not have a Slave Clear option. For these reasons I have added two 555's to allow precise control of the S-100 Reset and Slave Clear functions. These can be triggered by push button switches.

An IOBYTE Port. This is a simple 8 bit switch/input port that allows the user to configure the console, printer reader etc. for IO in hardware. It's a carry over of the early Teletype days and was used in early versions of CPM, but it is useful is setting BIOS options because the data is not lost when

the power is removed. The port can reside anywhere within the range of E8-EFH

Master/Slave Switching Circuitry. This is an input port that allows one to bring low the S-100 TMA0 or TMA1 bus lines, useful for CPU master/slave switching. The port can reside anywhere within the range of E8-EFH

A System Tick Clock. This is just a simple timer that allows one to pulse one of the S-100 interrupt lines ~10 times a second. This is useful for CPM-86 background processing and multi-user systems. Again a 555 time is used but the "normal" circuitry is modified (with a diode) to give a narrow sharp low going pulse which can be jumpered to any one of the S-100 vector interrupt lines V0-V7. The clock it reset by the S-100 INTA or by inputting from a port (one of E8H -EFH) via a jumper.

A 2MHz signal for the S-100 Bus line #49. This line is not often used but when it is, it is typically used for things like a UART baud rate generator. The IEEE-696 specs specify exactly and always 2MHz. Some older S-100 boards tap into the master CPU clock (4MHz) and divide by two. This limits these boards to being always used at 4MHz.

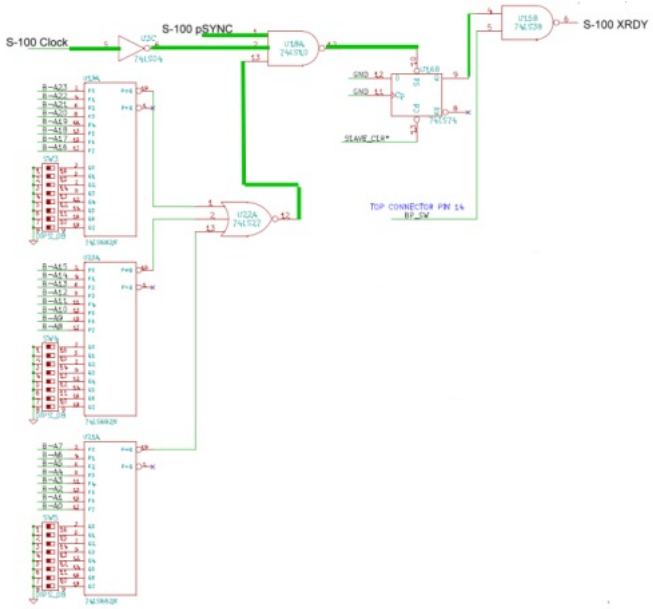
Inverse of the Master Clock signal (S-100 line #25). Some older S-100 boards count of the pSTVAL signal being a simple inverse of the main clock signal. The IEEE-696 signal is not exactly like this. The most well known example of this is the SD Systems Versafloppy II FDC board. There are modifications that can be made to that board to get it to work but the simplest thing to do is just supply the inverse clock signal on an unused S-100 bus line and redirect the Versafloppy to that line. I use the unused S-100 line #27, "RFU".

The Circuitry.

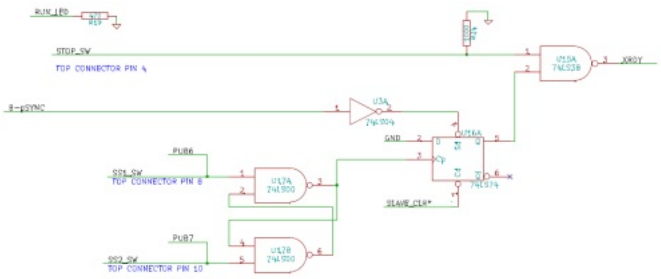
Unlike most S-100 boards describe here the electronic circuitry on the board is really quite simple. There is just a lot of it. All the indicator LED's or HEX displays are driven by inverters. Here is a full schematic of the board.

The board utilizes 74LS682's for port addressing. If your are unfamiliar with this technique click here.

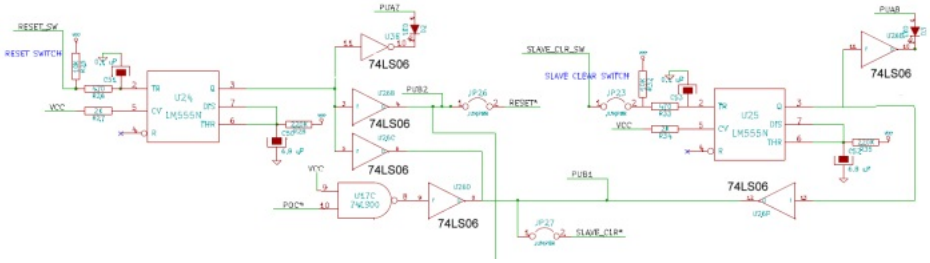
The breakpoint circuitry is set by adjusting the top leftmost 3 eight position dip switches to be 24 bit binary address you want the bus master CPU to stop at. It does not matter if its an 8080, Z80, 8086, 80286... It will stop the CPU if ever that address is accessed. The core of the circuitry is just three 8 bit comparators and a flip flop to latch the S-100 bus XRDY line low.



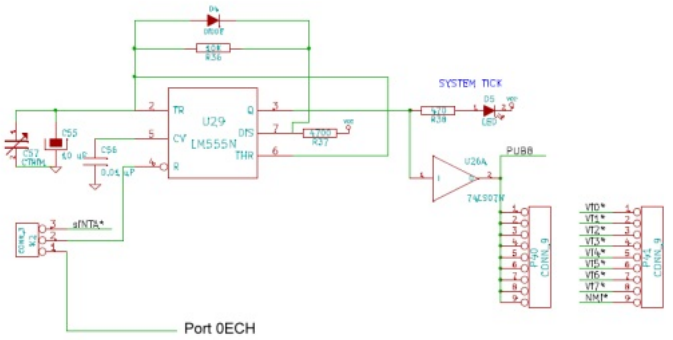
The Stop and Single Step circuitry is equally simple. Here it is:-



The 555 Reset and Slave Clear circuitry is shown here. Note Power On Clear (POC) also brings low the slave clear line - something some boards do not do:-



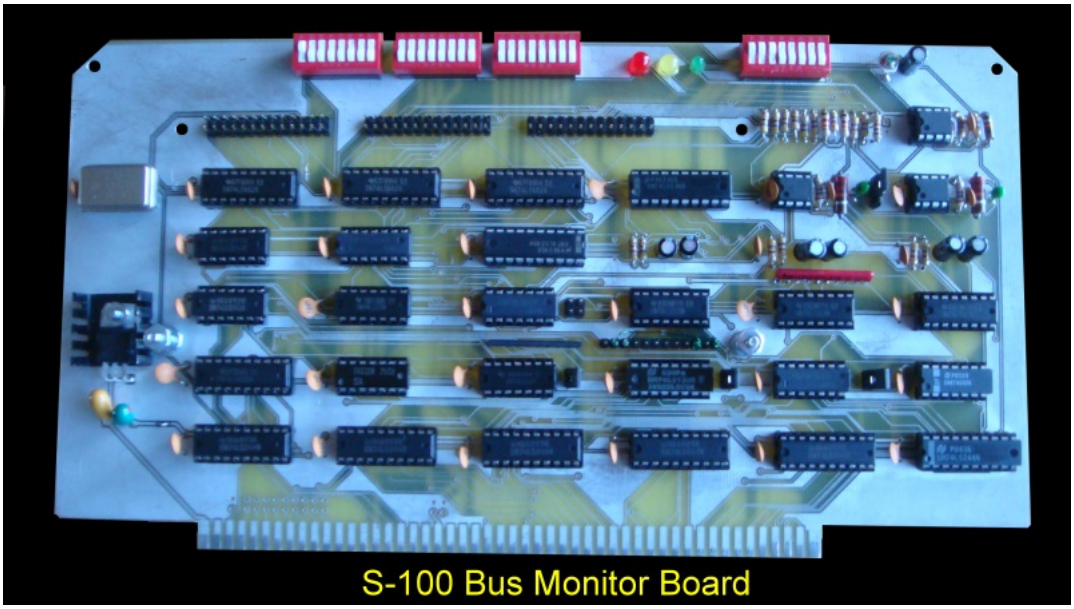
Finally here is the "System Tick" Interrupt clock circuitry:-



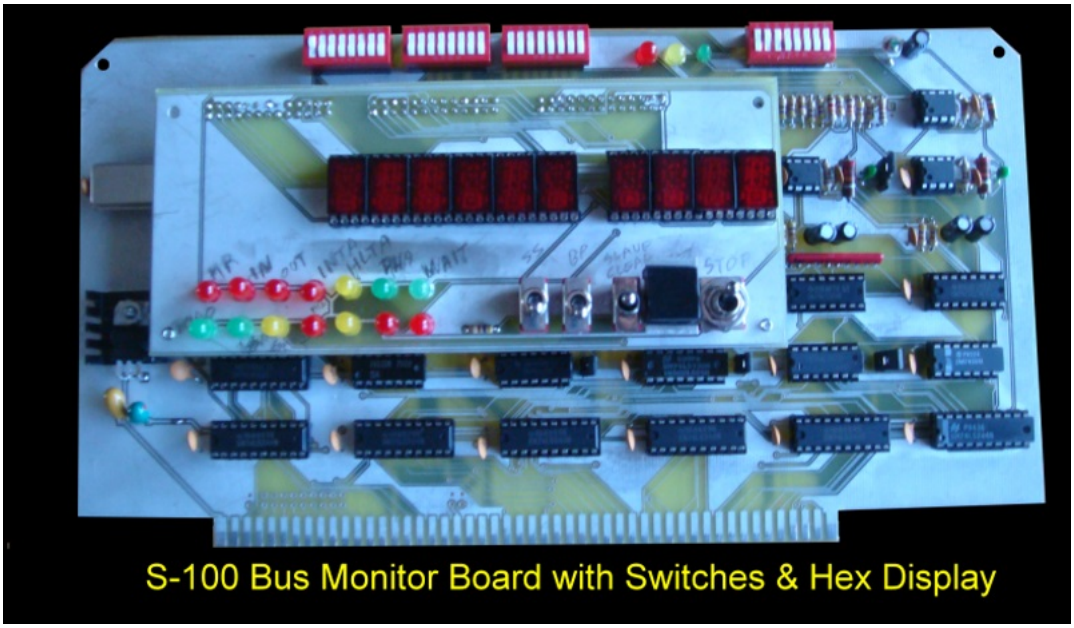
This simple circuit is used to generate a periodic pulse for an interrupt for things like CPM-86's interrupt driven system. It can be used as a CPM "poor man's" real time clock (the trim cap lets you fine tune the system) but I am currently doing a full blown PIC (Priority Interrupt Controller) and Clock/Calendar S-100 board which will eventually replace this component.

I collected all of the above and a few other things and squeezed them all on to one board.

Here is a picture of the prototype board:-



In my second (testing S-100 system), I do not have a front panel box to house the S-100 boards. In this system I attach the HEX displays (see below), LED's and switches on a small daughter board that is piggy backed on to the main S-100 board as shown here:-

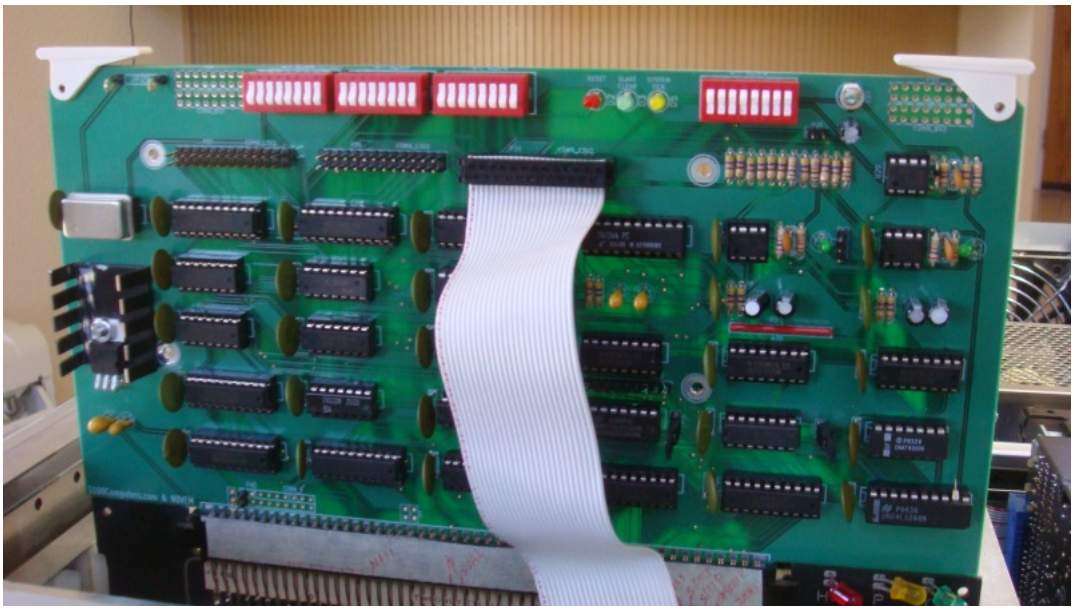


The 10 HP 5082-7340 displays are excellent displays for presenting the address lines and Hex data on 8 bit lines. They have their own internal circuitry to convert 8 bits into an HEX character 0-F. Unfortunately these displays are now fairly hard to find and somewhat expensive (typically $8-$12 each on eBay). Most modern similar displays unfortunately only decode 8 bits from 0 to 9!

Understanding that not everybody would want such a luxury, I have decided to make three versions of the board. In all three cases the actual LED display itself is on a small daughter board that attaches to the front of the single main S-100 board. One daughter board is designed to accept the above HP 5082-7340 HEX displays. Another board (exactly the same dimensions) will accept the common bar graph LED's. These work by displaying the 24 address and 8/16 data lines as binary bits. There is still another daughter board for the somewhat easier to find (and cheaper) TIL 311 HEX displays. All three daughter boards also contain the necessary switches to stop, reset and single step etc. the CPU.

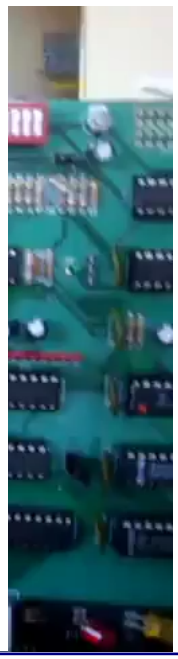
In my case (and I'm sure for others) I wanted to bring out the HEX display and switches to the front panel of my S-100 box. To allow this, the board has a three ribbon cable connectors which connect directly to the switches and HEX displays.

A Second Prototype.
A second prototype board was done to improve on some of the functions of the first board. Here is a picture of that board.



As you can see it almost the same as the first board but has a few minor corrections and is constructed on a commercial layout/material. Construction of a final board is now under way.

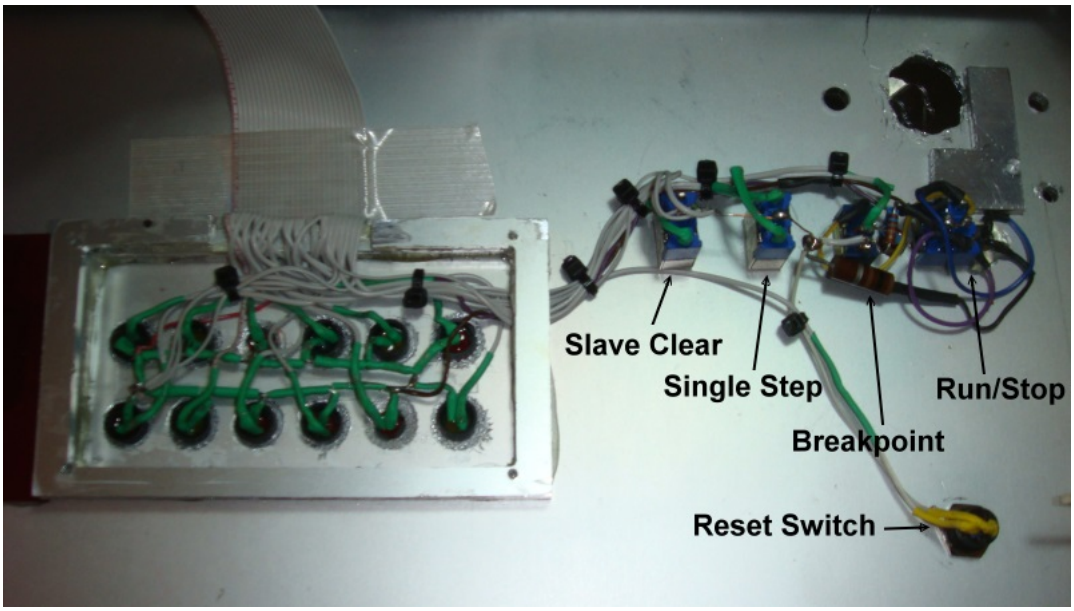
The S-100 Bus SMB in Action:-
The short (amateur) video below shows the S-100 SMB board in action. The system is powered up. CPM-80 is first booted from the IDE S100 Board. Then CPM-86 is then booted from the same IDE board. I then stop the CPU and single step along the 8086 one instruction at a time. You can see the address and 16 bit data lines being displayed. I then reset the computer and do the same thing with the Z80.



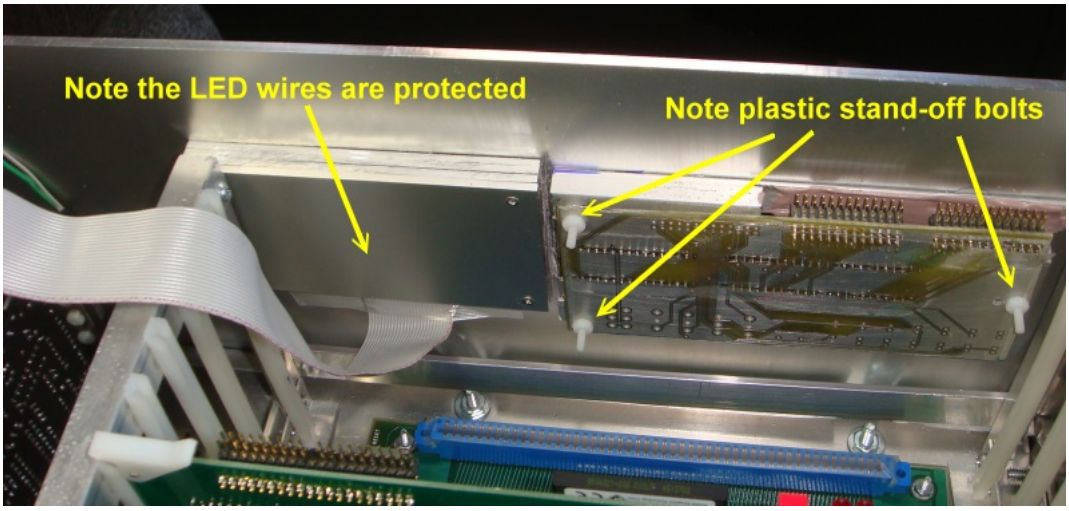
To play the video click on the arrow within the window.
(Note: You must have Adobe Flash installed on your computer to see this video. You can download the latest Adobe Flash Player for your Browser here).

External LED's and Switches

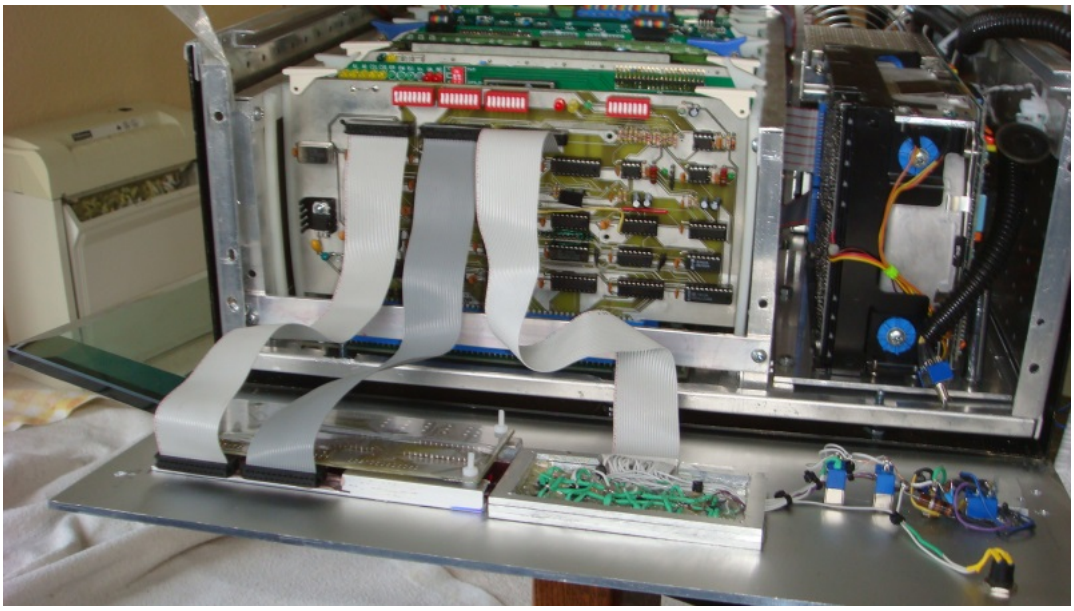
In my main S-100 system, which is a custom hand built box, I have brought the bus indicator LED's out to the front panel of the box. I also relocated the HEX display LED's to a small window cut into the front of the box. Finally the switches were also brought to the front of the box. This required that three 26 conductor ribbon cables with sockets be connected to the board. Below are a few pictures of my construction process.



It is obviously very important to make sure all wires are insulated and not exposed. The LED's tend to be fragile and the wires can break over time. For this reason they are enclosed in their own "Box", sealed and covered.



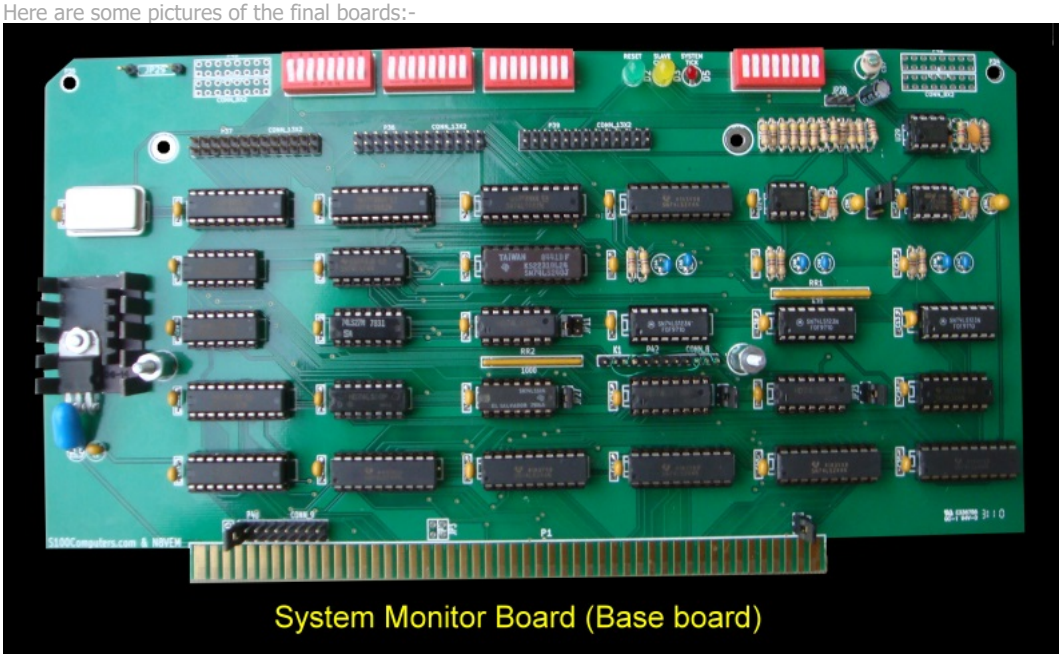
Below is a picture of the complete set-up before the front panel is re-attached to the system.

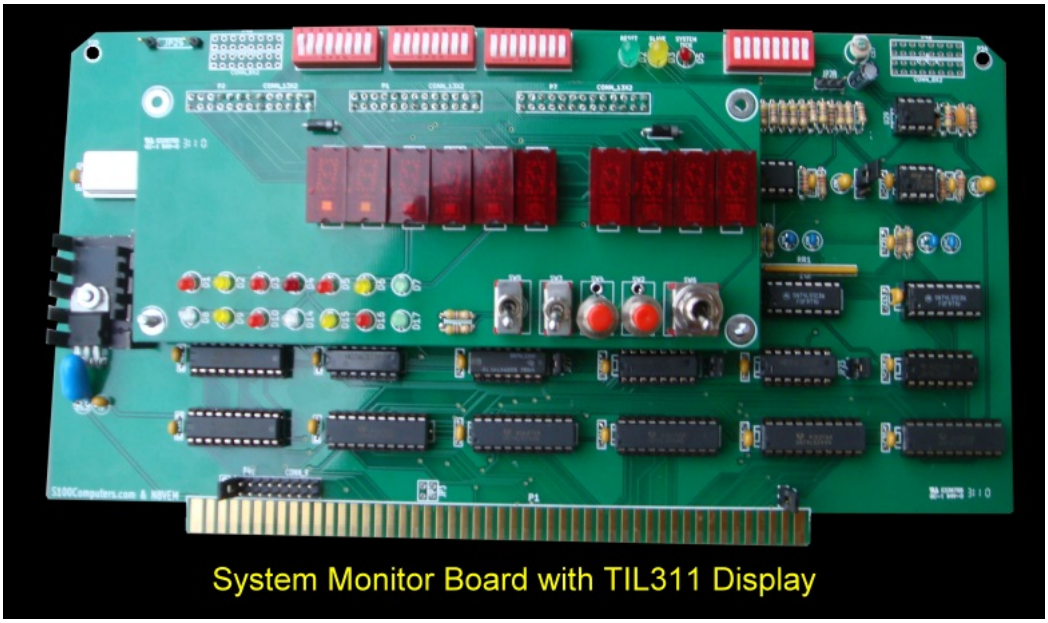


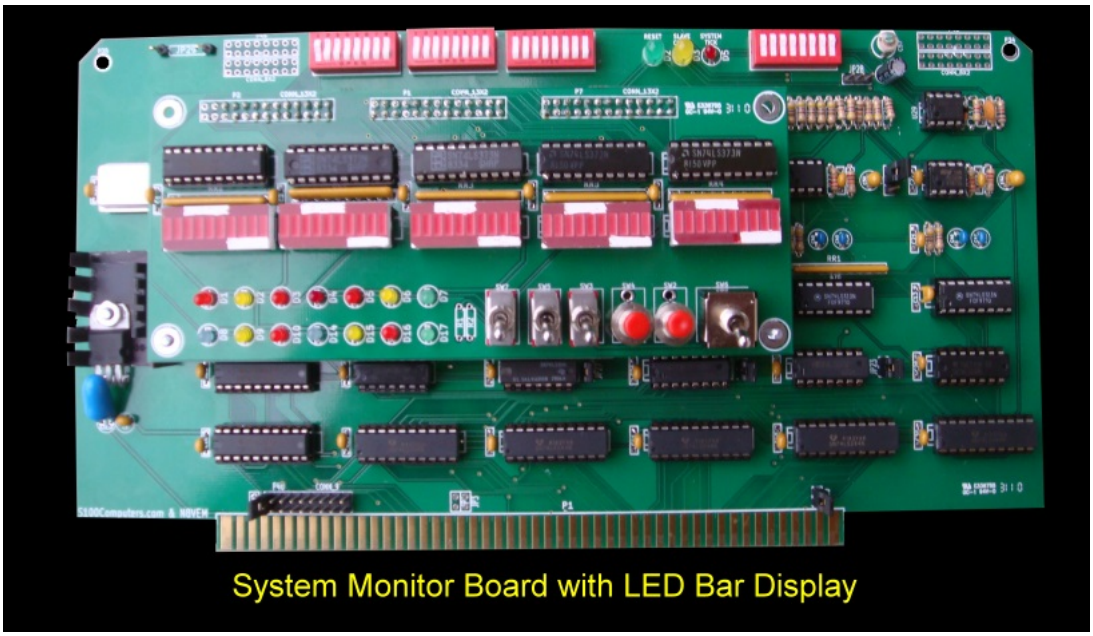
Below shows the final arrangement.

The final schematics for this board and the daughter add-on boards can be downloaded from here:-
SMB Schematic
SMB Board Layout
HP-5082 Display Board Schematic
TIL311 Display Board Schematic
LED Bar Display Board Schematic

Here are some pictures of the final boards:-







Assembly Notes for those constructing the board.

Assembly of the board is fairly straightforward. That said, here are some useful hints:-

While the logic on the board is simple there is a lot of it. The traces on the board are narrow and closely spaced. Take care not to scratch the board when soldering connections. The solder mask on the board is excellent so there should not be a problem. In splitting apart the three display boards

take care in cutting the small connecting bridges not to damage the nearby traces. When soldering in caps and resistors, after placing them in the board, solder in one side first for all then the other side. This way the component has time to cool.

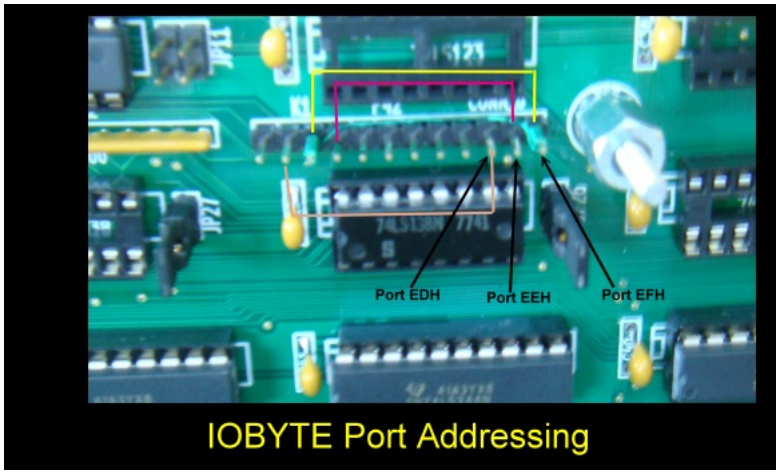
All components can be commonly obtained. I obtained mine from Jameco (except the 74LS682's unfortunately -- which I use a lot and got as a batch from DigiKey). Here are Jameco catalogue numbers of some components:-

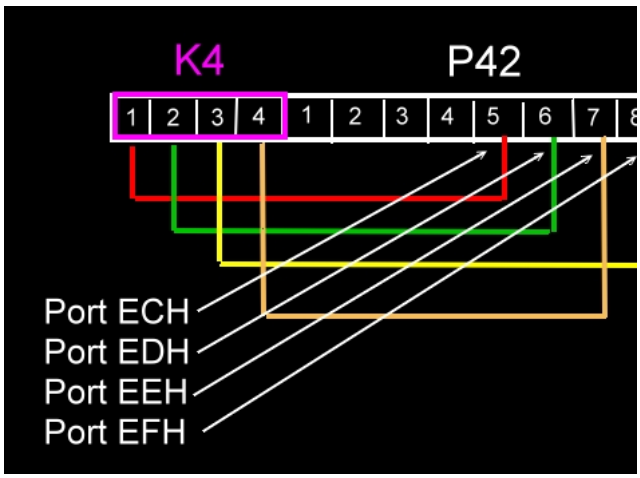
1.5A, 5Volt voltage regulator #924570
2MHz Oscillator #27924
TIL 311 displays #32951
10 Bar LED Display #697477
470 Ohm Resistor pack #97869
1K Ohm Resistor pack #78777
Trimmer Cap #32839
Dual Row jumper headers #67821
Single Row Jumpers (cut to size) #527654
Bypass capacitors #25525
Note, this is not a complete list.

First solder in all IC sockets, resistors, capacitors and jumper sockets. You should use a 1.5A, 5V voltage regulator. The TIL and HP displays use a fair amount of power. I like to keep my voltage regulators (and boards) cool, so I used a somewhat larger heat sink. In order for it to fit I had to drill a hole in it left of center. This way I could actually place one cooling fin within the larger one. See below. Make sure the cooling fins do not contact the add-on display board. See the picture above.

Before you add any IC's place the board in your system. Check it starts up normally. If not find the solder bridge.
Next add the Voltage regulator and the U29 555. This is the system tick circuitry. Insert the D5 LED and check that it flashes rapidly (1-3 times a second). Make sure you have the LED polarity correct. The longer lead to the square pad. Jiggle them around to check before soldering them in.
Then add the other two 555's (U24 & U25) as well as U17 and U26. Also LED's D2 and D3. Insert jumpers JP27 and JP23. Then **carefully** touch the connector P39 pin 12 (RESET_SW) or pin 18(SLAVE_CLEAR_SW) to ground. The system should reset (pin#12) and the appropriate LED's should come on briefly.

Next we will add the IOBYTE circuitry Input port address and Master/Slave switch circuitry. We will do this in stages. First add the bus drivers, U8,U9,U11,U12,U13 and U14 (do not add U28). Place the card in the bus and check the system is OK. Next add U18,U3,U27 and U28 and switch SW9 if it is already not there. We will input data from the switch settings from a port in the range (E8H to EFH). You can choose your own. I will use port EFH as the IOBYTE port (EDH as the master/slave switch (see below) and port EEH to reset the timer tick (see below). To do this we need to carefully jumper connections between connector P42 and K1 (both are side by side). Currently I use Port EDH to switch in my 8086/80386 boards and Port ECH to switch in my 68K CPU board. (In theory up to 16 CPUs can reside on the S100 bus). I prefer to use wire wrap connections with a wire wrap tool but solder connections are also fine. Here is an enlarged picture:-

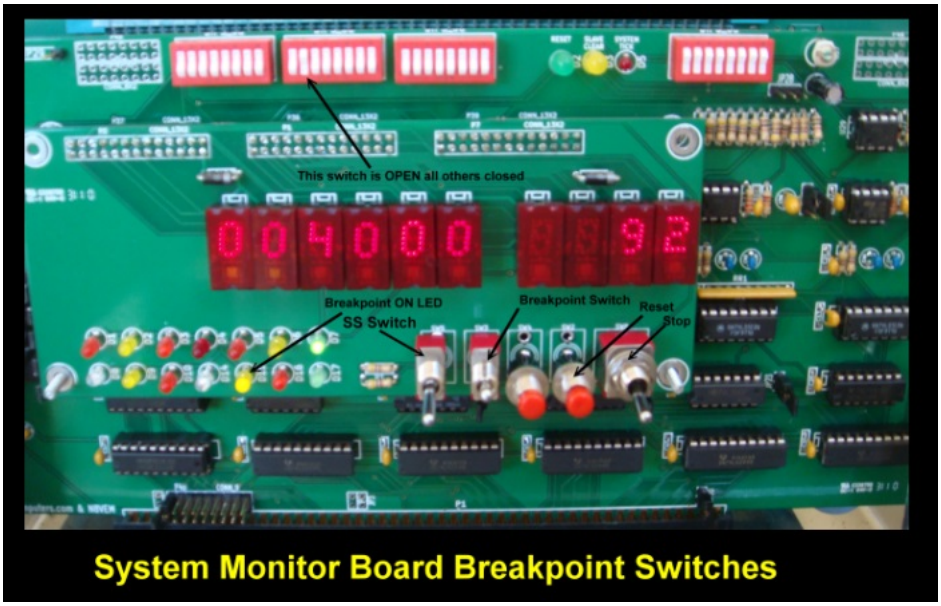




After you have setup the above connections insert the board in your system and input from port EF. The value returned should reflect the switch settings of switch SW9. Normally they are closed returning a value of 0FFH. You can use various bits to redirect your BIOS outputs as discussed above. There is circuitry on this board to control the S-100 TMA0 and TMA1 lines. These are used to switch in other bus CPU's such as an 8086. We will use them later with future boards. For now insert U2 and U15. Inputting from port EDH should bring the S-100 line #55 low. (The connection for port EEH is to reset the system tick when utilized with interrupt driven systems like CPM86).

Next insert all the rest of the IC's on the board. Assemble the display board. Be sure to get the polarity of the LED's correct. If in doubt don't solder them in yet. Just tape them in and we will jiggle them around. Attach the Display Board. Take care to make sure the back side of the display board is not contacting anything on the S-100 board itself. Insert in your system. Check that the appropriate LED's are lighting up correctly. For example inputting from any port should flash the PORT IN LED. Placing 76H in RAM and jumping to that address should cause the HALT LED to stay on.

Next check the Single Step circuitry as I described in the above video. Finally you can test the hardware breakpoint circuitry. Setup the board breakpoint switches so that all are closed except the one shown here in the picture.



System Monitor Board Breakpoint Switches

Flip the Breakpoint switch up and reset your system. Then display memory from say 3F00H to 4100H. The computer will stop as shown above at address 4000H. Flip the Stop switch up. Then lower the Breakpoint switch. Then by flipping the Single Step switch up and down you can step (the current S100 CPU bus master), one opcode at a time from this location. What is nice about this arrangement is that it is software independent and will work with any CPU anywhere in the 16MG address space of the S-100 Bus.

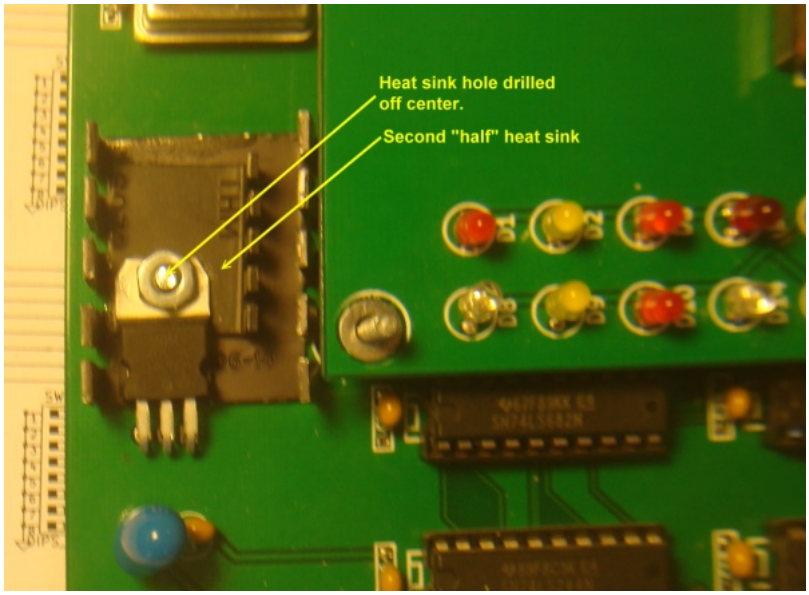
There are a few jumpers on the board to add flexibility for different systems:-
JP30. This puts a 2MHz clock signal on S-100 pin #49. Some old CPU boards do this by reducing their clock speed. The signal is sometimes used by UARTS/Serial IO. By adding it here you are free to run your CPU board at any speed you like.
JP11. Older CPU S-100 systems generated the MWRT signal from the Front Panel circuitry. Almost all later CPU boards generated it on-board. If you need to use an older CPU card use this option.
JP10. This is a specialized jumper for older boards that count on S-100 pSTVAL signal being an exact inverse copy of the clock signal. I discussed this above.
P41/P41. This jumper block allows any one of the S-100 eight interrupt lines to carry the system tick interrupt signal.
JP28. This is a connection if you need one to activate a Hard Disk indicator LED. Normally used if you are bringing your indicator LED out to a front panel *via* a cable as described above.
JP26. Jumper, if you wish this board to generate a S-100 bus reset signal. If your box/motherboard does it for you no need to be connected -- though the 555 here will create a nice long reliable signal.
JP23. Jumper for a separate S-100 Slave Clear reset signal. Normally jumpered as few if any boxes/motherboards brought this signal out.

Note the LED Bar display board has an extra switch to the left of the SS Switch. For this board you need to switch on/off the 8/16 data lines display manually depending on which CPU you are using. The HP and TIL displays have an internal latch, so this is done automatically.

Tracing Through EPROMS's
This board can be used to single step through EPROMS (or EEPROMS) and set a breakpoint at an location within the EPROM **provided** the EPROM is "visible" on the S-100 bus to the SMB. In most cases you will not be able to trace through code that resides in an EPROM that is local to (say a Z80), on its own S-100 CPU board. This is because the S-100 signals required to read such an EPROM are usually not actually placed on the S-100 bus, (There is no need, it simpler and faster for the Z80 to locally address the EPROM). The SMB board has no way of seeing the required EPROM data. This has the advantage that in a multiprocessor S-100 system, (say with an 8086 CPU), that the Z80 PROM does not "create a hole " in the 1M address space. It is completely invisible to the 8086.

If you wish to debug Z80 code for a EPROM, the easiest thing to do is assemble a debugging version at a low RAM location, test it there using the SMB, and then burn an EPROM for the final location.

The Voltage Regulator Heat Sink.
Two things, first the hole for the 5V voltage regulator is quite far to the left on this board (so the daughter board will fit). This means that to use a reasonable size heat sink you need to drill a new hole in the heat sink a little off center. See below.



Second, because of all the Hex Display LED's the Voltage regulator on this board runs a little hot. Not to the point of damaging the regulator. However I like to run a cool box. I actually add a second heat sink vane within the larger one to dissipate the heat more effectively.

Note. Please use a 1.5 Amp. regulator (e.g. Jameco #1130069), not an 1 Amp. one. Also C51 and C53 are regular 0.1uF filter caps (not electrolytics).

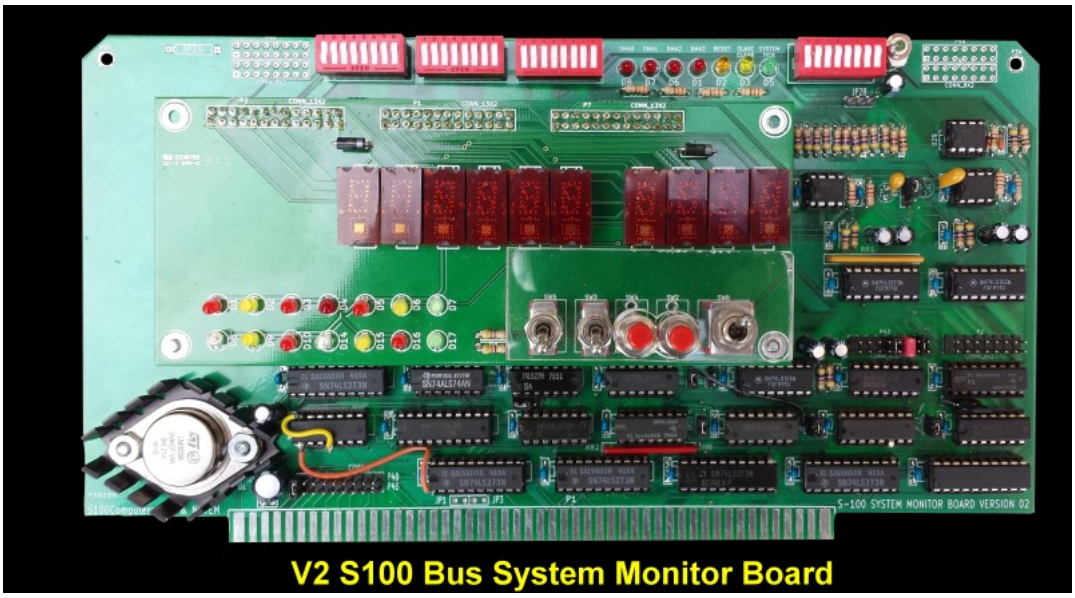
A Prototype V2 Version of the S100 Bus System Monitor Board.

The above board has proven to be popular and reliable. Over 20 boards were already made and distributed to users. I decided to add a few small changes to the board to improve it's functionality. We are calling this board the **V2-System Monitor Board**.

The changes/additions are:-

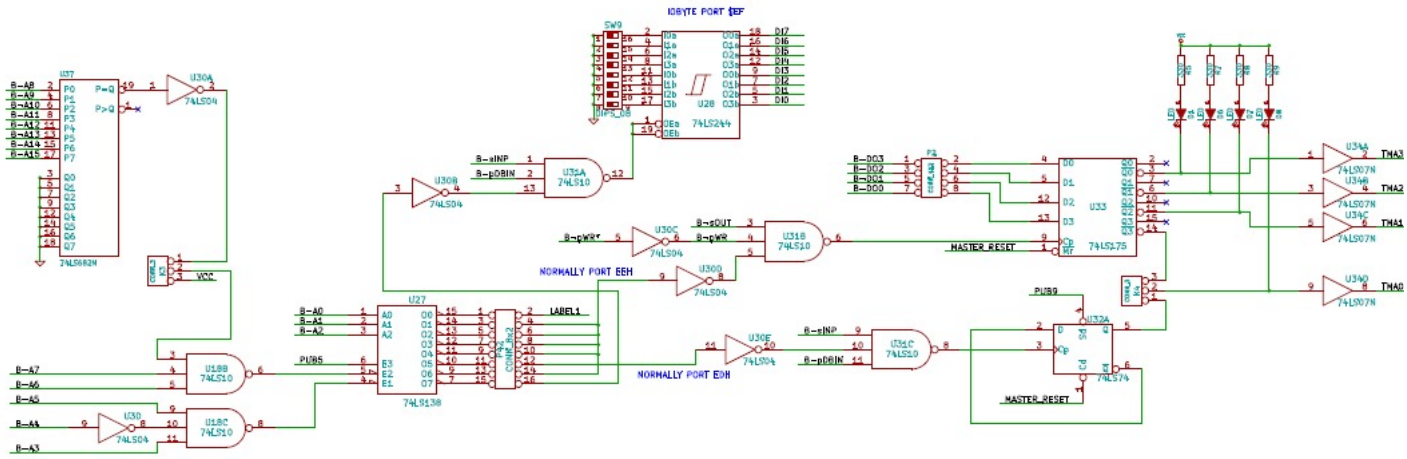
1. Add a circuit to allow any combination of the four TMA0-3 lines to be activated. This allows up to 15 slave devices on the bus
2. Latch all address lines for better display reliability with high speed CPU's
3. Optimize the reset circuit
4. Optimize the single step circuit.
5. Add a circuit to ground address lines A16-A23 when (older) non IEEE-696 CPU boards are the bus master.
6. Switch to a 3A TO-3 voltage regulator

Here is a picture of the new V2 Prototype board:-



Please excuse the few wire patches. They will be removed in the final version of the board.

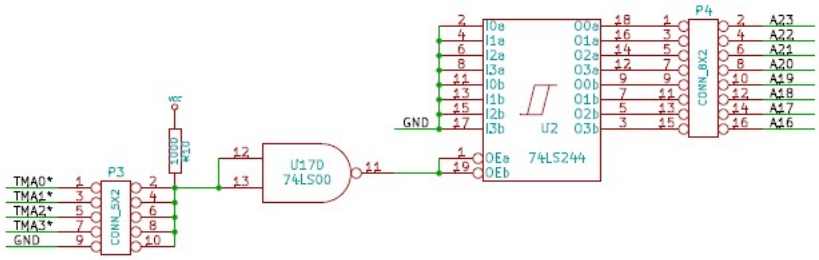
The main change to this board was the addition of a circuit to lower any combination of the S100 busses TMA0-3 lines. This will allow you in theory to have up to 15 temporary bus masters/DMA controllers. The old board had two simple one bit 74LS74 flip-flops which when one inputted from (the default) 0EDH or EEH ports activated TMA0* or DMA1*. This limited the busses ability to handle DMA controllers. Also while some of our master/slave CPU boards had their own circuit to do this having one central location/port to handle all the bus controllers in one spot makes sense. Also the port now can be an 8 or 16 bit port address.
Here is the relevant circuit:-



As you can see on this board we output to a port (default EEH) an 4 bit value. In order to allow the transition of older software the single bit activation by inputting port (default) EDH is also retained with a jumper (K4). For convenience I have also include a LED to indicate the active TMA lines.

One thing I found with high speed CPU's (68000 & 80386), was the need to latch the address lines. Apparently the HP5082 Hex displays were not able to reliable latch these signals in single step mode. The address lines are now latched for display with 74LS273's (and pSYNC/pSTVAL).

Another small addition was the inclusion of a circuit to allow old S100 bus CPU boards to work correctly with non IEEE-696 boards that do not control the high address lines A16-A23. With our own, and other RAM boards that decode all the address lines, such CPU's would either write into every 64K RAM block in the 16MB address space or some cases not at all, depending on how the high address lines float or are jumpered. This becomes a problem if other CPU's are also in the bus. Here is the relevant circuit:-

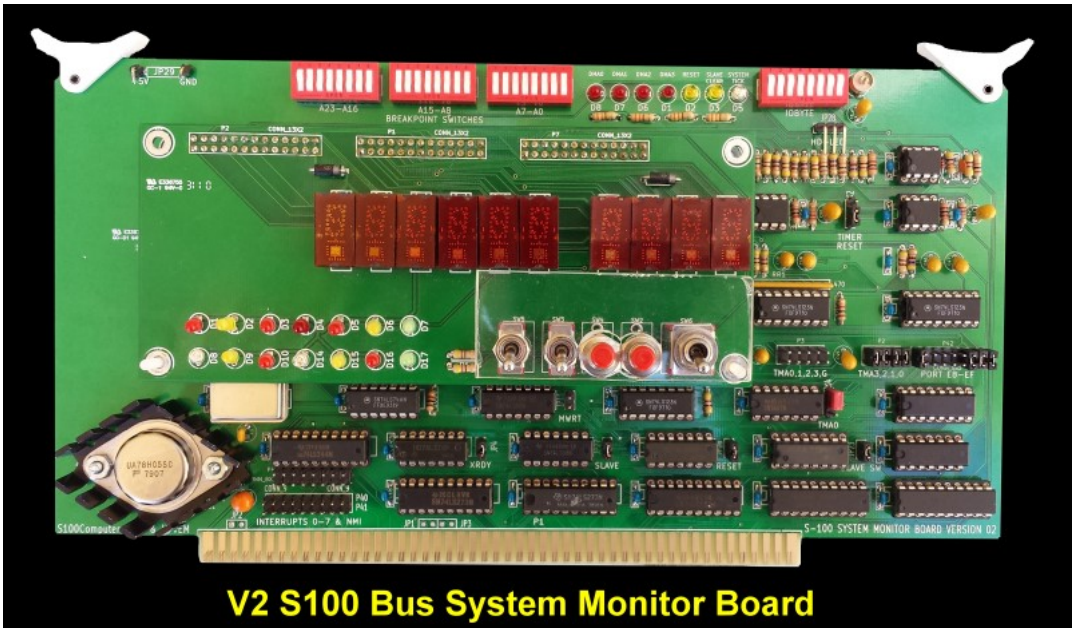


If none of the TMA0-3 lines are active (low), and all the P4 jumpers are installed, then the upper address lines will always be low. This for example would be required with the Cromemco ZPU Z80 CPU board if you want to utilize the "Dazzler" video board, see here. When you switch to another (usually a 16 bit) CPU, with the appropriate P3 jumper in place, then U2 is inactivated thereby allowing that CPU to control all the address lines.

Finally you will notice we added a T0-3 5V regulator. These regulators are a little bit more expensive and harder to find, but in this case, because of the high current requirements of the HEX displays a 3A regulator with a large heat sink makes more sense. Those old regulators just run cooler that the newer TO-220's.

Other than the above, everything else is the same as for the original SMB.

Step By Step Building the V2 SMB Board.



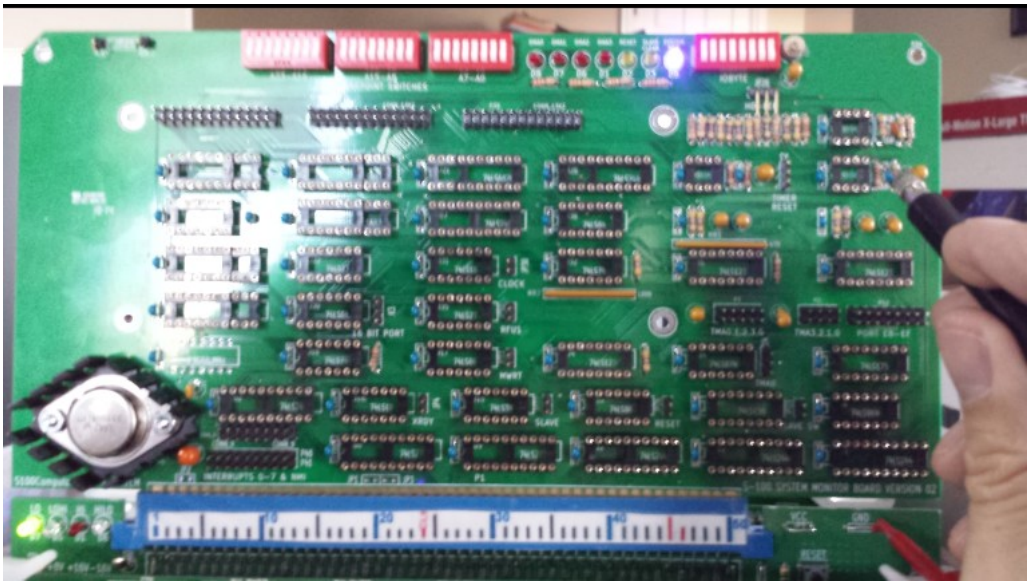
As always, first examine the bare board carefully for scratches or damaged traces, use a magnifying glass if need be. The quality of the boards we get is excellent. I must have done 50 by now, never had a problem, but there is always a first time. A broken trace is almost impossible to detect by eye on a completed board. I find it useful to carefully slide a file at 45 degrees along the edges (front & back) of the S100 connectors for easier insertion into the bus. Carefully, just one or two strokes.

Next solder in all the required IC sockets, resistors, resistor arrays, capacitors, jumpers, switches and the 5V voltage regulator. Because this board utilizes a fair amount of power to drive the Hex displays so I prefer to use a 78H05 regulator. Be sure you put the two resistor arrays are in with the correct orientation of pin 1. Check their values before soldering (they are difficult to remove).

For prototype boards I generally use "double swipe" IC sockets. For a critical board like this I prefer to use "Machine Tooled" IC sockets. However they are more expensive and you have to be particularly careful not to bend the IC pins.

Next insert all 7 LED's. Before soldering them in place ground the appropriate pin on the U33, U3, U29 and U26 sockets to make sure it lights up. I use blue for System tick,, yellow for Reset and Slave Clear and red for the 4 TMA lines.

Check the voltage to sockets on the board is above 5V by placing the board in your S-100 system using an extender board. With no load you will typically get 5.00V (+/- 0.25V). BTW, your system should boot and run correctly with its Z80 master CPU board. If not, you have a serious solder bridge somewhere on the board. Before you do anything else with a magnifying glass go over every socket on the board and examine for a proper solder joint. I like to "reheat" each joint just to be on the safe side. The silk screen/varnish on these boards us quite thick. It's easy not to have a good solder joint for the ground pins. Double check. Extra time here will save you hours later, (been there done that!).

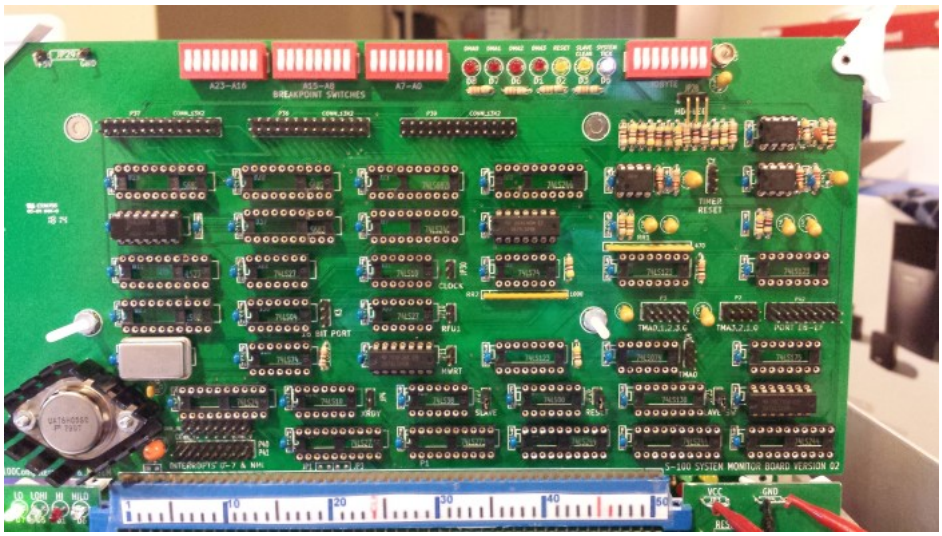


We will now build the board up in functional steps. Avoid the temptation of adding everything at once and popping it into your S-100 box. Step by step is faster in the end -- trust me. This assembly instruction assumes you have the earlier (V1) board and will be using one of the mezzanine boards that came with it. If not build one of those boards first.

Install the 2MHz Oscillator. You can solder this one in it will never need to be changed.

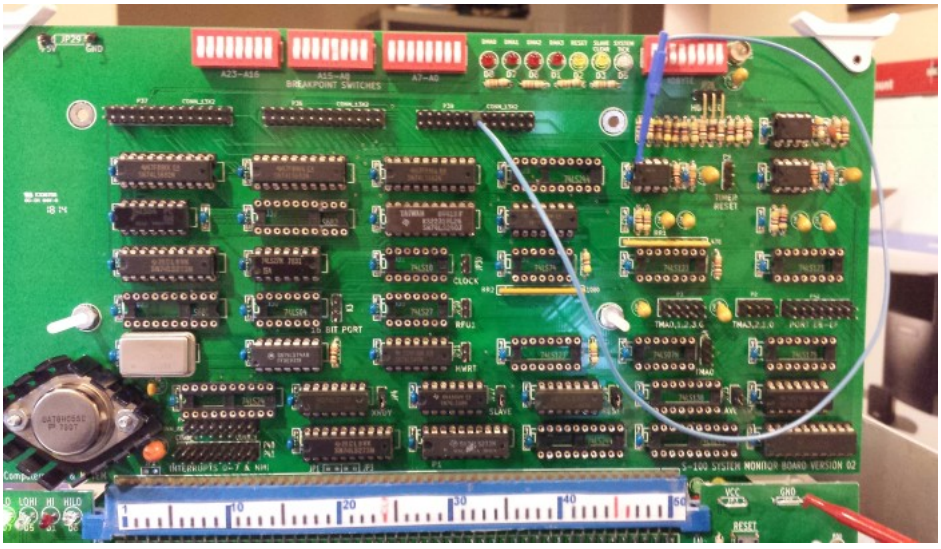
First we will install the reset, slave clear and system tick circuits.
Install the three 555's , U24, U25 & U29.
Add U17, U26, U36 & U3. Insert JP23, JP26 & JP27.

Insert the board into your system and power up. The System tick LED D5 should be flickering. Grounding pin 12 of P39 should light up the reset LED (D2). Grounding pin 18 of P39 should light up the Slave LED (D3). Here is a picture of the board at this stage:-



Next we will add the breakpoint, stop and single step circuit.
Add U22, U18, U16, U15, U14, U8, U9, U11, U19, U20, U21, U7 & U10. Insert jumper JP4.
Insert the board into your system and power up. Then carefully ground pin 14 of P39.

We will set a breakpoint in RAM at 800H. So set all the breakpoint switches to closed except switch 5 of SW4.
In your monitor display RAM 0 to FFFFH. The CPU should stop just before 800H is displayed. Later you can single step from this point onwards.
Note the ground connection on pin 14 should be made after the Z80 monitor signs on. This is because our Z80 CPU board on power on, scans the address lines from 0 up, to find the monitor ROM at F000H. It would pass through 800H and lock up. Here is a picture of the board at this stage:-



Next we will add the IOBYTE port and TMA selection circuit.
Add U37, U18, U27, U31, U28, U32, U33, U34, U27, U12, & U13.
Jumper K3, 1-2. Jumper K2 2-3. For P2 connect 1-2, 3-4, 5-6 & 7-8

BUGS

There is one minor error on this board. Pin 1 of K4 should be connected to pin 6 of U32 (not pin5). To correct, bend out pin 5 of U32 and on the back of the board connect U32 pins 5 & 6 together.

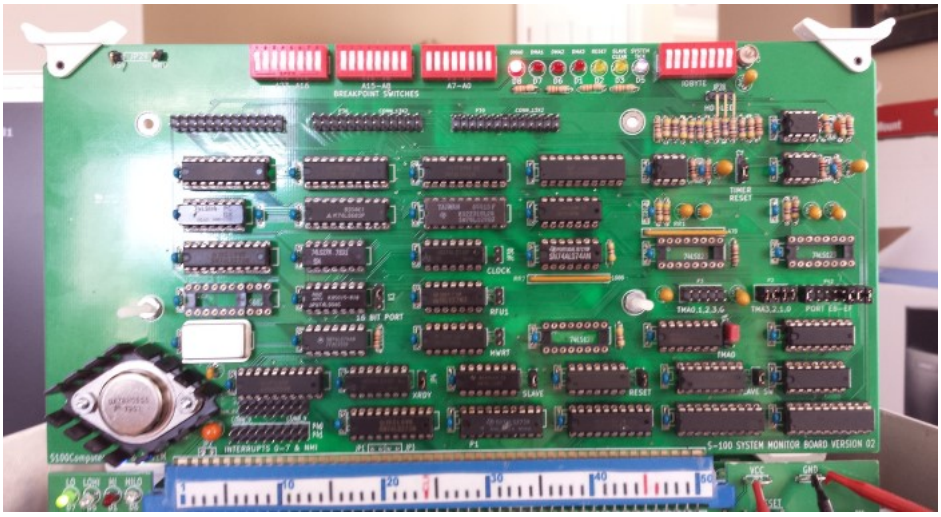
I also recommend you use a 74F138 and 74F10 for IC's U27 and U18. The problem is U31B is not getting the address lines up in time on its pin 5 to clock the correct data into U33 when you use a fast CPU like the Cyrix 80486 on our 80386 board.

Port Addressing

We will set the IOBYTE port to EFH, The new TMA select port to EEH and leave the old TMA0 select port at EDH. With the K4 jumpered to 2-3 we will be selecting our 8086 CPU based slave boards by outputting bit 0 as a 1 to port EEH. If you wish you can use the earlier approach/software by inputting (any bit) from port EDH. For this approach jumper K4 1-2. Remember in the latter case only TMA0 can be activated. Finally we will allow a reset of the U29 system tick (used for CPM86) by inputting port E8H.
So we add the following jumpers to P42, 1-2, 11-12, 13-14 & 15-16.
To check, insert into your system and power up. Input the IOBYTE port (EFH), and check the bit pattern received when you change the switches of SW9. (Note my Z80 master monitor looks at this switch after a reset, some bit patterns may redirect the console I/O).

Next check you can switch on/off the four TMA LED's. Outputting a 1 to port EEH should activate TMA0 (LED D8). If you have our 8086 CPU board in the system, control will be transferred to it. You should be able to activate individually each of the four TMA LED's. Here is a picture of the board

at this stage:-



Finally we will add the mezzanine display panel circuit.
Insert U23, U4, U5 & U6. Since I occasionally use the Versafloppy II FDC board, I will connect JP10 (to supply an inverted Phi signal on the S100 bus line RFU1).
Your board is now completely assembled and should behave as shown in the video above.

All components can be commonly obtained. I obtained mine from Jameco. Here are Jameco catalogue numbers of some components:-

2MHz Oscillator #27924
TIL 311 displays #32951
470 Ohm Resistor pack #97869
1K Ohm Resistor pack #78777
Trimmer Cap #32839
Dual Row jumper headers #67821
Single Row Jumpers (cut to size) #527654
Bypass capacitors #25525

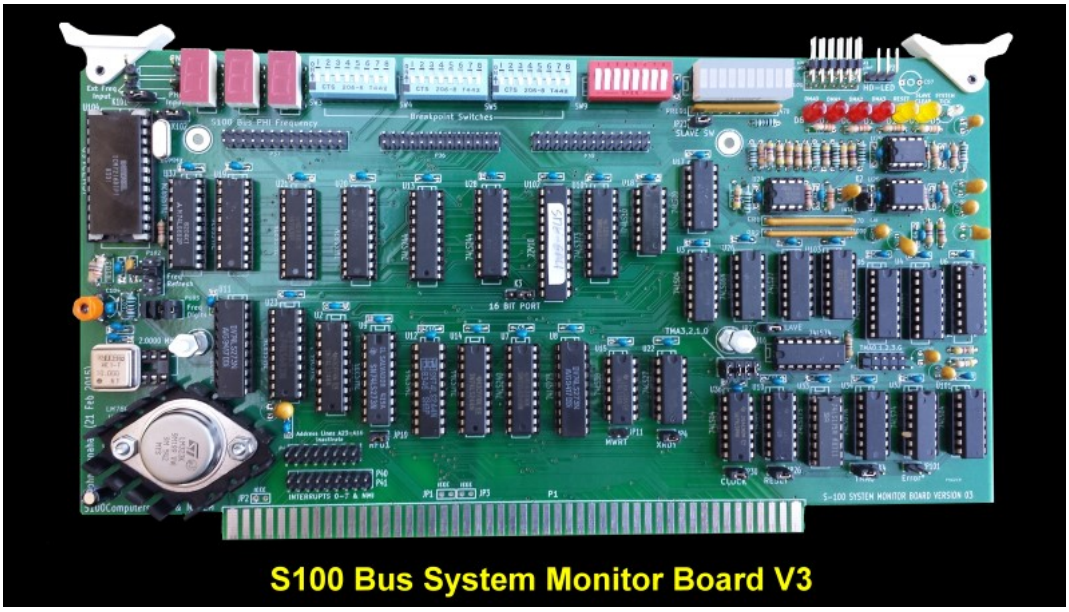
Note, this is not a complete list. The TO-3 78H05 's are hard to find. You can try the Jameco 3A LM323K's (#23667). However I had some units in the past from them that did not put out the correct voltage. Check your voltage before inserting the IC's as mentioned above.

Most of the IC's use on this board are common 74LSxx chips available from numerous sources such a Jameco, Mouser & DigiKey. A great source of chips like these is Unicorn Electronics who have very reasonable prices - though the chips are hard to find on their web site. Here in the San Francisco bay area I recommend Anchor Electronics. They take phone orders as well and are well stocked.

The TIL 311A's are expensive (~$20 each from places like Jameco). However overseas outfits like Utsource sell them in small quantities for ~$4 each. Utsource have about a 1 week delivery time to the US since they use DHL. It is indeed frustrating these LED HEX displays are so expensive. With the exception of the even rarer HP-5082's other such displays display only the digits 0-9. An alternative approach would be to use a 7 segment LED decoder such as a DM9368 and a simple 7 segment LED display. However these too are no longer normally stocked. Normal single line flat panel LCD displays just do not show up well when the card is in the S-100 bus card cage.

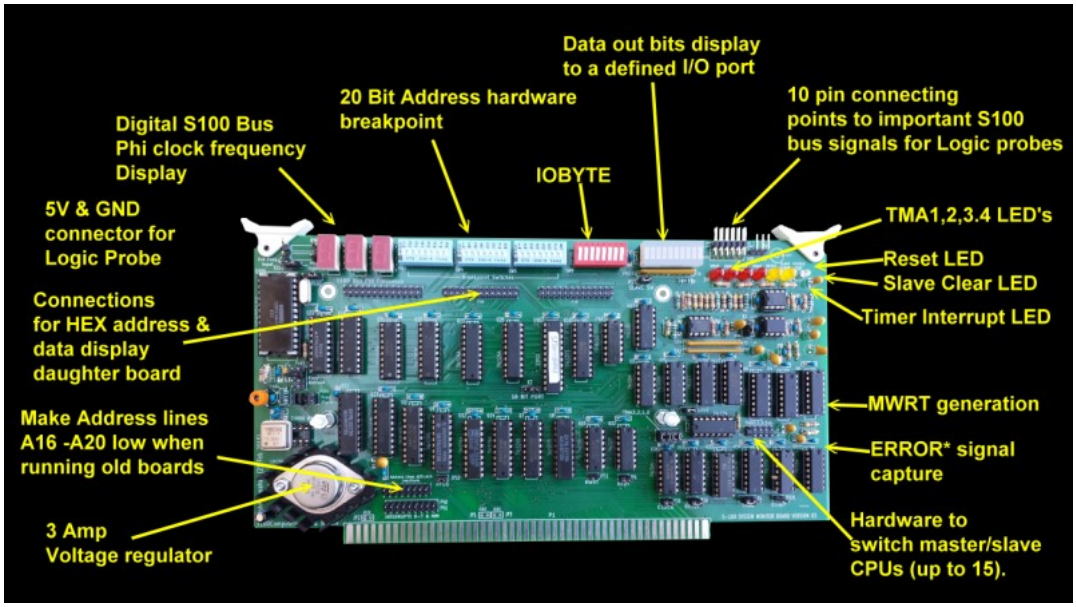
A V3 Version of this SMB Board.

The above two boards have worked well over the years and quite a number of them have gone out. It's time to consider doing another run. Before doing so I thought it would be nice to add a few extra bells and whistles in the little extra space that is on this board. If I have time I would completely redo the board and probably drive most of the circuitry with 4 or 5 GAL's, but I wanted to keep the changes to a minimum so that current users could just switch over chips to the new board. Here is a picture of this board (without the attached daughter board).



Changes:-
The was a minor error on the above V2 board (pin 5 of U32), this is now corrected.
There was a report of the TMA 0-3 switch circuit was not working reliably with the 80386 board at high speed
this circuit has now been completely replaced/simplified with a single GAL.
The LED's for the S100 bus signals TMA0-3 now display the actual state of these signals on the bus (not the output of U33).
An Intersil ICM7216D chip has been added along with 3 new Digital displays to continuously display the frequency of the bus master clock speed (PHI) in MHz.
A circuit and LED bar was added to continuously display the bit pattern of a user defined output port. (default, LED Diagnostic Port 05H).
A circuit was added to trigger an interrupt if the S100 bus ERROR* signal (pin 98) is activated.
A series of jumpers was added to easily attach a logic probe cable (such as a USBee or Saleae unit) for hardware debugging.
The LM323K 5Volt, 3Amp regulator can be replaced (in the same footprint/space) with a Pololu 5 Volt , 2.5 Amp D24V25 Switching regulator.

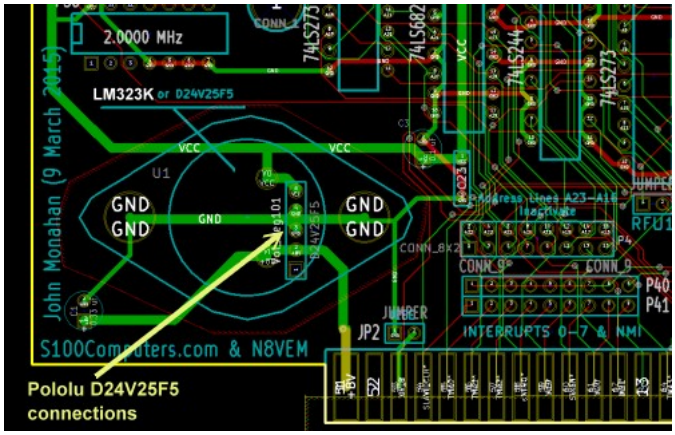
Here is a summary of the main changes:-



All the above facilities of the V1 and V2 boards were retained and are essentially unchanged.

The Pololu 5 Volt , 2.5 Amp D24V25 Switching regulator is a new addition to our boards. These switching regulators have a great reputation, no heat sink is required and take up less space than a nickel on a board. In this and most future boards I will include them as an option. Here is a picture of the regulator:-

Pololu 5V, 2.5A Step-Down Voltage Regulator D24V25F5.



There is also a D24V59F5, a 5 Amp version. A 5 pin header (0.1" centers) easily fits into the TO-3 footprint allowing either unit to be used on the same board.

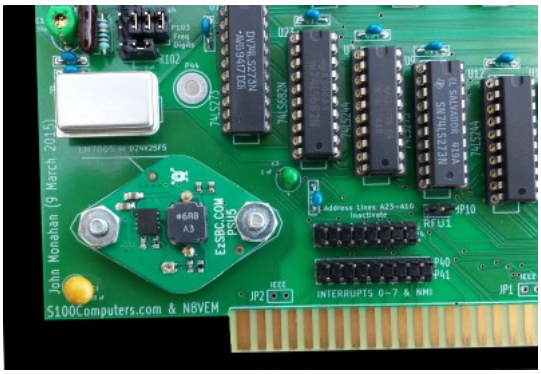
All of the IC's and components of the current V2 board can be transferred across to the V3 board. The main new chips are a 22V10 GAL, the three "common cathode" digital displays are widely available (e.g. Jameco #1955714). I solder them directly to the board, but you may like to use two single row sockets, their width is not standard. Be sure you obtain the "D" version of the Intersil ICM 7216D Frequency counter. Details about it can be found here. I got mine on eBay.

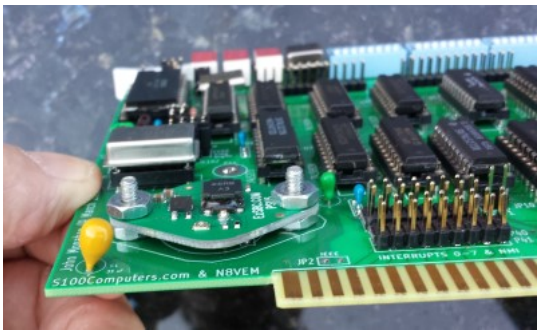
Step By Step Building the V3 SMB.

As always, first examine the bare board carefully for scratches or damaged traces, use a magnifying glass if need be. I find it useful to carefully slide a file at 45 degrees along the edges (front & back) of the S100 connector at the bottom of the board for easier insertion of the board into the bus. Carefully, just one or two strokes.

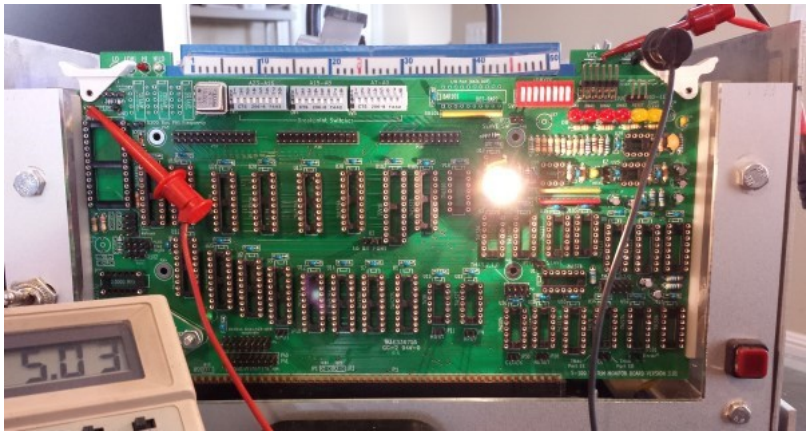
Next solder in all the required IC sockets, resistors, resistor arrays, capacitors, and jumper pins.
There are three possible options for the 5V Voltage regulator:--
You can use the traditional LM323K 3A TO-3 regulator as we did for the V2 board.
You can use the Pololu 5 Volt , 2.5 Amp D24V25 Switching regulator mentioned above.
Alternatively you can use these newer direct TO-3 footprint switching regulator by EzSBC.com called PSU5.

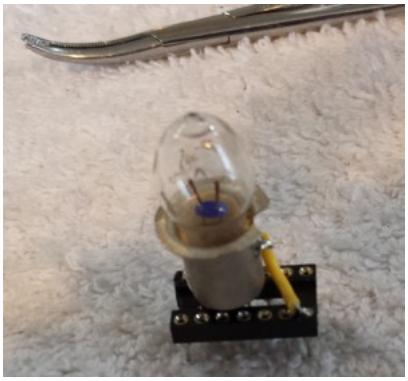
Both the Pololu and EzSBC units run cold, no heat sink is required. I have use the PSU5 for this V3 board. The TO-3 pin hole spacing on this board are very slightly too close together. To avoid snapping the fragile regulator leads I put a washer on the two nuts between the regulator and the board. This raised the regulator about 0.2" above the board. Here is a picture of the arrangement:-





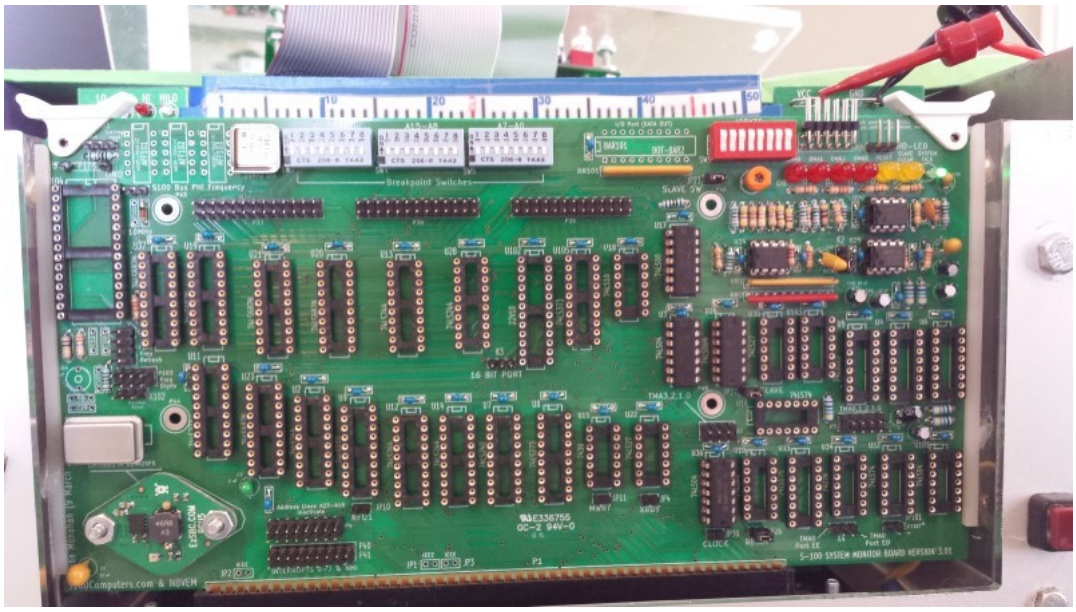
Before you add anything else check that Vcc is 5 volts on all the IC sockets. It's best to checkout the regulator under some load. I use an old flashlight bulb soldered to a 14 pin socket. Under load the regulator should still deliver 5 Volts. Here is a picture of the setup:-



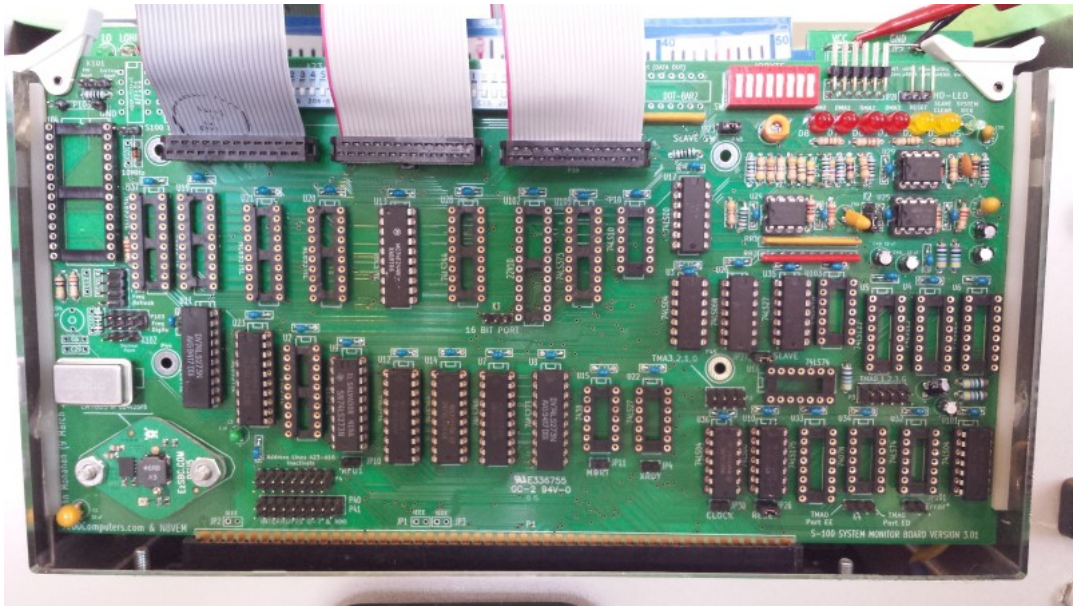


Next add the 10MHz and 2MHz Oscillators. Note the 10MHz oscillator pin 1 is the top left side of the socket.

Add the three 555's, U29, U24 and U26. Jumper (for now) K102 1-2. Jumper JP23, JP27,JP26.
Add U26,U36,U3,U17. Check Reset and Slave Clear switches. LED's D2 & D3 should light up when reset and slave clear buttons are pressed on daughter board. On Power on LED D2 should briefly flash.
The system tick LED D5 should flicker. The value of the trim cap (C57) is not critical. Its only used with things like CPM86 to trim the system interrupt time slightly. Any major pulse changes requires changing C55.
Here is a picture of the board at this stage.



Add U8, U9, U11, U12 & U13. Add U10, U35, U14, U23, & U101. The hex display lights should light up, but no single stepping etc. yet.
Add U19, U20 7 U21. Add U22, U18, U16 & U15. Its best to use a 7438 (rather than a 74LS38) for U15 if you have a large S100 bus. Add JP4.
You should now be able to single step the Z80 (see above).
Here is a picture of the board at this stage:-



We will now install the GAL and port I/O circuitry.
Burn a 22V10 Gal with the following code:-

```
;---------------------------------- PIN Declarations ---------------
PIN 1 bpSYNC              ;S100 bus address valid after returns low
PIN 2 pSTVAL              ;S100 Status Valid when HIGH (not currently used)
PIN 3 HIGH_8              ;S100 Bus Address line 8-15 for ports (Normally Zero) (See JP K3)
PIN 4 A7                  ;S100 Bus Address line 7
PIN 5 A6                  ;S100 Bus Address line 6
PIN 6 A5                  ;S100 Bus Address line 5
PIN 7 A4                  ;S100 Bus Address line 4
PIN 8 A3                  ;S100 Bus Address line 3
PIN 9 A2                  ;S100 Bus Address line 2
PIN 10 A1                 ;S100 Bus Address line 1
PIN 11 A0                 ;S100 Bus Address line 0
pin 13 sINP               ;S100 Bus sINP (Active High)
pin 14 pDBIN              ;S100 Bus pDBIN (Active High)
PIN 15 pWR                ;S100 Bus pWR* (Active low)
Pin 16 sOUT               ;S100 Bus sOUT(Active High)
Pin 17 MASTER_RES         ;Pulse low on S100 bus system wide reset
Pin 18 HIGH_LED           ;To LED Bar # 10 (Left most LED bar -- used here to indicate 16 bit port addressing).
Pin 19 Port_05            ;Diagnostic LED bar for Z80 "Master" monitor (Data IN when High)
Pin 20 Port_ED            ;In from this port activates TMA1 (IF, K4 1-2)
Pin 21 Port_EE            ;Low to High Output to this port activates (TMA1-4)
Pin 22 Port_E8            ;Resets system tick interrupt
```

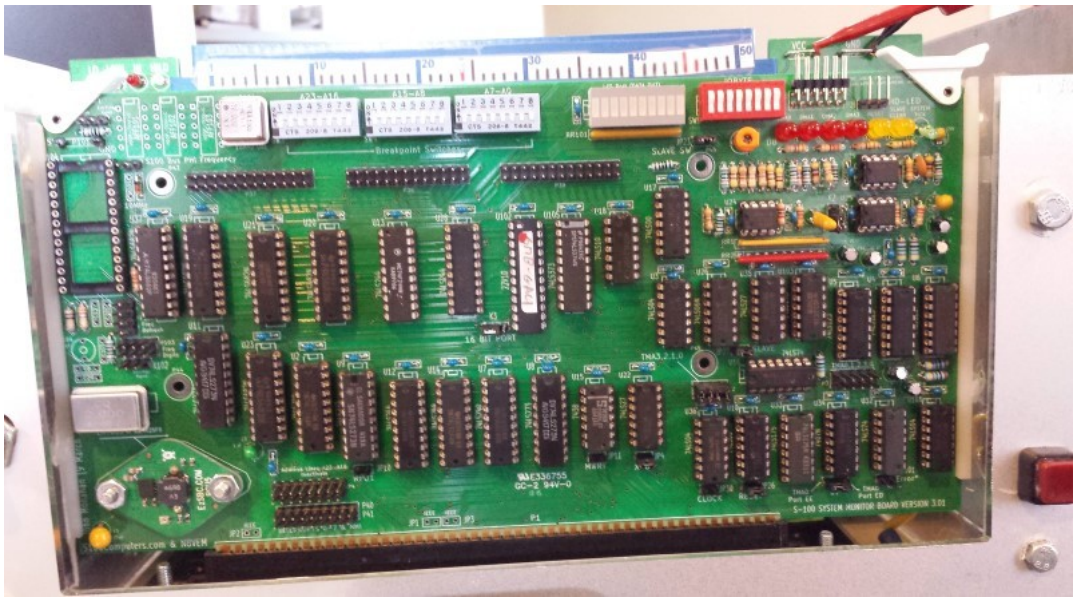
```
Pin 23 Port_EF          ;I0BYTE Port

;---------------------------------- Boolean Equation Segment ------

EQUATIONS

HIGH_LED = /HIGH_8

Port_05 = /bpSYNC * HIGH_LED * /A7 */A6 */A5 */A4 */A3 *A2 */A1 * A0 * sOUT * /pWR * MASTER_RES          ;PORT OUT (DIAGNOSTIC BAR DISPLAY)

/Port_EE = /bpSYNC * HIGH_LED * A7 * A6 * A5 * /A4 * A3 * A2 * A1 * /A0 * sOUT * /pWR * MASTER_RES          ;PORT OUT (TMA 0,1,2,3)

/Port_E8 = /bpSYNC * HIGH_LED * A7 * A6 * A5 * /A4 * A3 * /A2 * /A1 * /A0 * sOUT * /pWR * MASTER_RES          ;PORT OUT (Reset System Int)

/Port_ED = /bpSYNC * HIGH_LED * A7 * A6 * A5 * /A4 * A3 * A2 * /A1 * A0 * sINP * pDBIN * MASTER_RES          ;PORT IN (to activate TMA0)

/Port_EF = /bpSYNC * HIGH_LED * A7 * A6 * A5 * /A4 * A3 * A2 * A1 * A0 * sINP * pDBIN * MASTER_RES          ;PORT IN (IOBYTE)
```

The .JED file to burn the GAL is available below.

Insert the GAL in position U102. Add U37. Jumper K3 2-3. Add U2, U28, U105 and the LED Bar.
Do not solder the LED bars in until you are sure you have the correct orientation.
Power up the Z80 monitor. If you are using our MASTER.Z80 monitor (Version 5.0 or later), the "Diagnostic Display LED's" on port 05 should light up correctly.
You can check the GAL/Port by outputting a bit pattern to port 05H. 0FFH should light up the rightmost 8 LED bars.
Note as currently programmed in the GAL, the left most LED bar in ON when the board is configured for 16 bit Port/IO addressing (K3 1-2).

Next add U4, U5 & U6. The S100 bus status LED's on the daughter board should now be active.
Add U32, U34 & U103. Note the label for JP 103 is under U104. Its the two pin jumper above the diode D101.
Jumper P2 1-2, 3-4, 5-6 & 7-8. Jumper K101 1-2.

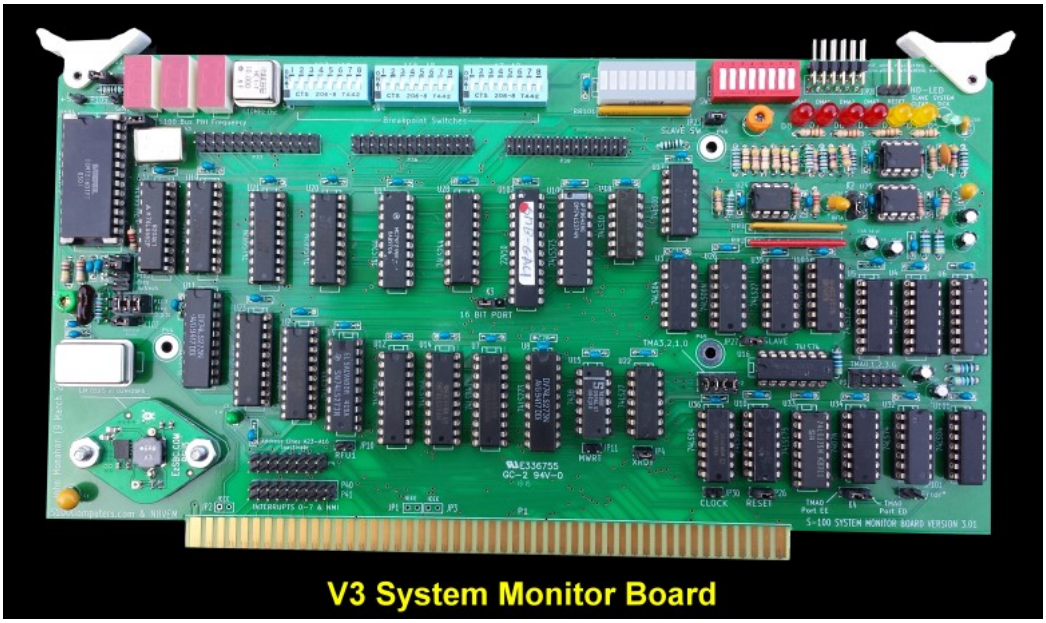
QOEE,01 lowers DMA0.
Here is a picture of the board at this stage.



Next we will add the S100 bus PHI frequency display section. Insert U104 the Intersil 7216D. Be careful of the pins they are quite fragile.
C104 is a 0-39pF cap (if you are using the crystal as the reference clock). P104 is a half size 10MHz Oscillator.
Insert the three Digital displays by hand, do not solder until they check out. The pins need to be bent out slightly. (The board footprint is too narrow).
Jumper P102, 2-3. Jumper P103, 2-3, 3-4 & 5-6. Jumper K102 2-3 (for Oscillator input), 1-2 for crystal input.
For some strange reason the PHI clock frequency displayed is always 1MHz less than expected. So our with our 10MHz Z80 board. 9.9 MHz is displayed. With our 8MH 8086 board 7.9 MHz is displayed. I don't have an explanation. It occurs with both the crystal and oscillator reference clocks.

You can shift the decimal point position and actual display range with jumpers K102 and P103. You can change the chip calculation time (0.1 sec to 10 seconds) with jumper P102. Please read the ICM-7216D data sheet for a full explanation.
BTW, you can remove the jumper K101 and use its pin 3 as an input to display the external frequency of a test circuit/device etc.

Here is a final picture of the completed V3 SMB:-



This board has been checked out with our Z80, 6502, 68K, 8088, 8086, 80286 and 80386 CPU boards. In all cases it seem to function correctly.

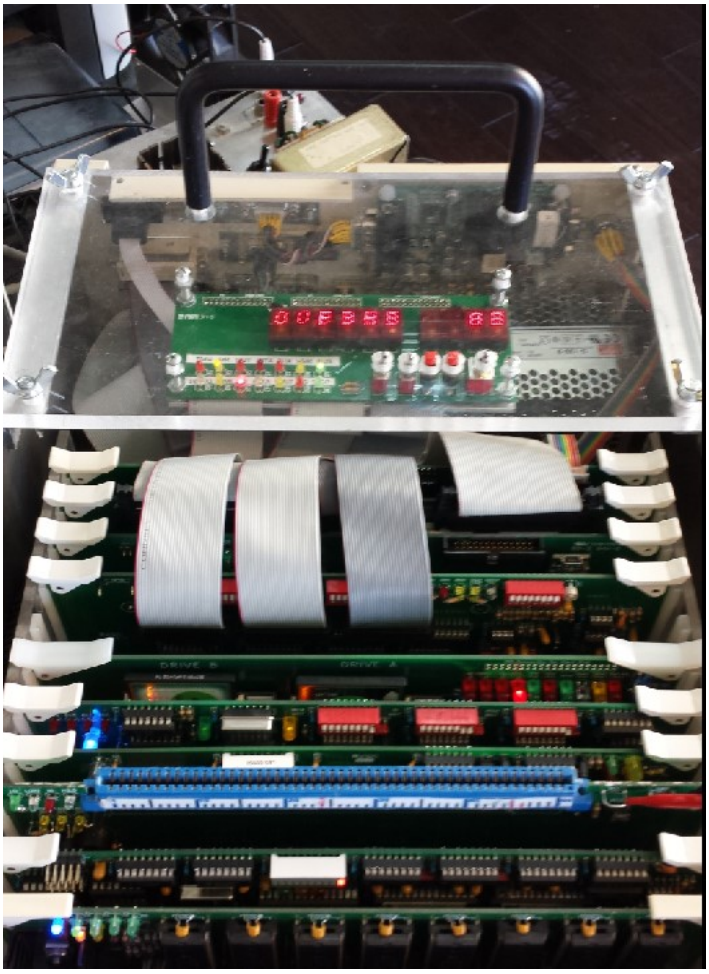
BUGS

Other than the bus clock frequency (PHI) display being 1 less than the actual value there are no reported bugs with this board so far.
Rick Bromagem noticed that the ICM7216D data sheet says pin 28 (frequency input) has a logic high minimum of 3.5 volts. That is not a normal TTL compatible level. You may need a 4.7K Ohm pull up on pin 28 (and pin 12) of the ICM7216D. This corrected the off by 1MHz issue I saw with the 8086 CPU board (see above).

I have also found that if you switch U7 from a 74LS240 to a 74F240 (or 74S240) you get a more accurate and stable frequency indication in the digital displays.

One thing to remember if you are building new CPU/hardware boards etc. The address displayed in the six HEX displays are latched when pSYNC is high and pSTVAL is low (i.e. When the S100 bus address lines are valid), but they are only displayed **when pDBIN goes high.** This board displays what the current bus master CPU board is actually reading on each bus cycle. So if you have hardware that alters the address lines only, they will not appear in the Hex display (unless you raise pDBIN).

BTW, instead of attaching the daughter HEX/Switch board to this SMB if you don't have a box with a front panel cutout, you may wish to consider relocating the display board elsewhere in your system by connecting it with ribbon cables. All the lines are buffered so the is no effect on the actual S100 bus signals.
Here is a picture of this setup in my "test" S100 box.



A Production S-100 Board

Realizing that a number of people might want to utilize a board like this together with a group of people on the Google Groups S100Computers Forum, "group purchases" are made from time to time. Please see here for more information.

The links below will contain the most recent schematic of this board.
Note, it may change over time and some IC part or pin numbers may not correlate *exactly* with the text in the article above.

MOST CURRENT SMB BOARD SCHEMATIC (FINAL, 9/23/2010)
MOST CURRENT SMB BOARD LAYOUT (FINAL, 9/23/2010)

MOST CURRENT HP5082 DISPLAY BOARD SCHEMATIC (V3, FINAL, 6/20/2010)
MOST CURRENT TIL311 DISPLAY BOARD SCHEMATIC (V3, FINAL, 6/20/2010)
MOST CURRENT LED BAR DISPLAY BOARD SCHEMATIC (V3, FINAL, 6/20/2010)

MOST CURRENT HP5082 DISPLAY BOARD LAYOUT (V3, FINAL, 6/20/2010)
Most current KiCAD files for this board (S100 FP display HP5082-002.zip 11/5/2014)

MOST CURRENT TIL311 DISPLAY BOARD LAYOUT (V3, FINAL, 6/20/2010)
Most current KiCAD files for TIL311 board (V1.1a 10/13/2017)
Most current Gerber files for TIL311 board (V1.1a 10/13/2017)

MOST CURRENT LED BAR DISPLAY BOARD LAYOUT (V3, FINAL, 6/20/2010)
Most current KiCAD files for this board (S100 FP display LED bars-002.zip 11/5/2014)

MOST CURRENT V2-SMB BOARD SCHEMATIC (V2 3/1/2014)
MOST CURRENT V2-SMB BOARD LAYOUT (V2 3/1/2014)
Most current KiCAD files for this board (S100 SMB V2.zip 11/5/2014)
KiCAD generated BOM file for V2 board (3/19/2014)
Excel BOM for V2 board (From Rick Bromagem, 3/23/2015)
PDF BOM for V2 board (From Rick Bromagem, 3/23/2015)

MOST CURRENT V3.1 BOARD SCHEMATIC (V3 3/20/2015)
MOST CURRENT V3.1 SMB BOARD LAYOUT (V3 3/20/2015)
Most current KiCAD files for V3. board (V3 3/20/2015)
Most current Gerber files for V3.1 board (V3 3/20/2015)
KiCAD generated BOM files for V3.1 board (V3 3/20/2015)
Excel BOM for V3.1 board (From Rick Bromagem, 2/13/2016)
JED code to burn the GAL (Zip File) (6/12/2015)

Other pages describing my S-100 hardware and software.
Please click here to continue...

This page was last modified on 11/29/2017