

Expand Your KIM!

Part 3: bus control board and memory

By now, I hope many of you have completed construction of the KIM-1 System cabinet, mainframe and power supply covered in my last article. Now, you will

see how easy it is to connect many Altair-compatible peripherals to a 6502 bus structure.

Let's begin by looking at the KIM bus and timing system.

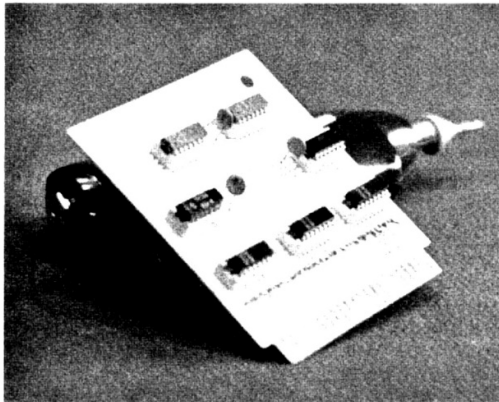


Photo 1. The bus control board is wire-wrapped on an inexpensive plug-in circuit board from Radio Shack.

ly 1000 ns exists between the leading edge of 01 and the trailing edge of 02, it can be assumed that the data will be stabilized before the trailing edge of 02. The processor therefore can use this edge to gate the data into its buffer register.

The same procedure can be used during a memory write, except that the trailing edge of 02 is used to gate the data supplied to the memory chips by the processor.

The R/W line is an output signal from the processor to tell the memory whether the present operation is a write or a read. A zero on the line indicates a write is to take place. If the inverse of the R/W line and the 02 clock are ANDed together, a negative edge is produced coincident with the trailing edge of 02, but only if the W/R line indicates a write operation. This signal, called a RAM R/W, is inverted on KIM and provides a positive edge, which latches most standard memory chips.

Most memory chips use either open collector or Tri-state outputs (1, 0, open circuit) so that they may have their outputs connected in parallel. This way, chips that are not selected (chip enable pin high) will not produce either a 1 or a 0 out.

Most memory boards will buffer the paralleled chip outputs with drivers. Typically, these drivers will also be Tri-state, and only the board selected will output its signals to the bus.

Let's look at a schematic of an S. D. Sales 4K memory board (Fig. 1) to see how it functions.

Address lines 12 through 15 are used to select the board. Since there are 16 combinations of these four lines, 16 4K boards could be used (total 64K of memory). The output of this board decoder (gate 34, pin 6) is used to allow the memory data to enter the bus if the SMEMR (which indicates a read operation on the Altair bus) is high and also (gate 35, pin 12) enables the outputs of a two-input decoder that will

The address bus is used by KIM to output the address of a memory location with which it wishes to communicate. The data to and from that location is transferred over the data bus. The timing and conditions of these transfers are maintained by the control bus.

The timing is controlled primarily by the phase 1 (01) and 02 clocks. The 01 and 02 signals are essentially square waves 180 degrees out of phase but, in addition, have a less-than-50-percent duty cycle so they are nonoverlapping.

First, let's look at a read operation. When the 01 clock goes high, the processor prepares to output the address, which stabilizes on the address bus within 200 ns. Another 100 ns or so is needed for the decoders to select a particular memory chip, and perhaps another 500 ns passes before the memory chip puts out stabilized data. Since near-

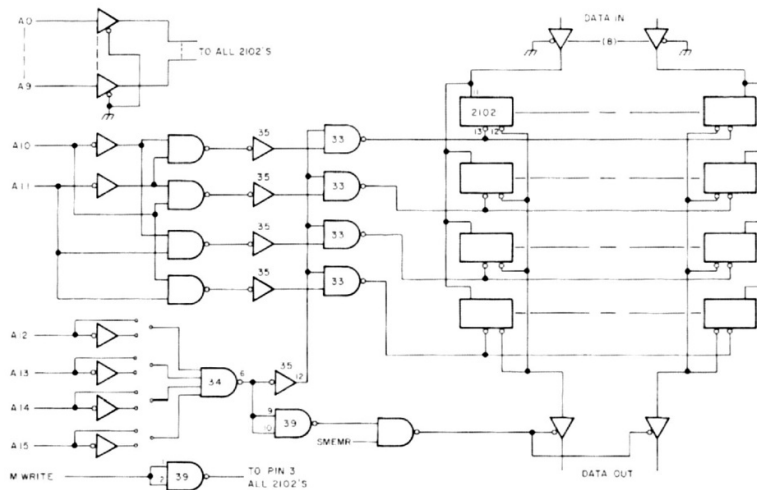


Fig. 1. Simplified schematic of S.D. Sales 4K memory board.

select one of the four banks of memory chips by bringing their chip enables (pin 13) low.

Once a bank of chips has been enabled, the address on lines 0-9 selects one of the 1024 words through internal decoding. Each chip that is enabled "reads" until its pin 3 is brought low, which will cause any data on the inputs to be gated into the locations specified by the address pins. The Altair bus provides this signal, and calls it an MWRITE.

This, or any Altair-compatible memory, can easily be connected to the KIM bus by using the inverse of KIM's RAM R/W as the MWRITE and KIM's R/W for a SMEMR. The address lines are connected KIM AB0 to Altair A0, etc. The data bus is slightly different, as KIM uses a bidirectional bus instead of the two single-direction buses found on the Altair. Since the data lines in and out are fully buffered, just tie the KIM DB0 to both the DO0 and DI0, etc., and memory interfacing is complete.

One of the major differences between the 6502 and the 8080 is the way they handle input/output. The 6502 uses memory-mapped I/O, which simply means that you build a memory location out of flip-flops and output to it with normal memory instructions.

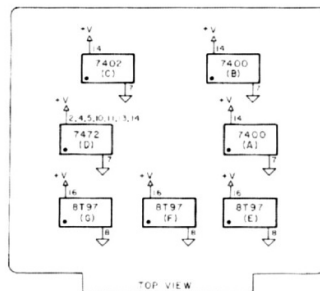
The 8080 instruction set provides separate instructions to manipulate several control lines that in essence are

duplicates of the R/W and RAM R/W functions. They are PWR, which acts as a RAM W/R for I/O; SINP, which indicates an

input operation; SOUT, which indicates an output operation; and PDBIN, which indicates that the data bus is to be used for an input operation. If it appears that the Altair I/O signals are needlessly complicated, I have to agree; after seeing how the 6502 can simulate the transfers, I'm sure several improvements might come to mind.

When the 8080 performs an I/O instruction, it uses the eight least significant bits as the address of the I/O port. Because of this, the only major change required to convert the 6502 bus is to reserve one page of memory as I/O ports. A decoder must indicate when this page is addressed and output a signal to simulate SINP and SOUT. Page FF (hex) cannot be used since KIM stores its interrupt vectors there; so the next logical choice is page FE.

As long as we've mentioned



AB0	out	A	1	AB0 in
AB1	out	B	2	AB1 in
AB2	out	C	3	AB2 in
AB3	out	D	4	AB3 in
AB4	out	E	5	AB4 in
AB5	out	F	6	AB5 in
AB6	out	H	7	AB6 in
AB7	out	J	8	AB7 in
AB8	out	K	9	AB8 in
AB9	out	L	10	AB9 in
AB10	out	M	11	AB10 in
AB11	out	N	12	AB11 in
AB12	out	P	13	AB12 in
AB13	out	R	14	AB13 in
AB14	out	S	15	AB14 in
AB15	out	T	16	AB15 in
I/O Enable	U	17	Lower 8K enable	
Int. Enable	V	18	Ram R/W	
R/W	W	19	Ready	
Sync	X	20	02	
Hold Req.	Y	21	+5 volts	
Hold Ackn.	Z	22	Ground	

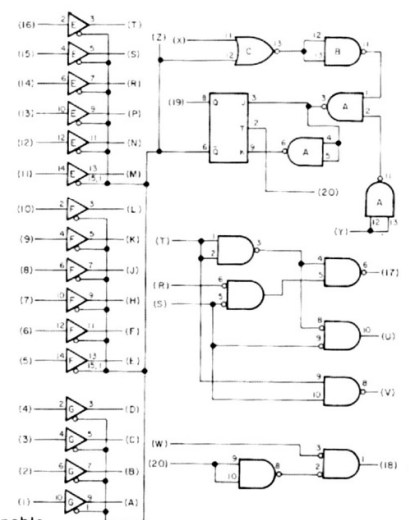


Fig. 2. Schematic, pinouts, and component placement for bus control board.

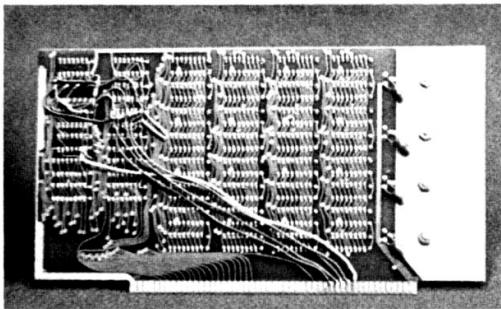


Photo 2. A 4K memory board modified for operation in K1 through K4 using the KIM decoders.

the interrupt vectors, page FF must also be decoded and ORed with the K7 to force the KIM ROM operating system to be executed whenever the RES button on the KIM keyboard is pressed (the KIM monitor vectors are in K7).

To decode both page FF and FE would require two eight-input gates and the appropriate inverters, so I decided to simplify matters. Rather than decode out all eight of the page bits, I checked only AB14 and AB15. If both are high, I assume page FF is being addressed, and if AB14 is low with AB15 high, an I/O operation is indicated. This of course wastes a lot of memory locations. An I/O port, for example, can be indicated by an instruction referencing page 80, 81, 90, AF, as well as 60 others. Actually, the top half of the memory locations cannot be used in this scheme. If you plan to have more than 32K of memory, you should decode out FF and FE with eight-input gates. With the decoding I suggest, you can add a 16K board to your system without additional modifications.

For details of how all the signals previously discussed are implemented, refer to Fig. 2, a reproduction of one page in my notebook. All pertinent information on the bus control board can be read from that sheet, including the schematic, pinouts and component placement. My circuits are all wire-wrapped on 44-pin boards available from Radio Shack. Photo 1 shows the completed

bus control board.

I'm usually asked why I did not simply invert the RAM R/W signal from KIM to get the needed RAM R/W. A little inspection will reveal that all 44 pins are being used, and I already had the R/W and 02 signals available for the DMA circuits, which are next on the list of topics to be covered.

DMA stands for direct memory access. In general, this means that the processor is forced into a wait, or hold, state while another processor (usually an intelligent controller) manipulates the bus lines to read and write directly from and to memory.

The processor must be effectively removed from both the address and the data bus to avoid conflict with the controller signals. When the 6502 is in a hold state, the data bus goes to an open-circuit condition since it is Tri-stated internally. The address bus, however, must have Tri-state drivers added to it so that when the processor goes into a hold, the drivers can be disabled, leaving the bus free for DMA.

The primary reason for adding DMA capability to the KIM is the Dazzler, which has the capability of mapping memory blocks onto a television screen. When the Dazzler needs to access memory to read a block of data, it issues a request signal and begins the read sequence when it receives an acknowledgment.

A circuit is needed that, upon request from the Dazzler, will put the processor into a

hold, disable the bus and generate an acknowledge.

There are several problems, however, that make this circuit more complicated than you might expect. First, the processor will react unpredictably if you try to put it into a hold during a 02 cycle. In order to insure that a request will not be recognized during the 02 cycle itself, I used a JK flip-flop that changes only on the trailing edge of the 02 clock. When this flip-flop clears, the address bus drivers are disabled, the hold line on the processor is pulled low, and an acknowledge is generated.

The second catch is that the processor will not stop if it is performing a write operation. Since the bus has been disabled, the Dazzler could take over for itself without problems. If, however, the processor was attempting to modify data in memory on the Dazzler side of the drivers, that data would not be modified and errors could result.

Therefore, I had to insure that the request could not be recognized unless the processor was in a non-write cycle.

You would expect that the R/W line could be used to determine if the present cycle was read or write. Unfortunately, the R/W line is not valid until well into the cycle. I chose to use the sync signal, which signifies that the processor is fetching an op code. A normal instruction always reads an address following an op-code fetch. The only exceptions are single-byte instructions, which

read another op code in the next cycle. The point is that if the sync line is high, the next cycle must be non-write. I use this principle to ignore the Dazzler's request until the sync is high, thus insuring that the processor will halt when the flip-flop is cleared.

I planned for the first two situations, as they are fairly obvious from a study of the 6502 hardware manual. The third problem, however, cost me a lot of sleep; it was not until I drew a timing diagram extending into the next cycle that I found a solution.

When the processor goes into a hold, the state of the sync line is not guaranteed. It may be either high or low, depending on the type of cycle it is in. If it happens to be low when the processor is in a hold, the next 02 pulse would enable the bus again by setting the flip-flop. To prevent this, I ORed the sync signal with Q output of the flip-flop. Even wired this way, the flip-flop will not clear until the sync is high; once cleared, however, it will remain so until the Dazzler ends its request. Without this latching circuit, the video tear is unbearable.

Since my only DMA operation is for the Dazzler, which never writes to memory, my circuits only allow for read-oriented DMA. Write-oriented DMA, such as a floppy disk, would need the R/W and RAM R/W lines Tri-stated along with the address bus. Two Tri-state drivers are available on the bus control board for this opera-

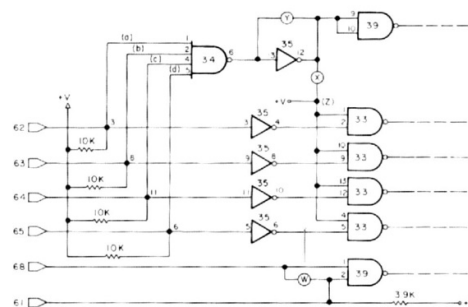


Fig. 3. Modifications to 4K memory board to allow KIM decoding.

tion, but since all 44 output pins are being used, other provisions would have to be made if you decide to add a disk in the future.

Recall that I used the S.D. Sales memory board as an example for explaining the bus conversion. I want to discuss the board again to show how it can be modified to fit in K1 through K4 using the on-board KIM decoder, for those who want this option. Since I designed the system, most of the K1 software has become available in K8 versions, making the following conversion totally a matter of choice. If you do not require memory in the K1 through K4 positions, the S.D. Sales board can be used on the KIM System Altair bus without modification.

Fig. 3 and Photo 2 show the modifications. The foil must be cut at points X and Y. Point Z can be wired to +V by connecting pin 13 and 14 on IC33. A wire must be added to connect pin 12 of IC35 to pin 9 on IC39.

Address	Instruction	Comments
0000	A9 10	Loads Acc. with mode (see Dazzler manual).
0002	8D OF 80	Sends mode to Dazzler input port OF.
0005	A9 90	Loads Acc. with starting address (page 20).
0007	8D OE 80	Sends starting address to Dazzler.
000A	4C 22 1C	Jump to KIM monitor.

Fig. 4. Program to start Dazzler.

ICs 36 and 37 (including the sockets) are not used. Refer to Fig. 3 and connect the points a, b, c and d to the designated pins where IC36 would have been installed, and on the gold fingers for the Altair pins 62 through 65. These were previously connected to the KIM decoder. If you have trouble understanding how the memory works with these modifications, re-read the explanations at the beginning of this article.

A write-protect modification is also shown in Fig. 3. A zero on pin 2 of gate 39 holds its output high. As indicated last month, switch number two

under AUXILIARY is used for this feature. The foil cut at point W is on the component side of the board, and is under the socket, so make it before beginning construction.

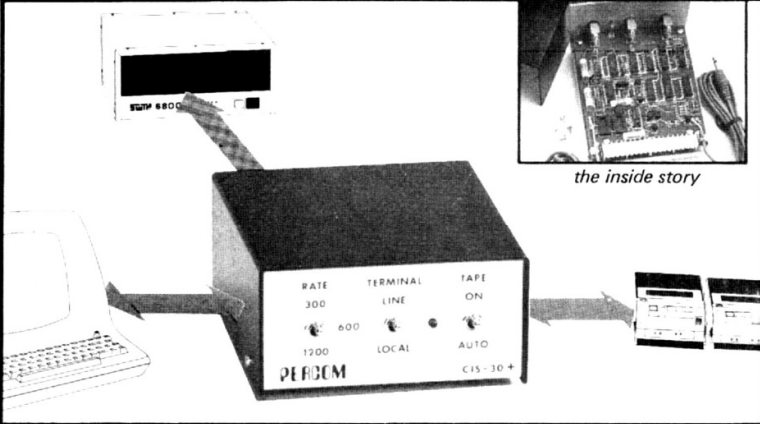
With the bus control board in place, the KIM should be operational. If you install the memory boards (the 8K board should start at page 20₁) and the Dazzler on the Altair bus, the program in Fig. 4 should put a display on the screen. Use the KIM keyboard to write data into page 20 locations, and watch the display change.

Remember, since the 4K board no longer uses the standard address lines, it cannot be

used for Dazzler displays.

I should add one comment on the Dazzler. Since my last article, it has come to my attention that random noise can build up on the Altair pins 54 and 99, causing some Dazzlers to turn off suddenly. If you are having this problem, it can be corrected by connecting each of these pins to +5 volts through 2200 Ohm resistors.

Next, I'll discuss the serial/parallel board, which will turn the SWTP printer and keyboard into a miniature teletypewriter. We'll also add an external display and keyboard to KIM, and the front panel will start to come alive. ■



- Record and playback at 120, 60 or 30 self-clocking bytes per second (extended Kansas City Standard)
- 1200, 600 or 300 baud data terminal interface
- Dual cassette operation
- Compatible with SWTPC cassette software
- Optional kit permits program control of cassettes
- Optional adaptor permits interfacing with any computer


Upgrade your SWTPC 6800 system to 1200 baud with PerCom's CIS-30+ dual-cassette/terminal interface

The CIS-30+ . . . four times as fast as SWTPC's AC-30 with the same dual-cassette capability . . . **plus** a 1200-baud data terminal interface . . . in a SWTPC color-compatible package that's only **1/10** the size of the AC-30.

Dependable? The simplicity of Harold Mauch PerCom Data designs says more than any well-chosen words. Simply put, for only **\$79.95*** you get the fastest, most dependable dual function interface you can buy for your SWTPC 6800.

See your nearest dealer or order direct from PerCom.

P7 **PerCom 'peripherals for personal computing'**



PERCOM DATA COMPANY, INC.
DEPT. K 318 BARNES GARLAND TX 75042
(214) 276-1968

*Kit price. Assembled and tested: \$99.95 + shipping. Tex. res. add 5% tax. BAC & MC available.