Micro Technology Unlimited

K-1032 BANKER

RAM ROM I/O SYSTEM BOARD

FOR KIM/MTU BUS SYSTEMS

32K READ/WRITE MEMORY
8/16K READ-ONLY MEMORY
4 I/O PORTS PLUS CONTROL LINES
EPROM PROGRAMMER
18 BIT ADDRESSING
BANK SWITCHING FOR RAM AND ROM

JANUARY, 1981 REV B

MICRO TECHNOLOGY UNLIMITED
P. O. BOX 12106
2806 HILLSBORO STREET
RALEIGH, NC 27605

TABLE OF CONTENTS

CONGRATULATIONS!

You have just purchased the most advanced, most flexible, most system oriented memory board available for 6502 based computers! Besides providing more memory and functions for less money, the Banker Memory goes a long way toward solving a problem us old-timers thought would never occur -- the problem of address space saturation.

When the 8008 microprocessor came out in 1972 its ability to address over 16 THOUSAND bytes of memory seemed to be incredibly lavish for a mere microprocessor. This was underlined by the fact that a board having this much memory would need 512 of the popular 1101 type static 256 bit memory chips. Even when using the most advanced 1103 type dynamic 1K bit memory chips, 128 were required for a full memory system; a quantity that completely dwarfed the microprocessor and its couple of dozen support chips. Just imagine the amazement when the second generation of microprocessors came out in 1975 that could address 64K of memory. This was as much memory as the typical campus mainframe (an IBM 360 model 30) had in 1970! Even with the 4K memory chips that were just being announced, it still took 128 packages (large 22-pin types at that) on several boards to make a full memory system.

Now the problem is different. 64K of memory requires only 32 memory chips that can be packed into 20 square inches of space. Whats more, read-only memory chips with permanent programming have grown in size as fast as read/write memory chips have. The typical system today devotes nearly half of its 64K of possible addresses to read-only memory, I/O port addresses, and other "system" functions in order to support user demands for sophisticated monitors, BASIC and other high level languages in ROM, disk operating systems, etc. "Only" 20K to 48K of addresses are left over for user programs and data and in many applications this is not enough. While the third generation of 16 bit microprocessors is once again expanding the available address space to as much as 16KK, those of us with limited budgets and large investments in 6502 hardware and software must find other ways.

The MTU Banker Memory is the best way! Its partitioned resources, each controlled by a software settable on-off switch, means that much more memory can be added to the system by overlapping resource addresses. This is particularly useful where a program must manipulate a large amount of data in memory at one time. For an example, lets assume that one has an AIM-65 that must be able to acquire 40,000 bytes of data in less than a second from, say, a car crash test and then thoroughly analyze it. This is too fast to dump on a floppy disk but can be accomodated neatly (and less expensively) with two Banker Memories. 16K (2 blocks) on one Banker Memory would be set to address from 0000 to 3FFF and would hold the program. The other 16K on the first Banker Memory and both 16K halves of the other Banker memory would all be set to address from 4000-7FFF. Finally to display the results and round out the system, a Visible Memory would be addressed from 8000-9FFF. With this arrangement, 48K is available 16K at a time to store the data and analysis results. If storage requirements increase, another Banker Memory can be added for a total of 80K of storage! The on-board ROM sockets can also be switched on and off under software control. If each ROM is a different application program, you can switch from one to the next in microseconds. The best part is that this bank switching feature can be used with any 6502 microcomputer having a KIM style bus and a 1MHz clock frequency.

Soon, MTU will be introducing its own high speed systems with 18 bit address busses and enhanced transparent bank switching 6502 CPU's. The Banker Memory (and future MTU bus interface products) is designed to operate in these advanced 18 address bit systems with simple jumper changes -- no obsolence! So there you have it, enable register on-off control for extended memory in today's systems and 18 bit addressing for even easier bank switch programming in tommorrow's systems all with the MTU Banker Memory!

1. K-1013 UNPACKING AND INSTALLATION

The K-1032 bank-switchable RAM-ROM-I/O board for KIM/MTU bus systems is a carefully engineered, manufactured, and tested product that should operate perfectly when handled and installed according to the following instructions. Note that the board is shipped in a black conductive plastic bag. Since MOS integrated circuits are used, damage from static discharge is possible. It is helpful to reduce static by working in an area with concrete floors and a reasonable humidity level. If this is impossible, at least avoid wearing rubber soled shoes and move slowly in the work area. When unpacking or handling the board, touch the screws sticking up in the middle of the heatsink first and release them last. Note that the preceeding comments apply equally to the user's computer board which of course contains MOS IC's also.

1.1 ADDRESS AND FUNCTION SELECT JUMPERS

Due to the variety of systems which may use this board, the K-1032 Banker Memory has a number of address and function select jumpers. BEFORE THE BOARD WILL FUNCTION AT ALL, THE ADDRESS SELECTION JUMPERS WILL HAVE TO BE INSTALLED! See sections 3, 4, and 5 for address jumper installation instructions. Certain function select jumpers however have been installed as listed below:

1. All ROM sockets configured for 2716 type EPROM's.
2. I/O interrupts disabled.
3. 16 bit addressing of all functions.

It is desirable that initial tests of the board in the user's system be done with these standard function select jumper options intact. After checkout, these jumpers may be changed if desired.

1.2 CONNECTION TO USER'S SYSTEM

As shipped the board requires a source of +8 (+7 to +12) volts unregulated input and a source of +16 (+14 to +20) volts unregulated for power. It is preferable to use the on-board regulators but if only regulated power is available, the regulators may be bypassed. Solder a jumper wire between the two outside pins of VR2 for regulated +5 input. Solder a jumper wire between the two outside pins of VR1 for regulated +12 input. Maximum current drain from +5 is 650MA and from +12 is 175MA. Note that the timing generator pot may have to be adjusted when using regulated +5 input. See section 14 for adjustment procedure.

The easiest method of connection to a KIM-1, SYM-1, or AIM-65 is to use an MTU K-1005 series motherboard/card file. With the card file one simply plugs the processor board into the top slot and the Banker memory into one of the other slots. The K-1032 may also be connected directly in parallel with the expansion connector of the processor (except for those signals marked with an * on page 41). If direct connection is used, the wires should not be longer than 4 inches and the ground wire should connect directly from the processor to the K-1032 and be 16 guage or heavier. See section 7 if you wish to use the 18 bit addressing feature of the K-1032.

2. INITIAL BOARD TEST

Perform the following steps to verify that the K-1032 has not suffered shipping damage and is in reasonable operating condition:

1. Inspect the board for shipping damage such as broken parts or bent-over component leads shorting PC traces. A maximum of three chipped or broken disk bypass capacitors (see the parts layout and schematic diagrams in sections 19 and 21) will not impair operation of the board.

2. Make sure that the RAM, ROM, and I/O addresses are set the way you wish them to be. See sections 3, 4, and 5 for instructions.

3. After making sure that power is turned off, that the user's power supply has discharged all residual voltages, and that there are no addressing conflicts between the K-1032 and the rest of the system, plug the board into the system and turn the power on.

4. Using the standard system monitor, examine the first RAM location. If 18 bit addressing is being used or the Default Enable jumpers have been set to disable the RAM, make sure that the correct bank is selected and that the RAM block is enabled before examining. The contents will probably be either 00 or FF and each examine operation should report the same value. If the results are different each time it is examined, either the board is not responding to its address or the board has failed to synchronize to the user's system clock which must be 1.0MHz. If the contents are always the same as the high byte of the address (try firstloc+100 and firstloc+200 as well), the board is definitely not responding to its address. Check that the address jumpers are set the way you expect. This can also be caused by having the RAM block disabled. Press Reset on the computer and try again. If the condition persists, refer to the section on Troubleshooting on page 39.

5. Using the standard system monitor, store something into the first RAM location and verify that it can be read back. Try several different values.

6. Repeat steps 4 and 5 for the other three 8K blocks of RAM if you have the full 32K of RAM. If you have the K-1032-2 16K RAM only version, there will be only two 8K blocks to check.

7. If the ROM sockets are present on the board, examine the first location of ROM 0. With no ROM in the socket, this location will probably read something different every time. With a ROM installed, it should read the first location in the ROM (a blank ROM will read FF). If instead the upper byte of the address is always read, the K-1032 is either not responding to the ROM addresses or the ROM sockets have been disabled. Press Reset and try again. If the condition persists, refer to the section on Troubleshooting on page 39.

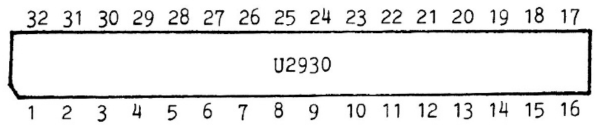
8. To perform a quick check of the enable/disable switching logic, write 00 into location I/OBASE+$20 (see section 5.1 for definition of I/OBASE). KIM-1 owners will have to enter the following program at location 0 to do this: A9 00 8D XX XX 4C 22 1C where XXXX is I/OBASE+$20 stored with the low byte first. After writing 00 into the Enable Register, the board should not respond to any RAM or ROM addresses which is evidenced by a tendency to read back the high byte of any address read. Pressing Reset should restore the Enable Register to its default value and restore previous operation.

9. To perform a quick check of the I/O section (if installed), look at the first I/O address (I/OBASE) which should read FF. The next address should also read FF while the following 2 addresses should read 00 (it is assumed that nothing is connected to the I/O pins). Write FF in location I/OBASE+2. Now location I/OBASE+1 should read 00. Repeat these tests with I/OBASE+$10 through I/OBASE+$13 to check the other I/O chip.

10. If desired, load the diagnostic program which starts on page 42. After loading, store the hex page number of each of the eight 4K RAM segments in locations 0010 through 0017. If you have the K-1032-2 with only 16K of RAM, store FF into locations 0010 - 0013 to bypass testing of the non-existant blocks. Make sure that the addresses of the two segments that make up an 8K block are stored in adjacent locations. Also make sure that the blocks are arranged in ascending Enable Register bit order, that is, the segment addresses for block 0 (controlled by Enable Register bit 0) in locations 0010 and 0011, etc. It is OK if 8K RAM blocks overlap each other since the diagnostic will enable only the block being tested at the time. Also the I/O base address (I/OBASE) must be stored in locations 0018 (low byte) and 0019. Note that the diagnostic does not allow for I/O addresses shadowing the RAM. If the I/O section is not present on the board, you must store FF in location 001A to bypass the I/O testing, otherwise store 00 there. The diagnostic program does not test the ROM sockets. The diagnostic program does a somewhat more thorough test of the I/O chips (if installed) and a very thorough RAM test using random data stored and retrieved in a scrambled order. The program should run for a couple of minutes and then stop by going to the system monitor. (If your monitor is not enter when a BRK instruction is executed, store a return jump in locations 03XX-03XX.) When it stops, look at location 0000 for an error code. Locations 0001-0005 contain details about the error which are explained in the program listing. If an error persists, see the trouble-shooting section (page 39).

3. RAM ADDRESS JUMPERS

The RAM address jumper section is implemented as a 32 pin IC socket (U2930 in the board layout and schematic drawings) and a 32 pin plug with solder terminals. The jumpers are actually wires connecting some of the terminals on the plug together. A sketch of the 32 pin plug is shown below to illustrate how the pins are numbered:



As shipped, the 32 pin plug has no jumper wires installed. The best type of wire to use on the jumper plug is the #30 wire-wrap wire with teflon insulation supplied with the K-1032. Although the teflon insulation does not melt under soldering heat, it does require a sharp stripping tool.

For most customers, one of the example procedures below should be suitable for configuring the RAM on their board. To set the jumpers for more specialized applications of the K-1032's RAM facililities, one should read section 3.5 and 3.6 which describes the theory behind the RAM address jumpers in detail. To disable all of the RAM (regardless of the Enable Register contents), simply remove the 32 pin plug.

3.1 CONTINUOUS RAM FROM 2000 TO 9FFF ON A KIM-1 SYSTEM

To provide continuous RAM from 2000 through 9FFF, install the following jumper wires on the 32 pin plug:

1.	2000-2FFF	pin 3 to pin 22	Enable Register bit 0
2.	3000-3FFF	pin 4 to pin 23	
3.	4000-4FFF	pin 5 to pin 18	Enable Register bit 1
4.	5000-5FFF	pin 6 to pin 19	
5.	6000-6FFF	pin 7 to pin 30	Enable Register bit 2
6.	7000-7FFF	pin 8 to pin 31	
7.	8000-8FFF	pin 9 to pin 26	Enable Register bit 3
8.	9000-9FFF	pin 10 to pin 27	

On the K-1032-2, wires 5 - 8 may be omitted since there is only 16K of RAM onboard. See section 7.1 for a discussion of the significance of the Enable Register.

3.2 CONTINUOUS RAM FROM 1000 TO 8FFF ON AN AIM-65 SYSTEM

To provide continuous RAM from 1000 through 8FFF, install the following jumper wires on the 32 pin plug:

1.	1000-1FFF	pin 2 to pin 22	Enable Register bit 0
2.	2000-2FFF	pin 3 to pin 23	
3.	3000-3FFF	pin 4 to pin 18	Enable Register bit 1
4.	4000-4FFF	pin 5 to pin 19	
5.	5000-5FFF	pin 6 to pin 30	Enable Register bit 2
6.	6000-6FFF	pin 7 to pin 31	
7.	7000-7FFF	pin 8 to pin 26	Enable Register bit 3
8.	8000-8FFF	pin 9 to pin 27	

On the K-1032-2, wires 5 - 8 may be omitted since there is only 16K of RAM onboard. See section 7.1 for a discussion of the significance of the Enable Register.

3.3 CONTINUOUS RAM FROM 1000 TO 7FFF ON A SYM-1 SYSTEM

To provide continuous RAM from 1000 through 7FFF, install the following jumper wires on the 32 pin plug:

1. 1000-1FFF pin 2 to pin 22 — Enable Register bit 0
2. 2000-2FFF pin 3 to pin 23
3. 3000-3FFF pin 4 to pin 18 — Enable Register bit 1
4. 4000-4FFF pin 5 to pin 19
5. 5000-5FFF pin 6 to pin 30 — Enable Register bit 2
6. 6000-6FFF pin 7 to pin 31
7. 7000-7FFF pin 8 to pin 26 — Enable Register bit 3

Note that since RAM can only go up to 7FFF on a SYM-1 that only 28K of the 32K is actually used. The remaining 4K is not used in this jumper configuration but could be put up in the ROM area of the SYM-1 (on an even 4K boundary to match the odd 4K of block 3, see section 3.6) if there is free space there. One could also remove the 4K of on-board RAM from the SYM-1 and install a wire from pin 1 to pin 27 to put that 4K in the K-1032. On the K-1032-2, all 16K is used and wires 5 - 7 may be omitted since there is only 16K of RAM on-board. See section 7.1 for a discussion of the significance of the Enable Register.

3.4 TWO 16K BANKS ON AN AIM-65

This configuration is most useful in an AIM-65 that already has much of its address space filled with a K-1013 Disk Controller (8000-9FFF and 4000-5FFF) and a K-1008 Visible Memory (6000-7FFF). The idea is to divide the 32K of RAM on the Banker memory into two 16K banks, each from 0000 to 3FFF. This would allow a program to switch banks (and also page zero and the stack) and thus greatly increase the effective amount of available memory. Many other combinations are possible; see section 3.6. For a complete discussion of bank switching, see section 7.

A. First 16K bank, enabled on power up and by writing 03 to Enable Register
   1. 0000-0FFF pin 1 to pin 22
   2. 1000-1FFF pin 2 to pin 23
   3. 2000-2FFF pin 3 to pin 18
   4. 3000-3FFF pin 4 to pin 19

B. Second 16K bank, enabled by writing 0C to Enable Register.
   5. 0000-0FFF pin 1 to pin 30
   6. 1000-1FFF pin 2 to pin 31
   7. 2000-2FFF pin 3 to pin 26
   8. 3000-3FFF pin 4 to pin 27

3.5 GENERAL PROCEDURE FOR SETTING RAM JUMPERS

For simple applications of the Banker RAM without overlapping addresses (i.e., bank switching not used) and 16 bit addressing, use the following procedure:

1. Make a list of the 4K segments of RAM you wish to assign to the K-1032. There must be 8 or fewer entries on this list for the K-1032-1 and 4 or fewer for the K-1032-2.

2. Associate each segment on the list with a pin number on the 32 pin plug for the U2930 socket according to the table below:

0000 - pin 1	4000 - pin 5	8000 - pin 9	C000 - pin 13
1000 - pin 2	5000 - pin 6	9000 - pin 10	D000 - pin 14
2000 - pin 3	6000 - pin 7	A000 - pin 11	E000 - pin 15
3000 - pin 4	7000 - pin 8	B000 - pin 12	F000 - pin 16

3. Arrange the segments into 4 or fewer even-odd pairs (2 pairs for the K-1032-2). For example, if the list is 1000, 2000, 3000, 4000, 5000, 6000, A000, B000, then the proper pairing would be: pair 0 1000:2000, pair 1 3000:4000, pair 2 5000:6000, pair 3 A000:B000. Note that the step 1 list cannot have more than 4 odd segment addresses or 4 even segment addresses, i.e., 5 odds and 3 evens cannot be accomodated.

4. Associate the odd and even members of each pair with pin numbers according to the table below:

Pair 0 -	even pin 22, odd pin 23
Pair 1 -	even pin 18, odd pin 19
Pair 2 -	even pin 30, odd pin 31
Pair 3 -	even pin 26, odd pin 27

5. Connect a jumper wire on the 32 pin plug from each pin established in step 2 to each pin established in step 4. For the example address configuration, the following pins would be connected together: 2 to 23, 3 to 22, 4 to 19, 5 to 18, 6 to 31, 7 to 30, 11 to 26, and 12 to 27. Use a minimum of soldering heat and have the plug installed in its socket (U2930) while soldering to prevent melting of the plug's plastic base.

The procedure when RAM bank switching is to be used is similar except that there will be duplicate segment addresses in the list generated in step 1. When pairing the segments in step 3, the segments within a pair must be at different addresses and one must be odd while the other is even. The pairs however may share addresses.

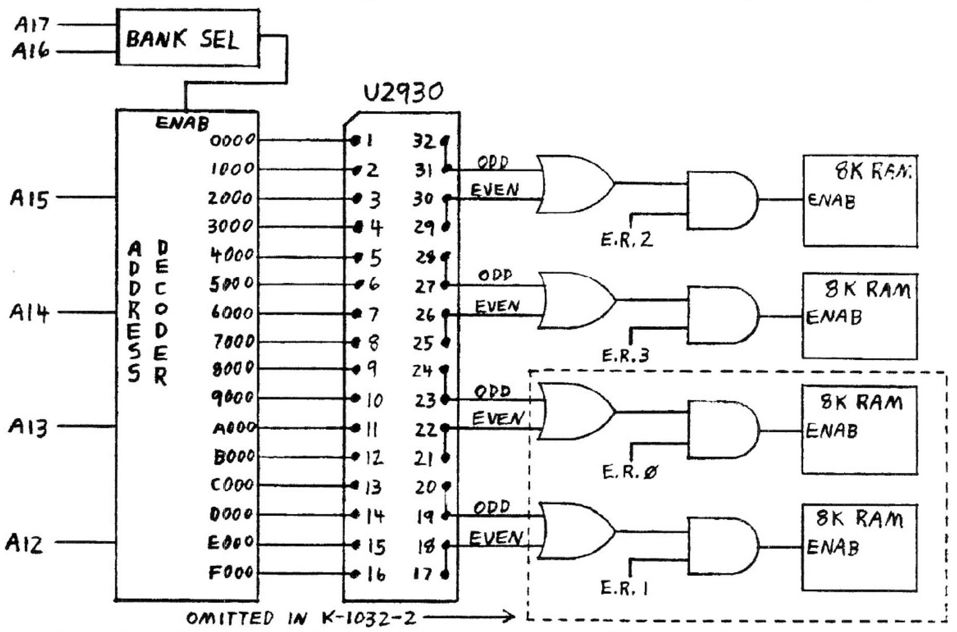
If 18 bit addressing is to be used, jumpers must be inserted into 8 pin socket U28 which is labelled RAM Bank Select on the schematic. 16 bit addressing is selected if there are <u>no</u> jumpers in this socket. The first step is to determine which 64K bank you wish <u>all</u> of the RAM on this board to reside in. Note that this bank selection applies only to the RAM section of the K-1032; the ROM and I/O sections have their own bank selection jumpers. The bank number may be between 0 and 3 according to the 4 possible combinations of the two extra address bits. Then, according to the table below, insert staple-shaped jumpers into the 8 pin U28 socket:

BANK	A17	A16	JUMPERS IN U28
0	0	0	2-7 4-5
1	0	1	2-7 3-6
2	1	0	1-8 4-5
3	1	1	1-8 3-6

3.6 RAM JUMPERS THEORY OF OPERATION

In order to take maximum advantage of the powerful bank switching capabilities of the Banker, RAM addressing has been made very flexible. For the purposes of addressing, the 32K of RAM is broken into 4 blocks of 8K each and each block may be individually addressed and enabled or disabled. Each block is in turn broken into two 4K segments for address jumpering purposes.

The drawing below is a conceptual view of the RAM addressing circuitry:



The Address Decoder on the left decodes all sixteen 4K segments of the possible 64K of addresses. If 18 bit addressing is used, the top left portion of the decoder recognizes the desired 64K "bank" that the RAM will be in. In either case, the 16 decoded 4K segments terminate in the left column of jumper terminals (U2930 pins 1-16). Each of the 4 RAM blocks on the right has two address selection inputs, one for the lower 4K segment and one for the upper 4K segment in the block. These 8 inputs terminate in the right column of jumper terminals (U2930 pins 17-32).

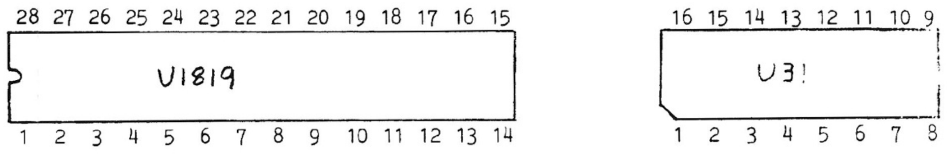
To activate a 4K segment of memory, a jumper must be wired from the desired decoder output (left column) to the desired memory segment input (right column). One has almost total freedom in this wiring. The only restriction is that one of the two inputs to a memory block must go to an even decoder output (0,2,4,6,8,A,C,E) while the other goes to an odd decoder output (1,3,5,7,9,B,D,F). Thus they cannot both be connected to odd outputs or both connected to even outputs. If one of the 8 segment selection inputs on the right is not connected to anything, then the corresponding 4K segment of memory is not active. Thus the RAM can be any size from 4K to 32K in 4K increments. For the K-1032-2 version of the Banker (16K of RAM), only the bottomost two blocks (4 segments) are available.

After an 8K block is selected, it still must be enabled to actually activate the RAM. This is represented by the logical AND gates towards the right above. Four bits of the Enable Register correspond to the four 8K blocks. As shipped, the Banker is set up to enable all 4 blocks after a Reset but the user can also write into the Enable Register to effect bank switching. For details, see section 7 which discusses RAM bank switching.

4. ROM ADDRESS JUMPERS

The ROM address jumper section is implemented in two sections. The first section is comprised of 28 pin socket U1819 and the second section is comprised of 16 pin socket U31 and its mating 16 pin plug. There are additional jumpers that select between 2K byte PROM's and 4K byte PROM's. Lastly, there is a jumper area that can convert each of the 4 sockets individually from programmable ROM's to masked ROM's. As shipped, the K-1032-1 is set up for 2K byte PROM's (Intel 2716 or equivalent).

The pin arrangements of the two ROM address jumper sockets are shown below:



Since all of the jumpers used in U1819 are straight across, the small staple-shaped jumpers supplied with the K-1032 should be used. The jumpers in U31 instead are usually scrambled up so it has a mating plug supplied. The best type of wire to use on the jumper plug is the #30 wire-wrap wire with teflon insulation supplied with the K-1032. Although the teflon insulation does not melt under soldering heat, it does require a sharp stripping tool. To disable all of the ROM regardless of the Enable Register contents, simply remove the plug from U31.

4.1 GENERAL PROCEDURE FOR SETTING 2K PROM ADDRESS JUMPERS

Use the following procedure for determining the proper 2K PROM address jumper settings for your system:

1. Verify that you desire to use 2K PROM's. If you wish to use 4K PROM's or masked ROM's, skip to section 4.2 or 4.3. If the board had been jumpered before, make sure that there is a jumper between U27 pins 1 and 8 and that there is no jumper between pins 2 and 7 of U28.

2. Decide which 16K block of address space all of your PROM's on this board will reside in. There are 4 choices: 0000-3FFF (Reference Number 0), 4000-7FFF (R.N. 1), 8000-BFFF (R.N. 2), or C000-FFFF (R.N. 3). If 18 bit addressing is to be used, also decide the 64K bank number (0 - 3).

3. First remove all jumpers (if any) from U1819. Then insert the following staple-shaped jumpers into U1819: pin 14 to pin 15, pin 12 to pin 17, and pin 10 to pin 19.

4. Refer to the table below for additional jumpers required in U1819 as a function of the Reference Number established in step 2:

Reference Number	Jumpers Required	
0	5-24	7-22
1	5-24	8-21
2	6-23	7-22
3	6-23	8-21

5. If 16 bit addressing is to be used, skip to step 6. Otherwise refer to the table below for setting the bank selection jumpers in socket U1819:

Bank Number	Jumpers Required	
0	2-27	4-25
1	2-27	3-26
2	1-28	4-25
3	1-28	3-26

6. Make a list of the starting addresses for each of the PROM's to be used. Note that each starting address must be on a 2K boundary. This list must have 4 or fewer entries.

7. Make a list of offset addresses from the step 6 list by subtracting the ROM block address established in step 2 from each of the starting addresses established in step 6.

8. Associate each offset address established in step 7 with a pin number in U31 according to the following table:

Offset Address	Associated Pin Number
0000	1
0800	2
1000	3
1800	4
2000	5
2800	6
3000	7
3800	8

9. Decide which ROM socket each ROM will plug into and associate it with a pin number in U31 according to the following list:

PROM Socket	U31 Pin Number	Enable Register Bit
U11	9	4
U3	11	5
U12	13	6
U4	15	7

10. Connect a jumper wire on the 16 pin plug from each pin established in step 8 to each pin established in step 9. Use a minimum of soldering heat and have the plug installed in its socket while soldering to prevent melting of the plug's plastic base.

4.2 GENERAL PROCEDURE FOR SETTING 4K PROM ADDRESS JUMPERS

Use the following procedure for determining the proper 4K PROM or ROM address jumper settings for your system (see also sect. 4.3 for 4K masked ROM's):

1. Verify that you desire to use 4K PROM's or masked ROM's. If you wish to use 2K PROM's, refer to section 4.1.

2. Remove the staple-shaped jumper between pins 1 and 8 of U27 and insert it between pins 2 and 7 of U27.

3. Decide which 32K block of address space <u>all</u> of your PROM's will reside in. There are 2 choices: 0000-7FFF (reference number 0) and 8000-FFFF (reference number 1). If 18 bit addressing is to be used, also decide the 64K bank number (0 - 3).

4. First remove <u>all</u> jumpers (if any) in socket U1819. Then insert the following staple-shaped jumpers into U1819: pin 13 to pin 16, pin 11 to pin 18, and pin 9 to pin 20.

5. If the reference number established in step 3 is 0, insert a jumper between U1819 pin 5 and pin 24. If it is 1, insert a jumper between pins 6 and 23 instead.

6. If 16 bit addressing is to be used, skip to step 7. Otherwise refer to the table below for setting the bank selection jumpers in socket U1819:

Bank Number	Jumpers Required	
0	2-27	4-25
1	2-27	3-26
2	1-28	4-25
3	1-28	3-26

7. Make a list of the starting addresses for each of the PROM's to be used. Note that each starting address must be on a 4K boundary. This list must have 4 or fewer entries.

8. Make a list of <u>offset</u> addresses from the step 7 list by subtracting the ROM <u>block</u> address established in step 3 from each of the starting addresses established in step 6.

9. Associate each offset address established in step 8 with a pin number in U31 according to the following table:

Offset Address	Associated Pin Number
0000	1
1000	2
2000	3
3000	4
4000	5
5000	6
6000	7
7000	8

10. Decide which ROM socket each ROM will plug into and associate it with a pin number in U31 according to the following list:

PROM Socket	U31 Pin Number	Enable Register Bit
U11	9	4
U3	11	5
U12	13	6
U4	15	7

11. Connect a jumper wire on the 16 pin plug from each pin established in step 8 to each pin established in step 9. Use a minimum of soldering heat and have the plug installed in its socket while soldering to prevent melting of the plug's plastic base.

4.3 CONVERSION FROM PROM'S TO MASKED ROM'S

Often desirable software functions are available in the form of 4K masked ROM's. In order to use these masked ROM's, the ROM address jumpering must have been set up for 4K PROM's as described in section 4.2 and the sockets to receive the masked ROM's must be modified to give the proper chip select signals to the masked ROM's. Although some ROM boards or systems may be able to use PROM's and masked ROM's interchangably, they accomplish this at the expense of high continuous power dissipation when PROM's are used. MTU has chosen to use both chip select inputs which reduces power consumption 75% when a PROM is not being accessed.

The jumper area for conversion from PROM's to ROM's is two sets of pads just above the U11 and U12 ROM sockets. The left set is numbered J1 through J12 while the right set is numbered J13 through J24. On the back of the board there are printed circuit traces between the pads corresponding to all of the even numbered J's. The following table relates these jumpers to the individual ROM sockets:

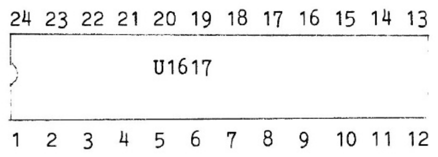
Jumpers for PROM (in PC)	Jumpers for Masked ROM	PROM Socket	Enable Register Bit
J2 J4 J6	J1 J3 J5	U11	4
J8 J10 J12	J7 J9 J11	U3	5
J14 J16 J18	J13 J15 J17	U12	6
J20 J22 J24	J19 J21 J23	U4	7

To change a socket to accept a masked ROM, first use a sharp knife to sever the printed circuit trace between the 3 even numbered J's associated with the socket. Then solder jumper wires into the 3 odd numbered J's associated with the socket.

5. I/O ADDRESSING JUMPERS

Setting the I/O address jumpers is considerably simpler than setting the RAM or ROM address jumpers. Note that even if you have the K-1032-2, which does not have the 6522 I/O chips, that the I/O address must still be set so that the Enable Register can be written into. If you do not plan to manipulate the Enable Register, the I/O section can be jumpered for a null address and the default enable jumpers set to enable everything after reset. See below for details.

The I/O address jumpers are staple-shaped pieces of wire that plug into U1617 which is a 24 pin IC socket. All jumpers go straight across the socket. The pin numbering of U1617 is shown below:



5.1 JUMPERING PROCEDURE

Use the following procedure for setting the I/O address jumpers:

1. Make sure that there are no jumpers inserted into socket U1617.

2. Select the desired I/O Base address in hexadecimal according to one of the following possibilities: XX00, XX40, XX80, XXC0, where X can be any hex digit. I/O addresses extend from the Base address to Base+2F. If 18 bit addressing is to be used, also select a 64K bank number (0 - 3). Note that I/O addresses can overlap addresses already assigned to the RAM or ROM. In this case, the 48 active I/O addresses take precedence over the RAM or ROM thus "shadowing" it.

3. Write out the selected Base Address in binary and then discard the rightmost 6 bits which will be zeroes. For 18 bit addressing, express the bank number in binary as well.

4. Use the table below to associate each of the resulting 10 address bits (12 with 18 bit addressing) with a jumper position in U1617:

Addr Bit 6	1-24	Addr Bit 10	5-20	Addr Bit 14	9-16
Addr Bit 7	2-23	Addr Bit 11	6-19	Addr Bit 15	10-15
Addr Bit 8	3-22	Addr Bit 12	7-18	Bank Bit 0	11-14
Addr Bit 9	4-21	Addr Bit 13	8-17	Bank Bit 1	12-13

5. For each Zero bit in the binary address representation established in step 3, insert a jumper into the corresponding position established in step 4. For each One, do nothing.

6. If 18 bit addressing is to be used, insert a jumper between pins 4 and 7 of U27. If this jumper is omitted, the I/O address decoder will ignore the 17th and 18th address bits.

7. The I/O section may be disabled (will not respond to any addresses regardless of the address jumpering), by connecting a wire between Test Point 3 (a little post marked TP3 just below U8) and ground. Note that this will prevent program access to the Enable Register as well.

6. OTHER JUMPER OPTIONS

Besides the addressing jumpers for RAM, ROM, and I/O functions, there are several other jumper options on the K-1032 Banker board. Each of these are set by inserting staple-shaped jumpers into the designated positions of IC sockets.

6.1 DEFAULT ENABLES

The Default Enable jumpers determine the setting of the Enable Register after power-up or a system Reset. With all of the jumpers removed (which is the way the board is shipped), all 4 RAM blocks and all 4 ROM sockets (if present) are enabled after power-up or reset. To have a block of RAM or a ROM socket come up disabled, it is necessary to insert the corresponding jumper into U32. The table below relates jumper pin numbers to bits in the Enable Register:

RAM CONTROL

Enable Reg Bit	U32
0	8-9
1	7-10
2	6-11
3	5-12

ROM CONTROL

Enable Reg Bit	ROM Socket	U32
4	U11	4-13
5	U3	3-14
6	U12	2-15
7	U4	1-16

6.2 I/O INTERRUPT ENABLE

The two 6522 I/O chips are capable of interrupting the host processor when they detect certain conditions. However a jumper must be inserted between pins 3 and 6 of U27 to connect the interrupt request outputs of these chips to the IRQ bus line. The K-1032 is shipped with the jumper removed which means that the I/O section cannot interrupt the host CPU unless this jumper is installed.

6.3 PROM PROGRAMMER PROM TYPE

Jumper socket U54 determines whether the PROM programming socket (if present) is set up for 2716 type PROM's (2K bytes) or 2732 type PROM's (4K bytes). As shipped, it is set up for 2716's since they are much less expensive and more available than 2732's. Use the following table for setting the PROM programmer PROM type:

2716 PROM	2732 PROM
U54 pins 2-7, 4-5	U54 pins 1-8, 3-6

If the programming socket is not configured for the type of PROM being programmed, the PROM can be damaged. The PROM programmer cannot program 2708 or TMS2716 type PROM's (multiple voltage type).

7. USE OF THE BANK SWITCHING FEATURES

The K-1032 Banker Memory incorporates two methods of extending the amount of memory that a 6502 based system may address. Internal bank switching logic allows various blocks of ROM and RAM on the K-1032 to be switched on and off under program control. This in turn allows 2 or more of these blocks to reside at the same address and thus extend the amount of data storage directly accessable by the microprocessor. The external bank switching feature makes the K-1032 capable of recognizing two additional address bits. This in turn allows external bank switching logic on the CPU board or elsewhere in the system to control up to 256K of memory addresses.

The advantage of the internal bank switching feature is the ability to immediately utilize bank switching to extend memory addressing range when using existing CPU boards such as the KIM-1, SYM-1, AIM-65, etc. The external bank switching feature allows the K-1032 to be used with advanced MTU CPU boards that will have 18 address bits.

7.1 USE OF INTERNAL BANK SWITCHING

The key to the internal bank switching feature of the K-1032 is the 8-bit Enable Register. Each bit of this register corresponds to a "resource" on the K-1032. Four of the resources are the four 8K blocks of RAM on the K-1032-1 (there are only two 8K blocks on the K-1032-2). The other four resources are the 4 ROM sockets (not present on the K-1032-2).

When a bit in the Enable Register is a Zero, the corresponding resource is disabled. This means that when the CPU attempts to read any addresses assigned to that resource that it will not respond and when the CPU attempts to write, data within the resource is not changed. If no resource on the K-1032 responds to an address, then the K-1032 bus buffers are not activated thus allowing other boards in the system to respond without conflict. When a bit in the Enable Register is a One, the corresponding resource is enabled and responds to read and write cycles just as if there was no internal bank switching logic on the K-1032.

When the system is powered up or reset, it is desirable that the Enable Register "come up" in a predictable state. This is doubly important if the system's operating software resides in ROM on-board the K-1032. A set of jumpers is used to establish the "Default Enable" state of the Enable Register whenever the system is reset. When a K-1032 is shipped from the factory, these jumpers are set to force the Enable Register to all One's after reset and thus enable all of the K-1032's resources. By inserting jumpers into socket U32, one can force selected Enable Register bits to Zero after power up and thus disable the associated resource. This is particularly important if some resources share common addresses. Please refer to section 6.1 for additional information on setting the Default Enable jumpers.

The first use example of the internal bank switching feature is using the full K-1032-1 in a system that does not have sufficient address space to address everything on the board at once. Lets assume an AIM-65 system that has a Visible Memory from 6000-7FFF, a K-1032 Disk Controller from 4000-5FFF and 8000-9FFF, the BASIC ROM from B000-CFFF, the assembler ROM from D000-DFFF, and the original 4K of RAM from 0000-0FFF. This leaves only 12K of address space from 1000-3FFF (E000-FFFF is the AIM monitor) for all of the Banker Memory's functions.

One way to organize the Banker Memory's resources is to address RAM block 0 from 1000-1FFF (4K of this block are unused; you could also remove the 4K of RAM from the AIM and replace it with Banker RAM and save a substantial amount of power in the process) and address RAM block 1 at 2000-3FFF. RAM block 2 would also be addressed at 2000-3FFF and RAM block 3 could be addressed at B000-CFFF. Next, the BASIC ROM's and assembler ROM would be removed from the AIM and plugged into ROM sockets 0, 1, and 2 on the Banker which are addressed at B000, C000, and D000 respectively. Finally, a user PROM with perhaps graphics subroutines (for the Visible Memory) in it would be put in ROM socket 3 addressed at D000. I/O on the Banker Memory would be put at AE00 (see application note 6 for AIM modifications to allow this). Now the Default Enable jumpers should be altered so that RAM blocks 0 and 1 and ROM sockets 0, 1, and 2 all come up <u>enabled</u> while RAM blocks 2 and 3 and ROM socket 3 comes up disabled.

An AIM system organized this way can be used in several different "modes". For example, when the system is first turned on or reset, the BASIC and assembler ROM's work normally. A machine language program could get an extra 8K of easily accessable memory at B000-CFFF by turning the BASIC ROM <u>off</u> and turning RAM block 3 <u>on</u>. For even more storage, it could alternate between RAM block 1 from 2000-3FFF and RAM block 2 there. A BASIC program that uses the user supplied graphics subroutine ROM mentioned earlier could turn the assembler ROM off and turn the user's ROM in socket 3 on. These and other system configurations can be selected merely by writing the appropriate bit pattern into the Enable Register addressed at I/OBASE+$20. If at any time the user program crashes, pressing reset on the AIM will <u>hardware</u> set the Enable Register for normal AIM operation.

One can also configure systems with massive amounts of memory by using two or more Banker Memory boards. Assuming that all that is desired is RAM expansion, several K-1032-3's (32K RAM only, no ROM or I/O) could be set up with the RAM on each board at the same address. By suitable manipulation of the Enable Registers, one board could be effectively "turned on" while the others were turned off. This sort of thing is ideal for multi-user or multi-task systems. Note however that the I/O addresses must be unique but that usually is not a problem.

<u>7.2</u> USE OF EXTERNAL BANK SWITCHING

The exact details of use of the 18 bit addressing feature of the K-1032 is dependent mainly on the system that is providing the 18 bit address bus. One could also view the two extra "address" bits as two "board enable" inputs that the user can program (by means of jumpers) to be active high or active low. Thus one could also connect these two extra address bits to I/O ports and control multiple Banker Memory boards (or other boards with 18 bit address busses) through a system I/O port. Sophisticated MTU CPU's of the future will have 18 bit addressing logic built into the CPU and possibly implemented in a transparent manner that does not even require a user program to be cognizant of bank switching taking place!

8. USING THE PROM PROGRAMMER

The easiest way to use the PROM programmer on the K-1032 is to use the PROM programmer program listed in section 18. The K-1032 is capable of programming 2716 (except TMS2716), TMS2516, and 2732 type PROM's. It cannot program the multi-voltage type PROM's such as 2708's, 2704's, TMS2716's or 1702's.

The programming socket on the K-1032 is at the rear of the board right next to the red light emitting diode. It is a high quality socket that will withstand hundreds of insertions and withdrawals although it does grip the ROM rather firmly. For use as a production programmer it is recommended that the user obtain a "zero insertion force" socket (such as Textool), and connect it to a 24 pin header through a short (6 inches) cable. This also allows easy access to the programming socket even when the K-1032 is installed in a card file.

It is recommended that the program listed in section 18 be used to program PROM's. If the user desires to write his own instead, consult that listing and also section 9.11 in the Principles of Operation writeup for programming details.

8.1 STEP-BY-STEP PROGRAMMING PROCEDURE

Use the following procedure for programming a PROM:

1. Be sure the PROM is thoroughly erased. Germacidal or ozone lamps generally require 30 minutes (longer if the lamp is old) with the PROM placed 1 inch from the arc. New PROM's should be erased since their history is unknown.

2. Disconnect any peripheral devices connected to VIA 2 port A or bits 0-4 of port B that may be disturbed by random signals or which significantly load these lines.

3. Load the data to be programmed into RAM somewhere other than pages 0-3. As written, the programmer program will program the entire PROM from memory at once. Note that the data need not reside at the same addresses that it will when the programmed PROM's are installed in the K-1032.

4. Load the PROM programmer program into RAM from the listing. Store the <u>base address</u> of the I/O section of the K-1032 into RAM at location 0007 (low) and 0008 (high). Store the address of the data loaded in step 3 into RAM at 0004 (low) and 0005 (high).

5. <u>Certify</u> that the PROM type jumpers in U54 on the K-1032 match the type of PROM to be programmed (see section 6.3). If you are going to program a 2K PROM (2716 type), store 00 in location 0006. If you are going to program a 4K PROM (2732 type), store FF in location 0006.

6. Insert the blank PROM into the programming socket. This should be done with the power <u>on</u> to prevent false programming The PROM should be tilted when inserted such that pin 12 engages the socket first. If a Textool socket is being used, close lock the socket quickly. If the environment is dry, discharge your body to ground before plugging the PROM in.

7. Verify that the PROM is blank by executing NEWPRM at location 0200. When the monitor is re-entered (verification only takes a fraction of a second), location 0000 will read FF if the PROM is indeed blank. Otherwise it will contain the contents of the first non-blank cell and locations 0002 (low) and 0003 (high) will contain the <u>RAM</u> address (see step 3 above) of the corresponding non-blank location in the PROM.

8. Execute PGMVFY at location 0203 to actually program the PROM. The program voltage LED should light during programming (although not very brightly) and may flicker to some extent. Programming will require about 5 minutes for a 2K PROM and 10 minutes for a 4K PROM.

9. After programming, the contents of the PROM are compared against RAM and the host system monitor is re-entered. If this verify is successful, locations 0000 through 0003 will all contain zeroes. If the verify was unsuccessful, location 0000 will contain the data actually in the PROM, location 0001 will contain the data that should have been in the PROM, and locations 0002 (low) and 0003 (high) will contain the RAM address of the data that did not program correctly.

10. Remove the PROM from the programming socket such that pin 12 breaks contact with the socket last. If a Textool type of socket is being used, unlock it quickly. If the PROM verified, set it aside for use later. If it did not verify, write a question mark on it and restock unless it already has a question mark in which case it should be discarded or returned to its point of purchase.

8.2 ADDITIONAL PROGRAMMER PROGRAM FEATURES

An unknown PROM may be compared against RAM contents by following steps 2-6 and then entering VERIFY at 0206. This is the same routine that is executed after a programming cycle and it signals errors in the same manner (see step 9 above).

A PROM may be copied by first reading it into RAM with READ at location 0209. Set up the RAM address in 0004 and 0005 and go to READ at 0209. The original PROM may now be removed and a blank one installed. Follow the programming procedure outlined above to program the copy PROM. There is no danger of damaging the original PROM as long as System Reset is not pressed and the programming routine is not entered.

9. PROGRAMMING THE 6522 I/O PORTS

The two 6522 VIA (Versitile Interface Adapter) chips provide 4 independent 8 bit ports plus 8 control lines for standard port-oriented I/O. In addition, each 6522 has two counter/timers and an 8 bit shift register. This section will describe how to use the parallel ports for simple I/O interfacing functions. The reader should refer to the 6522 data sheet in section 10 for detailed programming instructions for the control lines, counter/timers, and the shift register.

9.1 PORT ADDRESSING

Each 6522 VIA on the K-1032 uses 16 I/O addresses. VIA 1 is assigned addresses from I/OBASE through I/OBASE+15 (decimal) while VIA 2 uses addresses from I/OBASE+16 through I/OBASE+31. About half of these 16 addresses refer to registers inside the VIA and the other half are used to control the two counters. The table below gives the function of each of these addresses:

VIA 1 ADDR.	VIA 2 ADDR.	FUNCTION
I/OBASE	I/OBASE+16	Port B data register
I/OBASE+1	I/OBASE+17	Port A data register (normal handshake operation)
I/OBASE+2	I/OBASE+18	Port B direction register
I/OBASE+3	I/OBASE+19	Port A direction register
I/OBASE+4	I/OBASE+20	Write timer 1 low latch. Read timer 1 low count and clear timer 1 interrupt flag.
I/OBASE+5	I/OBASE+21	Write timer 1 high latch and high count, transfer low latch to low count. Read timer 1 high count.
I/OBASE+6	I/OBASE+22	Write timer 1 low latch. Read timer 1 low count.
I/OBASE+7	I/OBASE+23	Write timer 1 high latch. Read timer 1 high count.
I/OBASE+8	I/OBASE+24	Write timer 2 low latch. Read timer 2 low count, clear timer 2 interrupt flag.
I/OBASE+9	I/OBASE+25	Write timer 2 high count, transfer low latch to low count, clear timer 2 interrupt flag. Read high count.
I/OBASE+10	I/OBASE+26	Shift register
I/OBASE+11	I/OBASE+27	Auxiliary control register (counter & shift reg. mode)
I/OBASE+12	I/OBASE+28	Peripheral control register (CA1, CA2, CB1, CB2 contrl)
I/OBASE+13	I/OBASE+29	Interrupt flag register
I/OBASE+14	I/OBASE+30	Interrupt enable register
I/OBASE+15	I/OBASE+31	Port A data register (no effect on handshake)

For simple port-oriented I/O, it is important that the Auxiliary control register, Peripheral control register, Interrupt flag register, and Interrupt enable register all be zeroes. Normally system reset can be counted upon to establish this condition. However if the I/O program must be completely self-initializing after a crash, it can simple write 00 into these registers.

The next step is to determine whether the port is to be an input port or an output port. If Port A is to be input, for example, then the program should write 00 into the Port A direction register. This is also accomplished by system reset which sets both direction registers to 00 and thus programs both data registers for inputs. To program a port for all outputs, FF must be written into its corresponding direction register. One can also mix inputs and outputs on the same port if desired; simply set bits in the direction register to ones that are desired to be outputs in the data register.

Examination of the chart above will reveal that the Port A data register can be reached at two different addresses. With the Peripheral control register set to 00 there is no significant difference between the action of the two addresses.

IRQ (Interrupt Request)
The Interrupt Request output goes low whenever an internal interrupt flag is set and the corresponding interrupt enable bit is a logic 1. This output is "open-drain" to allow the interrupt request signal to be "wire-or'ed" with other equivalent signals in the system.

PA0-PA7 (Peripheral A Port)
The Peripheral A port consists of 8 lines which can be individually programmed to act as inputs or outputs under control of a Data Direction Register. The polarity of output pins is controlled by an Output Register and input data may be latched into an internal register under control of the CA1 line. All of these modes of operation are controlled by the system processor through the internal control registers. These lines represent one standard TTL load in the input mode and will drive one standard TTL load in the output mode. Figure 7 illustrates the output circuit.

CA1, CA2 (Peripheral A Control Lines)
The two Peripheral A control lines act as interrupt inputs or as handshake outputs. Each line controls an internal interrupt flag with a corresponding interrupt enable bit. In addition, CA1 controls the latching of data on Peripheral A port input lines. CA1 is a high-impedance input only while CA2 represents one standard TTL load in the input mode. CA2 will drive one standard TTL load in the output mode.

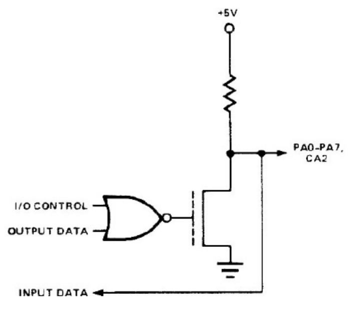


Figure 7. Peripheral A Port Output Circuit

PB0-PB7 (Peripheral B Port)
The Peripheral B port consists of eight bi-directional lines which are controlled by an output register and a data direction register in much the same manner as the PA port. In addition, the polarity of the PB7 output signal can be controlled by one of the interval timers while the second timer can be programmed to count pulses on the PB6 pin. Peripheral B lines represent one standard TTL load in the input mode and will drive one standard TTL load in the output mode. In addition, they are capable of sourcing 1.0mA at 1.5VDC in the output mode to allow the outputs to directly drive Darlington transistor circuits. Figure 8 is the circuit schematic.

CB1, CB2 (Peripheral B Control Lines)
The Peripheral B control lines act as interrupt inputs or as handshake outputs. As with CA1 and CA2, each line controls an interrupt flag with a corresponding interrupt enable bit. In addition, these lines act as a serial port under control of the Shift Register. These lines represent one standard TTL load in the input mode and will drive one standard TTL load in the output mode. Unlike PB0-PB7, CB1 and CB2 cannot drive Darlington transistor circuits.

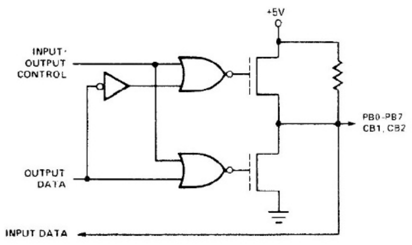


Figure 8. Peripheral B Port Output Circuit

FUNCTIONAL DESCRIPTION

Port A and Port B Operation
Each 8-bit peripheral port has a Data Direction Register (DDRA, DDRB) for specifying whether the peripheral pins are to act as inputs or outputs. A 0 in a bit of the Data Direction Register causes the corresponding peripheral pin to act as an input. A 1 causes the pin to act as an output.

Each peripheral pin is also controlled by a bit in the Output Register (ORA, ORB) and an Input Register (IRA, IRB). When the pin is programmed as an output, the voltage on the pin is controlled by the corresponding bit of the Output Register. A 1 in the Output Register causes the output to go high, and a "0" causes the output to go low. Data may be written into Output Register bits corresponding to pins which are programmed as inputs. In this case, however, the output signal is unaffected.

Reading a peripheral port causes the contents of the Input Register (IRA, IRB) to be transferred onto the Data Bus. With input latching disabled, IRA will always reflect the levels on the PA pins. With input latching enabled, IRA will reflect the levels on the PA pins at the time the latching occurred (via CA1).

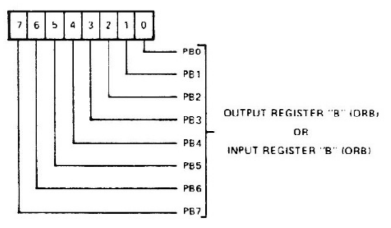
The IRB register operates similar to the IRA register. However, for pins programmed as outputs there is a difference. When reading IRA, the level on the pin determines whether a 0 or a 1 is sensed. When reading IRB, however, the bit stored in the output register, ORB, is the bit sensed. Thus, for outputs which have large loading effects and which pull an output "1" down or which pull an output "0" up, reading IRA may result in reading a "0" when a "1" was actually programmed, and reading a "1" when a "0" was programmed. Reading IRB, on the other hand, will read the "1" or "0" level actually programmed, no matter what the loading on the pin.

Figures 9, 10, and 11 illustrate the formats of the port registers. In addition, the input latching modes are selected by the Auxiliary Control Register (Figure 16.)

Handshake Control of Data Transfers

The SY6522 allows positive control of data transfers between the system processor and peripheral devices

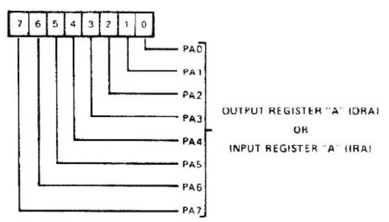
REG 0 – ORB/IRB



Pin Data Direction Selection	WRITE	READ
DDRB = "1" (OUTPUT)	MPU writes Output Level (ORB)	MPU reads output register bit in ORB. Pin level has no affect
DDRB = "0" (INPUT) (Input latching disabled)	MPU writes into ORB, but no effect on pin level, until DDRB changed	MPU reads input level on PB pin
DDRB = "0" (INPUT) (Input latching enabled)		MPU reads IRB bit, which is the level of the PB pin at the time of the last CB1 active transition

Figure 9. Output Register B (ORB), Input Register B (IRB)

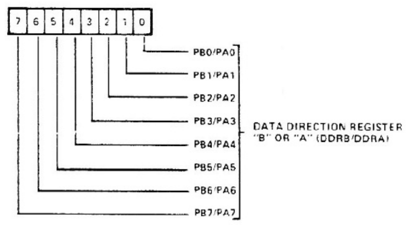
REG 1 – ORA/IRA



Pin Data Direction Selection	WRITE	READ
DDRA = "1" (OUTPUT) (Input latching disabled)	MPU writes Output Level (ORA)	MPU reads level on PA pin
DDRA = "1" (OUTPUT) (Input latching enabled)		MPU reads IRA bit which is the level of the PA pin at the time of the last CA1 active transition
DDRA = "0" (INPUT) (Input latching disabled)	MPU writes into ORA, but no effect on pin level, until DDRA changed	MPU reads level on PA pin
DDRA = "0" (INPUT) (Input latching enabled)		MPU reads IRA bit which is the level of the PA pin at the time of the last CA1 active transition

Figure 10. Output Register A (ORA), Input Register A (IRA)

REG 2 (DDRB) AND REG 3 (DDRA)



"0"	ASSOCIATED PB/PA PIN IS AN INPUT (HIGH IMPEDANCE)
"1"	ASSOCIATED PB/PA PIN IS AN OUTPUT, WHOSE LEVEL IS DETERMINED BY ORB ORA REGISTER BIT

Figure 11. Data Direction Registers (DDRB, DDRA)

through the operation of "handshake" lines. Port A lines (CA1, CA2) handshake data on both a read and a write operation while the Port B lines (CB1, CB2) handshake on a write operation only.

Read Handshake

Positive control of data transfers from peripheral devices into the system processor can be accomplished very effectively using Read Handshaking. In this case, the peripheral device must generate the equivalent of a "Data Ready" signal to the processor signifying that valid data is present on the peripheral port. This signal normally interrupts the processor, which then reads the data, causing generation of a "Data Taken" signal. The peripheral device responds by making new data available. This process continues until the data transfer is complete.

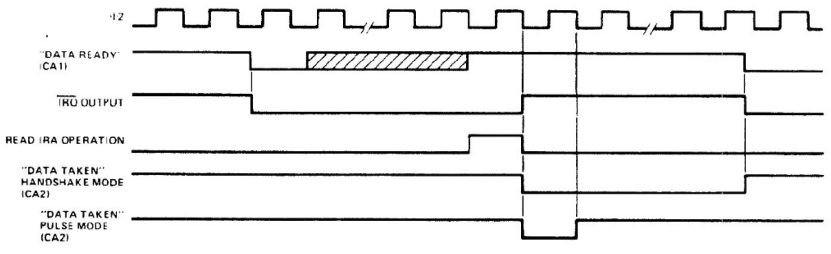

φ2
"DATA READY" (CA1)
IRQ OUTPUT
READ IRA OPERATION
"DATA TAKEN" HANDSHAKE MODE (CA2)
"DATA TAKEN" PULSE MODE (CA2)

Figure 12. Read Handshake Timing (Port A, Only)

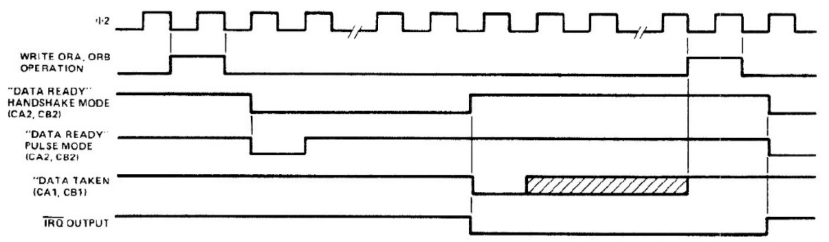

φ2
WRITE ORA, ORB OPERATION
"DATA READY" HANDSHAKE MODE (CA2, CB2)
"DATA READY" PULSE MODE (CA2, CB2)
"DATA TAKEN (CA1, CB1)
IRQ OUTPUT

Figure 13. Write Handshake Timing

In the SY6522, automatic "Read" Handshaking is possible on the Peripheral A port only. The CA1 interrupt input pin accepts the "Data Ready" signal and CA2 generates the "Data Taken" signal. The "Data Ready" signal will set an internal flag which may interrupt the processor or which may be polled under program control. The "Data Taken" signal can either be a pulse or a level which is set low by the system processor and is cleared by the "Data Ready" signal. These options are shown in Figure 12 which illustrates the normal Read Handshaking sequence.

Write Handshake

The sequence of operations which allows handshaking data from the system processor to a peripheral device is very similar to that described for Read Handshaking. However, for Write Handshaking, the SY6522 generates the "Data Ready" signal and the peripheral device must respond with the "Data Taken" signal. This can be accomplished on both the PA port and the PB port on the SY6522. CA2 or CB2 act as a "Data Ready" output in either the handshake mode or pulse mode and CA1 or CB1 accept the "Data Taken" signal from the peripheral device, setting the interrupt flag and cleaning the "Data Ready" output. This sequence is shown in Figure 13.

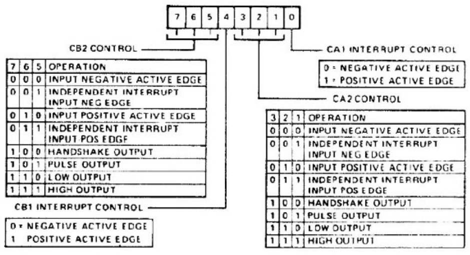
Selection of operating modes for CA1, CA2, CB1, and CB2 is accomplished by the Peripheral Control Register (Figure 14).

Timer Operation

Interval Timer T1 consists of two 8-bit latches and a 16-bit counter. The latches are used to store data which is to be loaded into the counter. After loading, the counter decrements at φ2 clock rate. Upon reaching zero, an interrupt flag will be set, and $\overline{\text{IRQ}}$ will go low if the interrupt is enabled. The timer will then disable any further interrupts, or will automatically transfer the contents of the latches into the counter and will continue to decrement. In addition, the timer may be programmed to invert the output signal on a peripheral pin each time it "times-out". Each of these modes is discussed separately below.

The T1 counter is depicted in Figure 15 and the latches in Figure 16.

REG 12 – PERIPHERAL CONTROL REGISTER



CB2 CONTROL

7	6	5	OPERATION
0	0	0	INPUT NEGATIVE ACTIVE EDGE
0	0	1	INDEPENDENT INTERRUPT INPUT NEG EDGE
0	1	0	INPUT POSITIVE ACTIVE EDGE
0	1	1	INDEPENDENT INTERRUPT INPUT POS EDGE
1	0	0	HANDSHAKE OUTPUT
1	0	1	PULSE OUTPUT
1	1	0	LOW OUTPUT
1	1	1	HIGH OUTPUT

CB1 INTERRUPT CONTROL

0 = NEGATIVE ACTIVE EDGE
1 POSITIVE ACTIVE EDGE

CA1 INTERRUPT CONTROL

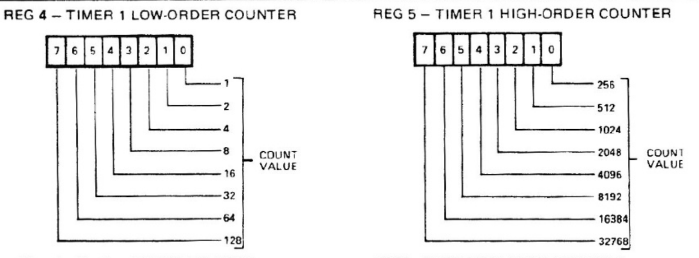
0 = NEGATIVE ACTIVE EDGE
1 = POSITIVE ACTIVE EDGE

CA2 CONTROL

3	2	1	OPERATION
0	0	0	INPUT NEGATIVE ACTIVE EDGE
0	0	1	INDEPENDENT INTERRUPT INPUT NEG EDGE
0	1	0	INPUT POSITIVE ACTIVE EDGE
0	1	1	INDEPENDENT INTERRUPT INPUT POS EDGE
1	0	0	HANDSHAKE OUTPUT
1	0	1	PULSE OUTPUT
1	1	0	LOW OUTPUT
1	1	1	HIGH OUTPUT

Figure 14. CA1, CA2, CB1, CB2 Control

Two bits are provided in the Auxiliary Control Register (bits 6 and 7) to allow selection of the T1 operating modes. The four possible modes are depicted in Figure 17.



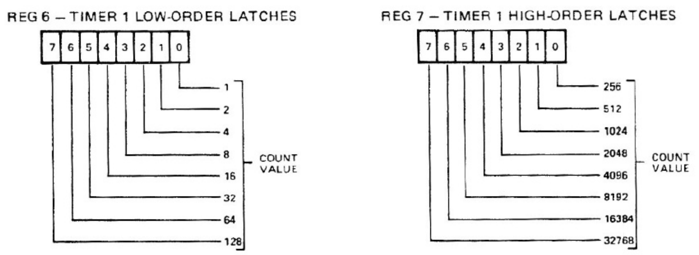
WRITE – 8 BITS LOADED INTO T1 LOW-ORDER LATCHES. LATCH CONTENTS ARE TRANSFERRED INTO LOW-ORDER COUNTER AT THE TIME THE HIGH ORDER COUNTER IS LOADED (REG 5).

READ – 8 BITS FROM T1 LOW-ORDER COUNTER TRANSFERRED TO MPU. IN ADDITION, T1 INTERRUPT FLAG IS RESET (BIT 6 IN INTERRUPT FLAG REGISTER).

WRITE – 8 BITS LOADED INTO T1 HIGH-ORDER LATCHES. ALSO, AT THIS TIME BOTH HIGH AND LOW-ORDER LATCHES TRANSFERRED INTO T1 COUNTER T1 INTERRUPT FLAG ALSO IS RESET

READ – 8 BITS FROM T1 HIGH-ORDER COUNTER TRANSFERRED TO MPU.

Figure 15. T1 Counter Registers



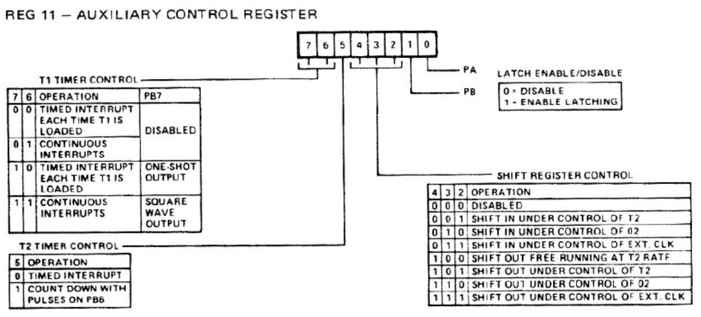
WRITE – 8 BITS LOADED INTO T1 LOW-ORDER LATCHES. THIS OPERATION IS NO DIFFERENT THAT A WRITE INTO REG 4.

READ – 8 BITS FROM T1 LOW-ORDER LATCHES TRANSFERRED TO MPU. UNLIKE REG 4 OPERATION, THIS DOES NOT CAUSE RESET OF T1 INTERRUPT FLAG.

WRITE – 8 BITS LOADED INTO T1 HIGH-ORDER LATCHES. UNLIKE REG 4 OPERATION NO LATCH-TO-COUNTER TRANSFERS TAKE PLACE

READ – 8 BITS FROM T1 HIGH-ORDER LATCHES TRANSFERRED TO MPU.

Figure 16. T1 Latch Registers



T1 TIMER CONTROL

7	6	OPERATION	PB7
0	0	TIMED INTERRUPT EACH TIME T1 IS LOADED	DISABLED
0	1	CONTINUOUS INTERRUPTS	
1	0	TIMED INTERRUPT EACH TIME T1 IS LOADED	ONE-SHOT OUTPUT
1	1	CONTINUOUS INTERRUPTS	SQUARE WAVE OUTPUT

T2 TIMER CONTROL

5	OPERATION
0	TIMED INTERRUPT
1	COUNT DOWN WITH PULSES ON PB6

SHIFT REGISTER CONTROL

4	3	2	OPERATION
0	0	0	DISABLED
0	0	1	SHIFT IN UNDER CONTROL OF T2
0	1	0	SHIFT IN UNDER CONTROL OF 02
0	1	1	SHIFT IN UNDER CONTROL OF EXT. CLK
1	0	0	SHIFT OUT FREE RUNNING AT T2 RATE
1	0	1	SHIFT OUT UNDER CONTROL OF T2
1	1	0	SHIFT OUT UNDER CONTROL OF 02
1	1	1	SHIFT OUT UNDER CONTROL OF EXT. CLK

Figure 17. Auxiliary Control Register

Note: The processor does not write directly into the low order counter (T1C-L). Instead, this half of the counter is loaded automatically from the low order latch when the processor writes into the high order counter. In fact, it may not be necessary to write to the low order counter in some applications since the timing operation is triggered by writing to the high order counter.

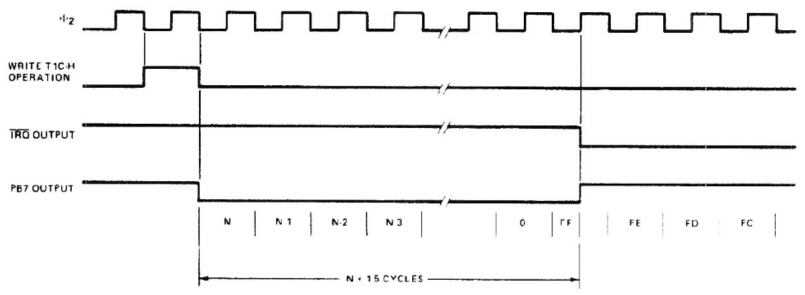


Figure 18. Timer 1 and Timer 2 One-Shot Mode Timing

Timer 1 One-Shot Mode

The interval timer one-shot mode allows generation of a single interrupt for each timer load operation. As with any interval timer, the delay between the "write T1C-H" operation and generation of the processor interrupt is a direct function of the data loaded into the timing counter. In addition to generating a single interrupt, Timer 1 can be programmed to produce a single negative pulse on the PB7 peripheral pin. With the output enabled (ACR7=1) a "write T1C-H" operation will cause PB7 to go low. PB7 will return high when Timer 1 times out. The result is a single programmable width pulse.

In the one-shot mode, writing into the high order latch has no effect on the operation of Timer 1. However, it will be necessary to assure that the low order latch contains the proper data before initiating the count-down with a "write T1C-H" operation. When the processor writes into the high order counter, the T1 interrupt flag will be cleared, the contents of the low order latch will be transferred into the low order counter, and the timer will begin to decrement at system clock rate. If the PB7 output is enabled, this signal will go low on the phase two following the write operation. When the counter reaches zero, the T1 interrupt flag will be set, the $\overline{IRQ}$ pin will go low (interrupt enabled), and the signal on PB7 will go high. At this time the counter will continue to decrement at system clock rate. This allows the system processor to read the contents of the counter to determine the time since interrupt. However, the T1 interrupt flag cannot be set again unless it has been cleared as described in this specification.

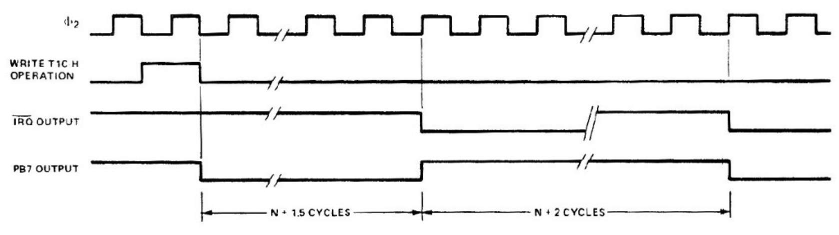
Timing for the SY6522 interval timer one-shot modes is shown in Figure 18.

Timer 1 Free-Run Mode

The most important advantage associated with the latches in T1 is the ability to produce a continuous series of evenly spaced interrupts and the ability to produce a square wave on PB7 whose frequency is not affected by variations in the processor interrupt response time. This is accomplished in the "free-running" mode.

In the free-running mode, the interrupt flag is set and the signal on PB7 is inverted each time the counter reaches zero. However, instead of continuing to decrement from zero after a time-out, the timer automatically transfers the contents of the latch into the counter (16 bits) and continues to decrement from there. The interrupt flag can be cleared by writing T1C-H, by reading T1C-L, or by writing directly into the flag as described later. However, it is not necessary to rewrite the timer to enable setting the interrupt flag on the next time-out.

All interval timers in the SY6522 are "re-triggerable". Rewriting the counter will always re-initialize the time-out period. In fact, the time-out can be prevented completely if the processor continues to rewrite the timer before it reaches zero. Timer 1 will operate in this manner if the processor writes into the high order counter (T1C-H). However, by loading the latches only, the processor can access the timer during each down-counting operation without affecting the time-out in process. Instead, the data loaded into the latches will determine the length of the next time-out period. This capability is particularly valuable in the free-running mode with the output enabled. In this mode, the signal on PB7 is inverted and the interrupt flag is set with each time-out. By responding to the interrupts with new data for the latches, the processor can determine the period of the next half cycle during each half cycle of the output signal on PB7. In this manner, very complex waveforms can be generated. Timing for the free-running mode is shown in Figure 19.



Note: A precaution to take in the use of PB7 as the timer output concerns the Data Direction Register contents for PB7. Both DDRB bit 7 and ACR bit 7 must be 1 for PB7 to function as the timer output. If one is 1 and the other is 0, then PB7 functions as a normal output pin, controlled by ORB bit 7.

Figure 19. Timer 1 Free-Run Mode Timing

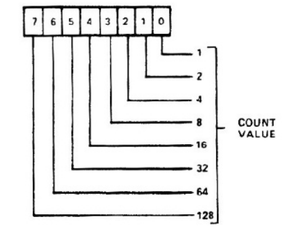
Timer 2 Operation

Timer 2 operates as an interval timer (in the "one-slot" mode only), or as a counter for counting negative pulses on the PB6 peripheral pin. A single control bit is provided in the Auxiliary Control Register to select between these two modes. This timer is comprised of a "write-only" low-order latch (T2L-L), a "read-only" low-order counter and a read/write high order counter. The counter registers act as a 16-bit counter which decrements at Φ2 rate. Figure 20 illustrates the T2 Counter Registers.

Timer 2 One-Shot Mode

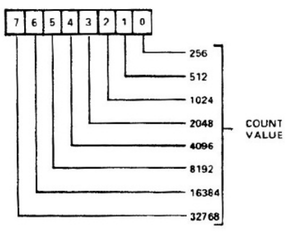
As an interval timer, T2 operates in the "one-shot" mode similar to Timer 1. In this mode, T2 provides a single interrupt for each "write T2C-H" operation. After timing out, the counter will continue to decrement. However, setting of the interrupt flag will be disabled after initial time-out so that it will not be set by the counter continuing to decrement through zero. The processor must rewrite T2C-H to enable setting of the interrupt flag. The interrupt flag is cleared by reading T2C-L or by writing T2C-H. Timing for this operation is shown in Figure 18.

REG 8 – TIMER 2 LOW-ORDER COUNTER



WRITE – 8 BITS LOADED INTO T2 LOW-ORDER LATCHES.
READ – 8 BITS FROM T2 LOW-ORDER COUNTER TRANSFERRED TO MPU. T2 INTERRUPT FLAG IS RESET.

REG 9 – TIMER 2 HIGH-ORDER COUNTER



WRITE – 8 BITS LOADED INTO T2 HIGH-ORDER COUNTER. ALSO, LOW-ORDER LATCHES TRANSFERRED TO LOW-ORDER COUNTER. IN ADDITION, T2 INTERRUPT FLAG IS RESET.
READ – 8 BITS FROM T2 HIGH-ORDER COUNTER TRANSFERRED TO MPU.

Figure 20. T2 Counter Registers

Timer 2 Pulse Counting Mode

In the pulse counting mode, T2 serves primarily to count a predetermined number of negative-going pulses on PB6. This is accomplished by first loading a number into T2. Writing into T2C-H clears the interrupt flag and allows the counter to decrement each time a pulse is applied to PB6. The interrupt flag will be set when T2 reaches zero. At this time the counter will continue to decrement with each pulse on PB6. However, it is necessary to rewrite T2C-H to allow the interrupt flag to set on subsequent down-counting operations. Timing for this mode is shown in Figure 21. The pulse must be low on the leading edge of Φ2.

Shift Register Operation

The Shift Register (SR) performs serial data transfers into and out of the CB2 pin under control of an internal modulo-8 counter. Shift pulses can be applied to the CB1 pin from an external source or, with the proper mode selection, shift pulses generated internally will appear on the CB1 pin for controlling external devices.

The control bits which select the various shift register operating modes are located in the Auxiliary Control Register. Figure 22 illustrates the configuration of the SR data bits and the SR control bits of the ACR.

Figures 23 and 24 illustrate the operation of the various shift register modes.

Interrupt Operation

Controlling interrupts within the SY6522 involves three principal operations. These are flagging the interrupts, enabling interrupts and signaling to the processor that an active interrupt exists within the chip. Interrupt flags are set by interrupting conditions which exist within the chip or on inputs to the chip. These flags normally remain set until the interrupt has been serviced. To determine the source of an interrupt, the microprocessor must examine these flags in order from highest to lowest priority. This is accomplished by reading the flag register into the processor accumulator, shifting this register either right or left and then using conditional branch instructions to detect an active interrupt.

Associated with each interrupt flag is an interrupt enable bit. This can be set or cleared by the processor to enable interrupting the processor from the corresponding interrupt flag. If an interrupt flag is set to a logic 1 by an interrupting condition, and the corresponding interrupt enable bit is set to a 1, the Interrupt Request Output ($\overline{IRQ}$) will go low. $\overline{IRQ}$ is an "open-collector" output which can be "wire-or'ed" with other devices in the system to interrupt the processor.

In the SY6522, all the interrupt flags are contained in one register. In addition, bit 7 of this register will be read as a logic 1 when an interrupt exists within the chip. This allows very convenient polling of several devices within a system to locate the source of an interrupt.

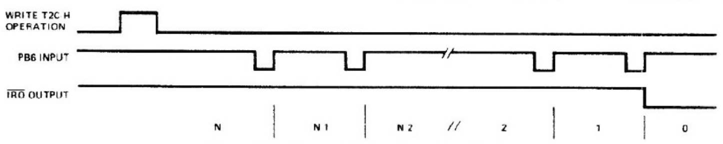
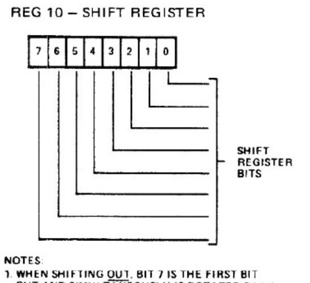
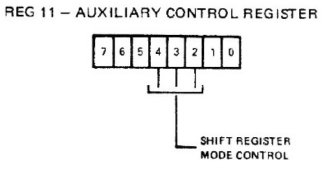


Figure 21. Timer 2 Pulse Counting Mode



NOTES:
1. WHEN SHIFTING OUT, BIT 7 IS THE FIRST BIT OUT AND SIMULTANEOUSLY IS ROTATED BACK INTO BIT 0.
2. WHEN SHIFTING IN, BITS INITIALLY ENTER BIT 0 AND ARE SHIFTED TOWARDS BIT 7.


REG 11 – AUXILIARY CONTROL REGISTER
7 6 5 4 3 2 1 0
SHIFT REGISTER MODE CONTROL

4	3	2	OPERATION
0	0	0	DISABLED
0	0	1	SHIFT IN UNDER CONTROL OF T2
0	1	0	SHIFT IN UNDER CONTROL OF Φ_2
0	1	1	SHIFT IN UNDER CONTROL OF EXT CLK
1	0	0	SHIFT OUT FREE-RUNNING AT T2 RATE
1	0	1	SHIFT OUT UNDER CONTROL OF T2
1	1	0	SHIFT OUT UNDER CONTROL OF Φ_2
1	1	1	SHIFT OUT UNDER CONTROL OF EXT CLK

Figure 22. SR and ACR Control Bits

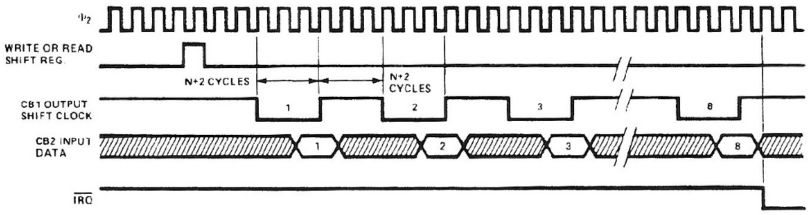
SR Disabled (000)

The 000 mode is used to disable the Shift Register. In this mode the microprocessor can write or read the SR, but the shifting operation is disabled and operation of CB1 and CB2 is controlled by the appropriate bits in the Peripheral Control Register (PCR). In this mode the SR Interrupt Flag is disabled (held to a logic 0).

Shift in Under Control of T2 (001)

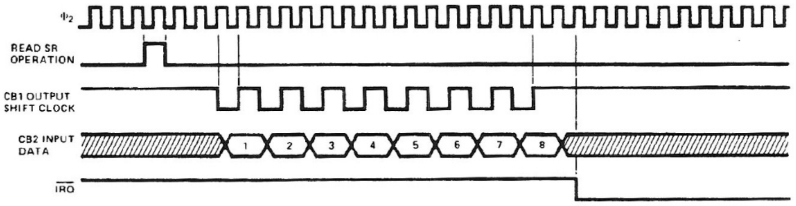
In the 001 mode the shifting rate is controlled by the low order 8 bits of T2. Shift pulses are generated on the CB1 pin to control shifting in external devices. The time between transitions of this output clock is a function of the system clock period and the contents of the low order T2 latch (N).

The shifting operation is triggered by writing or reading the shift register. Data is shifted first into the low order bit of SR and is then shifted into the next higher order bit of the shift register on the negative-going edge of each clock pulse. The input data should change before the positive-going edge of the CB1 clock pulse. This data is shifted into the shift register during the ϕ_2 clock cycle following the positive-going edge of the CB1 clock pulse. After 8 CB1 clock pulses, the shift register interrupt flag will be set and $\overline{\text{IRQ}}$ will go low.



Shift in Under Control of ϕ_2 (010)

In mode 010 the shift rate is a direct function of the system clock frequency. CB1 becomes an output which generates shift pulses for controlling external devices. Timer 2 operates as an independent interval timer and has no effect on SR. The shifting operation is triggered by reading or writing the Shift Register. Data is shifted first into bit 0 and is then shifted into the next higher order bit of the shift register on the trailing edge of each ϕ_2 clock pulse. After 8 clock pulses, the shift register interrupt flag will be set, and the output clock pulses on CB1 will stop.



Shift in Under Control of External CB1 Clock (011)

In mode 011 CB1 becomes an input. This allows an external device to load the shift register at its own pace. The shift register counter will interrupt the processor each time 8 bits have been shifted in. However, the shift register counter does not stop the shifting operation; it acts simply as a pulse counter. Reading or writing the Shift Register resets the Interrupt flag and initializes the SR counter to count another 8 pulses.

Note that the data is shifted during the first system clock cycle following the positive-going edge of the CB1 shift pulse. For this reason, data must be held stable during the first full cycle following CB1 going high.

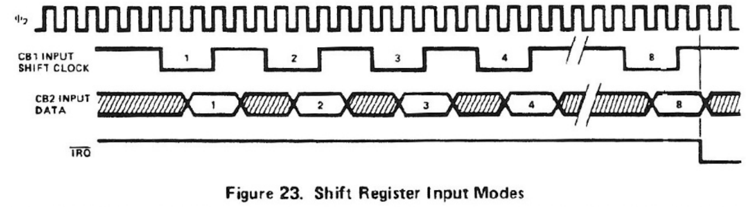
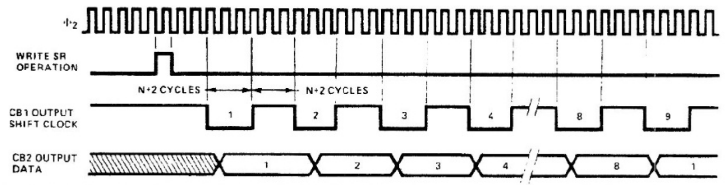


Figure 23. Shift Register Input Modes

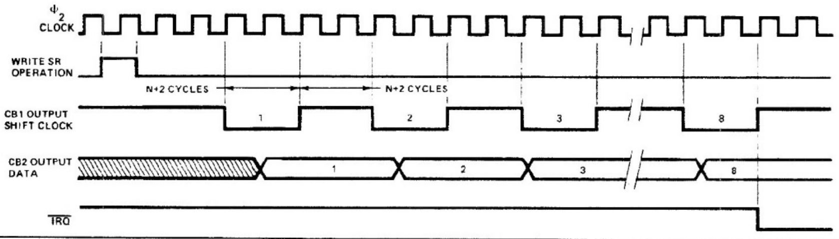
Shift Out Free-Running at T2 Rate (100)

Mode 100 is very similar to mode 101 in which the shifting rate is set by T2. However, in mode 100 the SR Counter does not stop the shifting operation. Since the Shift Register bit 7 (SR7) is recirculated back into bit 0, the 8 bits loaded into the shift register will be clocked onto CB2 repetitively. In this mode the shift register counter is disabled.



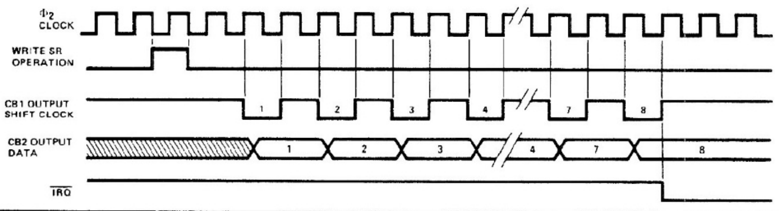
Shift Out Under Control of T2 (101)

In mode 101 the shift rate is controlled by T2 (as in the previous mode). However, with each read or write of the shift register the SR Counter is reset and 8 bits are shifted onto CB2. At the same time, 8 shift pulses are generated on CB1 to control shifting in external devices. After the 8 shift pulses, the shifting is disabled, the SR Interrupt Flag is set and CB2 remains at the last data level.



Shift Out Under Control of ϕ_2 (110)

In mode 110, the shift rate is controlled by the ϕ_2 system clock.



Shift Out Under Control of External CB1 Clock (111)

In mode 111 shifting is controlled by pulses applied to the CB1 pin by an external device. The SR counter sets the SR Interrupt flag each time it counts 8 pulses but it does not disable the shifting function. Each time the microprocessor writes or reads the shift register, the SR Interrupt flag is reset and the SR counter is initialized to begin counting the next 8 shift pulses on pin CB1. After 8 shift pulses, the interrupt flag is set. The microprocessor can then load the shift register with the next byte of data.

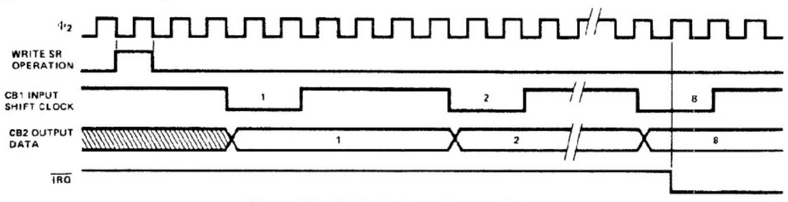


Figure 24. Shift Register Output Modes

The Interrupt Flag Register (IFR) and Interrupt Enable Register (IER) are depicted in Figures 25 and 26, respectively.

The IFR may be read directly by the processor. In addition, individual flag bits may be cleared by writing a "1" into the appropriate bit of the IFR. When the proper chip select and register signals are applied to the chip, the contents of this register are placed on the data bus. Bit 7 indicates the status of the IRQ output. This bit corresponds to the logic function: IRQ = IFR6 x IER6 + IFR5 x IER5 + IFR4 x IER4 + IFR3 x IER3 + IFR2 x IER2 + IFR1 x IER1 + IFR0 x IER0. Note: X = logic AND, + = Logic OR.

The IFR bit 7 is not a flag. Therefore, this bit is not directly cleared by writing a logic 1 into it. It can only be cleared by clearing all the flags in the register or by disabling all the active interrupts as discussed in the next section.

REG 13 – INTERRUPT FLAG REGISTER

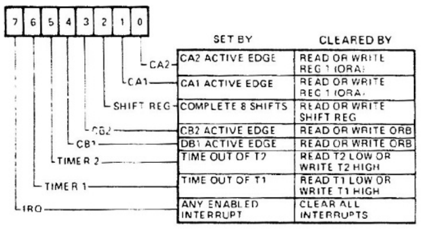


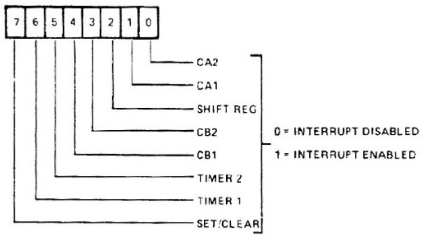
Figure 25. Interrupt Flag Register (IFR)

For each interrupt flag in IFR, there is a corresponding bit in the Interrupt Enable Register. The system processor can set or clear selected bits in this register to facilitate controlling individual interrupts without affecting others. This is accomplished by writing to address 1110 (IER address). If bit 7 of the data placed on the system data bus during this write operation is a 0, each 1 in bits 6 through 0 clears the corresponding bit in the Interrupt Enable Register. For each zero in bits 6 through 0, the corresponding bit is unaffected.

Setting selected bits in the Interrupt Enable Register is accomplished by writing to the same address with bit 7 in the data word set to a logic 1. In this case, each 1 in bits 6 through 0 will set the corresponding bit. For each zero, the corresponding bit will be unaffected. This individual control of the setting and clearing operations allows very convenient control of the interrupts during system operation.

In addition to setting and clearing IER bits, the processor can read the contents of this register by placing the proper address on the register select and chip select inputs with the R/W line high. Bit 7 will be read as a logic 0.

REG 14 – INTERRUPT ENABLE REGISTER



NOTES
1. IF BIT 7 IS A "0", THEN EACH "1" IN BITS 0 - 6 DISABLES THE CORRESPONDING INTERRUPT
2. IF BIT 7 IS A "1", THEN EACH "1" IN BITS 0 - 6 ENABLES THE CORRESPONDING INTERRUPT
3. IF A READ OF THIS REGISTER IS DONE, BIT 7 WILL BE "0" AND ALL OTHER BITS WILL REFLECT THEIR ENABLE/DISABLE STATE

Figure 26. Interrupt Enable Register (IER)

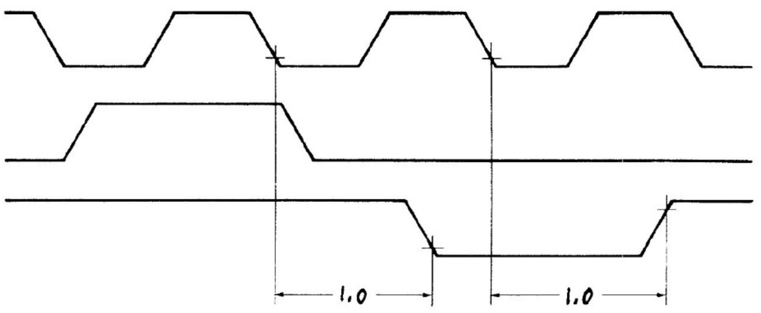


Figure 5a. CA2 Timing for Read Handshake, Pulse Mode

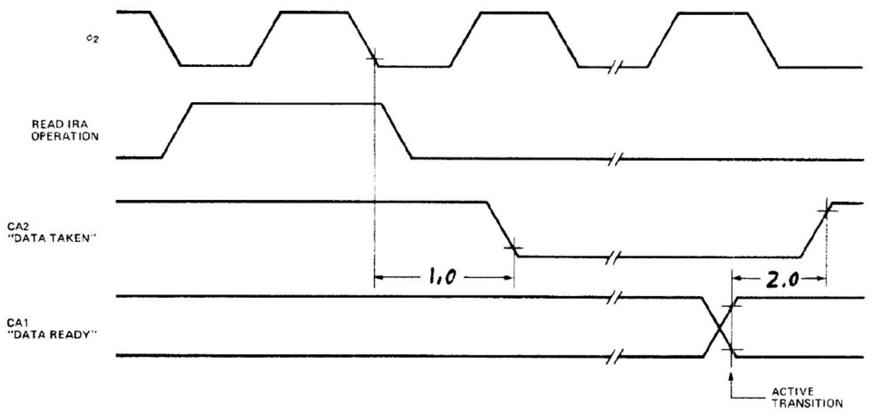


Figure 5b. CA2 Timing for Read Handshake, Handshake Mode

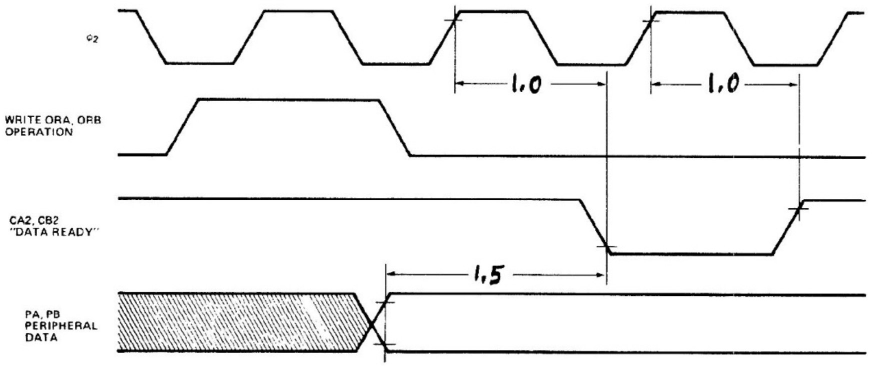


Figure 5c. CA2, CB2 Timing for Write Handshake, Pulse Mode

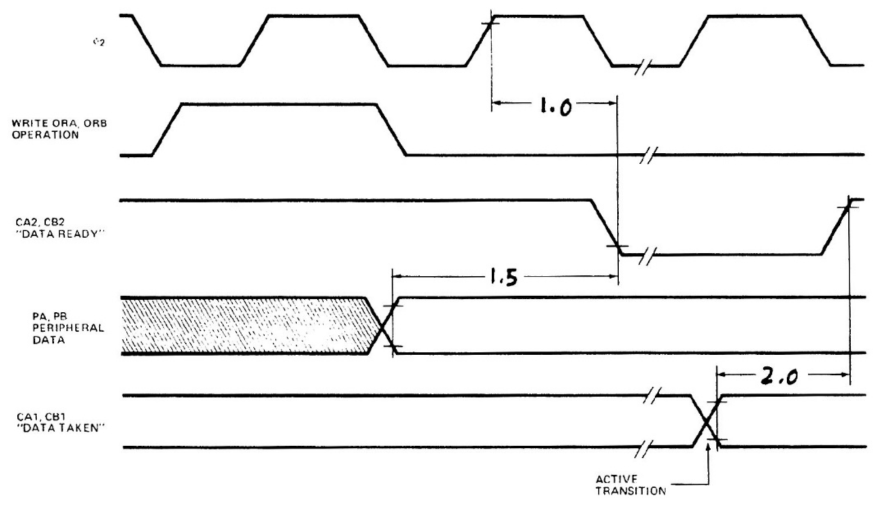


Figure 5d. CA2, CB2 Timing for Write Handshake, Handshake Mode

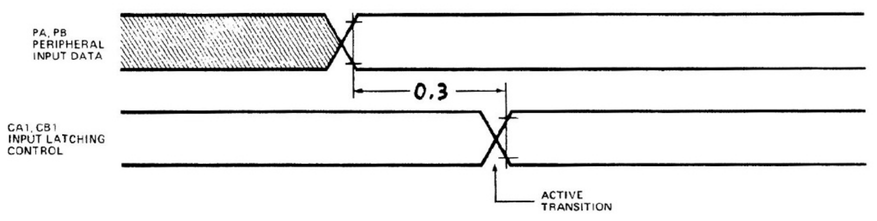


Figure 5e. Peripheral Data Input Latching Timing

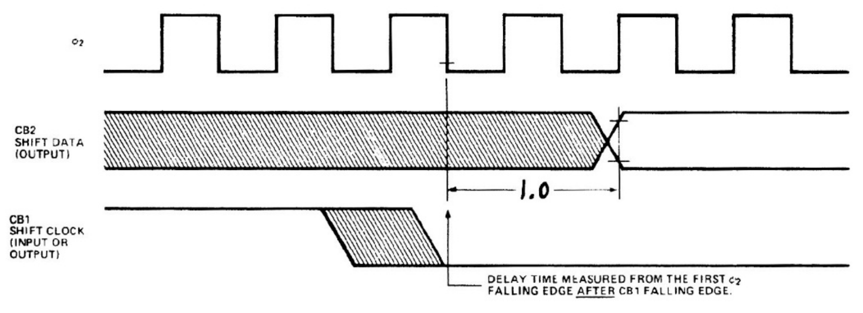

φ2
CB2 SHIFT DATA (OUTPUT)
CB1 SHIFT CLOCK (INPUT OR OUTPUT)
1.0
DELAY TIME MEASURED FROM THE FIRST φ2 FALLING EDGE AFTER CB1 FALLING EDGE.

Figure 5f. Timing for Shift Out with Internal or External Shift Clocking

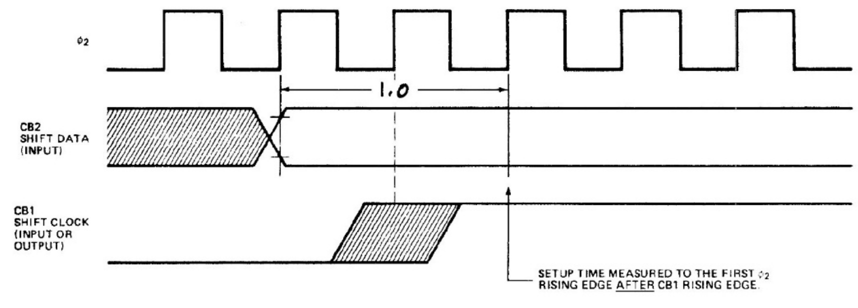

φ2
1.0
CB2
SHIFT DATA
(INPUT)
CB1
SHIFT CLOCK
(INPUT OR
OUTPUT)
SETUP TIME MEASURED TO THE FIRST φ2
RISING EDGE AFTER CB1 RISING EDGE.

Figure 5g. Timing for Shift In with Internal or External Shift Clocking

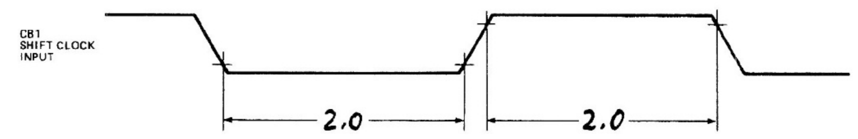


Figure 5h. External Shift Clock Timing

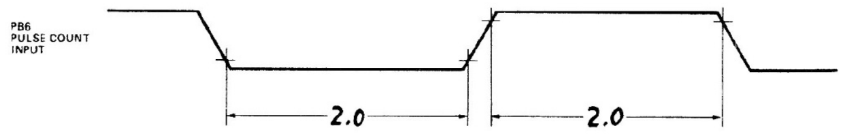


Figure 5i. Pulse Count Input Timing

11. PRINCIPLES OF OPERATION

The K-1032 Banker Memory is basically a 32K read/write memory board with PROM, I/O, TTL registers, and PROM programmer added. In operation it is synchronized to the 6502 bus cycle time which is expected to be 1.0MHz. Although there are a lot of parts on the board, its operation is straightforward. Half of the random logic is devoted to a very flexible addressing and bank switching capability. In addition, all address decoders can look at 18 address bits thus allowing the Banker Memory to be used in the MTU 18 bit address bus systems of the future. In addition to the digital logic, there is a small amount of analog circuitry in the power supply, PROM programmer, and timing generator. There should be enough information in this section to allow an experienced technician to understand the board's operation and make repairs and adjustments.

Page 57 shows a block diagram of the K-1032 Banker Memory board. Data are exchanged among the various elements of the board via a central bidirectional data bus. Because of the two-phase bus cycle of the 6502, this internal bus can be devoted to internal operations such as memory refreshing during Phase 1 and then turned over to the 6502 (if the board is addressed) during Phase 2. The memory IC's are therefore operated at a 2MHz cycle rate which these modern devices handle quite well. The address bus is unidirectional being simply a buffered version of the 6502 address bus. Note that there are three completely independent address decoders, one for the RAM, one for the ROM, and one for I/O. Each decoder looks at all 18 address bits which means that the Banker Memory is really 3 boards in one. The Enable Register is used to turn off any combination of the 4 ROM sockets or any combination of 4 blocks of RAM. I/O is always active however unless disabled by a jumper change (see section 5.1.7).

11.1 BUS BUFFERS

The lower 16 bits of the address bus are buffered by two octal buffers, U1 and U2, which are at the left edge of page 1 of the schematics. Since the enable inputs of the buffers are permantly tied to ground, they act simply as non-inverting buffers. Since address bit 14 and 15 must also be available in complemented form for the address recognizers, they are also passed through inverters after being buffered. The two "extra" address bits, A16 and A17 are buffered by 74LS04 inverters and then inverted again with another pair of inverters so that they also are available in true and complement form.

The data bus is buffered by an octal transceiver, U10. If something on the board is addressed and the 6502 is performing a Read cycle (6502 $R/\overline{W}$ high), the transmitters are activated by bringing both pins 1 and 19 low during PHASE 2 to drive the 6502 data bus with read data. If something on the board is addressed and the 6502 is performing a Write cycle (6502 $R/\overline{W}$ low), the receivers are enabled instead by forcing pin 1 high while pin 19 goes low during PHASE 2 to drive the internal K-1032 bus (LOC DB0 - LOC DB7) with write data. Neither buffer is activated during PHASE 1 in order to reduce noise generation. Resistors are also placed in series with the data bus connections to prevent excessive noise from being generated when the powerful bus drivers in the 74LS245 (U10) switch on.

11.2 RAM ADDRESS RECOGNIZER

The RAM address recognizer is in the top center and right portion of page 1 of the schematics. Since decoding is done in several levels, each level will be described seperately. The first level is the Bank Select logic which looks at the 17th and 18th address bits. Jumpers select which of the four 64K banks the RAM will be in and cause U41-11 to go low when that bank address appears on the address bus. By omitting the jumpers, the extra address bits are ignored and U41-13 and U41-12 are pulled high which causes U41-11 to be held low.

The next stage is to decode the selected 64K block of addresses into 16 4K blocks. This is accomplished by a 1-of-16 decoder built from two 74LS138 1-of-8 decoders (U22 and U23) interconnected. Note that the decoder is disabled through pin 5 whenever the I/O ADRD signal is true thus allowing I/O addresses to overlap RAM addresses without interference (I/O wins). Each of the 16 outputs then represents 4K of addresses within the chosen bank. These 16 outputs are then terminated on one side of the RAM address selection jumper array.

On the other side of the RAM address selection jumper array are 8 terminals, each representing 4K of actual RAM. To associate that 4K of RAM with 4K of addresses, a jumper is connected from one of the decoder outputs to one of these 8 encoder inputs. The 8 inputs are first ORed together in pairs (U35-6, U35-3, U36-6, and U36-3) making up four 8K blocks of RAM. The encoder assumes that one of the 4K halves of the block goes to an even-numbered decoder output while the other 4K half goes to an odd-numbered decoder output.

After being ORed together pairwise, the 4 resulting block select signals are ANDed with 4 bits from the Enable Register (U20). If the corresponding Enable Register bit is a one, the block select signal is passed to the final address encoder, otherwise it is blocked. The final address encoder encodes the 4 gated block select lines into the two most significant RAM address bits and also generates the RAM ADRD signal if any of them are active.

11.3 ROM ADDRESS RECOGNIZER

In many ways the ROM Address Recognizer, which is at the bottom of page 1 of the schematics, is similar to the RAM Address Recognizer. Assuming that 2K byte PROM's are to be used (2716 type), any true and complement combination of the most significant 4 address bits can be selected with jumpers to define a 16K block of addresses on a 16K boundary. U43-12 looks at 3 of the bits and an additional enable input on U24 looks at the fourth. When an address within the 16K block is on the address bus, U24 is enabled and acts as a decoder to further break down the 16K block into 8 2K blocks. As with the RAM, the I/O ADRD signal will inhibit the decoder. Each of the decoder's 8 outputs are terminated on one side of the ROM address selection jumper array.

On the other side of the ROM address selection jumper array are 4 terminals, each one representing a ROM socket. As with the RAM, to associate a PROM with a 2K block, a jumper is connected from a decoder output to a PROM enable input. If 4K PROM's or ROM's are to be used, the jumpers in U18 and U19 are rearranged so that a 32K major block is defined (instead of 16K), and the decoder decodes eight 4K blocks (instead of 2K).

The 4 outputs of the ROM address selection jumper array are inverted by U37 and then ANDed with the 4 bits remaining in the Enable Register. Finally they go to the individual chip select inputs of each PROM and are also ORed into the BOARD ADRD gate, U21.

11.4 I/O ADDRESS RECOGNIZER

The I/O address recognizer is situated between the RAM and ROM addressing circuitry in the middle of page 1. The first step in I/O address decoding is detect when the 18 bit address on the address bus is within the 64 byte block of addresses assigned to I/O. This is accomplished by comparing the most significant 12 address bus lines with a 12 bit value determined by the jumpers in U16 and U17. When the two 12 bit quantities are equal, decoder U26 is enabled. The comparison is accomplished with 12 open collector exclusive-nor gates. When the two inputs to each gate are equal, the output is allowed to float; if they are different, the output is driven to ground. If all 12 outputs float, which can only occur when the two 12 bit quantities match, R10 pulls them all high giving I/O ADRD.

Once an I/O address is recognized, U26 acts as a 1-of-4 decoder to break up the 64 byte I/O block into four 16 byte blocks. Two of these go on to enable the two 6522 I/O chips on the board. The third is used to select the Enable Register and the fourth is unused. The 3 used outputs of the decoder are also factored into the BOARD ADRD logic so in actuality only 48 of the 64 I/O addresses are actually used by the Banker Memory; the other 16 could be used by some other board without problems.

11.5 ENABLE REGISTER

The Enable Register consists of two 4-bit latches with multiplexors on their inputs. Normally the inputs connected to the internal data bus are selected so the register acts as a simple 8-bit write-only register. Normal writing into the register is controlled by U43-6 which detects the coincidence of ENAB ADRD from the I/O address decoder, PHASE 2, and $\overline{\text{LOC R/W}}$. During a system reset, the other set of inputs to the latches is selected. When reset is released, the settings of the Default RAM Enable jumpers and Default ROM Enable jumpers are written into the latches. R20 and C15 delay the input selection signal to the latches long enough for them to be clocked and capture the default enable data.

11.6 SPECIAL CIRCUITRY FOR KIM-1 SYSTEMS

Near the upper left corner of page 1 of the schematics is the circuitry that generates the Vector Fetch and Decode Enable signals needed by KIM-1 systems when their memory is expanded. For use in other systems these signals may be ignored The Vector Fetch bus line is pulled down to a logic zero whenever the upper 8 bits of the normal 16 bit address bus are all ones. This typically occurrs only when one of the 6502's three vectors in locations FFFA-FFFF are accessed. D1, which is a low-forward-drop germanium diode, makes U9 appear as if it was an open-collector gate. The Decode Enable bus line must be a logic zero whenever addresses between 0000 and 1FFF are on the address bus. This is accomplished with U43-8. Note that neither circuit takes the 17th and 18th address bits into account because KIM-1 based systems are highly unlikely to have been modified for 18 address bits.

11.7 TIMING GENERATOR

The timing generator is in the top half of schematic page 2. All timing is generated by counting down an 8mHz clock and then decoding the count in various ways to insure that accurate timing is always generated. The timing diagram on page 38 may be consulted for detailed timing relationships.

The timing generator is synchronized to the trailing edge of 6502 PHASE 2 by means of a phase-locked loop which is in the very center of the drawing. U77-6 is a simple Schmidt trigger oscillator with a nominal frequency (which can be adjusted with R29) of 8MHz. By connecting R27 to the R-C node, the frequency can be controlled by the application of a DC voltage to its free end. Although the linear control range is only 20% or so, it is ample for locking onto the crystal controlled system clock. The 8MHz output at U77-6 is normally asymetrical (35% high, 65% low) and goes to a number of places including the synchronous 4 bit counter, U33. U45-8 decodes the counter status to produce a low-going signal with duty cycle of 25% at a frequency of 1MHz. This is the comparison signal for the phase comparator. 6502 PHASE 2 is the reference signal and U48 is used as a phase comparator. The output of the phase comparator simply floats for 3/4 of each 1uS cycle since it is actually a tri-state gate. When it is enabled by U45-8, the output first goes high (since when clocked 6502 PHASE 2 would be high), then goes low when 6502 PHASE 2 terminates, and then floats when disabled by U45-8. The ratio of high-to-low time of this signal is averaged by lowpass filter R30 and C34 which is then the control voltage to the 8MHz oscillator.

Normally the trailing edge of 6502 PHASE 2 occurs midway in the "window" defined by the output of U45-8. Locking action can be understood by considering what would happen if 6502 PHASE 2 terminated later in the window, i.e., slowed down slightly. The output of the phase comparator would then be high for a longer time and low for a shorter time thus raising the averaged control voltage. Since a higher control voltage slows down the oscillator, the window frequency would decrease to match the input. The converse would occur if 6502 PHASE 2 should speed up. R29 can be adjusted to center the trailing edge of 6502 PHASE 2 in the window for accurate timing. This circuit has been found to be highly reliable and is in fact used on all MTU bus interface products to provide a phase locked high frequency clock for timing generation.

The most critical timing signals needed are those that operate the 16K dynamic RAM chips. U49-7 and both halves of U50 are set up as a 3 bit shift register delay line to generate the $\overline{\text{RAS}}$, address swap, and $\overline{\text{CAS}}$ sequence needed by the memories. A memory cycle is started if MEM CYC ENAB from $\overline{\text{U45-3}}$ is true when bit 0 of the 4 bit timing counter is high and bit 1 is low and a negative edge of 8 MHZ CLOCK is seen. U50-9 then goes high generating RAS which starts the timing chain for the memory cycle. The next positive edge of 8 MHZ CLOCK flips U50-6 which through the memory address multiplexor, switches from row address to column address. Finally, the next negative edge of 8 MHZ CLOCK after the address is switched generates $\overline{\text{CAS EVEN}}$ and $\overline{\text{CAS ODD}}$ which latches the column address in the RAM chips and activates the RAM I/O circuitry. At the end of the memory cycle, U50-9 is jammed back low by a negative edge of bit 1 of the timing counter (U33-14) through the network formed from C36, R32, and R33. Immediately afterward U50-6 is forced back high from U45-11. Finally, on the next negative 8 MHZ CLOCK edge U49-7 returns to its inactive state to complete the cycle.

MEM CYC ENAB, which is responsible for starting memory cycles, is generated whenever RAM ADRD is true during PHASE 2 or when the most significant bit of U33 is a ONE during PHASE 1 (note that U33-12 is very nearly the inverse of the system's PHASE 2 and is actually used in the timing logic). The former case occurs when a memory cycle must be performed for the host system and the latter occurs when a refresh cycle for the dynamic RAM's is needed. U40-6 and the gates preceeding it insure that write enable for the RAM chips is only generated when RAM ADRD is true, PHASE 2 (actually U44-6) is true, and $\overline{\text{LOC R/W}}$ is true. The RAM's are operated in an early write mode which allows their data-in and data-out pins to be tied directly to the internal data bus.

11.8 RAM ADDRESSING AND DRIVING

The RAM array itself is in the bottom half of page 2 of the schematics. A specialized chip is used to multiplex and drive address data to the RAM's. This chip is U5 and performs two distinct functions. When pin 2 is low, the chip acts as an address multiplexor and connects 7 of its inputs to the outputs when pin 3 is low and switches to the other 7 when pin 3 is high. When Pin 2 is high, the chip acts as a refresh address counter. In refresh mode the address inputs are ignored and an internal 7 bit counter is instead connected to the 7 outputs. This internal counter is incremented when a negative edge is applied to pin 1. The outputs of U5 are powerful MOS drivers and are damped by series resistors to prevent excessive noise generation and address line overshoot.

Logic at the upper right corner determines which row of RAM chips has been selected and generates the $\overline{\text{RAS}}$ signal for just that row. Refreshing is also done one row at a time to cut down on noise. During normal cycles, RAM A14 determines which row is to be driven. During refresh cycles, U49, which flips after each refresh cycle, determines the row. Although the CAS clock of both rows is driven on each cycle, the row that did not receive a RAS clock will simply ignore it.

11.9 PROMS AND ROMS

The left portion of page 3 of the schematics shows the ROM array circuitry. Essentially the 4 ROM sockets are wired in parallel except for the chip select pin. 2K (2716) and 4K (2732) EPROM's actually have 2 inputs that control the outputs. Chip Enable actually turns a substantial quantity of internal circuitry off when it is high thus cutting power consumption by 2/3. Output Enable just turns the outputs on and off without affecting the internal circuitry or power consumption. For a low power PROM array then, the address decoder must be connected to Chip Enable and Phase 2 must be connected to Output Enable.

The K-1032 is shipped set up for 2K EPROM's. When shifting to 4K EPROM's or ROM's, the address decoding jumpers must be reconfigured as described in sections 4.2 and 11.3. It is also necessary to move the jumper bridging U27-1 to U27-8 so that it bridges U27-2 to U27-7. This then sends LOC AB11 to the ROM array. When switching from PROM to ROM (ROM's have no transparent lids), three jumpers must be moved on each socket to be converted. This is necessary because of substantial pinout differences between PROM's and ROM's. Note that the ROM's used must have been masked so that pin 20 is an active-low enable and pin 21 is an active high enable. Only 4K masked ROM's (2332 type) are recommended.

11.10 PARALLEL I/O CHIPS

The right portion of schematic page 3 has the two parallel I/O chips. Like the ROM's, they are wired in parallel except for one of the chip selects. The 20 I/O signals from each 6522 are simply routed to 40 pins on the application connector of the K-1032. Many of the I/O signals of the topmost I/O chip (U7) also drive the PROM programmer which is described below.

11.11 PROM PROGRAMMER

The PROM programmer circuitry is in the upper right corner of page 3 of the schematics. U52 is the actual programming socket and should not have an PROM plugged into it unless actually programming it. The address for the PROM is supplied by a 12 bit CMOS counter which greatly reduces the number of connections between the programmer and the output port. This counter may be incremented and reset to zero under program control through U7 which is one of the parallel port chips. The programming voltage is developed from the 12 volt 500KHz square wave produced by the power supply with a voltage multiplier consisting of D6-D9 and C22, C23, and C25 and is stored in C26. This high programming voltage may be turned on and off under program control through the circuit made up of Q3, Q4, and Q5 and associated components. This circuit also regulates the +30 volts provided by the voltage multiplier to the 24 volts required by the PROM. Switching of the programming voltage is necessary because the voltage multiplier will not supply enough current for continuous programming. To maintain programming voltage regulation, the programming voltage must be switched on for a maximum of 50 milliseconds and then given a recovery period of 100 milliseconds for each PROM location programmed. D11, which is a light emitting diode, glows dimly when the programming voltage is turned.

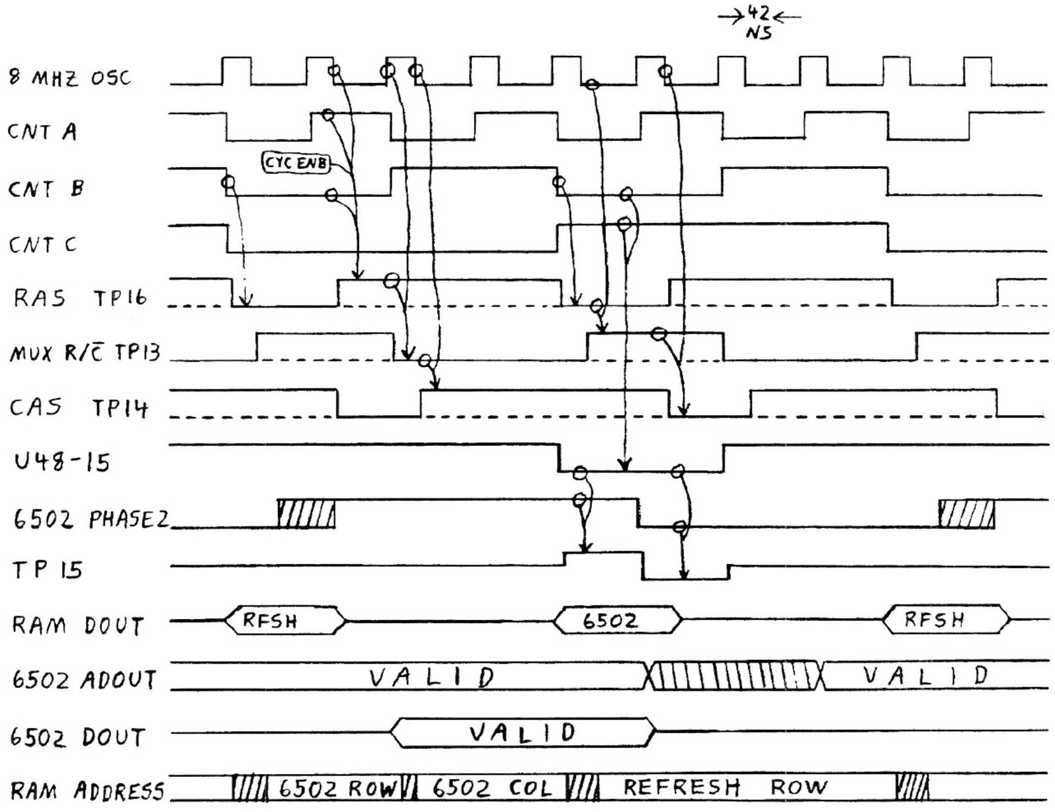
When shipped, the K-1032 is set up for programming 2K (2716 single 5 volt) PROM's. To program 4K PROM's, it is necessary to change the jumpers in U54. PROM's can be damaged if these jumpers are set inappropriately.

11.12 POWER SUPPLY

The on-board power supply is at the top left corner of schematic page 1. Positive 5 volts for the board logic is derived from the unregulated +8 input by VR2 which is a 1 amp 5 volt IC regulator. C17 prevents oscillation and numerous .05uF bypass capacitors maintain regulation during transient current drains.

Positive 12 volts for the memory IC's is provided by VR1. C12 suppresses oscillation and provides the additional filtering needed by MTU K-series power supplies on their 16 volt unregulated outputs. C13 absorbs the large current transients typical of dynamic memories. Negative 5 volts for the memory chips is provided by a charge pump circuit in the lower left corner of page 2 consisting of C21, D4, D5, and C24. The circuit is driven from a 12 volt amplitude .5MHz square wave provided by Q1, Q2, and associated circuitry. Shunt regulator D2 limits the negative voltage to -5 volts and also prevents the -5 bus from becoming substantially positive during component failure and thus prevents possible damage to the memory chips.

12. TIMING DIAGRAM



13. TROUBLESHOOTING

Factory assembled K-1032 boards have been carefully checked out and burned-in prior to shipment. However because of the scores of jumper options available, some of which involve PC trace cutting, it is impossible to test 100% of the board functions. Also since the customer supplies his own ROM's or PROM's, we have no control over their quality.

In the event of trouble first give the board a thorough visual inspection. Unclipped component leads may have bent over and shorted during shipment. A poor solder connection might have opened during shipping vibration. Although rare, IC's in sockets could have a lead bent under thus making intermittant contact with the socket pin. Check that all of the standard and user supplied jumpers are in place and not shorting against each other. Reread the sections on address jumpering and make sure that the resource you are trying to address is enabled (see section 7). It goes without saying that all connections between the processor board and the K-1032 should be checked. In particular a heavy ground lead (braid, large PC foil area or #18 hookup wire) should be used and the bus wire lengths should not exceed 4 inches unless run as twisted pairs with ground or alternating with ground in a ribbon cable. The best method of connection to KIM, SYM, and AIM processors is to use a K-1005 series motherboard/card file.

Following this the first area to check is the power supply. The incoming voltages should read a minimum of +8 and +15 volts with a voltmeter. If an oscilloscope is available, the negative peak of the ripple waveform must not drop below +7 and +14 volts. If regulated +5 input is used, make sure that the voltage measured accross an IC on the K-1032 is between +4.9 and +5.1. If it is outside that range, the timing generator may have to be adjusted (see section 15). Next check the output of the positive regulators. If the heatsink is blazing hot and one of the regulated voltages is low or zero, suspect a short, possibly through a user supplied PROM. Check the -5 volt output (TP7). If the PROM programmer does not function, check the +35 volt programming power supply (TP12). When not programming it should be at least +32 volts.

Addressing problems can be tracked down by noting which addresses the board does respond to. With most processors and monitors a non-existant address will read back the page number of the non-existant address (this is due to the operation sequence of indirect addressing). If it responds to too many addresses (for example, I/O can be activated at two different addresses), then either an address selection jumper is missing or a PC trace that was not needed by the factory test address is open. The most common cause of no response to addresses is likely to be improper jumpering or programming of the Enable Register. Remember that if all of the K-1032 resources are to be available simultaneously after power-up or reset that there should be no jumpers in the U32 socket.

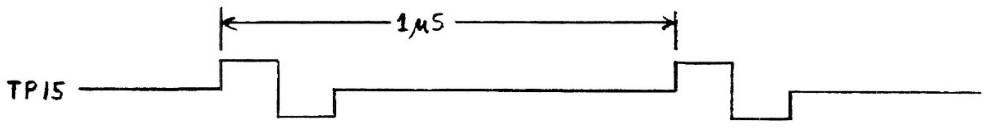
If the customer cannot find the problem or is unable to repair it, return the board to the factory for repair with the customer's jumpers in place. Also include a copy of the desired address settings.

14. ADJUSTMENTS

There is one adjustment on the K-1032 board. This adjustment affects the timing generator so that it can synchronize with the bus cycle of the host processor. All factory assembled boards have had this adjustment made at the factory and sealed. However if the user bypasses the on-board +5 volt regulator or replaces U47, or the system clock frequency is not exactly 1.0MHz, then readjustment may be necessary. Use the following procedure to check this adjustment:

1. With the board plugged into the system (the CPU board must be present) and powered on, look at the signal on test point 15 with an oscilloscope.

2. The waveform should resemble the drawing below with the positive and negative portions of the waveform equal in width and have a repetition frequency of 1.0MHz. Check the oscilloscope callibration to be sure the repetition frequency is not .5MHz or 2.0MHz.



3. If the waveform has an irregular shape or period, slowly adjust the potentiometer (R2) until the waveform stabilizes with a 1uS period. Further rotate the pot until the positive and negative portions of the cycle are equal.

4. Put a spot of nail polish or a piece of tape over the pot to prevent tampering.

5. Run the memory diagnostic on page 42 to verify proper board operation.

15. SPECIFICATIONS

RAM Capacity -	32K in 4 8K blocks using 4116 type 16K dynamic RAM chips.
PROM Capacity -	16K using 2732 type PROMS or 2332 ROM's, 8K using 2716 PROM's.
Parallel I/O -	Four 8-bit ports and 8 handshaking lines, each bit of each port may be programmed as an input or an output. Interrupt available for each group of 8 bits. 6522 VIA chips are used.
PROM Programmer -	Can program 5 volt 2716 PROMS or with a jumper change, 2732.
Access Time - -	550NS maximum as required by KIM-1, SYM-1, or AIM-65 when using 450NS PROM's.
Power Requirement -	+7.5 volts unregulated .65 amp, +16 volts unregulated .17 amp. maximum with all PROM's installed and programming. +25 required by the PROM programmer is generated on-board.
Output Power -	+5 volts at 100MA and +12 volts at 50MA regulated power provided for external interface circuitry.
Addressing - -	Three independent address decoders, up to 18 address lines recognized. RAM is in 4 individual 8K blocks each of which can be on any 4K boundary. ROM is 4 individual 2K or 4K blocks that may be anywhere in a 16K or 32K segment. I/O is 48 addresses starting at any address divisible by 64.
Bank Switching -	Each of the 4 RAM blocks and 4 ROM blocks has a bit in an Enable Register that can be turned on and off. Jumpers set the default Enable Register contents on power-up or reset.
Buffering - - -	Buffering for both address and data busses is provided. Maximum bus load is 1 LS TTL gate input and one LS TTL tri-state output.
Physical Size -	7.5" X 11" exclusive of edge fingers. Two sets of 44 edge fingers compatible with the KIM-1, SYM-1, or AIM-65 processors.

16. PIN CONNECTIONS

EXPANSION CONNECTOR				APPLICATION CONNECTOR			
PIN	SIGNAL	PIN	SIGNAL	PIN	SIGNAL	PIN	SIGNAL
E-1	N.C.	E-A	ADDR BUS 0	A-1	GROUND	A-A	+12 REG OUT
*E-2	ADDR BUS 16	E-B	ADDR BUS 1	A-2	N.C.	A-B	+5 REG OUT
*E-3	ADDR BUS 17	E-C	ADDR BUS 2	A-3	VIA 2 CA1	A-C	VIA 1 CA1
E-4	$\overline{\text{INT. REQ.}}$	E-D	ADDR BUS 3	A-4	VIA 2 CA2	A-D	VIA 1 CA2
E-5	N.C.	E-E	ADDR BUS 4	A-5	VIA 1 PA0	A-E	VIA 2 PA0
E-6	N.C.	E-F	ADDR BUS 5	A-6	VIA 1 PA1	A-F	VIA 2 PA1
E-7	$\overline{\text{RESET}}$	E-H	ADDR BUS 6	A-7	VIA 1 PA2	A-H	VIA 2 PA2
E-8	DATA BUS 7	E-J	ADDR BUS 7	A-8	VIA 1 PA3	A-J	VIA 2 PA3
E-9	DATA BUS 6	E-K	ADDR BUS 8	A-9	VIA 1 PA4	A-K	VIA 2 PA4
E-10	DATA BUS 5	E-L	ADDR BUS 9	A-10	VIA 1 PA5	A-L	VIA 2 PA5
E-11	DATA BUS 4	E-M	ADDR BUS 10	A-11	VIA 1 PA6	A-M	VIA 2 PA6
E-12	DATA BUS 3	E-N	ADDR BUS 11	A-12	VIA 1 PA7	A-N	VIA 2 PA7
E-13	DATA BUS 2	E-P	ADDR BUS 12	A-13	VIA 1 PB0	A-P	VIA 2 PB0
E-14	DATA BUS 1	E-R	ADDR BUS 13	A-14	VIA 1 PB1	A-R	VIA 2 PB1
E-15	DATA BUS 0	E-S	ADDR BUS 14	A-15	VIA 1 PB2	A-S	VIA 2 PB2
E-16	N.C.	E-T	ADDR BUS 15	A-16	VIA 1 PB3	A-T	VIA 2 PB3
E-17	N.C.	E-U	N.C.	A-17	VIA 1 PB4	A-U	VIA 2 PB4
*E-18	+7.5 VOLTS IN	E-V	READ/$\overline{\text{WRITE}}$	A-18	VIA 1 PB5	A-V	VIA 2 PB5
#E-19	$\overline{\text{VECTOR FETCH}}$	E-W	N.C.	A-19	VIA 1 PB6	A-W	VIA 2 PB6
#E-20	$\overline{\text{DECODE ENABLE}}$	*E-X	+16 VOLTS IN	A-20	VIA 1 PB7	A-X	VIA 2 PB7
E-21	N.C.	E-Y	$\overline{\text{PHASE 2}}$	A-21	VIA 1 CB1	A-Y	VIA 2 CB1
E-22	GROUND	E-Z	N.C.	A-22	VIA 1 CB2	A-Z	VIA 2 CB2

* Not to be connected to corresponding E pin of KIM, SYM, or AIM computers.
Connect VECTOR FETCH to A-J and DECODE ENABLE to A-K on the KIM-1.

18. DIAGNOSTIC PROGRAM LISTING

The following is a listing of a memory and I/O diagnostic program for the BANKER MEMORY. It occupies locations 0200-036C which should be suitable for KIM, SYM, and AIM computers. For general use instructions, see section 2.9, for details see the lsiting below.

```
                          .TITLE K32TS,'K-1032 CUSTOMER DIAGNOSTIC PROGRAM'
                ;         TEST AND EXERCISE PROGRAM FOR THE K-1032 RAM/ROM/IO BOARD.
                ;         THIS IS A SIMPLIFIED TEST THAT DOES NOT REQUIRE A LOOP-AROUND
                ;         PLUG ON THE APPLICATION CONNECTOR TO PERFORM.  IT THOROUGHLY
                ;         TESTS THE RAM USING RANDOM DATA AND IT TESTS THE TWO I/O CHIPS
                ;         FOR GROSS FUNCTION.

                ;         THE OVERALL TEST IS BROKEN DOWN INTO A NUMBER OF TESTS.  IF AN
                ;         ERROR IS DETECTED, THE TEST NUMBER WILL BE STORED INTO LOCATION
                ;         0000.  AN ERROR CODE OF 00 INDICATES THAT ALL TESTS COMPLETED
                ;         SATISFACTORILY.  ADDITIONAL ERROR INFORMATION IS STORED IN
                ;         MEMORY DEPENDING ON THE TEST.  SEE THE INDIVIDUAL TEST ERROR
                ;         LOG ROUTINE FOR DETAILS.

                ;         BEFORE EXECUTING THE TEST PROGRAM, THE BOARD CONFIGURATION WILL
                ;         HAVE TO BE SPECIFIED.  LOCATIONS 0010 - 0017 MUST CONTAIN THE
                ;         FIRST PAGE NUMBER OF EACH OF THE EIGHT 4K RAM SEGMENTS.  IF A
                ;         SEGMENT IS NOT ACTIVE, SPECIFY FF FOR THE PAGE NUMBER.
                ;         THE SEGMENT PAGE ADDRESSES MUST BE STORED IN ASCENDING CONTROL
                ;         REGISTER BIT ORDER.  THUS THE 8K BLOCK MADE OF 2 4K SEGMENTS
                ;         WHOSE PAGE ADDRESSES ARE IN 0010 AND 0011 MUST BE ASSOCIATED
                ;         WITH CONTROL REGISTER BIT 0.  THE I/O BASE ADDRESS MUST BE
                ;         STORED IN LOCATIONS 0018 (LOW) AND 0019.  IF THE 6522 I/O CHIPS
                ;         ARE INSTALLED, STORE 00 IN LOCATION 001A; IF THE I/O CHIPS ARE
                ;         NOT INSTALLED, STORE FF IN LOCATION 001A.

                ;         THIS PROGRAM DOES NOT CHECK THE ROM SOCKETS.

                ;         FAILURE CODES (LOCATION 0000 ON ERROR)

                ;        00  ALL TESTS PASSED.
                ;        01  RAM TEST FAILURE.
                ;        02  I/O REGISTER TEST FAILED. INDICATES A PROBLEM IN THE I/O
                ;            DATA OR ADDRESSING LOGIC OR A BAD 6522 VIA.
                ;         BASE PAGE DATA STORAGE
0000                      .=     0

0000 00         TSTNO:    .BYTE  0               ; TEST IN ERROR, ZERO IS OK
0001 00         REGAD:    .BYTE  0               ; RELATIVE REGISTER ADDRESS DURING I/O TEST
0002 00         ERRDTA:   .BYTE  0               ; ERROR DATA PETTERN
0003 00         OKDTA:    .BYTE  0               ; OK DATA PATTERN
                                                 ; WITH THE ERROR
0004 0000       SCMEMA:   .WORD  0               ; SCRAMBLED MEMORY ADDRESS DURING RAM TEST
                                                 ; BITS 12-14 IS RAM SEGMENT NUMBER
                                                 ; BITS 0-11 IS BYTE ADDRESS WITHIN SEGMENT
0006 0100       RANDNO:   .WORD  1               ; RANDOM NUMBER REGISTER (MUST NOT BE 0)
0008 0000       SEED:     .WORD  0               ; TO SAVE RANDOM NUMBER SEED
000A 00         TEMP:     .BYTE  0               ; TEMPORARY STORAGE
000B 00         ITCNT:    .BYTE  0               ; ITERATION COUNT
000C 0000       ADDRCT:   .WORD  0               ; ADDRESS COUNT FOR RAM TEST
```

```
000E                    .=      X'0010       ; SYSTEM CONFIGURATION PARAMETERS
                                             ; **** MUST BE SET PRIOR TO TEST ****
0010 00        SEG0:    .BYTE   0            ; RAM SEGMENT 0 PAGE ADDRESS---ENABLE REG 0
0011 00        SEG1:    .BYTE   0            ; RAM SEGMENT 1 PAGE ADDRESS_/
0012 00        SEG2:    .BYTE   0            ; RAM SEGMENT 2 PAGE ADDRESS---ENABLE REG 1
0013 00        SEG3:    .BYTE   0            ; RAM SEGMENT 3 PAGE ADDRESS_/
0014 00        SEG4:    .BYTE   0            ; RAM SEGMENT 4 PAGE ADDRESS---ENABLE REG 2
0015 00        SEG5:    .BYTE   0            ; RAM SEGMENT 5 PAGE ADDRESS_/
0016 00        SEG6:    .BYTE   0            ; RAM SEGMENT 6 PAGE ADDRESS---ENABLE REG 3
0017 00        SEG7:    .BYTE   0            ; RAM SEGMENT 7 PAGE ADDRESS_/

0018 0000      IOBASE:  .WORD   0            ; BASE ADDRESS FOR I/O SECTIOM
001A 0000      IOFLAG:  .WORD   0            ; =FF TO TEST I/O, =00 TO NOT TEST I/O

               ;        RAM TEST
               ;        WRITES RANDOM DATA INTO EACH 4K SEGMENT OF RAM IN A SCRAMBLED
               ;        ORDER AND THEN READS IT BACK.
               ;        THE ENABLE REGISTER IS EXERCISED SO IT IS OK IF RAM ADDRESSES
               ;        OVERLAP.

001C                    .=      X'200        ; START PROGRAM CODE AT 200

0200 D8        MTEST:   CLD                  ; INSURE BINARY ARITHMETIC
0201 A901               LDA     #X'01        ; INITIALIZE RANDOM NUMBER REGISTER
0203 8506               STA     RANDNO

               ;        TEST: 16 PASSES WITH RANDOM DATA, PAUSE IN 16TH PASS

0205 A90F      MAIN10:  LDA     #15          ; SET 16 ITERATION COUNT
0207 850B               STA     ITCNT
0209 20BD02    MAIN11:  JSR     RAND         ; NEW PASS, GET A RANDOM
020C A506               LDA     RANDNO       ; NUMBER IN RANDNO AND SAVE
020E 8508               STA     SEED         ; AS SEED FOR VERIFY
0210 A507               LDA     RANDNO+1
0212 8509               STA     SEED+1
0214 204D02             JSR     RNDGEN       ; GENERATE A RANDOM DATA PATTERN IN 32K
0217 A50B               LDA     ITCNT        ; TEST IF LAST PASS
0219 D011               BNE     MAIN15       ; SKIP OVER WAIT IF NOT
021B A200               LDX     #0           ; WAIT FOR ABOUT 15 SECONDS IN A TIGHT LOOP
021D A000      MAIN12:  LDY     #0           ; TO TEST REFRESH CIRCUITRY SINCE TESTING
021F A921      MAIN13:  LDA     #33          ; IS NORMALLY FAST ENOUGH TO KEEP THE
0221 18        MAIN14:  CLC                  ; MEMORY REFRESHED.
0222 69FF               ADC     #-1
0224 D0FB               BNE     MAIN14
0226 88                 DEY
0227 D0F6               BNE     MAIN13
0229 CA                 DEX
022A D0F1               BNE     MAIN12
022C A508      MAIN15:  LDA     SEED         ; RESTORE RANDOM SEED FOR VERIFY PHASE
022E 8506               STA     RANDNO
0230 A509               LDA     SEED+1
0232 8507               STA     RANDNO+1
0234 206C02             JSR     RNDVER       ; VERIFY
0237 D007               BNE     RMERLG       ; GO TO ERROR LOG IF ERROR
0239 C60B               DEC     ITCNT        ; DECREMENT AND CHECK ITERATION COUNT
023B 10CC               BPL     MAIN11       ; LOOP UNTIL 16 ITERATIONS DONE
023D 4CD002             JMP     IOTEST       ; IF OK, GO TO I/O TEST
```

```
0240 8502       RMERLG:  STA    ERRDTA        ; STORE ERROR BITS
0242 A506                LDA    RANDNO        ; STORE OK BITS
0244 8503                STA    OKDTA
0246 A901                LDA    #1            ; ESTABLISH TEST NUMBER
0248 8500                STA    TSTNO
024A 4C6A03              JMP    EXIT          ; EXIT TO MONITOR ON ERROR

                ;        RANDOM PATTERN STORED IN SCRAMBLED ORDER GENERATE ROUTINE

024D A900       RNDGEN:  LDA    #0            ; INITIALIZE ADDRESS COUNTER
024F 850C                STA    ADDRCT        ; TO 32768
0251 A980                LDA    #32768/256
0253 850D                STA    ADDRCT+1
0255 20BD02     STORPH:  JSR    RAND          ; GENERATE A RANDOM NUMBER
0258 208902              JSR    MADDR         ; FORM THE STORE ADDRESS
025B 9006                BCC    STORP1        ; SKIP STORE IF ADDRESS NOT ACTIVE
025D A506                LDA    RANDNO        ; STORE A RANDOM BYTE
025F A000                LDY    #0            ; INDIRECTLY THROUGH SCRAMBLED MEMORY
0261 9104                STA    (SCMEMA),Y    ; ADDRESS AT SCMEMA
0263 C60C       STORP1:  DEC    ADDRCT        ; DECREMENT ADDRESS COUNTER
0265 D0EE                BNE    STORPH        ; AND LOOP IF NOT ZERO
0267 C60D                DEC    ADDRCT+1
0269 D0EA                BNE    STORPH
026B 60                  RTS                  ; RETURN WHEN DONE


                ;        RANDOM PATTERN STORED IN SCRAMBLED ORDER VERIFY ROUTINE

026C A980       RNDVER:  LDA    #32768/256    ; INITIALIZE ADDRESS COUNTER
026E 850D                STA    ADDRCT+1
0270 20BD02     VERFPH:  JSR    RAND          ; GENERATE A RANDOM NUMBER
0273 208902              JSR    MADDR         ; FORM THE VERIFY ADDRESS
0276 9008                BCC    VERFP1        ; SKIP VERIFY IF ADDRESS NOT ACTIVE
0278 A000                LDY    #0
027A B104                LDA    (SCMEMA),Y    ; GET DATA FROM MEMORY INDIRECTLY
027C C506                CMP    RANDNO        ; THROUGH SCMEMA
027E D008                BNE    VERRET        ; GO RETURN ON UNEQUAL COMPARE
0280 C60C       VERFP1:  DEC    ADDRCT        ; DECREMENT ADDRESS COUNTER
0282 D0EC                BNE    VERFPH        ; AND LOOP IF NOT ZERO
0284 C60D                DEC    ADDRCT+1
0286 D0E8                BNE    VERFPH
0288 60         VERRET:  RTS                  ; RETURN

                ;        ADDRESS FORMATION AND BLOCK ENABLE ROUTINE
                ;        USES ADDRCT AND SEED TO FORM A SCRAMBLED ADDRESS IN SCMEMA
                ;        AND ALSO ENABLE THE CORRECT BLOCK
                ;        RETURNS WITH CARRY CLEAR IF SEGMENT ADDRESSED IS NOT ACTIVE

0289 A508       MADDR:   LDA    SEED          ; GET LOWER BYTE OF RANDOM NUMBER
028B 450C                EOR    ADDRCT        ; EXCLUSIVE-OR WITH LOWER ADDRESS
028D 8504                STA    SCMEMA        ; LOWER BYTE OF RESULT
028F A509                LDA    SEED+1        ; GET UPPER BYTE OF RANDOM NUMBER
0291 450D                EOR    ADDRCT+1      ; EXCLUSIVE-OR WITH UPPER ADDRESS
0293 48                  PHA                  ; SAVE FULL RESULT
0294 290F                AND    #X'0F         ; TRUNCATE TO 12 BITS TOTAL
0296 8505                STA    SCMEMA+1      ; SAVE AS ADDRESS WITHIN 4K SEGMENT
0298 68                  PLA                  ; RESTORE FULL RESULT
0299 4A                  LSRA                 ; ISOLATE AND POSITION HIGH 3 BITS
029A 4A                  LSRA
```

```
029B 4A                    LSRA
029C 4A                    LSRA
029D 2907                  AND    #X'07
029F AA                    TAX
02A0 B510                  LDA    SEGO,X        ; GET SEGMENT PAGE ADDRESS
02A2 C9FF                  CMP    #X'FF         ; TEST IF SEGMENT IS ACTIVE
02A4 18                    CLC
02A5 D001                  BNE    MADDR1        ; SKIP IF IT IS ACTIVE
02A7 60                    RTS                  ; RETURN WITH CARRY CLEAR IF NOT ACTIVE
02A8 6505        MADDR1:   ADC    SCMEMA+1      ; ADD SEGMENT PAGE ADDRESS TO SCMEMA
02AA 8505                  STA    SCMEMA+1
02AC BDB502                LDA    ENAB,X        ; GET PROPER ENABLE REGISTER MASK FOR THE
02AF A020                  LDY    #X'20         ; SEGMENT
02B1 9118                  STA    (IOBASE),Y    ; AND STORE INTO THE ENABLE REGISTER
02B3 38                    SEC                  ; SET CARRY FOR ACTIVE SEGMENT AND RETURN
02B4 60                    RTS

02B5 01010202    ENAB:     .BYTE  X'01,X'01,X'02,X'02  ; TABLE OF ENABLE REGISTER
02B9 04040808              .BYTE  X'04,X'04,X'08,X'08  ; VALUES INDEXED BY SEG. #

                 ;         RANDOM NUMBER GENERATOR SUBROUTINE
                 ;         ENTER WITH OLD RANDOM NUMBER IN RANDNO
                 ;         EXIT WITH NEW RANDOM NUMBER IN RANDNO
                 ;         USES 16 BIT INSIDE-OUT FEEDBACK SHIFT REGISTER METHOD
                 ;         METHOD OPTIMIZED FOR SPEED, ITERATE SEVERAL TIMES FOR REALLY
                 ;         GOOD RANDOM NUMBERS

02BD 0606        RAND:     ASL    RANDNO        ; SHIFT RANDNO LEFT BY 1
02BF 2607                  ROL    RANDNO+1
02C1 900C                  BCC    RAND1         ; GO RETURN IF A ZERO SHIFTED OUT
02C3 A506                  LDA    RANDNO        ; FLIP SELECTED BITS IN RANDNO
02C5 4987                  EOR    #X'87         ; IF A ONE WAS SHIFTED OUT
02C7 8506                  STA    RANDNO
02C9 A507                  LDA    RANDNO+1
02CB 491D                  EOR    #X'1D
02CD 8507                  STA    RANDNO+1
02CF 60          RAND1:    RTS                  ; RETURN

                 ;         REGISTER INTERFERENCE AND DATA RETENTION TEST
                 ;         TREATS THE 7 EASILY ACCESSABLE REGISTERS IN EACH 6522 AS 14
                 ;         "RAM" LOCATIONS AND DOES A "RAM TEST" USING RANDOM DATA.
                 ;         ANY I/O EQUIPMENT CONNECTED TO THE I/O PORTS MUST BE
                 ;         DISCONNECTED OR IT WILL BE THRASHED TO DEATH!
                 ;         CAUTION - A PROM MUST NOT BE IN THE PROGRAMMING SOCKET WHILE
                 ;                   THIS TEST IS RUN!

02D0 A51A        IOTEST:   LDA    IOFLAG        ; CHECK IF I/O TO BE TESTED
02D2 F003                  BEQ    IOTSTO
02D4 4C6A03                JMP    EXIT          ; EXIT IF I/O NOT TO BE TESTED
02D7 A900        IOTSTO:   LDA    #0            ; MAKE SURE PCR AND ACR REGISTERS ARE ZERO
02D9 A00A                  LDY    #X'0A         ; IN BOTH 6522'S
02DB 9118                  STA    (IOBASE),Y
02DD C8                    INY
02DE 9118                  STA    (IOBASE),Y
02E0 A01A                  LDY    #X'1A
02E2 9118                  STA    (IOBASE),Y
02E4 C8                    INY
02E5 9118                  STA    (IOBASE),Y
```

```
02E7 A900               LDA    #0            ; SET 256 PASS COUNT
02E9 850B               STA    ITCNT
02EB A506      IOTST1:  LDA    RANDNO        ; SAVE RANDOM NUMBER SEED
02ED 8508               STA    SEED
02EF A507               LDA    RANDNO+1
02F1 8509               STA    SEED+1
02F3 A200               LDX    #0            ; SET REGISTER COUNT
02F5 20BD02    IOTST2:  JSR    RAND          ; PRODUCE A RANDOM NUMBER
02F8 BC5C03             LDY    REGADR,X      ; GET A REGISTER ADDRESS INTO Y
02FB A506               LDA    RANDNO        ; STORE RANDOM BYTE INTO A 6522 REGISTER
02FD 9118               STA    (IOBASE),Y
02FF E8                 INX                  ; INCREMENT REGISTER POINTER
0300 E00E               CPX    #14
0302 D0F1               BNE    IOTST2        ; LOOP UNTIL ALL 14 REGISTERS FILLED

0304 A508               LDA    SEED          ; RESEED THE RANDOM NUMBER GENERATOR
0306 8506               STA    RANDNO
0308 A509               LDA    SEED+1
030A 8507               STA    RANDNO+1
030C A200               LDX    #0            ; SET REGISTER COUNT
030E 20BD02    IOTST3:  JSR    RAND          ; PRODUCE THE SAME RANDOM NUMBER AS ON
                                             ; STORE PASS
0311 E004               CPX    #4            ; TEST IF ONE OF THE DATA REGISTERS TO BE
0313 101C               BPL    IOTST4        ; READ, SKIP IF NOT
0315 BC6003             LDY    REGADR+4,X    ; TO READ A DATA REGISTER, CORRESPONDING
0318 B118               LDA    (IOBASE),Y    ; DIRECTION REGISTER MUST BE ALL OUTPUTS
031A 850A               STA    TEMP          ; SAVE OLD DIRECTION REG CONTENTS
031C A9FF               LDA    #X'FF         ; SET DIRECTION REGISTER TO OUTPUTS
031E 9118               STA    (IOBASE),Y
0320 BC5C03             LDY    REGADR,X
0323 B118               LDA    (IOBASE),Y    ; GET DATA REGISTER CONTENTS
0325 48                 PHA                  ; AND SAVE IT
0326 BC6003             LDY    REGADR+4,X    ; RESTORE DIRECTION REGISTER CONTENTS
0329 A50A               LDA    TEMP
032B 9118               STA    (IOBASE),Y
032D 68                 PLA                  ; RESTORE DATA READ
032E 4C3603             JMP    IOTST5        ; SKIP AHEAD
0331 BC5C03    IOTST4:  LDY    REGADR,X      ; GET REGISTER ADDRESS
0334 B118               LDA    (IOBASE),Y    ; READ THE REGISTER CONTENTS
0336 C506      IOTST5:  CMP    RANDNO        ; COMPARE WITH DATA THAT WAS WRITTEN
0338 D010               BNE    IOERR         ; JUMP OUT ON ERROR
033A E8                 INX                  ; INCREMENT REGISTER POINTER
033B E00E               CPX    #14
033D D0CF               BNE    IOTST3        ; LOOP UNTIL ALL 14 REGISTERS VERIFIED
033F C60B               DEC    ITCNT         ; COUNT ITERATIONS PERFORMED
0341 D0A8               BNE    IOTST1        ; LOOP IF NOT DONE
0343 A900               LDA    #0            ; SET SUCESSFUL TEST ERROR CODE
0345 8500               STA    TSTNO
0347 4C6A03             JMP    EXIT          ; EXIT TO THE SYSTEM MONITOR
                                             ; CAN BE CHANGED TO A JMP MTEST
                                             ; FOR CONTINUOUS TESTING
```

```
034A 8502      IOERR:   STA    ERRDTA          ; STORE ERRONEOUS DATA READ
034C A506               LDA    RANDNO          ; STORE CORRECT DATA
034E 8503               STA    OKDTA
0350 BD5C03             LDA    REGADR,X        ; STORE RELATIVE REGISTER ADDRESS
0353 8501               STA    REGAD
0355 A902               LDA    #2              ; SET ERROR CODE
0357 8500               STA    TSTNO
0359 4C6A03             JMP    EXIT            ; EXIT TO SYSTEM MONITOR

035C 01001110  REGADR:  .BYTE  X'01,X'00,X'11,X'10  ; DATA REGISTER ADDRESSES
0360 03021312           .BYTE  X'03,X'02,X'13,X'12  ; DIRECTION REGISTER ADDRESSES
0364 06071617           .BYTE  X'06,X'07,X'16,X'17  ; TIMER 1 LATCH ADDRESSES
0368 0A1A               .BYTE  X'0A,X'1A            ; SHIFT REG ADDRESSES

036A 000000    EXIT:    .BYTE  0,0,0           ; EXIT TO SYSTEM MONITOR VIA BREAK, CAN BE
                                               ; CHANGED TO A JMP IF NECESSARY

0000                    .END
```

18. PROM PROGRAMMER PROGRAM LISTING

The following is a listing for a program to operate the 2716/2732 EPROM programmer on the K-1032. Please see section 8 for use instructions for this program.

```
                ;       BURN PROGRAM FOR M.T.U. K-1032 RAM EPROM & I/O BOARD
                ;              BY KEITH SPROUL AND HAL CHAMBERLIN

                ;       WILL BURN:     INTEL 2716   OR EQUIVALENT
                ;                      T.I.  2516
                ;                      ANY   2732   OR EQUIVALENT

                ; +++++ THE STARTING ADDRESS OF BUFFER RAM HAS TO BE PUT
                ; +++++ INTO SAL,SAH BEFORE RUNNING ALL PARTS EXCEPT
                ; +++++ VERIFY NEW PROM.

                ; +++++ PRMTYP MUST BE SET TO 00 FOR 2716 OR TO FF FOR 2732 PROM'S.

                ; +++++ THE I/O BASE ADDRESS FOR THE K-1032 MUST BE STORED IN IOBASE
                ; +++++ (LOW BYTE FIRST) BEFORE EXECUTING ANY PART OF THIS PROGRAM.

                ; +++++ THE K-1032 IS SHIPPED SET UP TO PROGRAM 2716 PROMS.
                ; +++++ TO PROGRAM 2732'S, JUMPERS MUST BE CHANGED.  SEE THE JUMPER
                ; +++++ OPTIONS SECTION OF THE K-1032 MANUAL FOR INSTRUCTIONS.

                ;       ************************************************************
                ;       ********DANGER******* MAKE SURE THE PROGRAM SELECTION AND  *
                ;       ********DANGER******* PROM JUMPERING AGREE WITH THE TYPE   *
                ;       ********DANGER******* OF PROM TO BE PROGRAMMED! (SEE ABOVE)*
                ;       ************************************************************

                ;       THIS PROGRAM RETURNS TO THE SYSTEM MONITOR BY EXECUTING A BRK
                ;       INSTRUCTION.  IF YOUR MONITOR DOES NOT SUPPORT BRK ENTRY,
                ;       REPLACE THE 3 ZERO BYTES AT RESET WITH A JUMP TO THE MONITOR'S
                ;       ENTRY POINT.

                ;       NOTE THAT THE PROGRAM PULSE DUTY CYCLE IS 33% BECAUSE OF
                ;       PROGRAMMING POWER LIMITATIONS.  THUS IT WILL REQUIRE 300
                ;       SECONDS TO PROGRAM A 2716 AND 600 SECONDS TO PROGRAM A 2732.

0000                    .=      0            ; ALL DATA IN PAGE ZERO

0000 00        BADATA:  .BYTE   0            ; DATA THAT IS THERE
0001 00        OKDATA:  .BYTE   0            ; DATA THAT SHOULD BE THERE
0002 0000      BADADR:  .WORD   0            ; ADDRESS OF BAD BYTE
0004 00        SAL:     .BYTE   0            ; START ADDRESS LOW
0005 00        SAH:     .BYTE   0            ; START ADDRESS HIGH
0006 00        PRMTYP:  .BYTE   0            ; 00 FOR 2716, FF FOR 2732
0007 0000      IOBASE:  .WORD   0            ; BASE ADDRESS OF I/O SECTION OF K-1032
0009 00        EAL:     .BYTE   0            ; END ADDRESS +1 LOW
000A 00        EAH:     .BYTE   0            ; END ADDRESS +1 HIGH
000B 00        COUNT:   .BYTE   0            ; # OF TIMES THROUGH ADDRESS LOOP
000C 0000      TMPADR:  .WORD   0            ; ADDRESS POINTER
```

```
                    ;       I/O PORT EQUATES AS DISPLACEMENTS FROM IOBASE ON K-1032

0011                PORTAD  =       X'0011       ; VIA 2 PORT A DATA
0013                PORTAC  =       X'0013       ; VIA 2 PORT A DIRECTION
0010                PORTBD  =       X'0010       ; VIA 2 PORT B DATA
0012                PORTBC  =       X'0012       ; VIA 2 PORT B DIRECTION
001C                PRTPCR  =       X'001C       ; VIA 2 PERIPHERAL CONTROL REGISTER
001B                PRTACR  =       X'001B       ; VIA 2 AUXILIARY CONTROL REGISTER
001E                PRTIER  =       X'001E       ; VIA 2 INTERRUPT ENABLE REGISTER


                    ;       PORT A = DATA REGISTER  (READ/WRITE PROM)
                    ;                (SET TO INPUT EXCEPT WHEN PROGRAMMING)

                    ;       PORT B = CONTROL PORT
                    ;                BIT 0   0 = APPLY +24 VOLTS PROGRAMMING VOLTAGE
                    ;                        1 = GROUND PROGRAM VOLTAGE INPUT
                    ;                BIT 1   0 = OUTPUT ENABLE ON 2716
                    ;                        1 = PROGRAM PULSE ON 2716
                    ;                            NO FUNCTION WITH 2732
                    ;                BIT 2   0-TO-1 TRANSITION = INCREMENT ADDRESS COUNTER
                    ;                BIT 3   0 = NOTHING    1 = HOLD ADDRESS COUNTER AT ZERO
                    ;                BIT 4   0 = CHIP ENABLE

                    ;       1 MILLISECOND = 1000 MACHINE CYCLES  (1.0MHZ CLOCK ASSUMED)

000E                        .=      X'0200       ; START JUST ABOVE THE STACK

                    ;       TRANSFER VECTOR

0200 4C7C02                 JMP     NEWPRM       ; VERIFY NEW EPROM
0203 4C0F02                 JMP     PGMVFY       ; PROGRAM AND VERIFY
0206 4C3F02                 JMP     VERIFY       ; VERIFY ONLY
0209 4CB902                 JMP     READ         ; READ PROM INTO RAM
020C 000000  RESET:         .BYTE   0,0,0        ; BRK RETURN, ROOM FOR A JMP

                    ;       PROGRAM A EPROM

020F A9FF    PGMVFY:        LDA     #X'FF        ; INIT DATA PORT TO OUTPUT
0211 205403                 JSR     INIPRT       ; INIT PORTS AND IDLE PROM
0214 D8                     CLD
0215 A901                   LDA     #1
0217 850B                   STA     COUNT        ; SET TO 1  TIME  THROUGH ALL LOCATIONS

0219 202A03  LOOP1:         JSR     INIREG       ; INIT REGISTERS AND POINTERS
021C 208C03                 JSR     PROGM        ; SET PROGRAM MODE
021F A000    LOOP2:         LDY     #0
0221 B10C                   LDA     (TMPADR),Y
0223 20D602                 JSR     BURN         ; BURN THE DATA AT (TMPADR) INTO EPROM
0226 200703                 JSR     INCTMP       ; INCREMENT THE ADDRESS AND CHECK IF END
0229 90F4                   BCC     LOOP2        ; DO ALL LOCATIONS
022B 207203                 JSR     IDLE         ; IDLE THE PROM AFTER PROGRAMMING

022E A2FF                   LDX     #X'FF        ; WAIT FOR RECOVERY BEFORE CONTINUING
0230 A0FF    LOOP3:         LDY     #X'FF        ; LOOPS WILL WAIT FOR ABOUT 300MS
0232 88      LOOP4:         DEY
0233 D0FD                   BNE     LOOP4
0235 CA                     DEX
```

```
0236 D0F8              BNE    LOOP3
0238 C60B              DEC    COUNT          ; LOOP THROUGH ADDRESSES COUNT TIMES
023A D0DD              BNE    LOOP1
023C 4C3F02            JMP    VERIFY         ; GO VERIFY CORRECT PROGRAMMING

                ;      VERIFY THAT AN EPROM MATCHES RAM

023F A900     VERIFY:  LDA    #X'00          ; INIT DATA PORT TO INPUT
0241 205403            JSR    INIPRT         ; INIT PORTS AND IDLE PROM
0244 202A03            JSR    INIREG         ; INIT THE REGISTERS, POINTERS, & PORTB
0247 207F03            JSR    READM          ; SET PROM READ MODE
024A 200203   VERFY1:  JSR    LOAD           ; GET DATA CURRENTLY IN EPROM
024D A000              LDY    #0
024F D10C              CMP    (TMPADR),Y     ; COMPARE WITH RAM
0251 F014              BEQ    OKAY           ; JUMP IF MATCH

0253 8500              STA    BADATA         ; STORE THE BAD DATA
0255 A50C              LDA    TMPADR         ; STORE BAD ADDRESS OF PROM
0257 8502              STA    BADADR         ; IN BADADR
0259 A50D              LDA    TMPADR+1
025B 8503              STA    BADADR+1
025D B10C              LDA    (TMPADR),Y     ; GET WHAT SHOULD BE IN EPROM
025F 8501              STA    OKDATA         ; SAVE IT
0261 207203            JSR    IDLE           ; IDLE THE PROM
0264 4C0C02            JMP    RESET          ; RETURN TO THE MONITOR

0267 200703   OKAY:    JSR    INCTMP         ; INCREMENT REGISTERS
026A 90DE              BCC    VERFY1
026C A900              LDA    #0             ; SET ALL ERROR INFORMATION TO ZEROES IF OK
026E 8502              STA    BADADR
0270 8503              STA    BADADR+1
0272 8500              STA    BADATA
0274 8501              STA    OKDATA
0276 207203            JSR    IDLE           ; IDLE THE PROM
0279 4C0C02            JMP    RESET          ; RETURN TO THE MONITOR

                ;      VERIFY NEW EPROM

027C A900     NEWPRM:  LDA    #X'00          ; INIT DATA PORT TO INPUT
027E 205403            JSR    INIPRT         ; INITIALIZE PORTS AND IDLE THE PROM
0281 202A03            JSR    INIREG         ; RESET REGISTERS AND POINTERS
0284 207F03            JSR    READM          ; SET PROM READ MODE

0287 200203   NWPRM1:  JSR    LOAD           ; READ A PROM LOCATION
028A C9FF              CMP    #X'FF          ; TEST IF IN THE ERASED STATE
028C D017              BNE    OLDPRM         ; JUMP OUT IF NOT
028E 200703            JSR    INCTMP         ; INCREMENT REGISTERS AND POINTERS
0291 90F4              BCC    NWPRM1         ; LOOP UNTIL ALL LOCATIONS EXAMINED

0293 A900              LDA    #0             ; ZERO ALL ERROR ADDRESS IF OK
0295 8502              STA    BADADR
0297 8503              STA    BADADR+1
0299 A9FF              LDA    #X'FF          ; SET ERROR DATA TO FF IF OK
029B 8500              STA    BADATA
029D 8501              STA    OKDATA
029F 207203            JSR    IDLE           ; IDLE THE PROM
02A2 4C0C02            JMP    RESET          ; RETURN TO THE MONITOR
```

```
02A5 8500     OLDPRM:  STA   BADATA        ; SET BAD DATA WITH PROM CONTENTS IF NOT OK
02A7 A9FF              LDA   #X'FF         ; SET OKDATA TO FF
02A9 8501              STA   OKDATA
02AB A50C              LDA   TMPADR        ; SET ERROR ADDRESS IF NOT OK
02AD 8502              STA   BADADR
02AF A50D              LDA   TMPADR+1
02B1 8503              STA   BADADR+1
02B3 207203            JSR   IDLE          ; IDLE THE PROM
02B6 4C0C02            JMP   RESET         ; RETURN TO THE MONITOR

              ;        READ EPROM CONTENTS INTO RAM

02B9 A900     READ:    LDA   #X'00         ; INIT DATA PORT TO INPUT
02BB 205403            JSR   INIPRT        ; INIT PORTS AND IDLE THE PROM
02BE 202A03            JSR   INIREG        ; INIT REGISTERS
02C1 207F03            JSR   READM         ; SET PROM READ MODE
02C4 200203   READ1:   JSR   LOAD          ; GET PROM DATA
02C7 A000              LDY   #0
02C9 910C              STA   (TMPADR),Y    ; STORE IT INTO RAM
02CB 200703            JSR   INCTMP        ; INCREMENT POINTERS
02CE 90F4              BCC   READ1         ; LOOP UNTIL DONE
02D0 207203            JSR   IDLE          ; SET PROM IDLE MODE
02D3 4C0C02            JMP   RESET         ; RETURN TO THE MONITOR

              ;        BURN DATA AT (TMPADR) INTO EPROM AT (ADDRESS COUNTER)

02D6 A011     BURN:    LDY   #PORTAD
02D8 9107              STA   (IOBASE),Y    ; STORE DATA AT PROM
02DA A010              LDY   #PORTBD
02DC B107              LDA   (IOBASE),Y    ; WAIT 10 MICROSECONDS AND
02DE 4910              EOR   #X'10
02E0 EA                NOP
02E1 EA                NOP
02E2 EA                NOP
02E3 9107              STA   (IOBASE),Y    ; TURN ON PROGRAM PULSE
02E5 20F702            JSR   DELAY         ; HOLD THE PULSE ON FOR 50 MILLISECONDS
02E8 A010              LDY   #PORTBD
02EA B107              LDA   (IOBASE),Y
02EC 4910              EOR   #X'10         ; TURN THE PROGRAM PULSE OFF
02EE 9107              STA   (IOBASE),Y
02F0 20F702            JSR   DELAY         ; HOLD IT OFF FOR 100 MILLISECONDS FOR A
02F3 20F702            JSR   DELAY         ; 33% DUTY CYCLE (ALLOWS +24V POWER SUPPLY
                                           ; TO RECOVER)
02F6 60                RTS                 ; RETURN

02F7 A032     DELAY:   LDY   #50           ; WAIT FOR 50 MILLISECONDS
02F9 A2C8     DELAY1:  LDX   #200          ; LEAVES X AND Y SET TO 0
02FB CA       DELAY2:  DEX
02FC D0FD              BNE   DELAY2
02FE 88                DEY
02FF D0F8              BNE   DELAY1
0301 60                RTS                 ; RETURN

              ;        LOAD DATA AT PROM (ADDRESS COUNTER) INTO ACC

0302 A011     LOAD:    LDY   #PORTAD
0304 B107              LDA   (IOBASE),Y
0306 60                RTS
```

```
               ;         INCREMENT AND TEST TMPADR, INCREMENT EPROM ADDRESS COUNTER

0307 E60C      INCTMP:   INC    TMPADR         ; INCREMENT LOW BYTE
0309 D002                BNE    INC1
030B E60D                INC    TMPADR+1       ; INCREMENT HIGH BYTE IF NECESSARY
030D A50C      INC1:     LDA    TMPADR         ; TEST IF TMPADR IS GREATER THAN OR EQUAL
030F C509                CMP    EAL            ; TO EAL,EAH
0311 A50D                LDA    TMPADR+1
0313 E50A                SBC    EAH
0315 9001                BCC    NOTFIN         ; JUMP IF NOT
0317 60                  RTS                   ; IF DONE, RETURN WITH CARRY SET
0318 201D03    NOTFIN:   JSR    INCREG         ; INCREMENT TMPADR ONLY IF NOT FINISHED
031B 18                  CLC
031C 60                  RTS                   ; RETURN WITH CARRY CLEAR IF NOT DONE

031D A010      INCREG:   LDY    #PORTBD
031F B107                LDA    (IOBASE),Y     ; INCREMENT HARDWARE ADDRESS COUNTER
0321 0904                ORA    #X'04          ; SET INCREMENT PULSE HIGH
0323 9107                STA    (IOBASE),Y
0325 29FB                AND    #X'FB          ; SET INCREMENT PULSE LOW
0327 9107                STA    (IOBASE),Y
0329 60                  RTS                   ; RETURN

               ;         INIT REGISTERS

032A 204703    INIREG:   JSR    RESADR         ; RESET ADDRESS COUNTER
032D A504                LDA    SAL            ; TRANSFER STARTING RAM ADDRESS TO TMPADR
032F 850C                STA    TMPADR
0331 A505                LDA    SAH
0333 850D                STA    TMPADR+1
0335 18                  CLC                   ; ADD PROM SIZE TO STARTING ADDRESS TO GET
0336 A504                LDA    SAL            ; ENDING ADDRESS+1
0338 8509                STA    EAL
033A A505                LDA    SAH
033C 6908                ADC    #8             ; 2716 = 8 256 BYTE PAGES
033E 2406                BIT    PRMTYP         ; TEST PROM TYPE
0340 1002                BPL    INIRG1         ; SKIP IF 2716
0342 6908                ADC    #8             ; ADD ANOTHER 8 PAGES FOR 2732
0344 850A      INIRG1:   STA    EAH
0346 60                  RTS

0347 A010      RESADR:   LDY    #PORTBD
0349 B107                LDA    (IOBASE),Y
034B 0908                ORA    #X'08          ; RESET THE HARDWARE ADDRESS COUNTER TO 0
034D 9107                STA    (IOBASE),Y
034F 29F7                AND    #X'F7
0351 9107                STA    (IOBASE),Y
0353 60                  RTS

               ;         INITIALIZE THE PORTS AND IDLE THE PROM

0354 48        INIPRT:   PHA                   ; SAVE DATA PORT DIRECTION ON STACK
0355 A900                LDA    #0             ; INITIALIZE THE TWO VIA CONTROL REGISTERS
0357 A01C                LDY    #PRTPCR
0359 9107                STA    (IOBASE),Y     ; PERIPHERAL CONTROL REGISTER
035B A01B                LDY    #PRTACR
035D 9107                STA    (IOBASE),Y     ; AUXILIARY CONTROL REGISTER
```

```
035F A01E                LDY    #PRTIER
0361 9107                STA    (IOBASE),Y    ; INTERRUPT ENABLE REGISTER
0363 207203              JSR    IDLE          ; SET PORT B DATA FOR IDLE PROM
0366 A91F                LDA    #X'1F         ; SET LOW 5 BITS OF PORT B TO OUTPUTS
0368 A012                LDY    #PORTBC
036A 9107                STA    (IOBASE),Y
036C 68                  PLA                  ; RESTORE DATA PORT DIRECTION
036D A013                LDY    #PORTAC
036F 9107                STA    (IOBASE),Y    ; SET DATA DIRECTION FOR PORT A
0371 60                  RTS                  ; RETURN

               ;         SET PORT B FOR IDLE PROM

0372 A903      IDLE:     LDA    #X'03         ; IDLE CODE FOR 2716
0374 2406                BIT    PRMTYP        ; CHECK PROM TYPE
0376 1002                BPL    IDLE1         ; SKIP IF 2716
0378 A911                LDA    #X'11         ; IDLE CODE FOR 2732
037A A010      IDLE1:    LDY    #PORTBD
037C 9107                STA    (IOBASE),Y    ; SET IDLE STATE
037E 60                  RTS                  ; RETURN

               ;         SET PORT B FOR READ MODE

037F A900      READM:    LDA    #X'00         ; READ MODE CODE FOR 2716
0381 2406                BIT    PRMTYP        ; CHECK PROM TYPE
0383 1002                BPL    READM1        ; SKIP IF 2716
0385 A901                LDA    #X'01         ; READ MODE CODE FOR 2732
0387 A010      READM1:   LDY    #PORTBD
0389 9107                STA    (IOBASE),Y    ; SET READ MODE
038B 60                  RTS                  ; RETURN

               ;         SET PORT B FOR PROGRAM MODE

038C A902      PROGM:    LDA    #X'02         ; PROGRAM MODE FOR 2716
038E 2406                BIT    PRMTYP        ; CHECK PROM TYPE
0390 1002                BPL    PROGM1        ; SKIP IF 2716
0392 A910                LDA    #X'10         ; PROGRAM MODE FOR 2732
0394 A010      PROGM1:   LDY    #PORTBD
0396 9107                STA    (IOBASE),Y    ; SET PROGRAM MODE
0398 60                  RTS                  ; RETURN

0000                     .END
```

19. K-1032 PARTS LIST

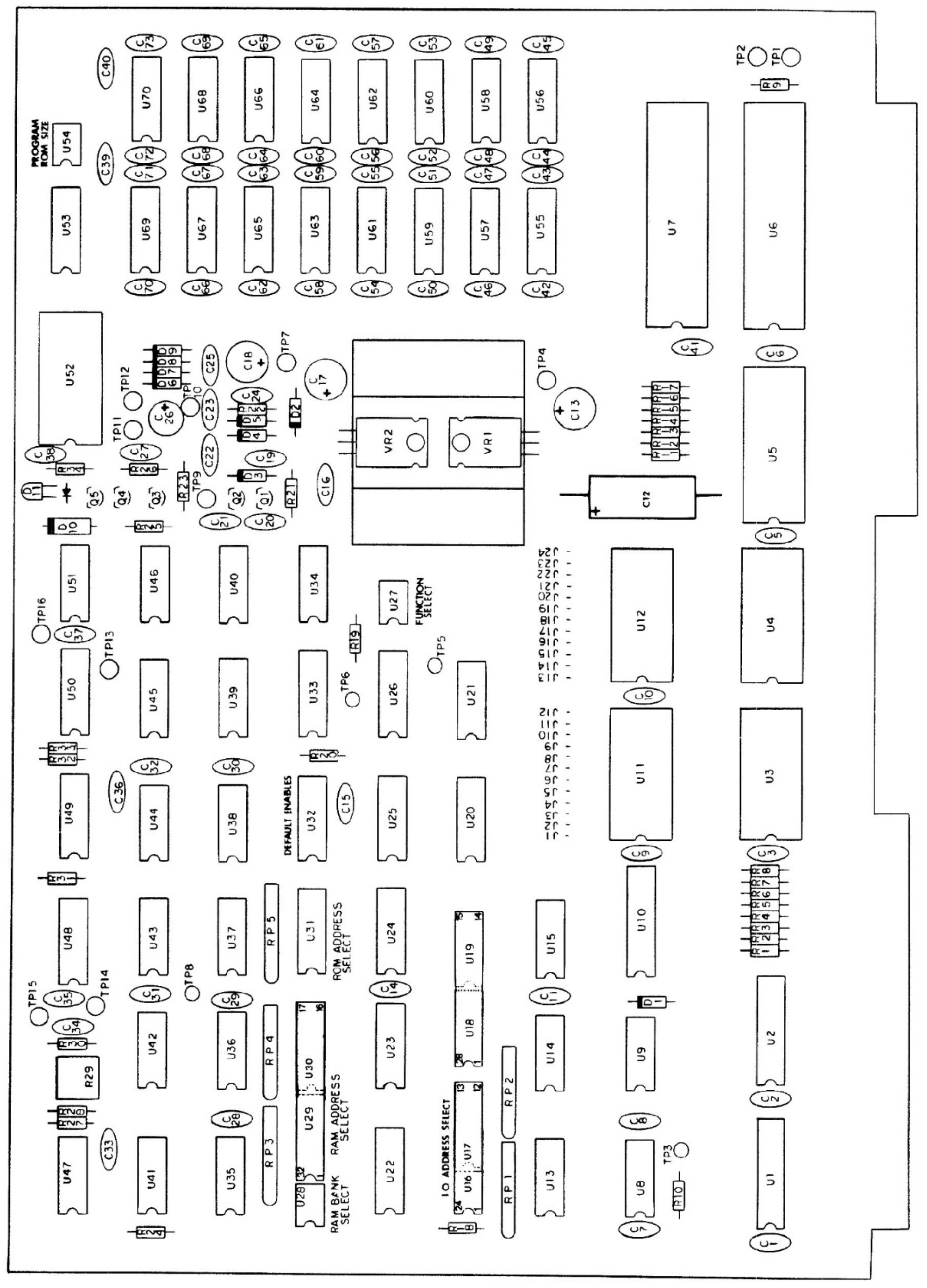
DESIGNATION[1]	PART TYPE	K1032-1 QTY	K1032-2 QTY
U35,36,(38),40,41,45	LOGIC, 74LS00	6	5
U15,34,37,44	LOGIC, 74LS04	4	4
U46	LOGIC, 74LS08	1	1
U39,43	LOGIC, 74LS10	2	2
U47	LOGIC, 74LS13	1	1
U42	LOGIC, 74LS20	1	1
U9,(21)	LOGIC, 74LS30	2	1
U51	LOGIC, 74LS51	1	1
U49,50	LOGIC, 74LS109	2	2
U22,23,(24),26	LOGIC, 74LS138	4	3
U33	LOGIC, 74LS161	1	1
U10	LOGIC, 74LS245	1	1
U8,13,14	LOGIC, 74LS266	3	3
U20,(25)	LOGIC, 74LS298	2	1
U48	LOGIC, 74LS368	1	1
U1,2	LOGIC, 81LS95	2	2
U55,(56),57,(58), 59,(60),61,(62), 63,(64),65,(66), 67,(68),69,(70)	MEMORY, 4116 TYPE 200ns	16	8
U5	LOGIC, 3242	1	1
U53	LOGIC, 4040	1	0
U6,7	LOGIC, 6522	2	0
U31	HEADER, SOLDER 16 PIN	1	0
U29/30	HEADER, SOLDER 24 PIN	1	1
XU17,27,28,(54)	SOCKET, PC 8 PIN	4	3
XU8,9,13-15,(18,19) (21),34-36,(37,38) 39-42,(43),44-47, 51	SOCKET, PC 14 PIN	24	20
XU16,20,22,23,(24,25), 26,29,30,(31),32, 33,48-50,(53),55, (56),57,(58),59, (60),61,(62),63, (64),65,(66),67, (68),69,(70)	SOCKET, PC 16 PIN	32	20
XU1,2,10	SOCKET, PC 20 PIN	3	3
XU(3,4,11,12,52)	SOCKET, PC 24 PIN	5	0
XU5	SOCKET, PC 28 PIN	1	1
XU(6,7)	SOCKET, PC 40 PIN	2	0

NOTES: 1. Items in parentheses are omitted on the K-1032-2 assembly.

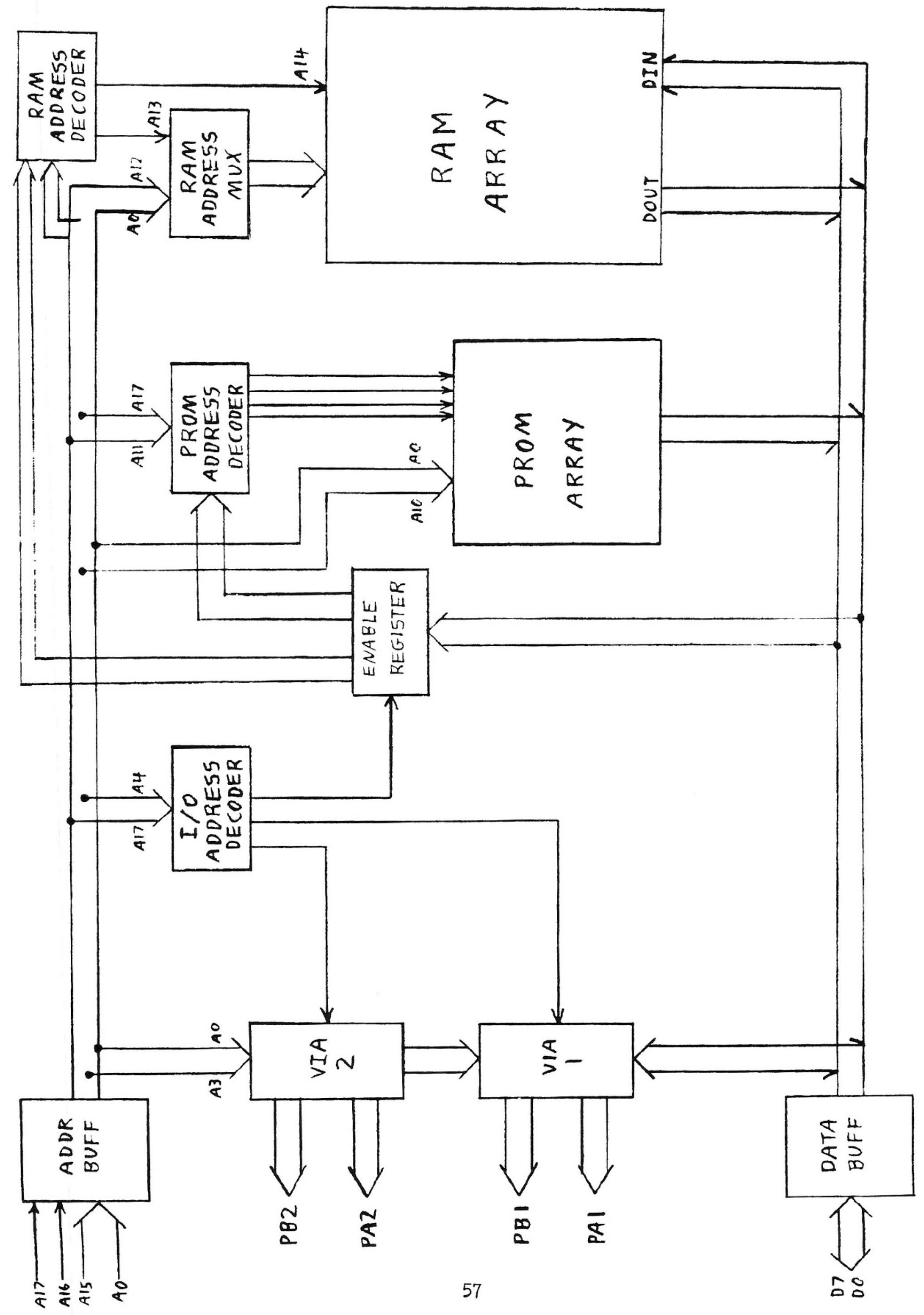
DESIGNATION[1]	PART TYPE	K1032-1 QTY	K1032-2 QTY
VR2	VOLT REG. +5V LM340T-5	1	1
VR1	VOLT REG. +12V LM341P-12	1	1
D1	DIODE, GERMANIUM 1N270	1	1
D3-5,(6-9)	DIODE, SILICON 1N4148	7	3
D(10)	DIODE, ZENER 24.5V 1N4749A 2%	1	0
D2	DIODE, ZENER 5.1V 1N5231	1	1
D(11)	DIODE, LED RED	1	0
Q(3,4),5	TRANSISTOR, NPN PN2222	3	1
Q1	TRANSISTOR, NPN PN3646	1	1
Q2	TRANSISTOR, PNP PN4916	1	1
HQ1/2	HEATSINK 1" X 1.5" AAVID 5072B	1	1
TP1-7,(8,9),10, (11,12),13-16	TEST POSTS, .062" DIA	16	12
R11-17	RESISTOR, 1/4W 5% 47 OHM	7	7
R1-8,20	RESISTOR, 1/4W 5% 270 OHM	9	9
R(26),28,30	RESISTOR, 1/4W 5% 470 OHM	3	2
R(9),10,19,21,(23) 31	RESISTOR, 1/4W 5% 1K	6	4
R22,27,33	RESISTOR, 1/4W 5% 2.2K	3	3
R32	RESISTOR, 1/4W 5% 3.3K	1	1
R(25,34)	RESISTOR, 1/4W 5% 4.7K	2	0
R18,24	RESISTOR, 1/4W 5% 10K	2	2
R29	TRIMPOT, 500 OHM SQUARE	1	1
RP1-4,(5)	RES. PACK 5% 10K 8 TO 5V	5	4
C(26)	CAP,ELECTROLYTIC 47UF 50V RADIAL	1	0
C13,17,18	CAP,ELECTROLYTIC 100UF 16V RADIAL	3	3
C12	CAP,ELECTROLYTIC 1000UF 25V AXIAL	1	1
C33,36	CAP, DISK, NPO 68PF 12V	2	2
C20	CAP, DISK, Y5F 100PF 12V	1	1
C15	CAP, DISK, Y5F 1000PF 12V	1	1
C19	CAP, DISK, Y5F 2200UF 12V	1	1
C34	CAP, DISK, Z5U .01UF 12V	1	1
C1-3,5-11,14,16, 28-32,35,37-39, (40,41),42,43, (44,45),46,47, (48,49),50,51, (52,53),54,55, (56,57),58,59, (60,61),62,63, (64,65),66,67, (68,69),70,71, (72,73)	CAP, DISK, Z5U .05UF 12V	55	37
C21,(22),24	CAP, DISK, Z5U .1UF 12V	3	2
C(23,25,27)	CAP, DISK, Z5U .1UF 50V	3	0

NOTES: 1. Items in parentheses are omitted on the K-1032-2 assembly.

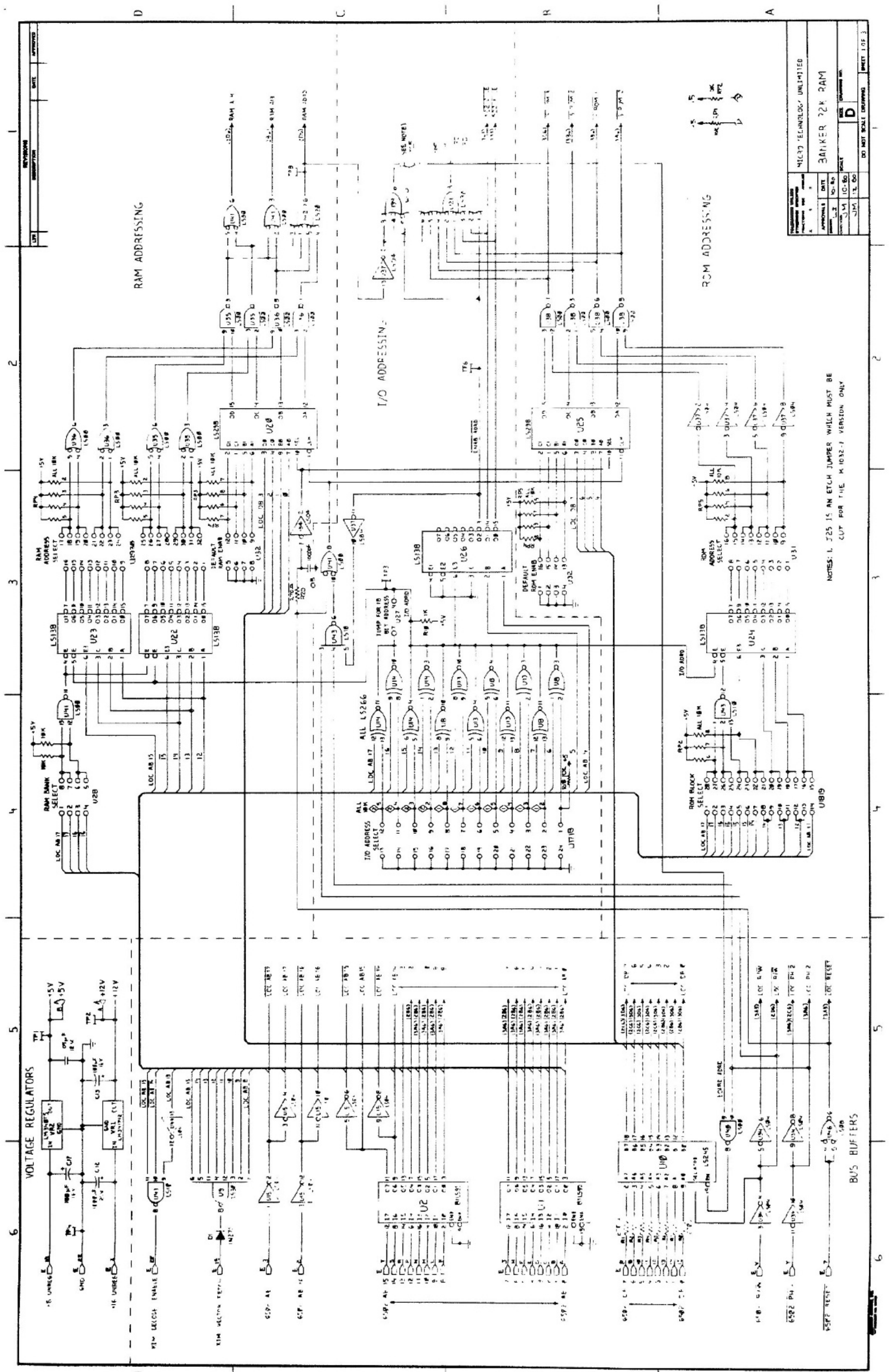
20. K-1032 PARTS LAYOUT

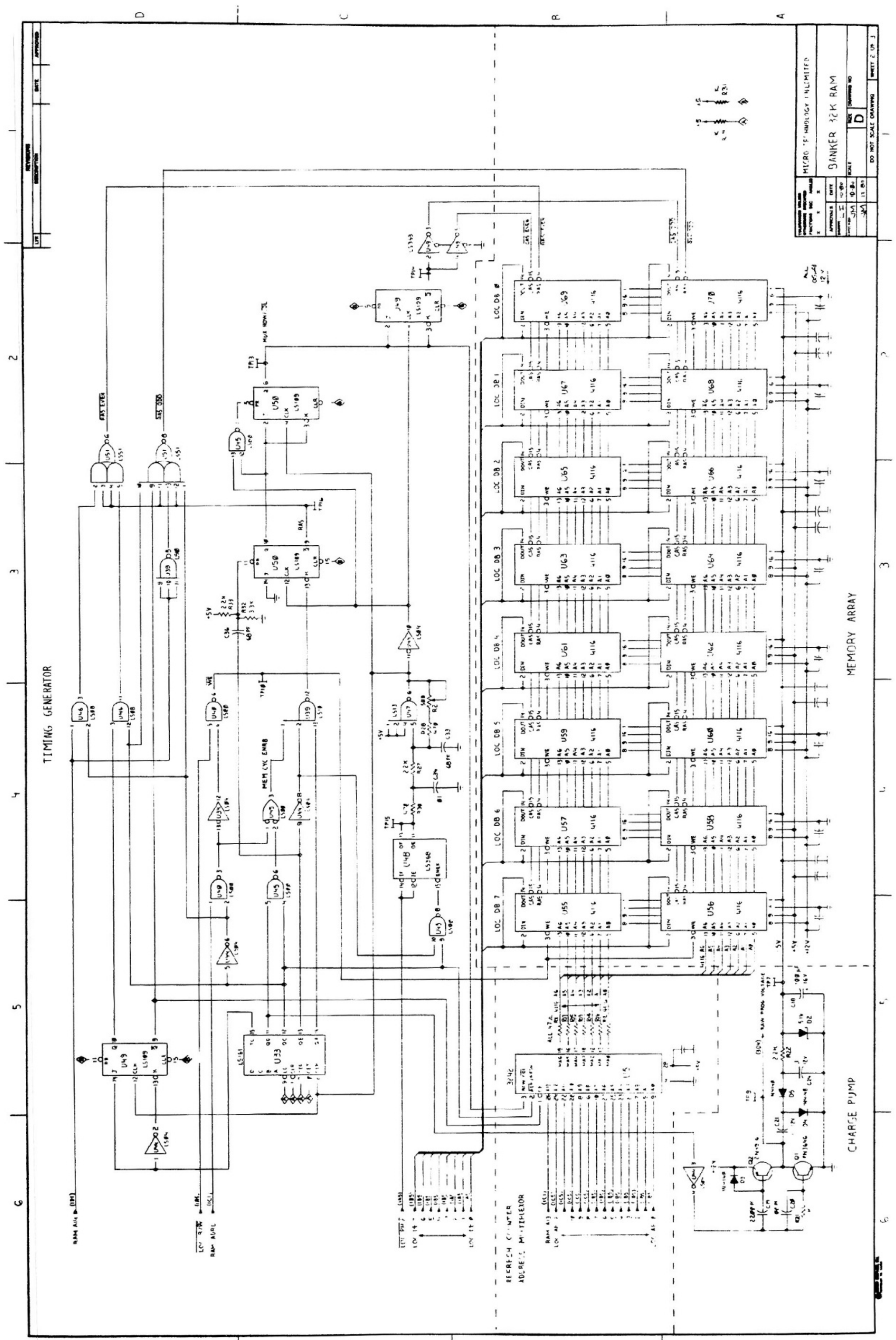


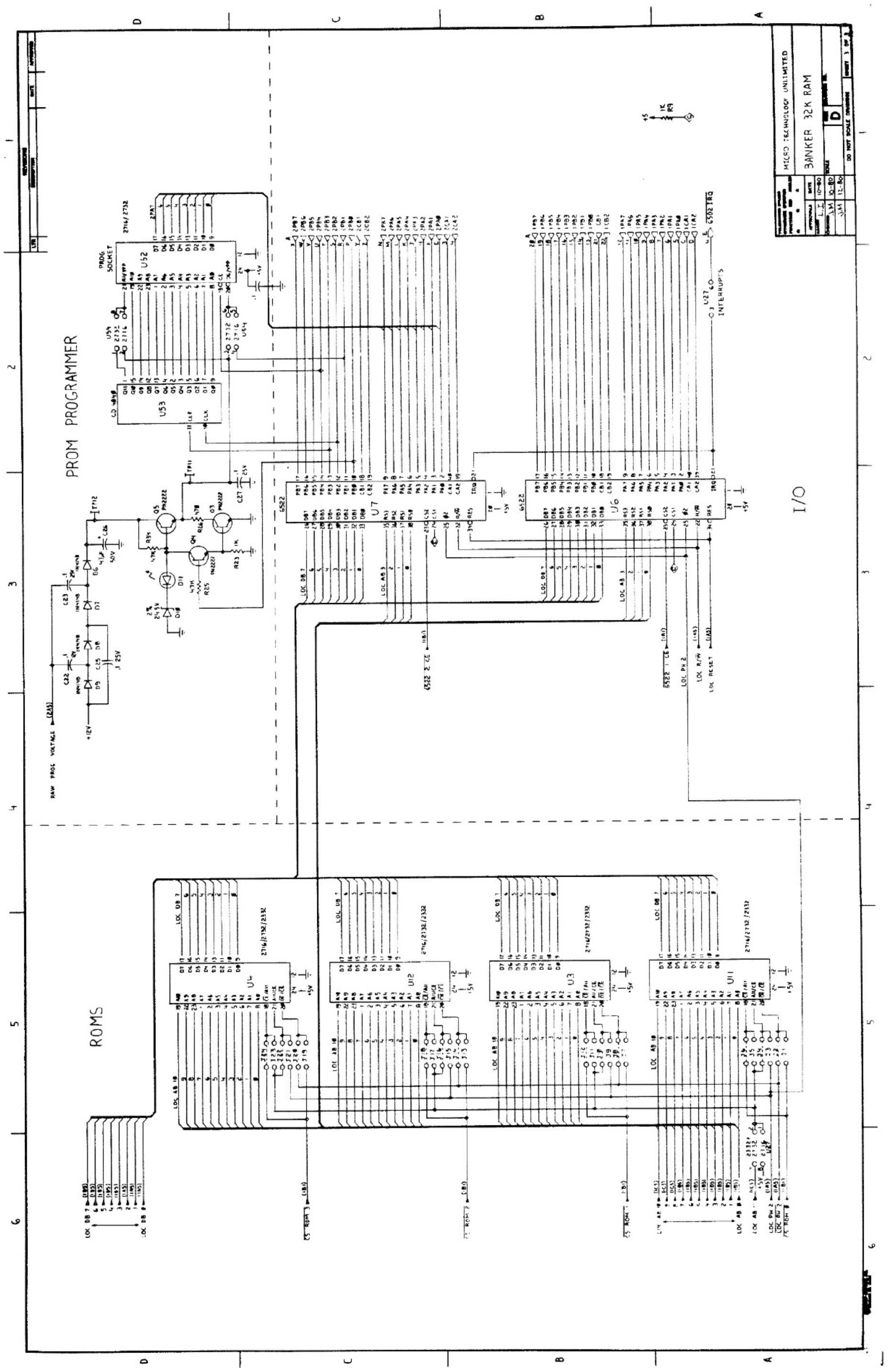
21. K-1032 BLOCK DIAGRAM



K-1032 SCHEMATIC DIAGRAM




TIMING GENERATOR
MEMORY ARRAY
CHARGE PUMP
REFRESH COUNTER ADDRESS MULTIPLEXOR
MICRO TECHNOLOGY UNLIMITED
BANKER 32K RAM
D


PROM PROGRAMMER
ROMS
I/O
MICRO TECHNOLOGY UNLIMITED
BANKER 32K RAM