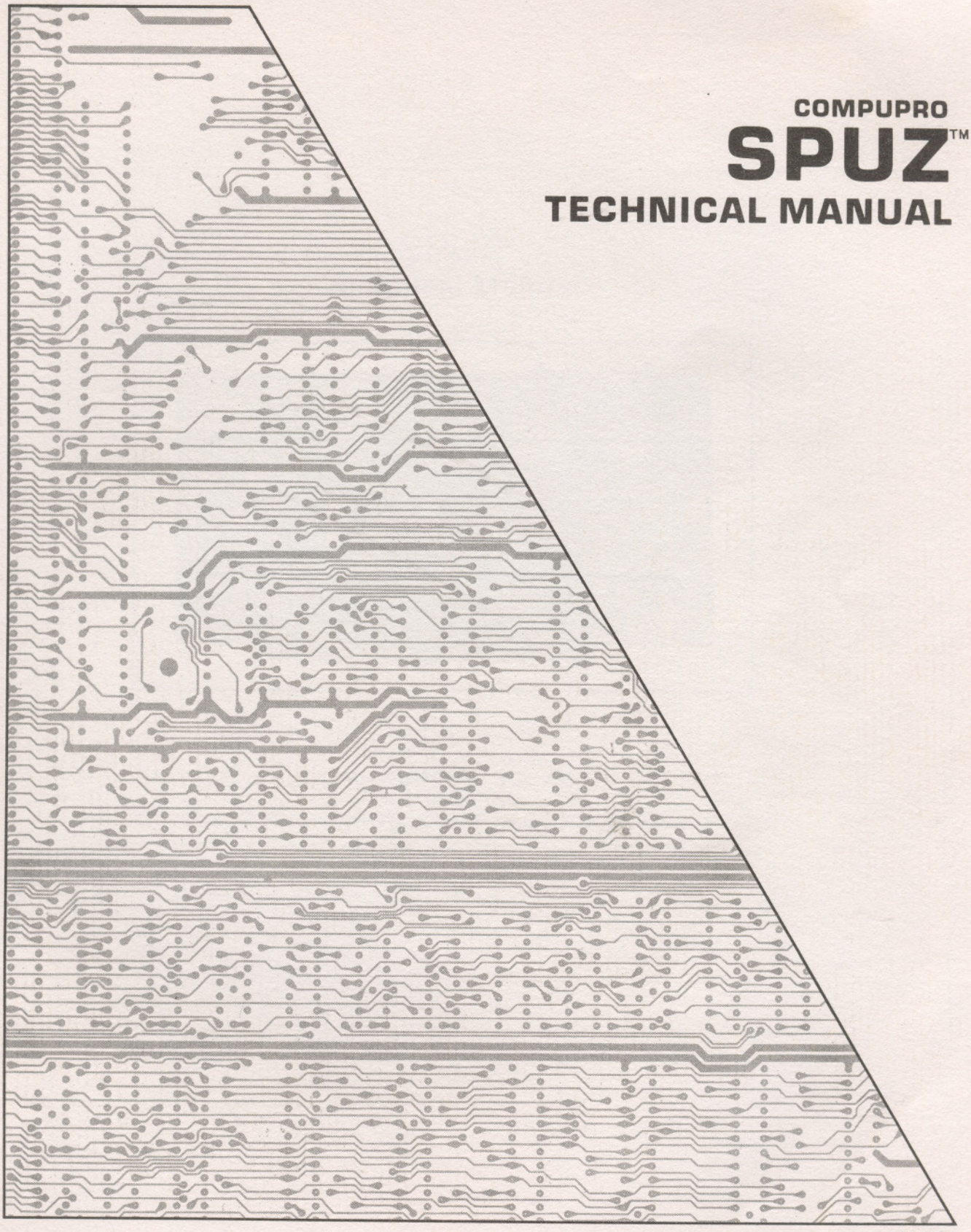
CompuPro



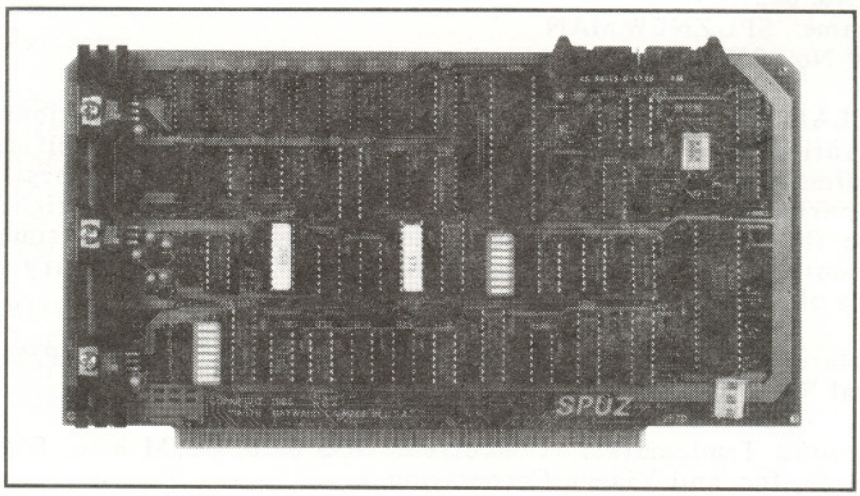
COMPUPRO

SPUZ™

TECHNICAL MANUAL

$20.00

SPUZ
Technical Manual



HIGH PERFORMANCE Z80
64 or 256K RAM
DUAL ASYNCHRONOUS COMMUNICATION CHANNELS

SPUZ TECHNICAL MANUAL

Hayward, CA 94545

First Edition: April, 1985
Latest Printing: July, 1985

Part No. 8261-0030
Filename: SPUZNEW.MAN
Board No: 257E

DISCLAIMER - Viasyn Corporation makes no representations or warranties with respect to the contents hereof and specifically disclaims any implied warranties of merchantablity or fitness for any particular purpose. Further, VIASYN reserves the right to revise this publication and to make changes from time to time in the content hereof without obligation of VIASYN to notify any person of such revision or changes.

CONTENTS

ABOUT THE SPUZ

VIASYN's SPUZ board combines an 8 MHz Z80H, 64K or 256K of memory and two fully bi-directional RS-232C serial I/O channels to perform a number of "slave processing" tasks.

Because each user has their own processor, memory and serial ports, the overhead on the host system is reduced when the slave is running an 8-bit task. When running 16- or 32-bit programs, the SPUZ can be used as a front end I/O processor adding 8-bit processing capability to your system (as with CompuPro 816/D1, E1, F1, G1 and 286 computers). Up to 16 SPUZs can be resident in the same mainframe.

In systems where high system throughput is necessary, this **multi-processing** configuration, in which host and slaves may run 8- and 16-bit software simultaneously, is faster and more efficient than dated dual processing systems where one of the processors is stopped while the other is running.

The SPUZ meets all IEEE 696/S-100 bus specifications and includes the following features:

- Fast 8MHz Z80H for high throughput.
- 64K or 256K bytes of RAM. Up to 16 SPUZ boards occupy only 64K of system memory.
- Dual bi-directional RS-232C serial ports.
- Fully software selectable serial port parameters.
- Supported by CompuPro's Concurrent DOS™ 8-16™ multi-user, multi-tasking operating system.
- True multi-processing instead of dual processing.
- Ideal as an 8-bit node in a "processor-per-user" system.
- Adds 8-bit capability to 16- or 32-bit systems.

HOW TO GET YOUR SPUZ UP AND RUNNING WITHOUT READING MUCH OF THE MANUAL

This section is for those of you who are so anxious to see the SPUZ running that you don't want to read the manual. This section will tell you how to set up the SPUZ board so that it can run under Concurrent DOS 8-16 in your system.

STEP 1. UNPACK THE SPUZ BOARD.

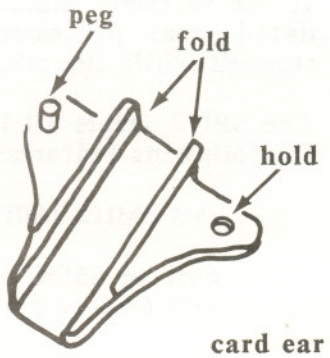
Along with the board, you will find an extra jumper shunt and two card ears in the plastic bag.

STEP 2. INSTALL CARD EARS.

a) Hold the board so the component side is toward you. (See diagram below.)

b) Insert the peg on the card ear into the hole in the right corner of the board. Fold the ear over the board's edge until the ear's hole snaps over the peg (make sure the long edge of the ear is along the top edge of the board extending toward the center.)

c) Repeat for left ear.



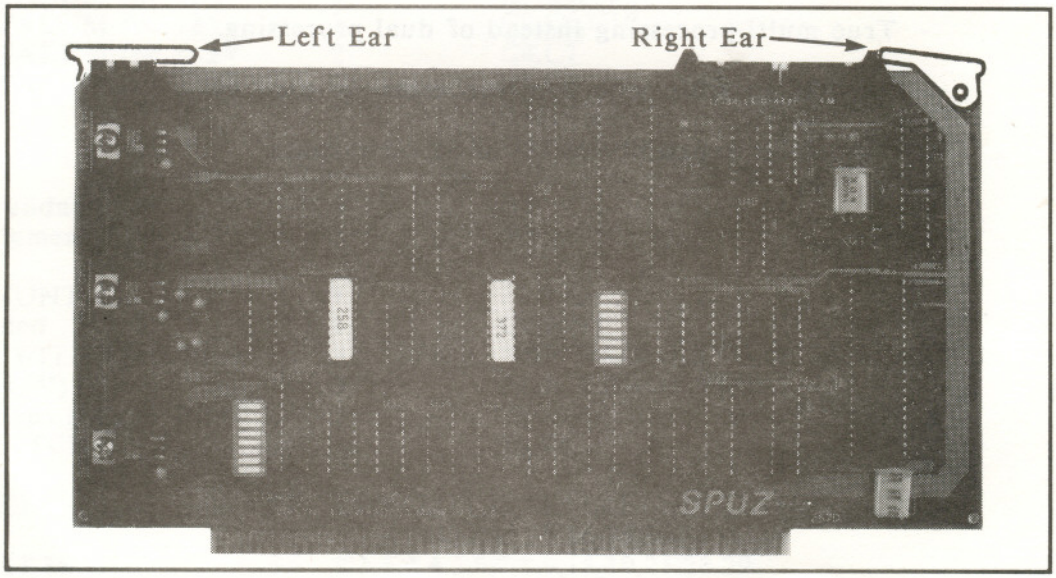


Figure 1. SPUZ (Component Side)

STEP 3. SET SWITCHES. Set the SPUZ switches according to the information in the Hardware Section. (See figure 1 for the location of the switches.)

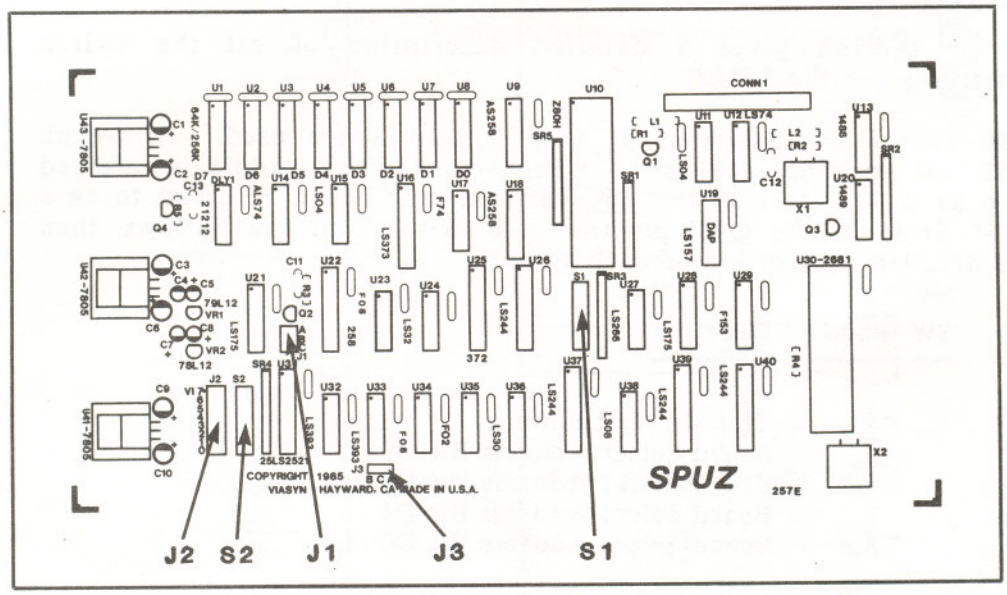
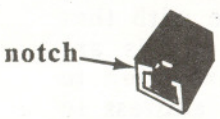


Figure 2. SPUZ (jumper and switch location)

STEP 4. SET JUMPER SHUNT CONNECTIONS

Install the jumper shunts according to the information in the Hardware Section. See figure 2 for the location of jumper connectors.

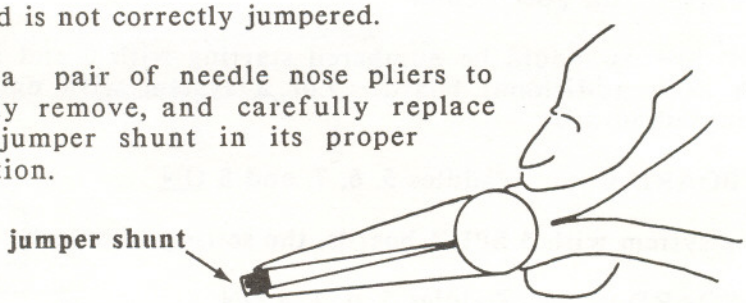
JUMPER SHUNTS



A jumper shunt is a small plastic part used to connect two pins on the jumper connector. Jumper shunts should be installed notch side up.

IF: The board is not correctly jumpered.

THEN: Use a pair of needle nose pliers to gently remove, and carefully replace the jumper shunt in its proper location.



INSERT THE SPUZ INTO THE MOTHERBOARD.

Place the board into a slot close to the back of the enclosure. The edge connector is offset, so that the SPUZ will fit only with the component side facing the front panel of the enclosure. **PUSH DOWN GENTLY UNTIL THE BOARD IS FIRMLY INSTALLED.**

SWITCHES

This section gives a detailed description of all the switch settings for the SPUZ.

In switch S1, positions 4, 5, 6, 7, and 8 are used. In switch S2, all positions are used. Whenever a switch paddle is referred to as a "0", it is in the ON position, and when referred to as a "1", it is in the OFF position. A switch overview follows, then a detailed description of each switch.

SW	Paddle	Function
1	4	Z80 M1 Wait State
	5	Board Select Address Bit D3
	6	Board Select Address Bit D2
	7	Board Select Address Bit D1
	8	Board Select Address Bit D0
2	1-8	Memory Address Bits A23-A16

Switch 1 (SW1) - The function of switch 1 positions 5, 6, 7, and 8 is to set the 4-bit **Select Address** of this board. Up to 16 SPUZ boards can be in a single system, with their memory all addressed at the same 64K page in system memory space, and their 18 I/O ports all overlapping at FC30h-FC41h. Each one must have a different select address. To choose the board that you wish to communicate with, write its select address in the lower nibble to the SLAVESEL port FC40h. You can now communicate with that board's DUART and memory, and the other SPUZ boards are invisible. It is important to always start your communication to a particular SPUZ with a write of the correct select address to the SLAVESEL port FC40h.

SPUZ boards should be numbered starting with 0 and incrementing with each additional board. For a system with one board, the settings should be:

BOARD 0 -- Paddles 5, 6, 7, and 8 ON

For a system with 5 SPUZ boards, the settings should be:

BOARD 0 -- Paddles 5, 6, 7, 8 ON
BOARD 1 -- Paddles 5, 6, 7 ON, paddle 8 OFF
BOARD 2 -- Paddles 5, 6, 8 ON, paddle 7 OFF
BOARD 3 -- Paddles 5, 6 ON, paddles 7, 8 OFF
BOARD 4 -- Paddles 5, 7, 8 ON, paddle 6 OFF

Finally, SW1 position 4 controls the M1 wait states to the Z80

processor. This switch must be ON when the Z80 is run at 6MHz and above. For 5MHz and below, this switch can be turned off. Since all standard SPUZ boards shipped by VIASYN are 8MHz, leave this switch on. Recommended setting:

Z80 M1 Wait -- SW1, paddle 4 ON

Switch 2 (SW2) - Switch 2 determines where the 64K page window of SPUZ memory is to appear to the system host and DMA devices. It can appear in any one of 256 pages of memory to CPUs and DMA devices with 24 bits of addressing, and in any one of 16 pages to CPUs and DMA devices with 20 bits of addressing. Paddle 1 is the most significant bit and corresponds with A23, while paddle 8 is the least significant bit and corresponds with A16. For standard CompuPro systems, all SPUZ boards in a mainframe appear in one page. SPUZ boards with 256K of memory also appear as 64K blocks, just the same as a 64K board, and they can be mixed in a system.

The CompuPro standard address is page 0Eh. For the CompuPro standard, all SPUZ boards in a system should be set to:

SW2 - MEMORY ADDRESS -- Paddles 1, 2, 3, 4, 8 ON,
Paddles 5,6, 7 OFF.

JUMPERS

This section gives a detailed description of all the jumper settings for the SPUZ.

Jumper J1 is a 6 pin jumper. A shunt can be placed horizontally across A (top), B (middle), C (bottom), or be left as shipped with a closed connection on the solder side across position B. Jumper J2 is a 16 pin jumper. A shunt should be placed horizontally at one of the 8 positions. J2 is shipped with pins installed and a shunt placed across VI0. Jumper J3 is a three pin jumper. A shunt can be placed across A-C (right), B-C (left), or be left as shipped with a closed connection on the solder side across A-C. A jumper overview is next with a detailed explanation of each jumper following.

J	Position	Function
1	A	One S-100 Bus Wait State
	B	Two S-100 Bus Wait States (as shipped)
	C	Three S-100 Bus Wait States
2	VI7	Z80 Interrupt Output to VI7*
	VI6	Z80 Interrupt Output to VI6*
	VI5	Z80 Interrupt Output to VI5*
	VI4	Z80 Interrupt Output to VI4*
	VI3	Z80 Interrupt Output to VI3*
	VI2	Z80 Interrupt Output to VI2*
	VI1	Z80 Interrupt Output to VI1*
	VI0	Z80 Interrupt Output to VI0* (as shipped)
3	A-C	Four Cycle Hold Off (as shipped)
	B-C	Eight Cycle Hold Off

Jumper J1 - This jumper controls how many wait states the SPUZ inserts into S-100 bus accesses to the memory or I/O ports on the SPUZ. It is a three position jumper. As the total time required to read or write the SPUZ memory is constant, the number of wait states is dependent only on the speed of the host processor. Position A is for one wait state and can be safely used only when the speed of the host processor is 6MHz or less. Position B is for two wait states, and should be used when the processor is 12MHz or less. Position C is for systems faster than 12MHz.

Jumper J1 is shipped with no pins, but with a closed connection on the solder side across position B. This should be left as it is shipped for installation in all CompuPro systems. If special circumstances require this jumper be changed, the trace connecting position B should be cut, pins installed, and a shunt placed in the desired position. Recommended position:

J1 -- Across position B (as shipped).

Jumper J2 - This jumper controls which Vectored Interrupt Line (VI) on the S-100 bus can be asserted by the SPUZ. It is an 8 position jumper, corresponding to 8 VI lines. The Z80 can assert an interrupt to the host by writing a bit pattern to a particular port. The host can clear the interrupt by writing a different bit pattern to a particular port. Any one Vectored Interrupt can be generated.

Jumper J2 is shipped with pins and a shunt across position VI0. VI0 is the CompuPro standard for all SPUZ boards. If it is ever desirable to change this, the shunt can be moved to any position. Recommended position:

J2 -- Across position VI0 (as shipped).

Jumper J3 - Jumper J3 controls the bus acknowledge hold off, after access of the Z80's bus. When the host or a DMA device accesses the SPUZ memory or I/O ports, the Z80 is issued a BUSREQ, and sometime later it gives up its internal bus for an external cycle with a BUSACK. If this process took place every time the host or DMA device wanted to access the SPUZ memory, the requestor would waste substantial time in a wait state waiting for the Z80 to issue BUSACK. For this reason, the Z80 is kept in a BUSACK state for a certain time after the last time it was accessed. Thus, when rapid access of the SPUZ memory is occuring (such as when the system is filling SPUZ memory with a program to run), it can happen with a minimum of wait time. Increasing the hold off decreases the host or DMA device wait time when accessing the SPUZ, but reduces the processing throughput of the Z80, and vice-versa.

Our experience and measurements indicate that for most systems, a hold off of 4 bus cycles is a good compromise. If you feel that a larger hold off would be beneficial to your system (for example if your processor was just missing the hold off time in its routine to load programs into the SPUZ memory), the hold off can be increased to 8 bus cycles.

Jumper J3 is normally closed on the solder side of the board across A-C (right side). If it is desirable to increase the hold off to 8 bus cycles, the A-C jumper should be cut, pins installed, and a shunt placed across B-C. Recommended position:

J3 -- Across A-C (as shipped).

THEORY OF OPERATION

This section of the manual will explain, in general, how the circuitry on the SPUZ works. In the following discussions, it will be helpful to refer to the schematic diagrams contained in the Hardware Description section of this manual.

OVERVIEW

The SPUZ looks like 64K bytes of memory and 18 I/O ports to the S-100 host processor or temporary bus master (DMA device). It does not become a temporary master and is therefore restricted from accessing other user's memory spaces or system peripherals. The S-100 bus has complete access rights to all of the SPUZ's resources including all 64K/256K bytes of memory, both bi-directional serial ports and all internal control ports.

Inter-processor communication is accomplished through a bi-directional interrupt structure. Message passing is through the memory located on the SPUZ. DMA devices can transfer data directly to or from the SPUZ's memory.

There is no EPROM on the SPUZ. When the system is powered up, the on-board Z80 is reset until the S-100 host processor has put a program in the SPUZ's memory and allows the Z80 to run. The host processor can reset the SPUZ at any time.

THE CIRCUITRY

A 16MHz processor-speed-times-two clock is generated by two sections of inverter U11, inductor L2, resistor R2, capacitor C12, and crystal X1. This clock is fed to the divide by two flip flop U12 to generate a symmetrical 8MHz clock. The signal CLK8* is then fed to the booster network consisting of two sections of inverter U11, inductor L1, resistor R1, and transistor Q1. This booster network generates the large swing, fast rise/fall time clock required by the Z80. The CLK8 signal is also used by flip flop U12 to generate SYNCACK*. SYNCACK* is the BUSACK* signal delayed by a portion of a clock width. It is used to start an external access to the memory or I/O after the S-100 addresses driven onto the internal bus by BUSACK* have settled.

The Z80 needs one M1 wait state for processor speeds over 5MHz to guarantee that the DRAM will have data ready. The circuit consisting of OR gate U24 generates WAIT* to the Z80 during M1 cycles. Most dynamic RAMs (including the 256K parts) require 256 refresh addresses instead of the 128 put out by the Z80. The circuit of inverter U15, AND gate U34, OR gate U24, and counter U32 counts the number of refresh cycles, toggles ZA7 every 128 cycles, and asserts toggled ZA7 when RFSH* is asserted. Thus, the refresh address is counted from 0 to 256. Non-refresh cycles (when RFSH* is high) pass A7 from the Z80 to ZA7 directly.

The 2681 Dual Asynchronous Receiver/Transmitter (DUART) is I/O mapped into 16 ports of the Z80's I/O space. They are ports 0h to Fh, and are repeated, i.e. port 10h equals port 0h. A description of the ports is contained in the reprint in this manual of the 2681 data sheet.

The data bus is buffered by U39 or U40 depending on which direction the transfer is taking place. DIEN* is asserted on pDBIN when the proper status is met, and DOEN* is asserted on BUSACK* when the proper status is met. Part of the DO bus is buffered by U38 to go to the above buffer, the board select comparator (U27), and the parallel port PTLATCH (U28).

The S-100 address bus is buffered onto the internal address bus on BUSACK* by U26 and U37. PAL U25 (p372) along with AND gate U36, and NOR gate U35 decode the S-100 I/O ports from address FC30h to FC41h. Ports FC40h and FC41h are not mapped on to the internal SPUZ bus, and thus can be written any time regardless of the internal state of the Z80. Ports FC30h to FC3Fh are mapped on to the SPUZ bus, and thus the SPUZ must not be reset when these ports (or SPUZ memory) are accessed or the S-100 bus will hang. The reason for this is that host accesses to the SPUZ internal ports and memory must be arbitrated by the Z80, and if the Z80 is reset when these ports are requested, the Z80 can never issue a BUSACK* and allow the cycle to take place. Thus, the SPUZ will assert pRDY to wait for the Z80 to give a BUSACK* (that it will never give if it is reset), and the S-100 system will hang forever in a wait state.

Signal SLAVESEL* (write port FC40h) will be generated on every SPUZ board in the system when port FC40h is written. Only the one with switch SW1 positions 5, 6, 7, and 8, matching what is on the lower four bits of the DO bus will leave VPROC asserted following a write to the SLAVESEL port. Signal PTLATCH* (write port FC41h) will be generated only on the board with VPROC asserted when port FC41h is written. Through this port, the host controls ZRST* to the Z80, VDMA*, and PB0-PB1, the control for which bank of 64K out of 256K the S-100 bus host accesses. Finally, when VPROC is asserted on either an I/O read or write to ports FC30h to FC3Fh, INTI/O* is asserted, at which time the DUART is accessed. The S-100 port map is as follows:

Port	Function	
FC30h to FC3Fh	DUART	- Maps to ports 0h to Fh of the DUART. For Serial Channels and output bit manipulation. (Read/Write)
FC40h	SLAVESEL*	- selects which board within a bank of 16 is selected. (Write Only)
FC41h	PTLATCH*	- controls RESET, which board is the valid DMA board, and which 64K bank the S-100 bus accesses. (Write Only)

When the extended address (A16-A23) matches what is set on switch S2 and the signal VBD* is asserted, comparator U33 asserts VMA*. The system can access SPUZ internal memory when VMA* is low, the proper status is met on sMEMR, sWO* and sOUT, and when pDBIN or pWR* is asserted. Thus to access a memory location on the SPUZ (when it is running or halted but not reset), first write the proper data pattern to SLAVESEL port FC40h to clock in VPROC on the trailing edge of pWR*, then access the 64K SPUZ memory window.

The port PTLATCH controls ZRST*, VDMA*, PB0, and PB1. Only the host can access this port made from flip flop U28. The bit map of this port is as follows:

Bit		Function (Write Only)
D3	=	ZRST* - Write a one to allow the Z80 to run. Write a zero to reset the Z80.
D2	=	VDMA* - Write a one to set DMA board. - Write a zero to turn off DMA board.
D1	=	Processor Bank 1 (PB1) - High bit of S-100 bank.
D0	=	Processor Bank 0 (PB0) - Low bit of S-100 bank.

On SPUZ boards with 256K of DRAM, the Z80 can be running out of one of four 64K pages and the S-100 bus can simultaneously access any 64K page. This makes it ideal in multi-user or multi-tasking environments, as the Z80 could be running one program while the host is fetching and loading another program. Furthermore, the host can set up a DMA device to load directly to or from any SPUZ memory page, and simultaneoulsy access any other SPUZ page. Thus, the Z80('s) can be running from one page, the host reading and writing a different page, and a DMA device directly accessing still another page. The pages are controlled by bits PB0 and PB1 (Processor Bank 0 and 1), DB0 and DB1 (DMA Bank 0 and 1), and ZB0 and ZB1 (Z80 Bank 0 and 1). When 64K of memory is installed, these bits don't matter. Thus, when the host is accessing the memory, bits PB0 and PB1 from PTLATCH U28 control which 64K page the host sees. When a DMA device is accessing the memory (pHLDA is asserted), DB0 and DB1 from the 2681 OP2 and OP3 control which 64K page the DMA device sees. Finally, when the Z80 is accessing the memory, ZB0 and ZB1 from the 2681 OP4 and OP5 control which 64K page the Z80 sees.

Just as a write to port FC40h (SLAVESEL) selects which SPUZ board of a possible 16 the host (not DMA) will access, setting the VDMA* signal on one SPUZ selects which board of a possible 16 when a DMA device accesses the SPUZ resources. When a host decides that DMA is to take place to a paticular memory page in a particular SPUZ, the host first selects the particular board via SLAVESEL. Second, the host asserts the VDMA* signal by writing a 04h (OR'ed with the proper ZRST, PB1, and PB0 bits) to the

PTLATCH on the selected board. Third, the host writes to the 2681 on the selected board to choose the proper DB0 and DB1. Now, any DMA to the memory window of the SPUZ(s) will access the properly selected board and the selected page, regardless of further host writes to the SLAVESEL port FC40h.

Warning: Only one SPUZ can have VDMA* asserted at a time. The host MUST turn off the previous VDMA* before asserting a new one. Permanent damage to SPUZ boards may result if more than one board has VDMA* asserted and DMA occurs to the SPUZ window page. Care must also be taken when DMA devices with real time requirements access the SPUZ as it is not guaranteed to give up its internal bus in any specified amount of time. Furthermore, it will lock out the bus for several microseconds every eight consecutive accesses in order to run a refresh cycle.

Once the bits PB0 and PB1, DB0 and DB1, and ZB0 and ZB1 have been selected, multiplexers U29 and U19 control which bits get to the DRAM on what kind of cycle. Depending on the status of BUSACK* from the Z80 and pHLDA from the S-100, two of these bits will be chosen to produce A8 on RAS and A8 on CAS. If BUSACK* is not asserted, ZB0 and ZB1 are used. If BUSACK* is asserted and pHLDA is not, PB0 and PB1 are used. If BUSACK* and pHLDA are asserted, DB0 and DB1 are used. Multiplexer U19 also selects whether VPROC* or VDMA* produces VBD*. If pHLDA is asserted, VDMA* is used; if pHLDA is not asserted, VPROC* is used. This allows the host processor to access a completely different SPUZ than the DMA device is accessing.

The 2681 DUART handles: 1)the serial receive/transmit function, 2)several output bits necessary on the board, 3)several input bits, and 4)the timer/counter. The 16 ports of the DUART control these functions and are defined in the reprint of the 2681 data sheet included in this manual. The output bits of the 2681 are defined in the following table. All the bits in the OPR are low on reset, and the remarks in parentheses tell what this means to the circuitry.

Bit	Function	
OP7	ZINT*	OP7 and INTRN of the 2681 are tied together to Z80 ZINT*. (interrupt not asserted to Z80).
OP6	S100 VI	S-100 Vectored Interrupt output. (interrupt asserted to S-100 bus).
OP5 OP4	ZB1 ZB0	Controls which 64K bank Z80 sees when 256K parts are installed. (Z80 starts running from bank 3 - high bank).
OP3 OP2	DB1 DB0	Controls which 64K bank DMA sees when 256K parts installed on valid DMA board. (page 3 accessed on reset - high bank).
OP1	RTSB	Request to Send on B channel of serial port. (inhibits terminal or other device from transmitting to DUART).
OP0	RTSA	Request to Send on A channel of serial port. (inhibits terminal or other device from transmitting to DUART).

The following table defines the input bits of the 2681.

Bit	Function	
IP6	HALT*	From HALT* output pin on the Z80. Used to monitor when Z80 is in HALT state.
IP5 IP4	SB1 SB0	From bits selecting which page the host accesses or last accessed.
IP3 IP2	OP3 OP2	Not used and not connected. Not used and not connected.
IP1	CTSB	Clear to Send on channel B. Low when terminal or other device is ready to take characters.
IP0	CTSA	Clear to Send on channel A. Low when terminal or other device is ready to take characters.

S-100 bus access to SPUZ resources is controlled by PAL U22 (part p258). It controls the necessary internal status and strobes to run an internal bus cycle. It generates pRDY through capacitor C11 and resistor R5 when the proper status is met, and does not end pRDY until ENDWAIT is generated by counter U21. Thus the chain of events is:

1) Host or DMA requests resource (memory or I/O port),
2) SPUZ puts host or DMA device in wait state and asserts BREQ* to the Z80,
3) Z80 issues BUSACK* which starts counter U21, and puts data and address on the internal bus,
4) SYNCACK* is asserted which asserts MRQ*, IORQ*, RD*, or WR* depending on input status by PAL U22,
5) DRAM or DUART cycle is performed, and
6) ENDWAIT is asserted which terminates pRDY.

Half of counter U33 controls the hold off which keeps the Z80 in hold by asserting BREQ* for a certain number of strobes after the last access. The other half of the counter insures that the host does not keep the Z80 in hold too long. If this happened, refresh of the DRAM could be lost. Therefore, the host or DMA device can run no more than 8 cycles on the internal bus before the SPUZ forces the host or DMA device to wait and runs a cycle of its own. In this way, refresh will be maintained. If it is ever necessary to defeat this feature, U24 pin 8 can be removed from the socket, and a small solder bridge can be made from pin 8 to pin 9.

The 2681 interface is straightforward. When the host or the Z80 asserts IORQ* and either WR* or RD*, the 2681 will either accept data or present data to the bus ZD7-ZD0. The address ZA3-ZA0 controls what section of the 2681 is accessed. The port map for the 2681 can be found in the reprint of the 2681 data sheet. The time base for the 2681's baud rate generator and counter/timer is the 3.6864 MHz crystal X2. The 2681 is reset whenever the Z80 is reset as ZRST goes to both chips. Finally, the OP7 bit and INTRN are both able to generate ZINT* to the Z80, enabling the Z80 to be interrupted by the host, by both serial channels, or by the timer/counter.

The Dynamic RAM circuit provides liberal timing to the RAM chips, while maintaining high speed. The MRQ* signal's falling edge provides the clock to U17 to start the cycle. The delay line DLY1 then controls the timing, asynchronously from the 8MHz clock. When the data is ready on a read cycle, it is latched in U16 to be read by the Z80 or host. On a write, data flows off the ZD0-ZD7 lines into the DRAM when W* is asserted.

Either eight 64K parts or eight 256K parts can be used in the SPUZ. The 64K board has multiplexer U29 missing. If you decide to upgrade your 64K board to 256K, a factory upgrade can be obtained from VIASYN. We will install the 256K parts, multiplexer U29, and run a complete confidence test.

THE DUART

Part of Signetics' data sheet for the SC2681 Dual Asynchronous Receiver/Transmitter (DUART) is reprinted here with permission. The complete data sheet can be found in the Signetics MOS Microprocessor Data Manual for 1982 or later, and can be obtained by writing or calling Signetics, 811 East Arques Avenue, P.O. Box 409, Sunnyvale, California 94086, (408)739-7700.

DUAL ASYNCHRONOUS RECEIVER/TRANSMITTER (DUART) SCN2681 SERIES

Preliminary

DESCRIPTION

The Signetics SCN2681 Dual Universal Asynchronous Receiver/Transmitter (DUART) is a single chip MOS-LSI communications device that provides two independent full-duplex asynchronous receiver/transmitter channels in a single package. It interfaces directly with microprocessors and may be used in a polled or interrupt driven system.

The operating mode and data format of each channel can be programmed independently. Additionally, each receiver and transmitter can select its operating speed as one of eighteen fixed baud rates, a 16x clock derived from a programmable counter/timer, or an external 1x or 16x clock. The baud rate generator and counter/timer can operate directly from a crystal or from external clock inputs. The ability to independently program the operating speed of the receiver and transmitter make the DUART particularly attractive for dual-speed channel applications such as clustered terminal systems.

Each receiver is quadruply buffered to minimize the potential of receiver overrun or to reduce interrupt overhead in interrupt driven systems. In addition, a flow control capability is provided to disable a remote DUART transmitter when the buffer of the receiving device is full.

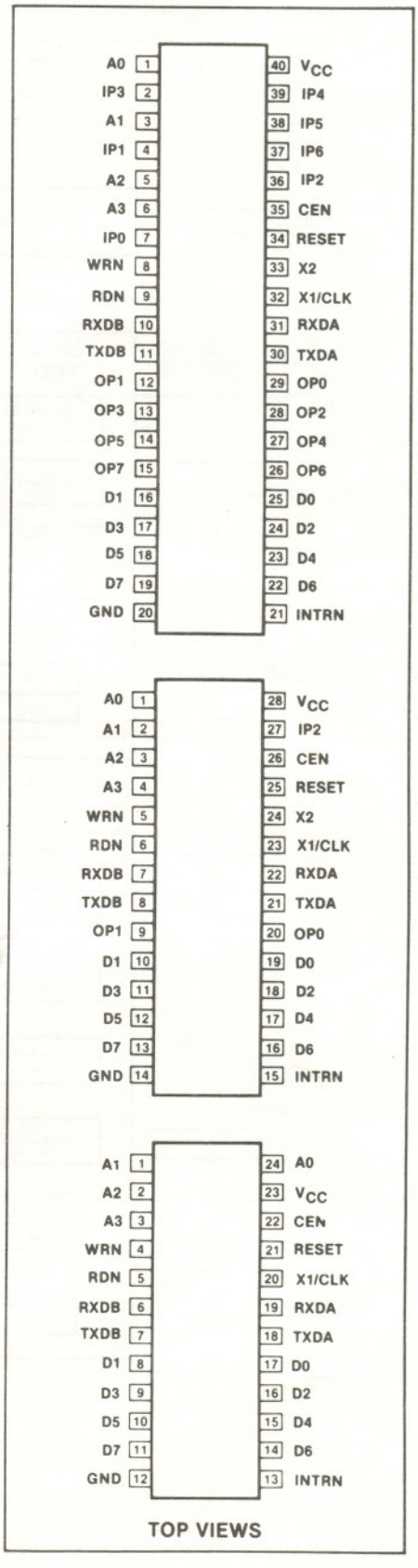
Also provided on the SCN2681 are a multipurpose 7-bit input port and a multipurpose 8-bit output port. These can be used as general purpose I/O ports or can be assigned specific functions (such as clock inputs or status/interrupt outputs) under program control.

The SCN2681 is available in three package versions to satisfy various system requirements: 40-pin and 28-pin, both 0.6″ wide DIPs, and a compact 24-pin, 0.4″ wide, DIP.

FEATURES

- **Dual full-duplex asynchronous receiver/transmiter**
- **Quadruple buffered receiver data registers**
- **Programmable data format**
  - **—5 to 8 data bits plus parity**
  - **—Odd, even, no parity or force parity**
  - **—1, 1.5 or 2 stop bits programmable in 1/16 bit increments**
- **Programmable baud rate for each receiver and transmiter selectable from:**
  - **—18 fixed rates: 50 to 38.4K baud**
  - **—One user defined rate derived from programmable timer/counter**
  - **—External 1x or 16x clock**
- **Parity, framing, and overrun error detection**
- **False start bit detection**
- **Line break detection and generation**
- **Programmable channel mode**
  - **—Normal (full duplex)**
  - **—Automatic echo**
  - **—Local loopback**
  - **—Remote loopback**
- **Multi-function programmable 16-bit counter/timer**
- **Multi-function 7-bit input port**
  - **—Can serve as clock or control inputs**
  - **—Change of state detection on four inputs**
- **Multi-function 8-bit output port**
  - **—Individual bit set/reset capability**
  - **—Outputs can be programmed to be status/interrupt signals**
- **Versatile interrupt system**
  - **—Single interrupt output with eight maskable interrupting conditions**
  - **—Output port can be configured to provide a total of up to six separate wire-OR'able interrupt outputs**
- **Maximum data transfer: 1X — 1MB/sec, 16X — 125KB/sec**
- **Automatic wake-up mode for multidrop applications**
- **Start-end break interrupt/status**
- **Detects break which originates in the middle of a character**
- **On-chip crystal oscillator**
- **TTL compatible**
- **Single +5V power supply**

PIN CONFIGURATION



TOP VIEWS

ORDERING CODE

PACKAGES	V_{CC} = 5V ±5%, T_A = 0°C to 70°C		
	24 Pin[1]	28 Pin[2]	40 Pin[2]
Ceramic DIP	Not available	SCN2681AC1I28	SCN2681AC1I40
Plastic DIP	SCN2681AC1N24	SCN2681AC1N28	SCN2681AC1N40

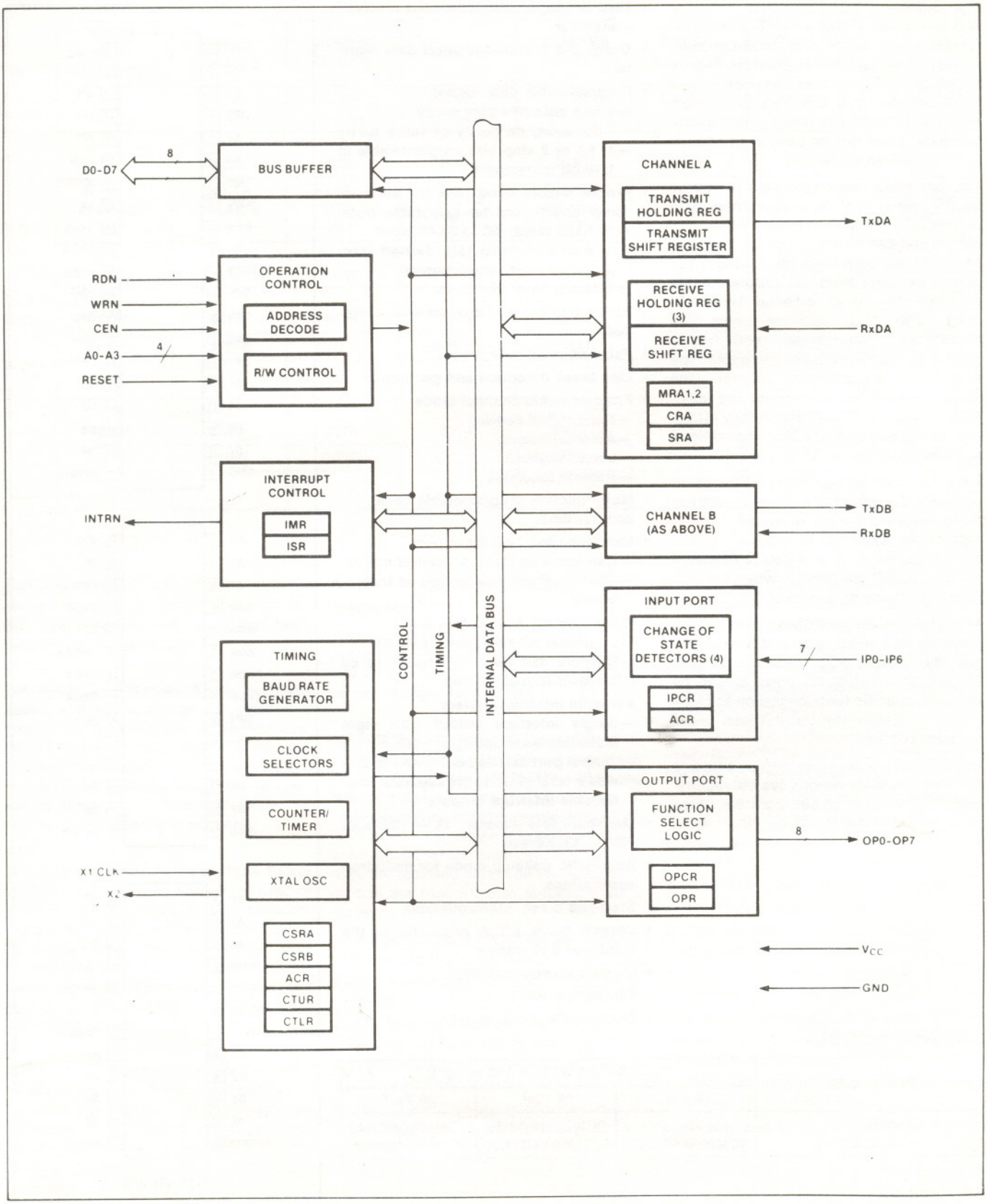
[1] 400 mil wide DIP

[2] 600 mil wide DIP

DUAL ASYNCHRONOUS RECEIVER/TRANSMITTER (DUART) SCN2681 SERIES

Preliminary

BLOCK DIAGRAM



DUAL ASYNCHRONOUS RECEIVER/TRANSMITTER (DUART) SCN2681 SERIES

Preliminary

PIN DESIGNATION

MNEMONIC	APPLICABLE			TYPE	NAME AND FUNCTION
	40	28	24		
D0-D7	X	X	X	I/O	**Data Bus:** Bidirectional 3-state data bus used to transfer commands, data and status between the DUART and the CPU. D0 is the least significant bit.
CEN	X	X	X	I	**Chip Enable:** Active low input signal. When low, data transfers between the CPU and the DUART are enabled on D0-D7 as controlled by the WRN, RDN and A0-A3 inputs. When high, places the D0-D7 lines in the 3-state condition.
WRN	X	X	X	I	**Write Strobe:** When low and CEN is also low, the contents of the data bus is loaded into the addressed register. The transfer occurs on the rising edge of the signal.
RDN	X	X	X	I	**Read Strobe:** When low and CEN is also low, causes the contents of the addressed register to be presented on the data bus. The read cycle begins on the falling edge of RDN.
A0-A3	X	X	X	I	**Address Inputs:** Select the DUART internal registers and ports for read/write operations.
RESET	X	X	X	I	**Reset:** A high level clears internal registers (SRA, SRB, IMR, ISR, OPR, OPCR), puts OP0-OP7 in the high state, stops the counter/timer, and puts channels A and B in the inactive state, with the TxDA and TxDB outputs in the mark (high) state.
INTRN	X	X	X	O	**Interrupt Request:** Active low, open drain, output which signals the CPU that one or more of the eight maskable interrupting conditions are true.
X1/CLK	X	X	X	I	**Crystal 1:** Crystal or external clock input. A crystal or clock of the specified limits must be supplied at all times. When a crystal is used, a capacitor must be connected from this pin to ground (see figure 5).
X2	X	X		O	**Crystal 2:** Connection for other side of the crystal. Should be connected to ground if a crystal is not used. When a crystal is used, a capacitor must be connected from this pin to ground (see figure 5).
RxDA	X	X	X	I	**Channel A Receiver Serial Data Input:** The least significant bit is received first. 'Mark' is high, 'space' is low.
RxDB	X	X	X	I	**Channel B Receiver Serial Data Input:** The least significant bit is received first. 'Mark' is high, 'space' is low.
TxDA	X	X	X	O	**Channel A Transmitter Serial Data Output:** The least significant bit is transmitted first. This output is held in the 'mark' condition when the transmitter is disabled, idle, or when operating in local loopback mode. 'Mark' is high, 'space' is low.
TxDB	X	X	X	O	**Channel B Transmitter Serial Data Output:** The least significant bit is transmitted first. This output is held in the 'mark' condition when the transmitter is disabled, idle, or when operating in local loopback mode. 'Mark' is high, 'space' is low.
OP0	X	X		O	**Output 0:** General purpose output, or channel A request to send (RTSAN, active low). Can be deactivated on receive or transmit.
OP1	X	X		O	**Output 1:** General purpose output, or channel B request to send (RTSBN, active low). Can be deactivated on receive or transmit.
OP2	X			O	**Output 2:** General purpose output, or channel A transmitter 1X or 16X clock output, or channel A receiver 1X clock output.
OP3	X			O	**Output 3:** General purpose output, or open drain, active low counter/timer output, or channel B transmitter 1X clock output, or channel B receiver 1X clock output.
OP4	X			O	**Output 4:** General purpose output, or channel A open drain, active low, RxRDYA/FFULLA output.
OP5	X			O	**Output 5:** General purpose output, or channel B open drain, active low, RxRDYB/FFULLB output.
OP6	X			O	**Output 6:** General purpose output, or channel A open drain, active low, TxRDYA output.
OP7	X			O	**Output 7:** General purpose output, or channel B open drain, active low, TxRDYB output.
IP0	X			I	**Input 0:** General purpose input, or channel A clear to send active low input (CTSAN).
IP1	X			I	**Input 1:** General purpose input, or channel B clear to send active low input (CTSBN).
IP2	X	X		I	**Input 2:** General purpose input, or counter/timer external clock input.
IP3	X			I	**Input 3:** General purpose input, or channel A transmitter external clock input (TxCA). When the external clock is used by the transmitter, the transmitted data is clocked on the falling edge of the clock.

DUAL ASYNCHRONOUS RECEIVER/TRANSMITTER (DUART) SCN2681 SERIES

Preliminary

PIN DESIGNATION (Continued)

MNEMONIC	APPLICABLE			TYPE	NAME AND FUNCTION
	40	28	24		
IP4	X			I	**Input 4:** General purpose input, or channel A receiver external clock input (RxCA). When the external clock is used by the receiver, the received data is sampled on the rising edge of the clock.
IP5	X			I	**Input 5:** General purpose input, or channel B transmitter external clock input (TxCB). When the external clock is used by the transmitter, the transmitted data is clocked on the falling edge of the clock.
IP6	X			I	**Input 6:** General purpose input or channel B receiver external clock input (RxCB). When the external clock is used by the receiver, the received data is sampled on the rising edge of the clock.
V_{CC}	X	X	X	I	**Power Supply:** + 5V supply input
GND	X	X	X	I	**Ground**

BLOCK DIAGRAM

The 2681 DUART consists of the following eight major sections: data bus buffer, operation control, interrupt control, timing, communications channels A and B, input port and output port. Refer to the block diagram.

Data Bus Buffer

The data bus buffer provides the interface between the external and internal data busses. It is controlled by the operation control block to allow read and write operations to take place between the controlling CPU and the DUART.

Operation Control

The operation control logic receives operation commands from the CPU and generates appropriate signals to internal sections to control device operation. It contains address decoding and read and write circuits to permit communications with the microprocessor via the data bus buffer.

Interrupt Control

A single active low interrupt output (INTRN) is provided which is activated upon the occurence of any of eight internal events. Associated with the interrupt system are the interrupt mask register (IMR) and the interrupt status register (ISR). The IMR may be programmed to select only certain conditions to cause INTRN to be asserted. The ISR can be read by the CPU to determine all currently active interrupting conditions.

Outputs OP3-OP7 can be programmed to provide discrete interrupt outputs for the transmitters, receivers, and counter/timer.

Timing Circuits

The timing block consists of a crystal oscillator, a baud rate generator, a programmable 16-bit counter/timer, and four clock selectors. The crystal oscillator operates directly from a 3.6864MHz crystal connected across the X1/CLK and X2 inputs. If an external clock of the appropriate frequency is available, it may be connected to X1/CLK. The clock serves as the basic timing reference for the baud rate generator (BRG), the counter/timer, and other internal circuits. A clock signal within the limits specified in the specifications section of this data sheet must always be supplied to the DUART.

The baud rate generator operates from the oscillator or external clock input and is capable of generating 18 commonly used data communications baud rates ranging from 50 to 38.4K baud. The clock outputs from the BRG are at 16X the actual baud rate. The counter/timer can be used as a timer to produce a 16X clock for any other baud rate by counting down the crystal clock or an external clock. The four clock selectors allow the independent selection, for each receiver and transmitter, of any of these baud rates or an external timing signal.

The counter/timer (C/T) can be programmed to use one of several timing sources as its input. The output of the C/T is available to the clock selectors and can also be programmed to be output at OP3. In the counter mode, the contents of the C/T can be read by the CPU and it can be stopped and started under program control. In the timer mode, the C/T acts as a programmable divider.

Communications Channels A and B

Each communications channel of the 2681 comprises a full duplex asynchronous receiver/transmitter (UART). The operating frequency for each receiver and transmitter can be selected independently from the baud rate generator, the counter timer, or from an external input.

The transmitter accepts parallel data from the CPU, converts it to a serial bit stream, inserts the appropriate start, stop, and optional parity bits and outputs a composite serial stream of data on the TxD output pin. The receiver accepts serial data on the RxD pin, converts this serial input to parallel format, checks for start bit, stop bit, parity bit (if any), or break condition and sends an assembled character to the CPU.

Input Port

The inputs to this unlatched 7-bit port can be read by the CPU by performing a read operation at address D_{16}. A high input results in a logic 1 while a low input results in a logic 0. D_7 will always be read as a logic 1. The pins of this port can also serve as auxiliary inputs to certain portions of the DUART logic.

Four change-of-state detectors are provided which are associated with inputs IP3, IP2, IP1, and IP0. A high-to-low or low-to-high transition of these inputs lasting longer than 25-50μs will set the corresponding bit in the input port will change register. The bits are cleared when the register is read by the CPU. Any change of state can also be programmed to generate an interrupt to the CPU.

Preliminary

Output Port

The 8-bit multi-purpose output port can be used as a general purpose output port, in which case the outputs are the complements of the output port register (OPR). OPR[n] = 1 results in OP[n] = low and vice-versa. Bits of the OPR can be individually set and reset. A bit is set by performing a write operation at address E_{16} with the accompanying data specifying the bits to be set (1 = set, 0 = no change). Likewise, a bit is reset by a write at address F_{16} with the accompanying data specifying the bits to be reset (1 = reset, 0 = no change).

Outputs can be also individually assigned specific functions by appropriate programming of the channel A mode registers (MR1A, MR2A), the channel B mode registers (MR1B, MR2B), and the output port configuration register (OPCR).

OPERATION

Transmitter

The 2681 is conditioned to transmit data when the transmitter is enabled through the command register. The 2681 indicates to the CPU that it is ready to accept a character by setting the TxRDY bit in the status register. This condition can be programmed to generate an interrupt request at OP6 or OP7 and INTRN. When a character is loaded into the transmit holding register (THR), the above conditions are negated. Data is transferred from the holding register to the transmit shift register when it is idle or has completed transmission of the previous character. The TxRDY conditions are then asserted again which means one full character time of buffering is provided. Characters cannot be loaded into the THR while the transmitter is disabled.

The transmitter converts the parallel data from the CPU to a serial bit stream on the TxD output pin. It automatically sends a start bit followed by the programmed number of data bits, an optional parity bit, and the programmed number of stop bits. The least significant bit is sent first. Following the transmission of the stop bits, if a new character is not available in the THR, the TxD output remains high and the TxEMT bit in the status register (SR) will be set to 1. Transmission resumes and the TxEMT bit is cleared when the CPU loads a new character into the THR. If the transmitter is disabled, it continues operating until the character currently being transmitted is completely sent out. The transmitter can be forced to send a continuous low condition by issuing a send break command.

The transmitter can be reset through a software command. If it is reset, operation ceases immediately and the transmitter must be enabled through the command register before resuming operation. If CTS operation is enabled, the CTSN input must be low in order for the character to be transmitted. if it goes high in the middle of a transmission, the character in the shift register is transmitted and TxDA then remains in the marking state until CTSN goes low. The transmitter can also control the deactivation of the RTSN output. If programmed, the RTSN output will be reset one bit time after the character in the transmit shift register and transmit holding register (if any) are completely transmitted, if the transmitter has been disabled.

Receiver

The 2681 is conditioned to receive data when enabled through the command register. The receiver looks for a high to low (mark to space) transition of the start bit on the RxD input pin. If a transition is detected, the state of the RxD pin is sampled each 16X clock for 7-1/2 clocks (16X clock mode) or at the next rising edge of the bit time clock (1X clock mode). If RxD is sampled high, the start bit is invalid and the search for a valid start bit begins again. If RxD is still low, a valid start bit is assumed and the receiver continues to sample the input at one bit time intervals at the theoretical center of the bit, until the proper number of data bits and the parity bit (if any) have been assembled, and one stop bit has been detected. The least sigificant bit is received first. The data is then transferred to the receive holding register (RHR) and the RxRDY bit in the SR is set to a 1. This condition can be programmed to generate an interrupt at OP4 or OP5 and INTRN. If the character length is less than eight bits, the most significant unused bits in the RHR are set to zero.

After the stop bit is detected, the receiver will immediately look for the next start bit. However, if a non-zero character was received without a stop bit (framing error) and RxD remains low for one half of the bit period after the stop bit was sampled, then the receiver operates as if a new start bit transition had been detected at that point (one-half bit time after the stop bit was sampled).

The parity error, framing error, overrun error and received break state (if any) are strobed into the SR at the received character boundary, before the RxRDY status bit is set. If a break condition is detected (RxD is low for the entire character including the stop bit), a character consisting of all zeros will be loaded into the RHR and the received break bit in the SR is set to 1. The RxD input must return to a high condition for at least one-half bit time before a search for the next start bit begins.

The RHR consists of a first-in-first-out (FIFO) stack with a capacity of three characters. Data is loaded from the receive shift register into the topmost empty position of the FIFO. The RxRDY bit in the status register is set whenever one or more characters are available to be read, and a FFULL status bit is set if all three stack positions are filled with data. Either of these bits can be selected to cause an interrupt. A read of the RHR outputs the data at the top of the FIFO. After the read cycle, the data FIFO and its associated status bits (see below) are 'popped' thus emptying a FIFO position for new data.

In addition to the data word, three status bits (parity error, framing error, and received break) are also appended to each data character in the FIFO (overrun is not). Status can be provided in two ways, as programmed by the error mode control bit in the mode register. In the 'character' mode, status is provided on a character-by-character basis: the status applies only to the character at the top of the FIFO. In the 'block' mode, the status provided in the SR for these three bits is the logical OR of the status for all characters coming to the top of the FIFO since the last 'reset error' command was issued. In either mode reading the SR does not affect the FIFO. The FIFO is 'popped' only when the RHR is read. Therefore the status register should be read prior to reading the FIFO.

If the FIFO is full when a new character is received, that character is held in the receive shift register until a FIFO position is available. If an additional character is received while this state exits, the contents of the FIFO are not affected: the character previously in the shift register is lost and the overrun error status bit (SR[4]) will be set upon receipt of the start bit of the new (overruning) character.

The receiver can control the deactivation of RTS. If programmed to operate in this mode, the RTSN output will be negated when a valid start bit was received and the FIFO is full. When a FIFO position becomes available, the RTSN output will be re-asserted automatically. This feature can be used to prevent an overrun, in the

receiver, by connecting the RTSN output to the CTSN input of the transmitting device.

If the receiver is disabled, the FIFO characters can be read. However, no additional characters can be received until the receiver is enabled again. If the receiver is reset, the FIFO and all of the receiver status, and the corresponding output ports and interrupt are reset. No additional characters can be received until the receiver is enabled again.

Multidrop Mode

The DUART is equipped with a wake up mode used for multidrop applications. This mode is selected by programming bits MR1A[4:3] or MR1B[4:3] to '11' for channels A and B respectively. In this mode of operation, a 'master' station transmits an address character followed by data characters for the addressed 'slave' station. The slave stations, with receivers that are normally disabled, examine the received data stream and 'wake-up' the CPU (by setting RxRDY) only upon receipt of an address character. The CPU compares the received address to its station address and enables the receiver if it wishes to receive the subsequent data characters. Upon receipt of another address character, the CPU may disable the receiver to initiate the process again.

A transmitted character consists of a start bit, the programmed number of data bits, an address/data (A/D) bit, and the programmed number of stop bits. The polarity of the transmitted A/D bit is selected by the CPU by programming bit MR1A[2]/MR1B[2]. MR1A[2]/MR1B[2] = 0 transmits a zero in the A/D bit position, which identifies the corresponding data bits as data, while MR1A[2]/MR1B[2] = 1 transmits a one in the A/D bit position, which identifies the corresponding data bits as an address. The CPU should program the mode register prior to loading the corresponding data bits into the THR.

In this mode, the receiver continuously looks at the received data stream, whether it is enabled or disabled. If disabled, it sets the RxRDY status bit and loads the character into the RHR FIFO if the received A/D bit is a one (address tag), but discards the received character if the received A/D bit is a zero (data tag). If enabled, all received characters are transferred to the CPU via the RHR. In either case, the data bits are loaded into the data FIFO while the A/D bit is loaded into the status FIFO position normally used for parity error (SRA[5] or SRB[5]). Framing error, overrun error, and break detect operate normally whether or not the receiver is enabled.

PROGRAMMING

The operation of the DUART is programmed by writing control words into the appropriate registers. Operational feedback is provided via status registers which can be read by the CPU. The addressing of the registers is described in table 1.

The contents of certain control registers are initialized to zero on RESET. Care should be exercised if the contents of a register are changed during operation, since certain changes may cause operational problems. For example, changing the number of bits per character while the transmitter is active may cause the transmission of an incorrect character. In general, the contents of the MR, the CSR, and the OPCR should only be changed while the receiver(s) and transmitter(s) are not enabled, and certain changes to the ACR should only be made while the C/T is stopped.

Mode registers 1 and 2 of each channel are accessed via independent auxiliary pointers. The pointer is set to MR1x by RESET or by issuing a 'reset pointer' command via the corresponding command register. Any read or write of the mode register while the pointer is at MR1x switches the pointer to MR2x. The pointer then remains at MR2x, so that subsequent accesses are always to MR2x unless the pointer is reset to MR1x as described above.

Mode, command, clock select, and status registers are duplicated for each channel to provide total independent operation and control. Refer to table 2 for register bit descriptions.

MR1A — Channel A Mode Register 1

MR1A is accessed when the channel A MR pointer points to MR1. The pointer is set to MR1 by RESET or by a 'set pointer' command applied via CRA. After reading or writing MR1A, the pointer will point to MR2A.

MR1A[7] — Channel A Receiver Request-to-Send Control — This bit controls the deactivation of the RTSAN output (OP0) by the receiver. This output is normally asserted by setting OPR[0] and negated by resetting OPR[0]. MR1A[7] = 1 causes RTSAN to be negated upon receipt of a valid start bit if the channel A FIFO is full. However, OPR[0] is not reset and RTSAN will be asserted again when an empty FIFO position is available. This feature can be used for flow control to prevent overrun in the receiver by using the RTSAN output signal to control the CTSN input of the transmitting device.

MR1A[6] — Channel A Receiver Interrupt Select — This bit selects either the channel A receiver ready status (RXRDY) or the channel A FIFO full status (FFULL) to be used for CPU interrupts. It also causes the selected bit to be output on OP4 if it is programmed as an interrupt output via the OPCR.

MR1A[5] — Channel A Error Mode Select — This bit selects the operating mode of the three FIFOed status bits (FE, PE, received break) for channel A. In the 'character' mode, status is provided on a character-by-character basis: the status applies only to the character at the top of the FIFO. In the 'block' mode, the status provided in the SR for these bits is the ac-

Table 1 2681 REGISTER ADDRESSING

A3	A2	A1	A0	READ (RDN = 0)	WRITE (WRN = 0)
0	0	0	0	Mode Register A (MR1A, MR2A)	Mode Register A (MR1A, MR2A)
0	0	0	1	Status Register A (SRA)	Clock Select Reg. A (CSRA)
0	0	1	0	*Reserved*	Command Register A (CRA)
0	0	1	1	RX Holding Register A (RHRA)	TX Holding Register A (THRA)
0	1	0	0	Input Port Change Reg. (IPCR)	Aux. Control Register (ACR)
0	1	0	1	Interrupt Status Reg. (ISR)	Interrupt Mask Reg. (IMR)
0	1	1	0	Counter/Timer Upper (CTU)	C/T Upper Register (CTUR)
0	1	1	1	Counter/Timer Lower (CTL)	C/T Lower Register (CTLR)
1	0	0	0	Mode Register B (MR1B, MR2B)	Mode Register B (MR1B, MR2B)
1	0	0	1	Status Register B (SRB)	Clock Select Reg. B (CSRB)
1	0	1	0	*Reserved*	Command Register B (CRB)
1	0	1	1	RX Holding Register B (RHRB)	TX Holding Register B (THRB)
1	1	0	0	*Reserved*	*Reserved*
1	1	0	1	Input Port	Output Port Conf. Reg. (OPCR)
1	1	1	0	Start Counter Command	Set Output Port Bits Command
1	1	1	1	Stop Counter Command	Reset Output Port Bits Command

Preliminary

Table 2 REGISTER BIT FORMATS

	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
	RX RTS CONTROL	RX INT SELECT	ERROR MODE	PARITY MODE		PARITY TYPE	BITS PER CHAR.	
MR1A MR1B	0 = no 1 = yes	0 = RXRDY 1 = FFULL	0 = char 1 = block	00 = with parity 01 = force parity 10 = no parity 11 = multi-drop mode		0 = even 1 = odd	00 = 5 01 = 6 10 = 7 11 = 8	

	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
	CHANNEL MODE		Tx RTS CONTROL	CTS ENABLE Tx	STOP BIT LENGTH*			
MR2A MR2B	00 = Normal 01 = Auto echo 10 = Local loop 11 = Remote loop		0 = no 1 = yes	0 = no 1 = yes	0 = 0.563 1 = 0.625 2 = 0.688 3 = 0.750	4 = 0.813 5 = 0.875 6 = 0.938 7 = 1.000	8 = 1.563 9 = 1.625 A = 1.688 B = 1.750	C = 1.813 D = 1.875 E = 1.938 F = 2.000

*Add 0.5 to values shown for 0-7 if channel is programmed for 5 bits/char.

	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
CSRA CSRB	RECEIVER CLOCK SELECT				TRANSMITTER CLOCK SELECT			
	See text				See text			

	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
CRA CRB	not used—must be 0	MISCELLANEOUS COMMANDS			DISABLE Tx	ENABLE Tx	DISABLE Rx	ENABLE Rx
		See text			0 = no 1 = yes	0 = no 1 = yes	0 = no 1 = yes	0 = no 1 = yes

	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
	RECEIVED BREAK	FRAMING ERROR	PARITY ERROR	OVERRUN ERROR	TxEMT	TxRDY	FFULL	RxRDY
SRA SRB	0 = no 1 = yes *	0 = no 1 = yes *	0 = no 1 = yes *	0 = no 1 = yes	0 = no 1 = yes	0 = no 1 = yes	0 = no 1 = yes	0 = no 1 = yes

*These status bits are appended to the corresponding data character in the receive FIFO. A read of the status register provides these bits (7:5) from the top of the FIFO together with bits 4:0. These bits are cleared by a 'reset error status' command. In character mode they are discarded when the corresponding data character is read from the FIFO.

	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
	OP7	OP6	OP5	OP4	OP3		OP2	
OPCR	0 = OPR[7] 1 = TxRDYB	0 = OPR[6] 1 = TxRDYA	0 = OPR[5] 1 = RxRDY/ FFULLB	0 = OPR[4] 1 = RxRDY/ FFULLA	00 = OPR[3] 01 = C/T OUTPUT 10 = TxCB (1X) 11 = RxCB (1X)		00 = OPR[2] 01 = TxCA (16X) 10 = TxCA (1X) 11 = RxCA (1X)	

	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
	BRG SET SELECT	COUNTER/TIMER MODE AND SOURCE			DELTA IP3 INT	DELTA IP2 INT	DELTA IP1 INT	DELTA IP0 INT
ACR	0 = set1 1 = set2	See table 4			0 = off 1 = on	0 = off 1 = on	0 = off 1 = on	0 = off 1 = on

	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
	DELTA IP3	DELTA IP2	DELTA IP1	DELTA IP0	IP3	IP2	IP1	IP0
IPCR	0 = no 1 = yes	0 = no 1 = yes	0 = no 1 = yes	0 = no 1 = yes	0 = low 1 = high	0 = low 1 = high	0 = low 1 = high	0 = low 1 = high

Table 2 REGISTER BIT FORMATS (Continued)

	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
ISR	INPUT PORT CHANGE	DELTA BREAK B	RxRDY/ FFULLB	TxRDYB	COUNTER READY	DELTA BREAK A	RxRDY/ FFULLA	TxRDYA
	0 = no 1 = yes	0 = no 1 = yes	0 = no 1 = yes	0 = no 1 = yes	0 = no 1 = yes	0 = no 1 = yes	0 = no 1 = yes	0 = no 1 = yes

	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
IMR	IN. PORT CHANGE INT	DELTA BREAK B INT	RxRDY/ FFULLB INT	TxRDYB INT	COUNTER READY INT	DELTA BREAK A INT	RxRDY/ FFULLA INT	TxRDYA INT
	0 = off 1 = on	0 = off 1 = on	0 = off 1 = on	0 = off 1 = on	0 = off 1 = on	0 = off 1 = on	0 = off 1 = on	0 = off 1 = on

	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
CTUR	C/T[15]	C/T[14]	C/T[13]	C/T[12]	C/T[11]	C/T[10]	C/T[9]	C/T[8]

	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
CTLR	C/T[7]	C/T[6]	C/T[5]	C/T[4]	C/T[3]	C/T[2]	C/T[1]	C/T[0]

cumulation (logical OR) of the status for all characters coming to the top of the FIFO since the last 'reset error' command for channel A was issued.

MR1A[4:3] — Channel A Parity Mode Select — If 'with parity' or 'force parity' is selected, a parity bit is added to the transmitted character and the receiver performs a parity check on incoming data. MR1A[4:3] = 11 selects channel A to operate in the special multidrop mode described in the Operation section.

MR1A[2] — Channel A Parity Type Select — This bit selects the parity type (odd or even) if the 'with parity' mode is programmed by MR1A[4:3], and the polarity of the forced parity bit if the 'force parity' mode is programmed. It has no effect if the 'no parity' mode is programmed. In the special multidrop mode it selects the polarity of the A/D bit.

MR1A[1:0] — Channel A Bits per Character Select — This field selects the number of data bits per character to be transmitted and received. The character length does not include the start, parity, and stop bits.

MR2A — Channel A Mode Register 2

MR2A is accessed when the channel A MR pointer points to MR2, which occurs after any access to MR1A. Accesses to MR2A do not change the pointer.

MR2A[7:6] — Channel A Mode Select — Each channel of the DUART can operate in one of four modes. MR2A[7:6] = 00 is the normal mode, with the transmitter and receiver operating independently. MR2A[7:6] = 01 places the channel in the automatic echo mode, which automatically retransmits the received data. The following conditions are true while in automatic echo mode:

1. Received data is reclocked and retransmitted on the TxDA output.
2. The receive clock is used for the transmitter.
3. The receiver must be enabled, but the transmitter need not be enabled.
4. The channel A TxRDY and TxEMT status bits are inactive.
5. The received parity is checked, but is not regenerated for transmission, i.e., transmitted parity bit is as received.
6. Character framing is checked, but the stop bits are retransmitted as received.
7. A received break is echoed as received until the next valid start bit is detected.
8. CPU to receiver communication continues normally, but the CPU to transmitter link is disabled.

Two diagnostic modes can also be configured. MR2A[7:6] = 10 selects local loopback mode. In this mode:

1. The transmitter output is internally connected to the receiver input.
2. The transmit clock is used for the receiver.
3. The TxDA output is held high.
4. The RxDA input is ignored.
5. The transmitter must be enabled, but the receiver need not be enabled.
6. CPU to transmitter and receiver communications continue normally.

The second diagnostic mode is the remote loopback mode, selected by MR2A[7:6] = 11. In this mode:

1. Received data is relocked and retransmitted on the TxDA output.
2. The receive clock is used for the transmitter.

3. Received data is not sent to the local CPU, and the error status conditions are inactive.
4. The received parity is not checked and is not regenerated for transmission, i.e., transmitted parity bit is as received.
5. The receiver must be enabled.
6. Character framing is not checked, and the stop bits are retransmitted as received.
7. A received break is echoed as received until the next valid start bit is detected.

The user must exercise care when switching into and out of the various modes. The selected mode will be activated immediately upon mode selection, even if this occurs in the middle of a received or transmitted character. Likewise, if a mode is deselected, the device will switch out of the mode immediately. An exception to this is switching out of autoecho or remote loopback modes: if the de-selection occurs just after the receiver has sampled the stop bit (indicated in autoecho by assertion of RxRDY), and the transmitter is enabled, the transmitter will remain in autoecho mode until the entire stop bit has been retransmitted.

MR2A[5] — Channel A Transmitter Request-to-Send Control — This bit controls the deactivation of the RTSAN output (OP0) by the transmitter. This output is normally asserted by setting OPR[0] and negated by resetting OPR[0]. MR2A[5] = 1 causes OPR[0] to be reset automatically one bit time after the characters in the channel A transmit shift register and in the THR, if any, are completely transmitted, including the programmed number of stop bits, if the transmitter is not enabled. This feature can be used to automatically terminate the transmission of a message as follows:

1. Program auto-reset mode: MR2A[5] = 1.
2. Enable transmitter.
3. Assert RTSAN: OPR[0] = 1.
4. Send message.
5. Disable transmitter after the last character is loaded into the channel A THR.
6. The last character will be transmitted and OPR[0] will be reset one bit time after the last stop bit, causing RTSAN to be negated.

MR2A[4] — Channel A Clear-to-Send Control — If this bit is 0, CTSAN has no effect on the transmitter. If this bit is a 1, the transmitter checks the state of CTSAN (IP0) each time it is ready to send a character. If IP0 is asserted (low), the character is transmitted. If it is negated (high), the TxDA output remains in the marking state and the transmission is delayed until CTSAN goes low. Changes in CTSAN while a character is being transmitted do not affect the transmission of that character.

MR2A[3:0] — Channel A Stop Bit Length Select — This field programs the length of the stop bit appended to the transmitted character. Stop bit lengths of 9/16 to 1 and 1-9/16 to 2 bits, in increments of 1/16 bit, can be programmed for character lengths of 6, 7, and 8 bits. For a character length of 5 bits, 1-1/16 to 2 stop bits can be programmed in increments of 1/16 bit. The receiver only checks for a 'mark' condition at the center of the first stop bit position (one bit time after the last data bit, or after the parity bit if parity is enabled) in all cases.

If an external 1X clock is used for the transmitter, MR2A[3] = 0 selects one stop bit and MR2A[3] = 1 selects two stop bits to be transmitted.

MR1B — Channel B Mode Register 1

MR1B is accessed when the channel B MR pointer points to MR1. The pointer is set to MR1 by RESET or by a 'set pointer' command applied via CRB. After reading or writing MR1B, the pointer will point to MR2B.

The bit definitions for this register are identical to the bit definitions for MR1A, except that all control actions apply to the channel B receiver and transmitter and the corresponding inputs and outputs.

MR2B — Channel B Mode Register 2

MR2B is accessed when the channel B MR pointer points to MR2, which occurs after any access to MR1B. Accesses to MR2B do not change the pointer.

The bit definitions for this register are identical to the bit definitions for MR2A, except that all control actions apply to the channel B receiver and transmitter and the corresponding inputs and outputs.

CSRA — Channel A Clock Select Register

CSRA[7:4] — Channel A Receiver Clock Select — This field selects the baud rate clock for the channel A receiver as follows:

				Baud Rate CLOCK = 3.6864MHz	
CSRA[7:4]				ACR[7] = 0	ACR[7] = 1
0	0	0	0	50	75
0	0	0	1	110	110
0	0	1	0	134.5	134.5
0	0	1	1	200	150
0	1	0	0	300	300
0	1	0	1	600	600
0	1	1	0	1,200	1,200
0	1	1	1	1,050	2,000
1	0	0	0	2,400	2,400
1	0	0	1	4,800	4,800
1	0	1	0	7,200	1,800
1	0	1	1	9,600	9,600
1	1	0	0	38.4K	19.2K
1	1	0	1	Timer	Timer
1	1	1	0	IP4—16X	IP4—16X
1	1	1	1	IP4—1X	IP4—1X

The receiver clock is always a 16X clock except for CSRA[7:4] = 1111.

CSRA[3:0] — Channel A Transmitter Clock Select — This field selects the baud rate clock for the channel A transmitter. The field definition is as per CSRA[7:4] except as follows:

				Baud Rate	
CSRA[3:0]				ACR[7] = 0	ACR[7] = 1
1	1	1	0	IP3—16X	IP3—16X
1	1	1	1	IP3—1X	IP3—1X

The transmitter clock is always a 16X clock except for CSRA[3:0] = 1111.

CSRB — Channel B Clock Select Register

CSRB[7:4] — Channel B Receiver Clock Select — This field selects the baud rate clock for the channel B receiver. The field definition is as per CSRA[7:4] except as follows:

				Baud Rate	
CSRB[7:4]				ACR[7] = 0	ACR[7] = 1
1	1	1	0	IP6—16X	IP6—16X
1	1	1	1	IP6—1X	IP6—1X

The receiver clock is always a 16X clock except for CSRB[7:4] = 1111.

CSRB[3:0] — Channel B Transmitter Clock Select — This field selects the baud rate clock for the channel B transmitter. The field definition is as per CSRA[7:4] except as follows:

				Baud Rate	
CSRB[3:0]				ACR[7] = 0	ACR[7] = 1
1	1	1	0	IP5—16X	IP5—16X
1	1	1	1	IP5—1X	IP5—1X

The transmitter clock is always a 16X clock except for CSRB[3:0] = 1111.

Preliminary

CRA — Channel A Command Register

CRA is a register used to supply commands to channel A. Multiple commands can be specified in a single write to CRA as long as the commands are non-conflicting, e.g., the 'enable transmitter' and 'reset transmitter' commands cannot be specified in a single command word.

CRA[6:4] — Channel A Miscellaneous Commands — The encoded value of this field may be used to specify a single command as follows:

CRA[6:4]	COMMAND
0 0 0	No command.
0 0 1	Reset MR pointer. Causes the channel A MR pointer to point to MR1.
0 1 0	Reset receiver. Resets the channel A receiver as if a hardware reset had been applied. The receiver is disabled and the FIFO is flushed.
0 1 1	Reset transmitter. Resets the channel A transmitter as if a hardware reset had been applied.
1 0 0	Reset error status. Clears the channel A Received Break, Parity Error, Framing Error, and Overrun Error bits in the status register (SRA[7:4]). Used in character mode to clear OE status (although RB, PE, and FE bits will also be cleared) and in block mode to clear all error status after a block of data has been received.
1 0 1	Reset channel A break change interrupt. Causes the channel A break detect change bit in the interrupt status register (ISR[2]) to be cleared to zero.
1 1 0	Start break. Forces the TXDA output low (spacing). If the transmitter is empty the start of the break condition will be delayed up to two bit times. If the transmitter is active the break begins when transmission of the character is completed. If a character is in the THR, the start of the break will be delayed until that character, or any others loaded subsequently are transmitted. The transmitter must be enabled for this command to be accepted.
1 1 1	Stop Break. The TXDA line will go high (marking) within two bit times. TXDA will remain high for one bit time before the next character, if any, is transmitted.

CRA[3] — Disable Channel A Transmitter — This command terminates transmitter operation and resets the TxRDY and TxEMT status bits. However, if a character is being transmitted or if a character is in the THR when the transmitter is disabled, the transmission of the character(s) is completed before assuming the inactive state.

CRA[2] — Enable Channel A Transmitter — Enables operation of the channel A transmitter. The TxRDY status bit will be asserted.

CRA[1] — Disable Channel A Receiver — This command terminates operation of the receiver immediately — a character being received will be lost. The command has no effect on the receiver status bits or any other control registers. If the special multidrop mode is programmed, the receiver operates even if it is disabled. See Operation section.

CRA[0] — Enable Channel A Receiver — Enables operation of the channel A receiver. If not in the special wakeup mode, this also forces the receiver into the search for start-bit state.

CRB — Channel B Command Register

CRB is a register used to supply commands to channel B. Multiple commands can be specified in a single write to CRB as long as the commands are non-conflicting, e.g., the 'enable transmitter' and 'reset transmitter' commands cannot be specified in a single command word.

The bit definitions for this register are identical to the bit definitions for CRA, except that all control actions apply to the channel B receiver and transmitter and the corresponding inputs and outputs.

SRA — Channel A Status Register

SRA[7] — Channel A Received Break — This bit indicates that an all zero character of the programmed length has been received without a stop bit. Only a single FIFO position is occupied when a break is received: further entries to the FIFO are inhibited until the RxDA line returns to the marking state for at least one-half a bit time (two successive edges of the internal or external 1x clock).

When this bit is set, the channel A 'change in break' bit in the ISR (ISR[2]) is set. ISR[2] is also set when the end of the break condition, as defined above, is detected.

The break detect circuitry can detect breaks that originate in the middle of a received character. However, if a break begins in the middle of a character, it must persist until at least the end of the next character time in order for it to be detected.

SRA[6] — Channel A Framing Error — This bit, when set, indicates that a stop bit was not detected when the corresponding data character in the FIFO was received. The stop bit check is made in the middle of the first stop bit position.

SRA[5] — Channel A Parity Error — This bit is set when the 'with parity' or 'force parity' mode is programmed and the corresponding character in the FIFO was received with incorrect parity.

In the special multidrop mode the parity error bit stores the received A/D bit.

SRA[4] — Channel A Overrun Error — This bit, when set, indicates that one or more characters in the received data stream have been lost. It is set upon receipt of a new character when the FIFO is full and a character is already in the receive shift register waiting for an empty FIFO position. When this occurs, the character in the receive shift register (and its break detect, parity error and framing error status, if any) is lost.

This bit is cleared by a 'reset error status' command.

SRA[3] — Channel A Transmitter Empty (TxEMTA) — This bit will be set when the channel A transmitter underruns, i.e., both the transmit holding register (THR) and the transmit shift register are empty. It is set after transmission of the last stop bit of a character if no character is in the THR awaiting transmission. It is reset when the THR is loaded by the CPU or when the transmitter is disabled.

SRA[2] — Channel A Transmitter Ready (TxRDYA) — This bit, when set, indicates that the THR is empty and ready to be loaded with a character. This bit is cleared when the THR is loaded by the CPU and is set when the character is transferred to the transmit shift register. TxRDY is reset when the transmitter is disabled and is set when the transmitter is first enabled, viz., characters loaded into the THR while the transmitter is disabled will not be transmitted.

Preliminary

SRA[1] — Channel A FIFO Full (FFULLA) — This bit is set when a character is transferred from the receive shift register to the receive FIFO and the transfer causes the FIFO to become full, i.e., all three FIFO positions are occupied. It is reset when the CPU reads the RHR. If a character is waiting in the receive shift register because the FIFO is full, FFULL will not be reset when the CPU reads the RHR.

SRA[0] — Channel A Receiver Ready (RxRDYA) — This bit indicates that a character has been received and is waiting in the FIFO to be read by the CPU. It is set when the character is transferred from the receive shift register to the FIFO and reset when the CPU reads the RHR, if after this read there are no more characters still in the FIFO.

SRB — Channel B Status Register

The bit definitions for this register are identical to the bit definitions for SRA, except that all status applies to the channel B receiver and transmitter and the corresponding inputs and outputs.

OPCR — Output Port Configuration Register

OPCR[7] — OP7 Output Select — This bit programs the OP7 output to provide one of the following:

- The complement of OPR[7]
- The channel B transmitter interrupt output, which is the complement of TxRDYB. When in this mode OP7 acts as an open collector output. Note that this output is not masked by the contents of the IMR.

OPCR[6] — OP6 Output Select — This bit programs the OP6 output to provide one of the followng:

- The complement of OPR[6]
- The channel A transmitter interrupt output, which is the complement of TxRDYA. When in this mode OP6 acts as an open collector output. Note that this output is not masked by the contents of the IMR.

OPCR[5] — OP5 Output Select — This bit programs the OP5 output to provide one of the following:

- The complement of OPR[5]
- The channel B receiver interrupt output, which is the complement of ISR[5]. When in this mode OP5 acts as an open collector output. Note that this output is not masked by the contents of the IMR.

OPCR[4] — OP4 Output Select — This bit programs the OP4 output to provide one of the following:

- The complement of OPR[4]
- The channel A receiver interrupt output, which is the complement of ISR[1]. When in this mode OP4 acts as an open collector output. Note that this output is not masked by the contents of the IMR.

OPCR[3:2] — OP3 Output Select — This field programs the OP3 output to provide one of the following:

- The complement of OPR[3]
- The counter/timer output, in which case OP3 acts as an open collector output. In the timer mode, this output is a square wave at the programmed frequency. In the counter mode, the output remains high until terminal count is reached, at which time it goes low. The output returns to the high state when the counter is stopped by a stop counter command. Note that this output is not masked by the contents of the IMR.
- The 1X clock for the channel B transmitter, which is the clock that shifts the transmitted data. If data is not being transmitted, a free running 1X clock is output.
- The 1X clock for the channel B receiver, which is the clock that samples the received data. If data is not being received, a free running 1X clock is output.

OPCR[1:0] — OP2 Output Select — This field programs the OP2 output to provide one of the following:

- The complement of OPR[2]
- The 16X clock for the channel A transmitter. This is the clock selected by CSRA[3:0], and will be a 1X clock if CSRA[3:0] = 1111.
- The 1X clock for the channel A transmitter, which is the clock that shifts the transmitted data. If data is not being transmitted, a free running 1X clock is output.
- The 1X clock for the channel A receiver, which is the clock that samples the received data. If data is not being received, a free running 1X clock is output.

ACR — Auxiliary Control Register

ACR[7] — Baud Rate Generator Set Select — This bit selects one of two sets of baud rates to be generated by the BRG:

Set 1: 50, 110, 134.5, 200, 300, 600, 1.05K, 1.2K, 2.4K, 4.8K, 7.2K, 9.6K, and 38.4K baud.

Set 2: 75, 110, 134.5, 150, 300, 600, 1.2K, 1.8K, 2.0K, 2.4K, 4.8K, 9.6K, and 19.2K baud.

The selected set of rates is available for use by the channel A and B receivers and transmitters as described in CSRA and CSRB. Baud rate generator characteristics are given in table 3.

Table 3 BAUD RATE GENERATOR CHARACTERISTICS CRYSTAL OR CLOCK = 3.6864MHz

NOMINAL RATE (BAUD)	ACTUAL 16X CLOCK (KHz)	ERROR (PERCENT)
50	0.8	0
75	1.2	0
110	1.759	-0.069
134.5	2.153	0.059
150	2.4	0
200	3.2	0
300	4.8	0
600	9.6	0
1050	16.756	-0.260
1200	19.2	0
1800	28.8	0
2000	32.056	0.175
2400	38.4	0
4800	76.8	0
7200	115.2	0
9600	153.6	0
19.2K	307.2	0
38.4K	614.4	0

NOTE
Duty cycle of 16X clock is 50% ± 1%

Preliminary

ACR[6:4]—Counter/Timer Mode and Clock Source Select — This field selects the operating mode of the counter/timer and its clock source as shown in table 4.

ACR[3:0] — IP3, IP2, IP1, IPO Change of State Interrupt Enable — This field selects which bits of the Input Port Change register (IPCR) cause the input change bit in the interrupt status register (ISR[7]) to be set. If a bit is in the 'on' state, the setting of the corresponding bit in the IPCR will also result in the setting of ISR[7], which results in the generation of an interrupt output if IMR[7] = 1. If a bit is in the 'off' state, the setting of that bit in the IPCR has no effect on ISR[7].

IPCR — Input Port Change Register

IPCR[7:4] — IP3, IP2, IP1, IP0 Change of State — These bits are set when a change of state, as defined in the Input Port section of this data sheet, occurs at the respective input pins. They are cleared when the IPCR is read by the CPU. A read of the IPCR also clears ISR[7], the input change bit in the interrupt status register.

The setting of these bits can be programmed to generate an interrupt to the CPU.

IPCR[3:0] — IP3, IP2, IP1, IP0 Current State — These bits provide the current state of the respective inputs. The information is unlatched and reflects the state of the input pins at the time the IPCR is read.

ISR — Interrupt Status Register

This register provides the status of all potential interrupt sources. The contents of this register are masked by the interrupt mask register (IMR). If a bit in the ISR is a '1' and the corresponding bit in the IMR is also a '1', the INTRN output will be asserted. If the corresponding bit in the IMR is a zero, the state of the bit in the ISR has no effect on the INTRN output. Note that the IMR does not mask the reading of the ISR — the true status will be provided regardless of the contents of the IMR. The contents of this register are initialized to 00_{16} when the DUART is reset.

ISR[7] — Input Port Change Status — This bit is a '1' when a change of state has occurred at the IP0, IP1, IP2, or IP3 inputs and that event has been selected to cause an interrupt by the programming of ACR[3:0]. The bit is cleared when the CPU reads the IPCR.

Table 4 ACR [6:4] FIELD DEFINITION

ACR[6:4]	MODE	CLOCK SOURCE
0 0 0	Counter	External (IP2)
0 0 1	Counter	TXCA — 1X clock of channel A transmitter
0 1 0	Counter	TXCB — 1X clock of channel B transmitter
0 1 1	Counter	Crystal or external clock (X1/CLK) divided by 16
1 0 0	Timer	External (IP2)
1 0 1	Timer	External (IP2) divided by 16
1 1 0	Timer	Crystal or external clock (X1/CLK)
1 1 1	Timer	Crystal or external clock (X1/CLK) divided by 16

ISR[6] — Channel B Change in Break — This bit, when set, indicates that the channel B receiver has detected the beginning or the end of a received break. It is reset when the CPU issues a channel B 'reset break change interrupt' command.

ISR[5] — Channel B Receiver Ready or FIFO Full — The function of this bit is programmed by MR1B[6]. If programmed as receiver ready, it indicates that a character has been received in channel B and is waiting in the FIFO to be read by the CPU. It is set when the character is transferred from the receive shift register to the FIFO and reset when the CPU reads the RHR. If after this read there are more characters still in the FIFO the bit will be set again after the FIFO is 'popped'. If programmed as FIFO full, it is set when a character is transferred from the receive holding register to the receive FIFO and the transfer causes the channel B FIFO to become full, i.e., all three FIFO positions are occupied. It is reset when the CPU reads the RHR. If a character is waiting in the receive shift register because the FIFO is full, the bit will be set again when the waiting character is loaded into the FIFO.

ISR[4] — Channel B Transmitter Ready — This bit is a duplicate of TxRDYB (SRB[2]).

ISR[3] — Counter Ready — In the counter mode, this bit is set when the counter reaches terminal count and is reset when the counter is stopped by a stop counter command.

In the timer mode, this bit is set once each cycle of the generated square wave (every other time that the counter/timer reaches zero count). The bit is reset by a stop counter command. The command, however, does not stop the counter/timer.

ISR[2] — Channel A Change in Break — This bit, when set, indicates that the channel A receiver has detected the beginning or the end of a received break. It is reset when the CPU issues a channel A 'reset break change interrupt' command.

ISR[1] — Channel A Receiver Ready or FIFO Full — The function of this bit is programmed by MR1A[6]. If programmed as receiver ready, it indicates that a character has been received in channel A and is waiting in the FIFO to be read by the CPU. It is set when the character is transferred from the receive shift register to the FIFO and reset when the CPU reads the RHR. If after this read there are more characters still in the FIFO the bit will be set again after the FIFO is 'popped'. If programmed as FIFO full, it is set when a character is transferred from the receive holding register to the receive FIFO and the transfer causes the channel A FIFO to become full, i.e., all three FIFO positions are occupied. It is reset when the CPU reads the RHR. If a character is waiting in the receive shift register because the FIFO is full, the bit will be set again when the waiting character is loaded into the FIFO.

ISR[0] — Channel A Transmitter Ready — This bit is a duplicate of TxRDYA (SRA[2]).

IMR — Interrupt Mask Register

The programming of this register selects which bits in the ISR cause an interrupt output. If a bit in the ISR is a '1' and the corresponding bit in the IMR is also a '1', the INTRN output will be asserted. If the corresponding bit in the IMR is a zero, the state of the bit in the ISR has no effect on the INTRN output. Note that the IMR does not mask the programmable interrupt outputs OP3-OP7 or the reading of the ISR.

Preliminary

CTUR and CTLR — Counter/Timer Registers

The CTUR and CTLR hold the eight MSBs and eight LSBs respectively of the value to be used by the counter/timer in either the counter or timer modes of operation. The minimum value which may be loaded into the CTUR/CTLR registers is 0002_{16}. Note that these registers are write-only and cannot be read by the CPU.

In the timer (programmable divider) mode, the C/T generates a square wave with a period of twice the value (in clock periods) of the CTUR and CTLR. If the value in CTUR or CTLR is changed, the current half-period will not be affected, but subsequent half periods will be. In this mode the C/T runs continuously. Receipt of a start counter command (read with A3–A0 = 1110) causes the counter to terminate the current timing cycle and to begin a new cycle using the values in CTUR and CTLR.

The counter ready status bit (ISR[3]) is set once each cycle of the square wave. The bit is reset by a stop counter command (read with A3–A0 = 1111). The command, however, does not stop the C/T. The generated square wave is output on OP3 if it is programmed to be the C/T output.

In the counter mode, the C/T counts down the number of pulses loaded into CTUR and CTLR by the CPU. Counting begins upon receipt of a start counter command. Upon reaching terminal count (0000_{16}), the counter ready interrupt bit (ISR[3]) is set. The counter continues counting past the terminal count until stopped by the CPU. If OP3 is programmed to be the output of the C/T, the output remains high until terminal count is reached, at which time it goes low. The output returns to the high state and ISR[3] is cleared when the counter is stopped by a stop counter command. The CPU may change the values of CTUR and CTLR at any time, but the new count becomes effective only on the next start counter command. If new values have not been loaded, the previous count values are preserved and used for the next count cycle.

In the counter mode, the current value of the upper and lower 8 bits of the counter (CTU, CTL) may be read by the CPU. It is recommended that the counter be stopped when reading to prevent potential problems which may occur if a carry from the lower 8-bits to the upper 8-bits occurs between the times that both halves of the counter are read. However, note that a subsequent start counter command will cause the counter to begin a new count cycle using the values in CTUR and CTLR.

If you are running the SPUZ under a normal CompuPro operating system (Concurrent DOS 8-16, CP/M-86®, CP/M 8-16™, etc.), the board should be installed in the system with the jumpers and switches set, and the new SPUZ SWITCH (SW!.CMD) placed on the correct drive. Eight-bit programs should now run by invoking the SPUZ through SW!.CMD. If you are trying to bring up the SPUZ in some other environment, the 8086 code below, written for the program HLT.CMD, provides an example that should be studied before you write any of your own code. An explanation as to why the steps are necessary follows the code. We are not saying that this is the best way to program the SPUZ; it simply shows you some of the possible pitfalls in getting the SPUZ running.

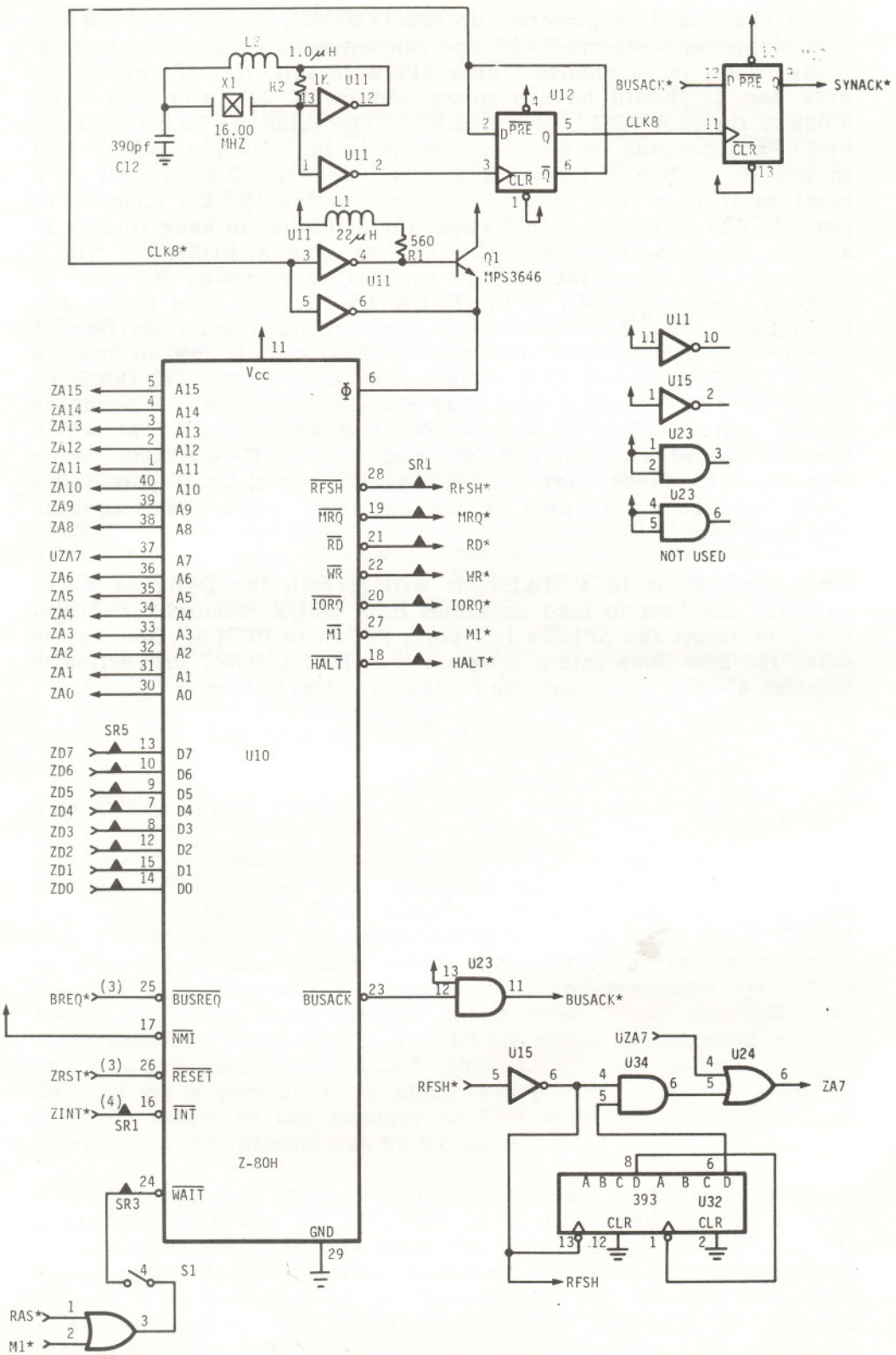
HLT.CMD -- Places Z80 into a halt state so that its memory is being refreshed and can be written to and read from.

```
START:
SELBD:    MOV  al,00h          ;00h is the board select of this
                               ;board. It should be between 00h
                               ;and 0Fh.
          MOV  dx,0FC40h       ;SLAVESEL port.
          OUT  dx,al           ;Select this board.

UNRES:    MOV  al,08h          ;Bit pattern to allow Z80 to run.
          MOV  dx,0FC41h       ;PTLATCH port.
          OUT  dx,al           ;Allow Z80 to execute starting at
                               ;location 0000h.

PUTHLT:   MOV  bx,0E000h       ;Memory address of SPUZ 64K segment.
          MOV  es,bx
          MOV  al,76h          ;Z80 HLT opcode is 76h.
          ES:MOV  [0h],al      ;Puts 76h at E0000h, 0h in SPUZ
                               ;memory.

RESET:    MOV  al,0h           ;Bit pattern to reset Z80.
          MOV  dx,0FC41h       ;PTLATCH port.
          OUT  dx,al           ;RESET Z80.

WAIT:     PUSH ax              ;Kill some time for Z80 to reset.
          POP  ax

RUN:      MOV  al,08h          ;Bit pattern to allow Z80 to run.
          MOV  dx,0FC41h       ;PTLATCH port.
          OUT  dx,al           ;Allow Z80 to execute starting at
                               ;location 0000h. Its first opcode
                               ;is a HLT.

CHECK: ES:MOV  al,[0h]         ;read back in 76h (halt opcode).
          CMP  al,76h          ;make sure it is.
          JNZ  START           ;retry if it is not.


EXIT:     BACK to OS.
```

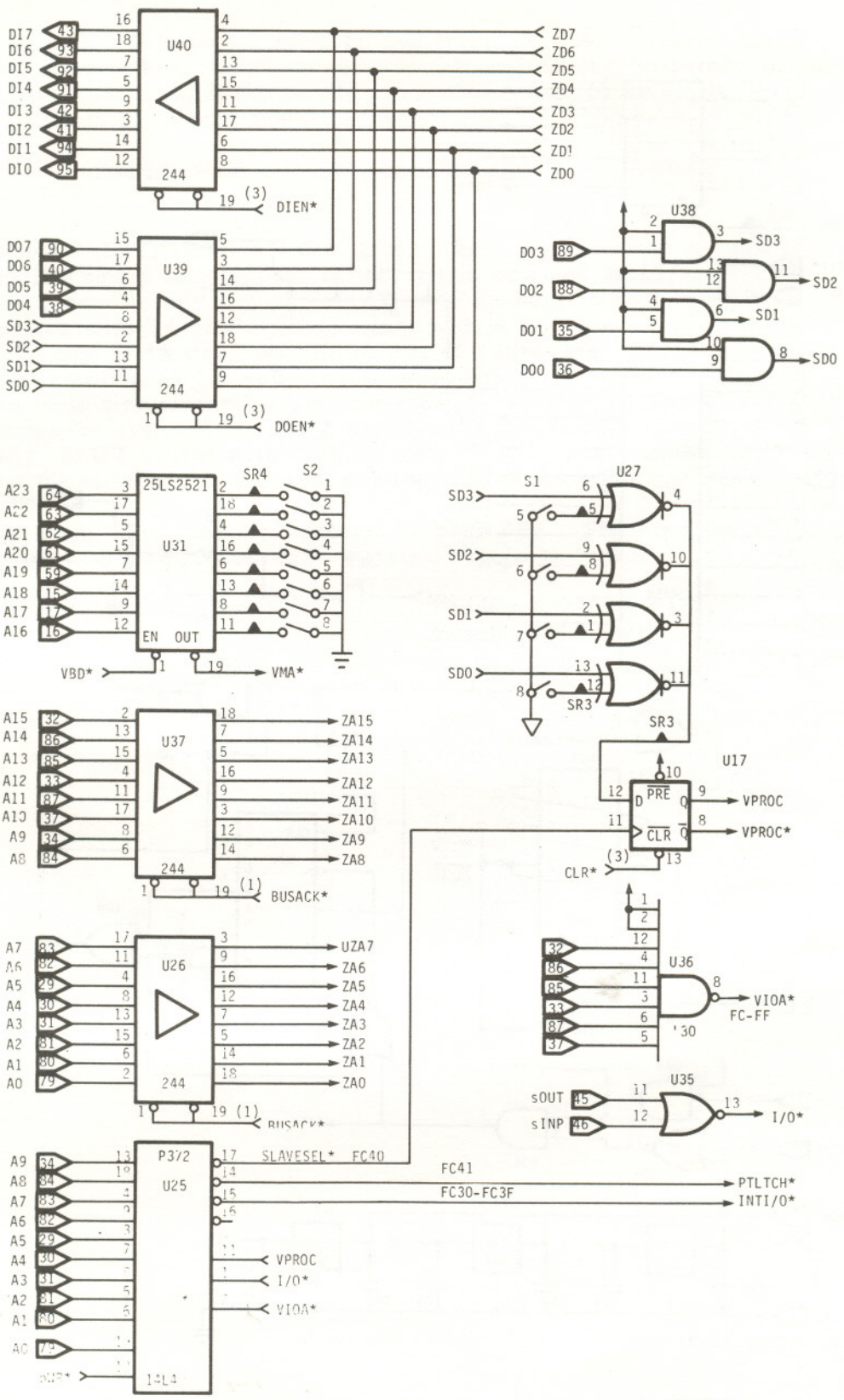
When the board is powered up, the Z80 is in a reset. It is not refreshing the Dynamic RAM, and can not issue BUSACK. The first thing to do is to choose which SPUZ to talk to. If this is the only one, it should have a select address of 00h, and thus write a 00h to the SLAVESEL port (SELBD). The value can be between 00h and 0Fh depending on which one of up to 16 SPUZ boards you want to access. When a board is selected and the Z80 is still in a reset as it is at this point, an access to the SPUZ's memory or ports FC42h to FC47h would cause the S-100 bus to hang forever in a wait state, waiting for the Z80 to give a BUSACK. It is imperative to allow the Z80 to run before accessing its ports or memory. Writing a 08h to the PTLATCH releases reset to the Z80 (UNRES). The Z80 is now running arbitrary code starting at 0000h. The program can then move a HLT (opcode 76h) to location 0000h (PUTHLT), and then issue a reset back to the Z80 (RESET). After waiting to make sure that the Z80 gets reset in even the fastest systems (WAIT), let the Z80 run again (RUN), at which time it should pick up the 76h, and enter a HALT state. It is important to check that the 76h is still there, as it potentially could have been corrupted by the Z80 itself when it was running arbitrary code.

Once the Z80 is in a HALT, it will refresh the Dynamic RAM, enabling the host to load programs into SPUZ memory. The host can also access the SPUZ's I/O ports FC30h to FC3Fh. One way to make the Z80 jump into a program is to load a JUMP instruction to location 0h, and then assert and release reset as above.

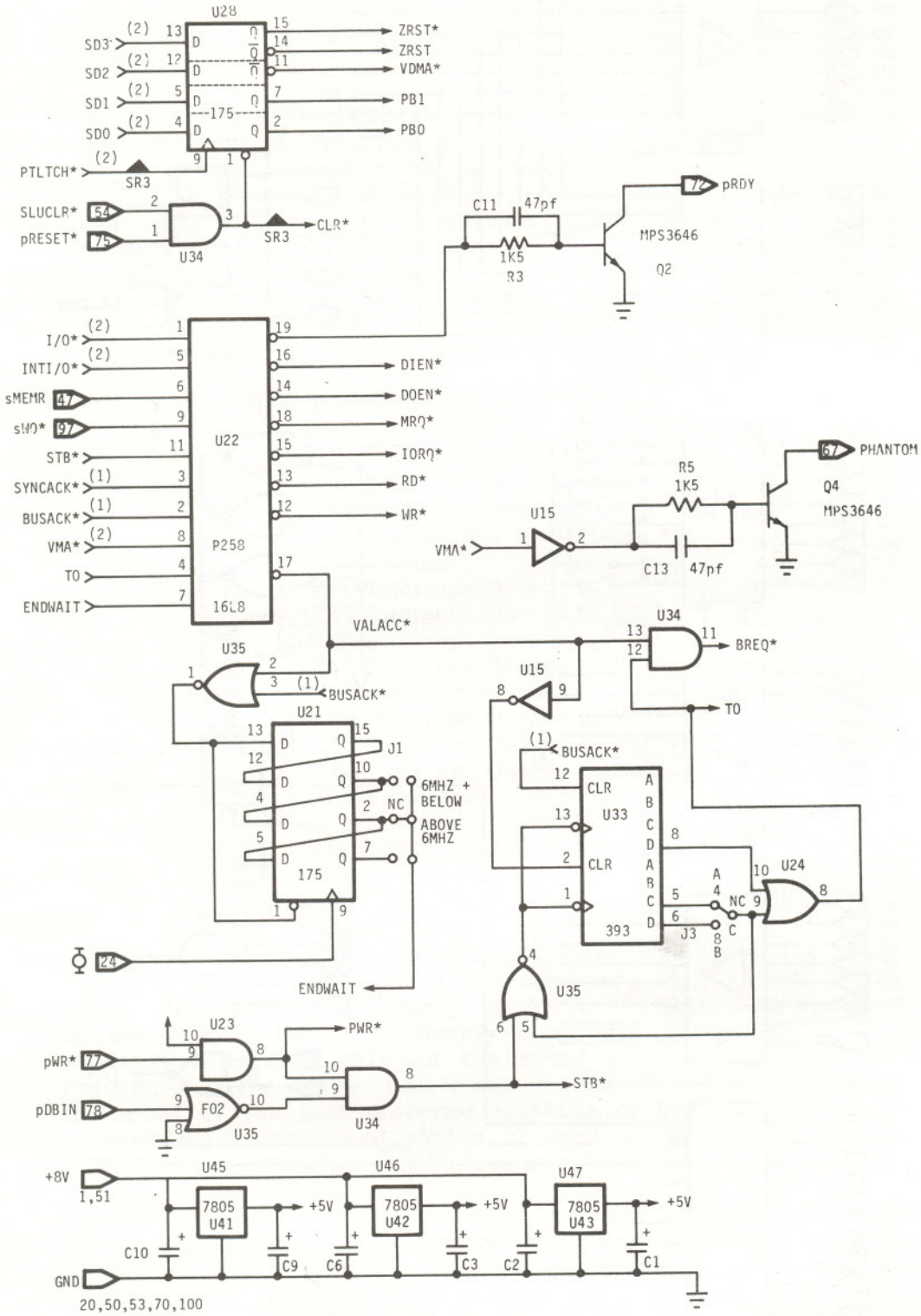
LOGIC DIAGRAM



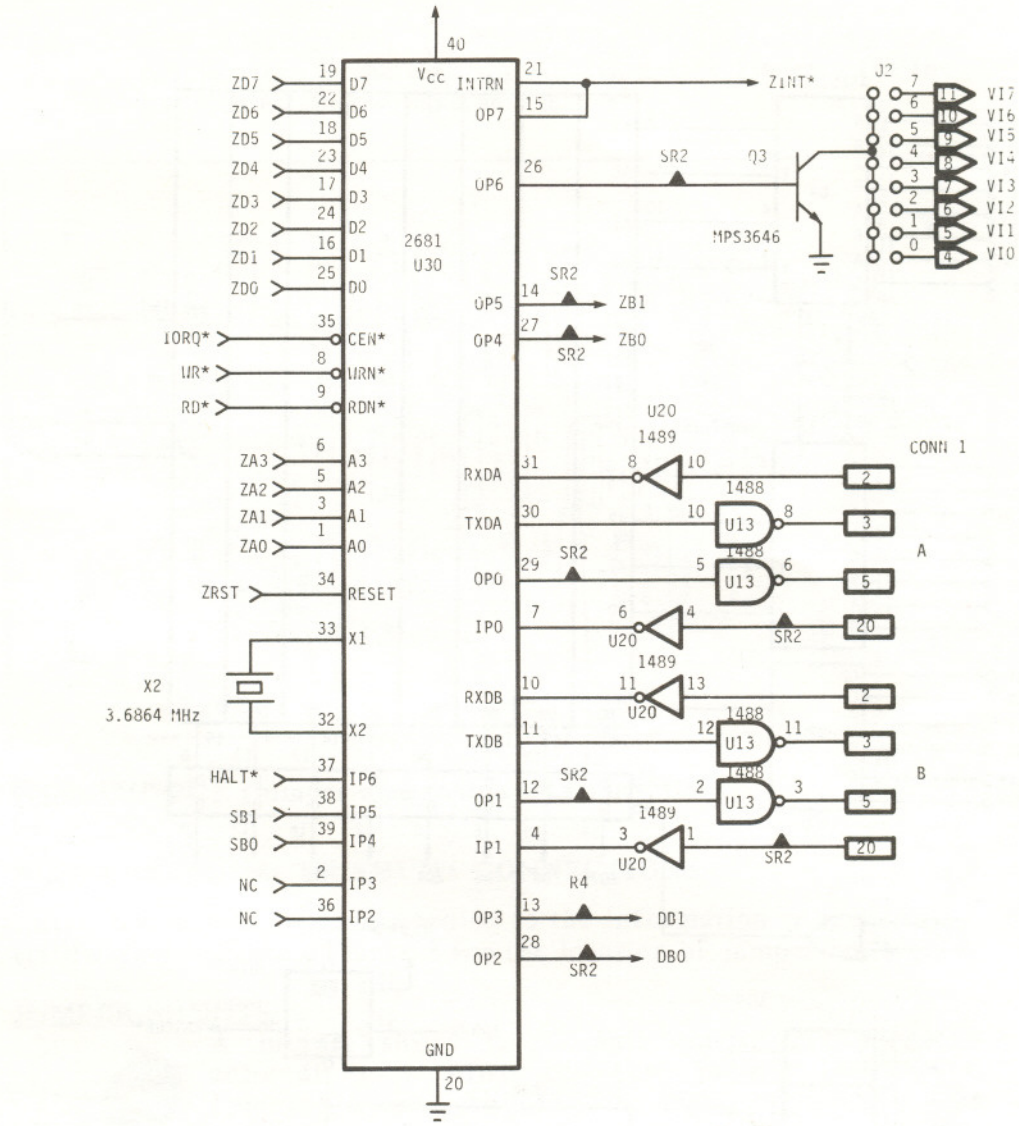
SPUZ
257E
Viasyn Corporation ©1985

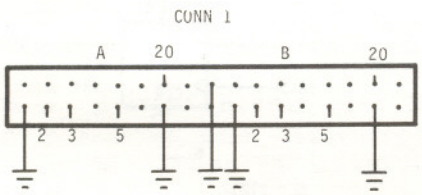


SPUZ
257E



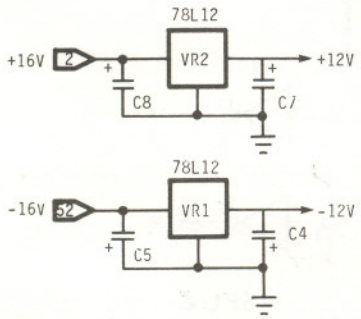
SPUZ
257E
Viasyn Corporation ©1985





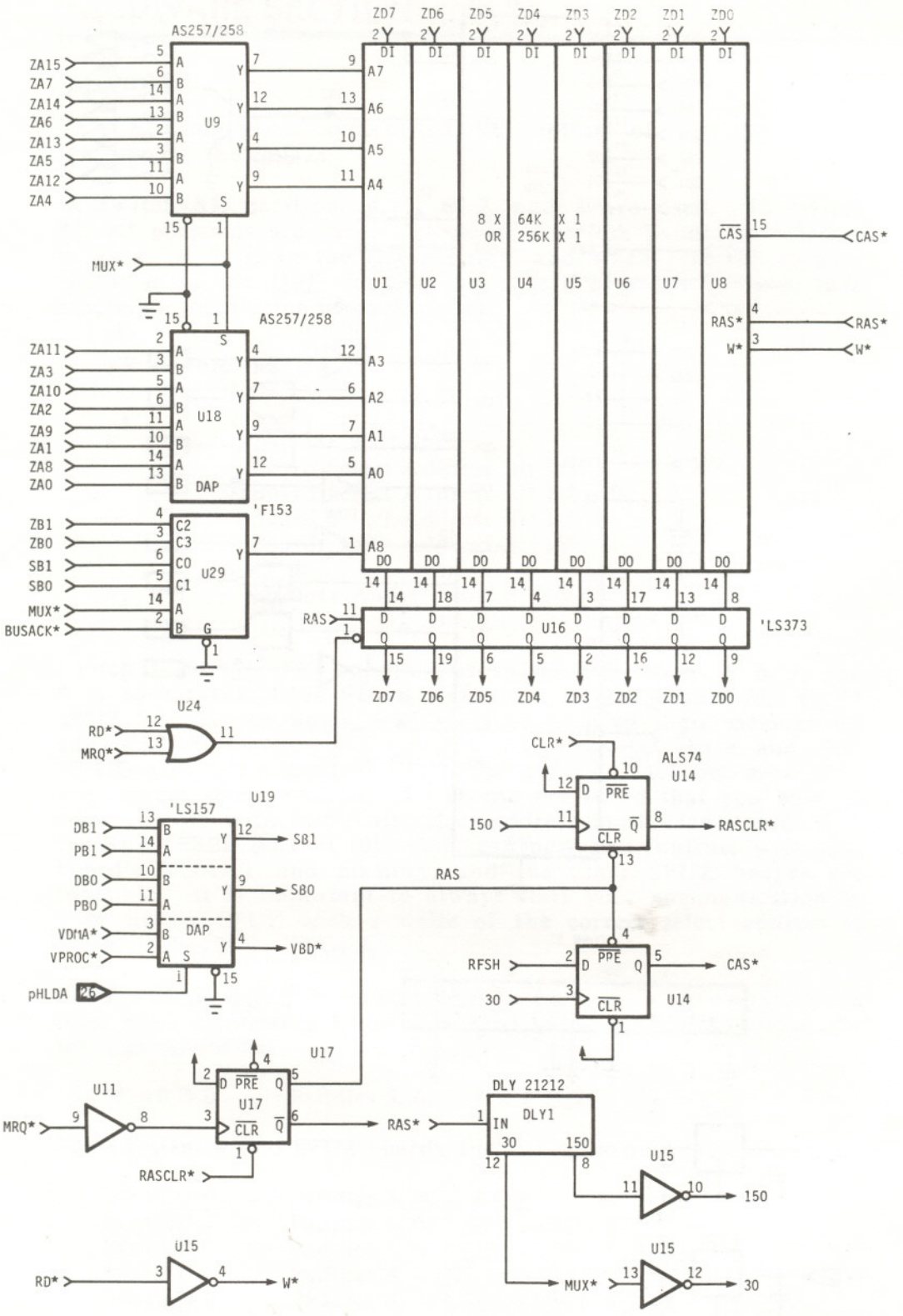
TOP VIEW - TOP OF BOARD

34 PIN
SHROUDED EDGE
CONNECTOR



38

SPUZ
257E
Viasyn Corporation ©1985



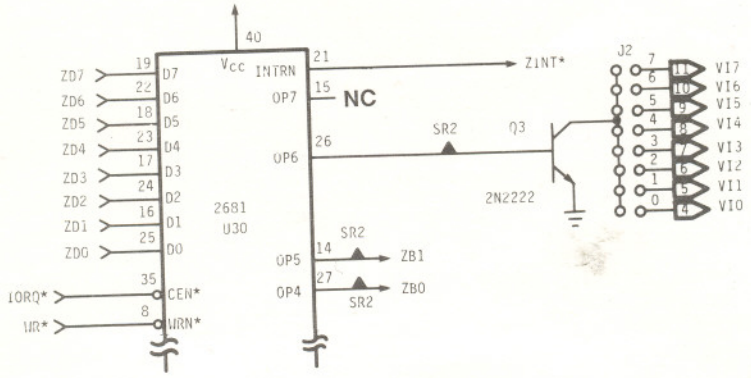
SPUZ
257E
Viasyn Corporation ©1985

CompuPro™

SPUZ 257D AND 257E ADDENDUM

This document is an addendum to the SPUZ Technical Manual No. 11990 for the SPUZ revision 257D and E. It is part of ECO 138 and should be attached to page 35 of the SPUZ manual.

The following change has been made to eliminate a possible hardware conflict. This change does not affect any CompuPro supplied software (SW!.CMD or SPZ.CMD). The hardware modification is to remove pin 15 of U30, the 2681 DUART from the socket. It should be done on every SPUZ revision 257D and 257E, and is corrected on 257E1 and later revisions.



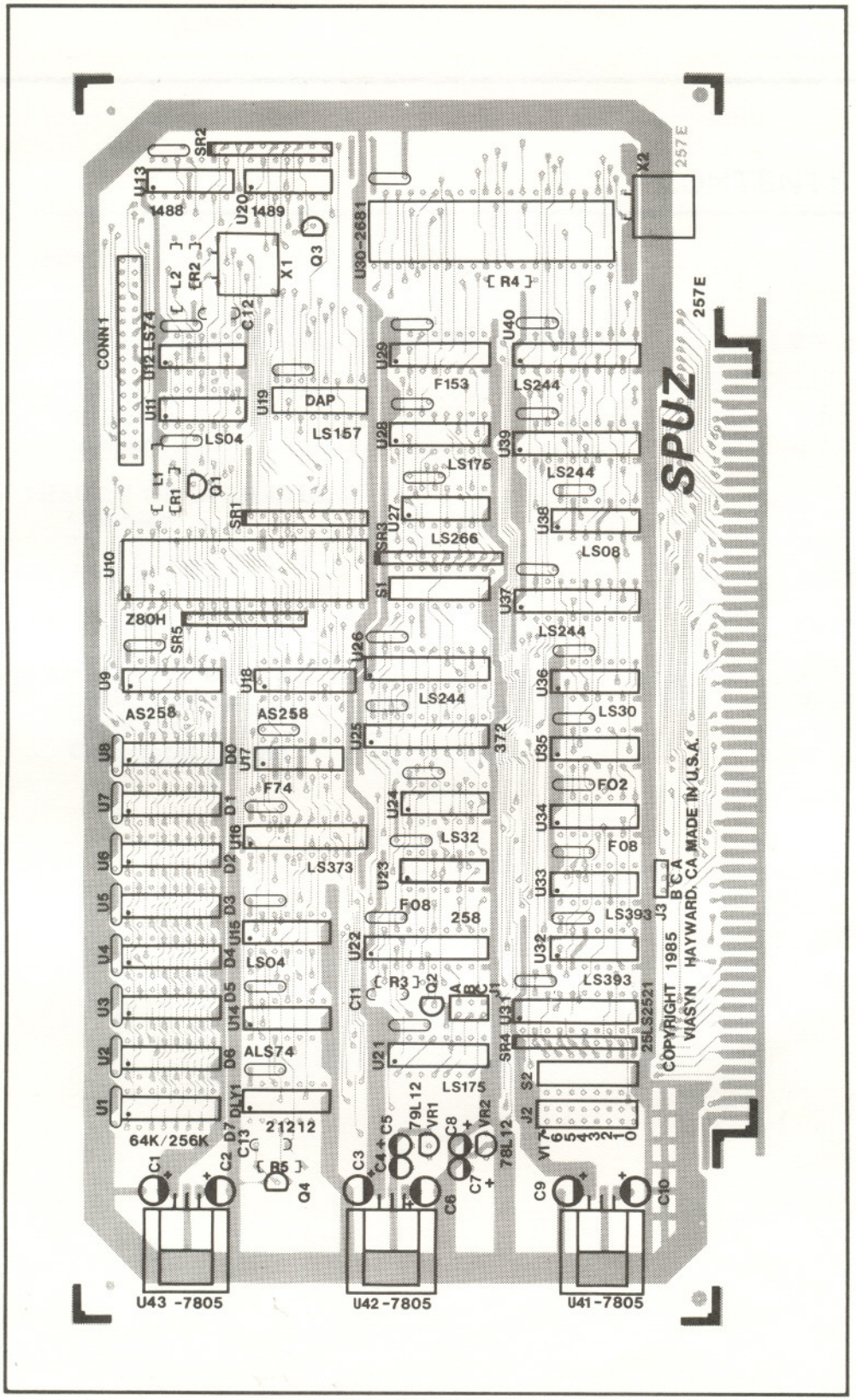
Viasyn Corporation
26538 Danti Court, Hayward, CA 94545–3999 (415) 786–0909

PARTS LIST

BD 257C	1	
74LS04	2	U11,U15
74F02	1	U35
74LS08	2	U38
74LS30	1	U36
74LS32	1	U24
74LS74	1	U12
74ALS74	1	U14
74F74	1	U17
74LS157	1	U19
74AS258	2	U9,U18
74LS175	2	U21,U28
74LS244	4	U26,U37,U39,U40
74LS266	1	U27
74LS373	1	U16
74LS393	2	U32,U33
25LS2521	1	U31
21212	1	DLY1
1488	1	U13
1489	1	U20
P258	1	U22 16L8 PAL
P372	1	U25 14L4 PAL
Z80H 8MHz	1	U10 CPU
2681	1	U30
74F08	2	U23,U34

For 64K DRAM:		
4164	8	U1,2,3,4,5,6,7,8
For 256K DRAM:		
41256	8	U1,2,3,4,5,6,7,8
74F153	1	U29
7805	3	U41,U42,U43
78L12	1	VR2
79L12	1	VR1
MPS3646	4	Q1, Q2, Q3, Q4
5.1KSIP	5	SR1,2,3,4,5
560	1	R1
1K	1	R2
4K7	1	R4
1K5	2	R3, R5
22UH	1	L1
1UH	1	L2

Ceramic Bypass	38	Unmarked
Radial TANTALUM for +5V	3	C1,C3,C9
Radial TANTALUM for +8V	3	C2,C6,C10
Radial TANTALUM for +12V	2	C4,C7
Radial TANTALUM for ±16V	2	C5,C8
47pF Ceramic	2	C11, C13
330pF Mica	1	C12
3.6864MHz Fundamental	1	X2
16MHZ Fundamental	1	X1
TANDEM SHUNT	1	J2-VI0
34 pin shrouded	1	CONN1
HEAT SINK WITH SPACER	3	FOR REGULATORS
6/32 BOLT (1/4")	3	FOR REGULATORS
8 LONG DOUBLE ROW PINS	1	J2
8 POS DIPSWITCH	2	S1,S2
14 PIN SOCKETS	17	DLY1,U11,12,13,14,15,17,20,23,24 U27,32,33,34,35,36,38
16 PIN SOCKETS	14	U1,2,3,4,5,6,7,8,9,18,19,21,28,29
20 PIN SOCKETS	8	U16,22,25,26,31,37,39,40
40 PIN SOCKET	2	U10,U30



COMPONENT LAYOUT

LIMITED WARRANTY

Viasyn Corporation warrants this computer product to be in good working order for a period of one (1) year, and six (6) months for drives from the date of purchase by the original end user. Should this product fail to be in good working order at any time during this warranty period, VIASYN will, at its option, repair or replace the product at no additional charge except as set forth below. Repair parts and replacement products will be furnished on an exchange basis and will be either reconditioned or new. All replaced parts and products become the property of VIASYN. This limited warranty does not include service to repair damage to the product resulting from accident, disaster, misuse, abuse, or unauthorized modification of the product.

If you need assistance, or suspect an equipment failure, always contact your System Center, or dealer first. System Center technicians are trained to provide prompt diagnosis and repair of equipment failures. If you are not satisfied by the actions taken by your System Center/dealer, please call VIASYN at (415) 786-0909 to obtain a **R**eturn **M**aterial **A**uthorization (**RMA**) number, or write to VIASYN at 26538 Danti Court, Hayward, California 94545-3999, Att.: RMA. Be sure to include a copy of the orginal bill of sale to establish purchase date. If the product is delivered by mail or common carrier, you agree to insure the product or assume the risk of loss or damage in transit, to prepay shipping charges to the warranty service location and to use the original shipping container or equivalent. Contact your System Center/dealer or write to VIASYN at the above address for futher information.

ALL EXPRESSED AND IMPLIED WARRANTIES FOR THIS PROCUCT, INCLUDING THE WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE LIMITED IN DURATION TO THE ABOVE LISTED PERIODS FROM THE DATE OF PURCHASE AND NO WARRANTIES, WHETHER EXPRESSED OR IMPLIED, WILL APPLY AFTER THIS PERIOD.

IF THIS PRODUCT IS NOT IN GOOD WORKING ORDER AS WARRANTED ABOVE, YOUR SOLE REMEDY SHALL BE REPAIR OR REPLACEMENT AS PROVIDED ABOVE. IN NO EVENT WILL VIASYN BE LIABLE TO YOU FOR ANY DAMAGES, INCLUDING ANY LOST PROFITS, LOST SAVINGS OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF OR INABILITY TO USE SUCH PRODUCT, EVEN IF VIASYN OR A SYSTEM CENTER/DEALER HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, OR FOR ANY CLAIM BY ANY OTHER PARTY.

IF THIS PRODUCT IS OUT OF WARRANTY, PLEASE CALL THE VIASYN RMA DEPARTMENT TO OBTAIN THE FLAT RATE LABOR QUOTATION FOR FACTORY SERVICE.

Viasyn Corporation
26538 Danti Court
Hayward, California 94545-3999
Telephone (415) 786-0909
TWX 510-100-3288

EFF 9/1/85: This warranty supersedes all previous warranties. All previous editions are obsolete.

Viasyn Corporation

8261-0030

26538 Danti Court, Hayward, CA 94545-3999 (415) 786-0909

385300/2