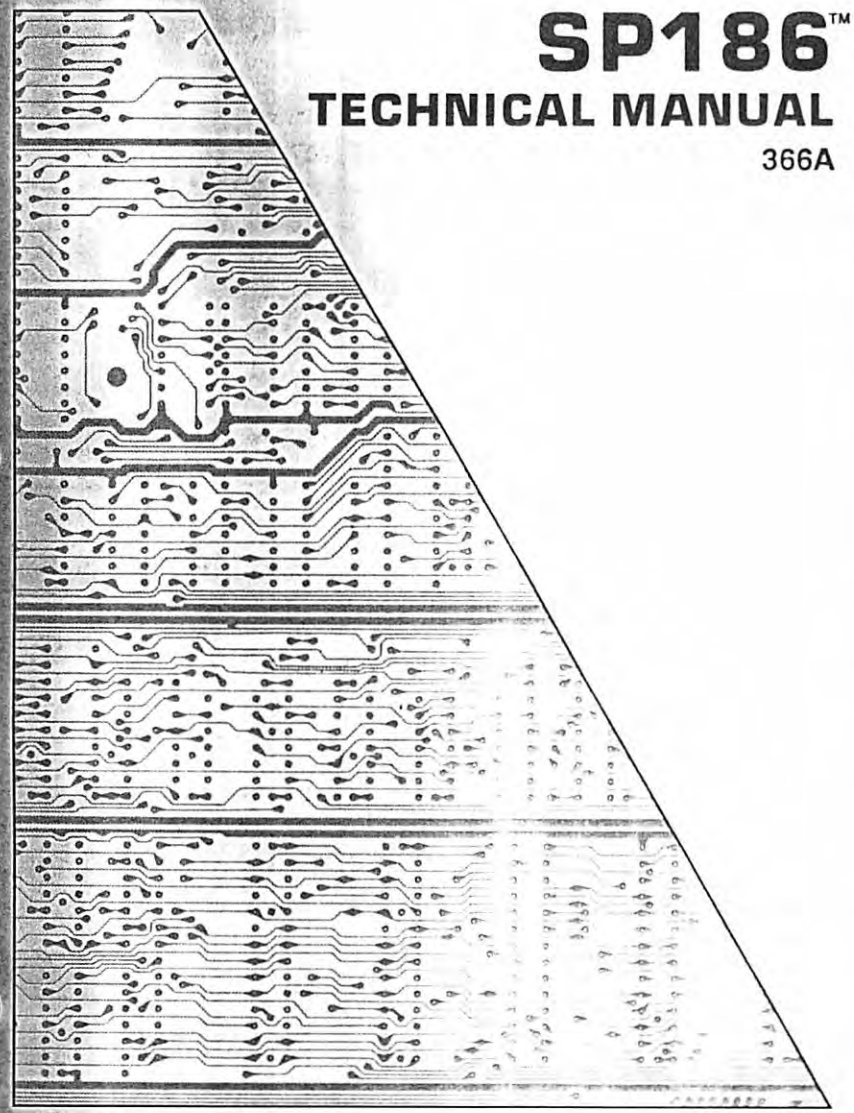
Viasyn Corporation
26538 Danti Court, Hayward, CA 94545-3999 (415) 786-0909

$20.00 8261-0031B 196125A/1

CompuPro

SP186™
TECHNICAL MANUAL
366A



8261-0031B September 1986

SP186
TECHNICAL MANUAL

HIGH PERFORMANCE 80186
512-KBYTE RAM
DUAL ASYNCHRONOUS COMMUNICATION CHANNELS

SP186 TECHNICAL MANUAL
Copyright 1986 Viasyn
Hayward, CA 94545

Document No. 8261-0031B
Filename: SP186.MAN
Board No: 366A

Contents

List of Tables

Specifications

Timing	Meets all IEEE 696/S-100 specifications
Processor.	80186
Clock Rate	10 MHz
Memory	512 Kbytes of RAM
S-100 Address Space .	Occupies 64-Kbyte memory space and 2 I/O ports
S-100 Memory Address .	Switch selectable to any 64-Kbyte page
Data Bus Width . .	16 bits internal, and fully conforms to IEEE 696/S100 16-bit protocol
Power Consumption. . .	2000 mA typical, 2500 mA maximum

About the SP186

The CompuPro SP186 Slave Processing Unit (80186) represents a significant advancement in multi-processing within the IEEE 696/S-100 environment. The SP186 combines a 10 MHz 80186, 512 Kbytes of memory, and two fully bi-directional RS-232C serial I/O channels to perform a number of "slave processing" tasks.

The SP186 supplies a high performance 16-bit node for a multi-user, multi-processing environment. Users can have their own 80186 processor, 512 Kbytes of memory, and their own serial ports. When used as an execution unit, the SP186 runs 16-bit user programs, dramatically reducing the requirements on the host. When used as a front-end I/O processor, the SP186 acts as an autonomous computer with serial ports, with access to all the resources of the host through operating system calls. Up to sixteen SP186s can reside within the same mainframe.

The SP186 is ideal for adding additional 16-bit processing power to 16-bit processor based systems. It is also ideal for adding 16-bit capability to 32-bit processor based systems. Where high throughput is necessary, this multi-processing configuration, in which host and slaves run simultaneously, is clearly superior to earlier dual processing systems where one of the processors is stopped while the other is running.

The SP186 meets all IEEE 696/S-100 bus specifications and is designed to run in S-100 based systems running at 10 MHz and up.

Features include:

- 10-MHz, 16-bit 80186 for high throughput
- 512 Kbytes of on-board RAM: sixteen SP186 boards occupy only 64 Kbytes of system memory
- Parity generation and detection for high reliability
- Dual bi-directional, asynchronous RS-232C serial ports for operation up to 38.4 Kbaud
- Fully software-selectable serial port parameters
- Supported by CompuPro's Concurrent DOS 8-16[tm] multi-user, multi-tasking operating system and by CP/M-68K[tm]

- True multi-processing instead of dual-processing
- 16-bit node in a "processor-per-user" system
- 16-bit capability in 16-bit and 32-bit systems

Introduction to Slave Processing

Tasks on a computer can be divided into two categories:

- I/O, including character and disk
- Execution of user programs

The SP186 is designed to process either character I/O or user programs effectively by dynamically changing roles under software control. When handling character I/O through its dual UART (DUART), the SP186 (80186, memory, and DUART) is in front-end I/O processor mode. When executing programs for the user, the SP186 is in execution unit mode. Due to the ability to change roles, the SP186's internal processor can timeshare between these two operations.

The major physical difference between these two modes is whether or not the task uses the on-board DUART.

The SP186 as an I/O Processor

When the SP186 is used as a front-end I/O processor, the task running on the SP186 uses the DUART for character I/O. It may also use the host for disk and character I/O through operating system calls.

Applications that use the SP186 as a front-end I/O processor require custom software that makes direct use of the DUART. Software development for custom applications can be done in a standard CP/M-86 format using the standard CP/M® tools (assemblers and debuggers) running on the SP186.

The SP186 as an Execution Unit

When the SP186 is used as an execution unit, any process running on the SP186 uses only system I/O; that is, it performs all of its character and disk I/O through the host CPU using operating system calls.

For example, the SP186 runs CP/M-86® programs as an execution unit. The 16-bit program is loaded into the SP186 and run by invoking SW86.CMD or SW86.68K under CompuPro's Concurrent DOS 8-16 or CP/M-68K, respectively. When the program requests disk or character I/O through CP/M-86 calls, the host provides it by using its I/O channels (terminals or disks).

One application of the SP186 in the front-end processor mode would be to monitor a stream of serial communication. Data would be brought in through a UART, processed by the 80186, and then retransmitted by the UART. It could change data under certain conditions, or collate statistics about the data stream and transfer these statistics to the host. Multiple SP186s can be used under one host, in one computer, to monitor many links.

Another way to think of the SP186 in the front-end processor mode is simply as an 80186 with a DUART and RAM, with two important additions:

- It can access the full resources of the host computer (e.g., disk) through operating system calls.
- The host computer can load the SP186 with programs. Any application in this category will use the Dual UART on the SP186 for communication.

In the execution unit mode, programs invoked by a user from the command line are loaded into the SP186 and executed. Input to the program can be from the system disk or system I/O ports through operating system (OS) calls. The input is acted upon by the program within the SP186, and the output is channeled back to the system disk or system I/O ports. For example, the SP186 is able to run 16-bit code that was written to operate under CP/M-86, on a host computer that may not support CP/M calls or doesn't run 16-bit 8086-type code (68000 or 32016 based systems). Another use is to add 16-bit multi-processing power to a 16-bit system that is heavily loaded.

An ideal application for the SP186 as an execution unit is as an upgrade to an existing multi-user CompuPro system running Concurrent DOS 8-16. In a system where the central CPU is bogging down under heavy loads, the SP186 can be added to enhance 16-bit performance. In addition, the SP186 relieves the host CPU of user programs, allowing it to execute the operating system functions more quickly and efficiently.

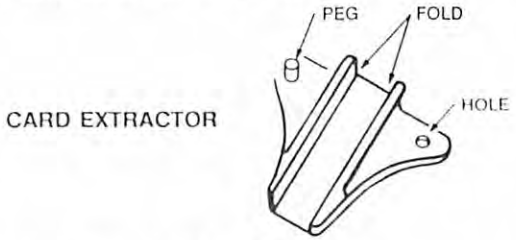
Another key application for the SP186 is found in systems where multiple tasks require large blocks of memory. Because the 1 Mbyte maximum addressable memory space of 8086 based uni-processor systems limits the number of programs that can be run, the SP186 provides an avenue for continued expansion of the memory space. Since each SP186 has 512 Kbytes of RAM, and sixteen SP186s can be in a system, far more memory is available to execute 16-bit programs.

A final application for the SP186 as an execution unit is to run 16-bit 8086-type code in a system that has a non-8086-type processor as a host. The SP186 can be installed in CompuPro 68K based systems running CP/M-68K to give the system the ability to run CP/M-86 programs. Thus, in a development environment, editing can be done on the SP186 using a CP/M-86 word processor such as 16-bit NewWord™, and compilation and debugging can be done by the host 68K processor. Finally, a CompuPro 68K system with a CompuPro Z80H based SPUZ™ and CompuPro SP186 can run CP/M-68K (68000), CP/M-86 (80186), and CP/M-80™ (Z80) programs.

Installing the SP186 Board

Step 1. Unpack the SP186 Board.

Along with the board, you will find extra jumper shunts and two card extractors in the plastic bag.



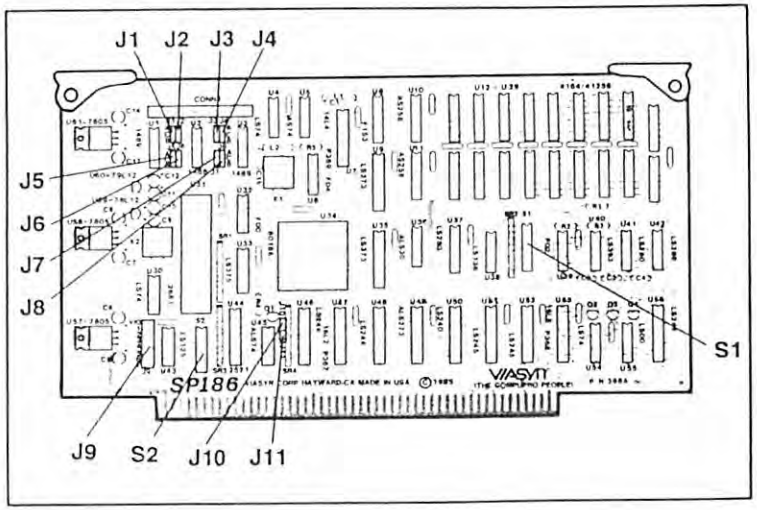
Step 2. Install the Card Extractors.

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

2. Insert the peg on the card extractor into the hole in the right corner of the board. Fold the extractor over the board's edge until the extractor's hole snaps over the peg.

3. NOTE: Make sure the long edge of the extractor is along the top edge of the board.

   Repeat for left extractor.



Step 3. Check Switch and Jumper Settings

For standard switch and jumper settings for use with Concurrent DOS 8-16, refer to the Concurrent DOS 8-16 **Installation and Customization Guide.** For standard switch and jumper settings for use with CP/M-68K, refer to the **CP/M-68K Installation and Customization Guide.**

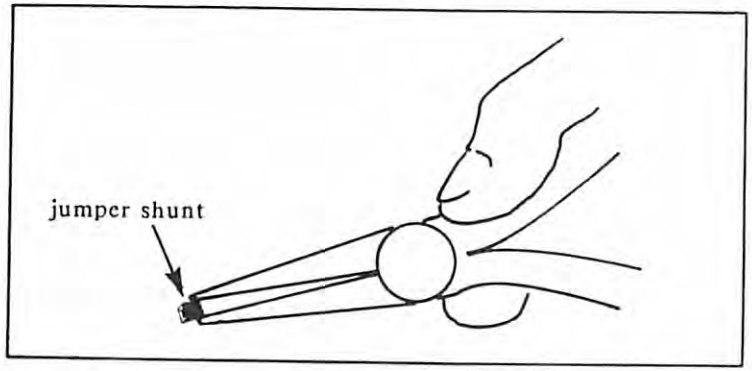
See Step 4 if you need to change jumper settings. Otherwise proceed to Step 5.

Step 4. How to Install Jumper Shunt Connectors

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.



Step 5. Insert the SP186 into the S-100 Bus.

The power to the system must be off. Place the board into a slot towards the middle of the enclosure. The edge connector is offset, so the SP186 fits only one way. Push down GENTLY until the board is firmly installed.

Switch and Jumper Summary

The following summary of switches and jumpers explains the function and logic of each option. For help in locating a switch or jumper, refer to the component layout diagram at the end of this manual.

The abbreviations under "RS-232 pin" are from the EIA (Electronic Industries Association) RS-232C standard, and are listed with their pin numbers. The abbreviations under "UART pin" are from the 2681 UART data sheet.

- TxD stands for transmitted data from the UART
- RxD stands for received data into the UART
- the OPx bits are output bits from the UART
- the IPx are input bits to the UART
- DCE stands for data communication equipment
- DTE stands for data terminal equipment

Table 1: Jumper Functions

J	Position	RS-232 pin	UART pin	SP186 function	
1	A-C	BB A (3)	TxD A	DCE	Channel A
	B-C	BB A (3)	RxD A	DTE	
2	A-C	BA A (2)	TxD A	DTE	
	B-C	BA A (2)	RxD A	DCE	
3	A-C	BB B (3)	TxD B	DCE	Channel B
	B-C	BB B (3)	RxD B	DTE	
4	A-C	BA B (2)	TxD B	DTE	
	B-C	BA B (2)	RxD B	DCE	
5	A-C	CB A (5)	OP0	DCE	Channel A
	B-C	CB A (5)	IP0	DTE	
6	A-C	CD A (20)	OP0	DTE	
	B-C	CD A (20)	IP0	DCE	
7	A-C	CB B (5)	OP1	DCE	Channel B
	B-C	CB B (5)	IP1	DTE	
8	A-C	CD B (20)	OP1	DTE	
	B-C	CD B (20)	IP1	DCE	
9	80186 Interrupt Output (Select VI0*-VI6* or VI7* Parity error output)				
10	Alternate S-100 port addresses				
11	Alternate S-100 port addresses				

Table 2: Switch Functions

S	Paddle	Function
1	1	Board Select Address Bit D3
	2	Board Select Address Bit D2
	3	Board Select Address Bit D1
	4	Board Select Address Bit D0
	5	Alternate S-100 I/O port addressing
2	1-8	Memory Address Bits A23-A16

Switch Settings

Switch S1 uses paddle positions 1, 2, 3, 4, and 5. Switch S2 uses all of its paddles. The following is an overview and detailed description of each switch.

Switch 1 paddles 1 through 4 (S1-1 through S1-4) - 4-bit Select Address

The function of Switch 1 paddles 1, 2, 3, and 4 is to set the 4-bit select address of the board. Up to sixteen SP186 boards can be installed in a single system, with all their memory addressed at the same 64 Kbyte page in system memory and their two I/O ports overlapping. Each SP186 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 SLVSEL port. You can now communicate with that board's memory and PORTL while the other SP186 boards are invisible. It is important always to start communication to a particular SP186 with a write of the correct select address to the SLVSEL port.

SP186 boards should be numbered starting with 0 and incremented with each additional board. For a system with one board, the settings should be:

BOARD 0 -- Paddles 1, 2, 3, and 4 ON

For a system with five SP186 boards, the settings should be:

BOARD 0 -- Paddles 1, 2, 3, 4 ON
BOARD 1 -- Paddles 1, 2, 3 ON, paddle 4 OFF
BOARD 2 -- Paddles 1, 2, 4 ON, paddle 3 OFF
BOARD 3 -- Paddles 1, 2 ON, paddles 3, 4 OFF
BOARD 4 -- Paddles 1, 3, 4 ON, paddle 2 OFF

If the SP186s in a system are configured for different I/O and memory addresses (through S1-5, J10, and J11), then the select address for all the boards can be the same. Normally, S1-1 through S1-4 of all the slaves are turned ON to give a select address of 0 for all the boards. In this way, the software can go to each board's SLVSEL port and write a 0. This allows all of the SP186's to have a memory window visible to the host at the same time. In this configuration, the software only has to write to each SP186 SLVSEL port once with a 0, and the boards will stay enabled and visible to the host until powered down.

Normally this approach is only practical in systems containing processors with large (>1 Mbyte) address spaces, such as a 32016-based system.

S1-5 Alternate S-100 Port Address

S1-5 selects an S-100 I/O port address for the SP186. It is used in conjunction with J10 and J11. For a complete description of the use of this switch and what ports are selected, refer to the section on Jumpers J10 and J11, especially Table 3: I/O Port Selection.

S2-1 through S2-8 Memory Address Bits A23-A16

Switch 2 determines where the 64 Kbyte page window of SP186 memory will appear to the system host and DMA devices. To CPU and DMA devices with 24 bits of addressing, the window appears in any one of 256 pages of memory. For CPU and DMA devices with 20 bits of addressing, the window is visible in any one of 16 pages. Paddle 1 is the most significant bit and corresponds with A23, while paddle 8 is the least significant bit and corresponds with A16.

The following are several examples of memory address selection:

MEMORY ADDRESS	S2 Paddles
0F0000h to 0FFFFFh	1,2,3,4 ON; 5,6,7,8 OFF.
450000h to 45FFFFh	1,3,4,5,7 ON; 2,6,8 OFF.
A70000h to A7FFFFh	2,4,5 ON; 1,3,6,7,8 OFF.

Jumper Settings

J1 through J8 DCE or DTE Select

Jumpers J1 through J8 allow the SP186 channel A and channel B serial ports to respond independently as either DTE or DCE. They are shipped with normally closed connections that allow both ports to act as DTE and connect directly to DTE devices such as terminals. As long as both ports are connected to DTE devices, these jumpers don't need to be changed. If you want to connect a DCE-type device like a modem to the SP186, a serial port must be changed to look like DCE. This can be accomplished either by installing a "null modem" cable in the line between the SP186 and the DCE device, or by changing the appropriate jumpers on the SP186.

To change channel A to look like a DTE device, the small traces connecting J1 A-C, J2 B-C, J5 A-C, and J6 B-C must be cut on the solder side of the board. Pins should then be installed in J1, J2, and J6 with shunts placed across J1 B-C, J2 A-C, J5 B-C, and J6 A-C.

To change channel B to look like DTE, use the same procedure on J3, J4, J7, and J8. Under all circumstances, make sure that either channel is set completely for DTE or DCE according to the settings given above before applying power. For example, if J1 is set for DTE, J2 must be set for DTE.

CAUTION: If a channel has jumpers mixed, component failure can occur.

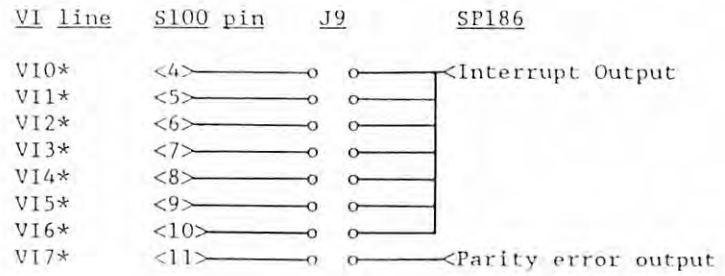
Jumpers J1 through J8 are 3-pin jumpers. Shunts are placed vertically across A-C (top 2 pins), or B-C (bottom 2 pins).

Jumpers J1, J3, J5, and J7 are shipped with a normally closed connection on the solder side of the board across A-C. Jumpers J2, J4, J6, and J8 are shipped with normally closed connections on the solder side of the board across B-C.

J9 Vectored Interrupt Select

Jumper J9 controls which vectored interrupt (VI) lines the SP186 asserts. There are two conditions under which the SP186 can assert a VI line. The first is through the normal setting of an interrupt to the host when the 80186 writes to the SETINT100 port. The second is when the SP186 detects an on-board DRAM (Dynamic Random Access Memory) parity error.

Jumper J9 can be set to allow both conditions to assert the same vectored interrupt or different interrupts. Through two pin shunts that are placed horizontally across one of the eight positions, the SETINT100 interrupt can assert VI0* through VI6* and the parity error can assert VI7*. Refer to the physical diagram of the jumper below. A vertical shunt should be placed across the bottom two pins on the right side of the jumper to allow both conditions to assert the same VI line.



J10 and J11 S-100 Port Address Selection

Jumpers J10 and J11, in conjunction with S1-5, control which S-100 I/O port addresses the SP186s occupy. Depending on the particular software configuration, all the SP186s in a system can occupy the same two I/O ports and have overlapping memory banks, or each of them can have different I/O ports and appear in different 64 Kbyte memory pages.

If all of the SP186s are set for the same I/O ports, then each must have a different select address chosen on Switch 1 paddles 1 through 4. In this way, up to sixteen SP186 boards can be in a single system, occupying only two I/O ports, and only one 64 Kbyte page of memory. This scheme is normally chosen when using 8088, 8086, or 80286 host processors due to their limited (1 Mbyte) memory addressing.

When using the 68000, 32016, or 80286 (protected) host processors with their expanded memory addressing (16 Mbytes), it is generally advantageous to give each SP186 a different set of I/O ports and 64 Kbyte memory page.

Up to eight different sets of I/O ports can be selected using J10, J11, and S1-5. It is also possible to have more than eight SP186 boards in a system at different I/O and memory locations. Please contact your Service Representative or CompuPro's technical assistance for information.

SP186 boards require two I/O ports; the SLVSEL port which is used to select one of the 16 boards, and the PORTL port which is used to control several functions of the currently selected board. The following table lists all possible locations for these two I/O ports.

Table 3: I/O Port Selection

S1-5	J11	J10 -->	SLVSEL	PORTL
ON	Removed	Removed	0FC40h	0FC41h
OFF	Removed	Removed	0FC42h	0FC43h
ON	Inserted	Removed	0FC44h	0FC45h
OFF	Inserted	Removed	0FC46h	0FC47h
ON	Removed	Inserted	0FC48h	0FC49h
OFF	Removed	Inserted	0FC4Ah	0FC4Bh
ON	Inserted	Inserted	0FC4Ch	0FC4Dh
OFF	Inserted	Inserted	0FC4Eh	0FC4Fh

In systems combining CompuPro SPUZ 8-bit Z80H based slave processors and SP186 16-bit 80186 based slaves, it may be desirable to overlap the I/O ports and 64 Kbyte memory page of the SP186 with the SPUZ. The SLVSEL port on both slaves makes this possible. On the SPUZ, the SLVSEL port is FC40h. Up to 16 slaves of either type can be in a system and use only one 64 Kbyte memory window by setting the SP186 SLVSEL port to 0FC40h, and the memory address of the SPUZ and SP186 to the same page.

Jumpers J10 and J11 are both 2-pin jumpers, with shunts that can either be inserted or removed.

Theory of Operation

This section of the manual explains how the circuitry on the SP186 works. In the following discussions, it will be helpful to refer to the schematic diagrams.

References

More detailed information about the 80186 MPU can be found in the 1985 Microsystems Components Handbook (order number 230843-002). Most of the information can be found in volume 1 of this 2 volume set. It can be obtained from Intel by contacting:

INTEL Literature Department
3065 Bowers Avenue
Santa Clara, CA. 95051
(800) 538-1876, or (800) 672-1833 (CA only)

Part of Signetics' data sheet for the SC2681 Dual Asynchronous Receiver/Transmitter (DUART) is reprinted with permission in Appendix A.

The 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 at:

811 East Arques Avenue
P.O. Box 409
Sunnyvale, CA. 94086
(408) 739-7700

Overview

The SP186 appears as two I/O ports and 64 Kbytes of memory to the S-100 host processor or the temporary bus master (DMA device). Because it doesn't become a temporary master, it is restricted from accessing other user memory space or system peripherals.

The S-100 bus has full access to the SP186's 512 Kbytes of memory. An innovative structure allows the host processor to address the SP186's 64 Kbyte memory window on any 32 Kbyte boundary, thus reducing boundary checking by the host.

Another unique feature of the SP186 is its ability to allow two programs running on the SP186 to have separate interrupt vector tables, simplifying multi-tasking.

Inter-processor communication is accomplished through a bi-directional interrupt structure. Message passing is available through the memory located on the SP186. DMA devices can transfer data directly to or from the SP186's memory.

There is no EPROM on the SP186. When the system is powered up, the on-board 80186 stays reset until the S-100 host processor has put a program in the SP186's memory and allows the 80186 to run. The host processor can reset the SP186 at any time.

Parity checking on the SP186 RAM ensures data integrity.

The on-board dual UART is accessible to the 80186 and can handle speeds up to 38.4 Kbaud asynchronously. This allows the SP186 to be an effective front-end I/O pre-processor for use in OEM or custom systems where high speed or real-time processing of large quantities of data is necessary. Compiled data is passed to the host system for mass storage or further processing.

Clock Generation

A 20 MHz clock is generated by two sections of inverter U6 (74F04), inductor L2, resistor R5, capacitor C15, and crystal X1. This clock is fed directly to the 80186, which divides it by 2 for its internal clock and generates the 80186's CLKOUT signal. CLKOUT is then inverted and used by flip-flop U45 (74ALS74) to generate SYNCHLDA*. SYNCHLDA* is the 86HLDA* signal delayed by a portion of a clock width. The SYNCHLDA* signal is used to start an external access to I/O or memory. This occurs after the S-100 addresses driven on to the internal bus by 86HLDA* have settled.

DRAM Refresh

The 80186 does not require any memory wait states if at least 150 nanosecond DRAM is used. Refresh to the DRAM chips is provided by programming an 80186 DMA channel to read every location in memory sequentially at about 16 microsecond intervals between words. RAS* is generated on every non-I/O cycle, regardless of address, allowing the DMA counter to scan through all of memory, refreshing it.

The DUART Ports

The 2681 Dual Asynchronous Receiver/Transmitter (DUART) is I/O mapped into 32 ports of the 80186's I/O space through PCS0* on the 80186. The base I/O port addresses can be selected in software. The 2681 is found only on the even Address/Data lines (AD0 through AD7 corresponding to ports 00h, 02h, 04h, 06h ... 01Eh) due to the SP186's 16-bit bus and the 2681's 8-bit bus. The ports listed in the 2681 data sheet should be doubled to get the actual port to the 80186. For example, to read 2681 port 06h (the CTU register), the 80186 must read port 0Ch. A description of the ports is contained in the reprint of the 2681 manual in Appendix A.

S-100 Access to SP186 Memory

The data bus is buffered by U51 and/or U52 (74LS245) depending on whether it's an 8- or 16-bit transfer and in which direction the transfer is taking place. The SP186 conforms to the IEEE 696/S-100 specification for 8- and 16-bit transfers and will transfer a full 16 bits when requested.

DOOE* and DIOE* are asserted to the S-100 bus on host memory reads (sMEMR) when pDBIN and ENDWAIT are asserted and the proper status is met. On writes, DOOE* and DIOE* are asserted to the 80186 A/D bus when pWR* and 86HLDA* are asserted and the proper status is met.

Crossover buffering that conforms to the DI bus and DO bus 8-bit S-100 protocol is provided by buffer U56 (74LS244) either when the host is writing an odd byte or when it is reading an even byte. The S-100 address bus is buffered onto the internal address bus (LA bus) on 86HLDA* by U46 and U48.

SP186 Control Ports

PAL U47 and NAND gate U36 (74ALS30) decode the S-100 I/O port addresses. The I/O addresses are decoded to produce PORTL* and SLVSEL* depending on the state of pins 13, 14, and 17 of PAL U47. SLVSEL* is produced regardless of the state of VBD*. This allows the compare circuitry, composed of U38 (74LS136) and flip-flop U45 (74ALS74), to select one of 16 slaves with a single I/O write when all of their SLVSEL* ports overlap. When VBD* is asserted low, VA* is asserted, enabling a memory cycle, and PORTL* is asserted,

writing control information to U49 (74ALS273). If all of the SP186s in a system have different SLVSEL* addresses, a single write to each board's SLVSEL* port with the correct select address will enable the board, turning on VBD*. Only if RESET* is asserted or SLVSEL* is written again with an incorrect select address will VBD* be negated, de-selecting the board.

The port map is shown following.

Table 4: SP186 Port Map

Port	Function		
Low	SLVSEL*	-	Selects which board within a bank of 16 is selected. (Write Only)
High	PORTL*	-	Controls 86RESET*, interrupt handshaking lines, and DRAM address for S-100 bus accesses. (Write Only)

The low port can be 0FC40h, 0FC42h, ..., and the high port can be 0FC41h, 0FC43h, ..., depending on the setting of J10, J11, and S1-5 (refer to the previous section on Jumpers J10 and J11). In systems with overlapping I/O ports, SLVSEL* is asserted on all the boards in the system, while PORTL* is asserted only on the board that has been selected by SLVSEL*. In systems with non-overlapping I/O ports, the proper select address must be written to each board separately through SLVSEL*, because PORTL* and memory accesses are not allowed until the board is selected.

When the extended address (A16-A23) matches what is set on Switch 2 and the signal VBD* is asserted, comparator U44 (25LS2521) asserts VA*. The system can access the SP186's internal memory when VA* is low, when pDBIN or pWR* is asserted, and when the proper status is met on sMEMR, sWO*, and sOUT. Thus, to access a memory location on the SP186 (when it is running, not reset), first write the proper data pattern to the SLVSEL* to clock in VBD* on the trailing edge of pWR*, then access the 64 Kbyte SP186 memory window.

The 8-bit write only port PORTL* is composed of U49 (74ALS273). Only the host can write to this port.

The bit map is as follows. The values given in parentheses are how the board comes up after a RESET* to the system.

Table 5: PORTL* Bit Map

Bit	Name	Function
D7	nc	Isn't used; output isn't connected.
D6	86RESET*	Write a 0 to reset the 80186. Write a 1 to allow 80186 to run. (0 - 86RESET* asserted low).
D5	SETINT86	Write a 0 then a 1 to cause an interrupt to the 80186. Rising edge triggered. (0 - ready to cause interrupt by writing 1).
D4	CLRINT100*	Write a 0 to clear the VI from the 80186. Write a 1 to enable the VI from the 80186. (0 - interrupt cleared).
D3	SB3*	Controls which memory the S-100 host accesses. These 4 bits control which of the sixteen 64 Kbyte pages in 512K DRAM (32 Kbyte boundaries) is selected. (0000b - comes up pointing to bottom memory.
D2	SB2*	
D1	SB1*	
D1	SB0*	

NOTE: These last 4 bits are inverted coming off the S-100 bus. That is, to access the bottom 64 Kbytes, the software must write a 1111b to the lower 4 bits.

SP186 Initialization

The initialization of the SP186 is critical. If the host accesses the SP186 memory when the 80186 is reset, the S-100 system crashes. The reason for this is that the host's access to the SP186 memory must be arbitrated by the 80186; if the 80186 is reset when memory is requested, the 80186 never issues an 86HLDA* allowing the cycle to take place. When the SP186 asserts pRDY to wait for the 80186 to give an 86HLDA*, the S-100 system hangs forever in a wait state.

To avoid this situation, when loading the SP186 memory, first allow the 80186 to run, load a small amount of code at the 80186's boot address 0FFFF0h, reset the 80186, wait, and allow the 80186 to run again. Check to make sure the code is intact and was not corrupted when the 80186 was running random code. Remember that the DRAM is not being refreshed except at the locations where the 80186 is running; the host must take this responsibility until the 80186 can be programmed to do it itself.

A programming example of how to do this is described in the software section and shown in Appendix B.

80186 Reset

The 86RESET* bit is set high to allow the 80186 to run. To reset the 80186 and 2681, set this bit low. This bit is set low on RESET* from the S-100 bus. The interrupt latches, consisting of both parts of U30 (74LS74), constitute a bi-directional interrupt structure. On RESET* from the S-100, the interrupt from the SP186 is cleared by CLRINT100*. No interrupt (VI) can be asserted, regardless of what the 80186 does, until the CLRINT100* bit is set high.

Interrupts

When CLRINT100* is set high, the 80186 can cause a VI with a rising edge of SETINT100 by first setting OP2 on the 2681 (U31) low and then setting it high (write a 1 and then a 0 to D2 of OPR). The host then clears the interrupt by lowering CLRINT100*.

NOTE: As long as CLRINT100* is low the 80186 cannot set an interrupt pending. Care must be taken to ensure that the 80186 never tries to SETINT100 while CLRINT100* is low, as the host does not get the interrupt, and the SETINT100 transition is lost.

A very similar setup forms the interrupt input to the 80186. Lowering the CLRINT86* bit by programming OP3 of the 2681 low clears the interrupt to the 80186. When CLRINT86* is high, a rising edge on the SETINT86 (first write a 0, then a 1 to bit D5 in PORTL) causes the INT0 line into the 80186 to be asserted. INT0 can then be negated by the 80186 by setting the CLRINT86* bit low. The same constraints on CLRINT100* and SETINT100 are present on CLRINT86* and SETINT86. If CLRINT86* is asserted low, no transitions on SETINT86 can cause an interrupt, and those transitions will be lost.

Memory Addressing on a 32 Kbyte Boundary by Host Processor

To eliminate the constant need for the host to check page boundaries when reading or writing to slave memory, adder U37 (74LS283) was included on the board to allow the internal DRAM 64 Kbyte window to be addressed at any 32 Kbyte boundary. By setting the SB(x)* bits correctly, the host is guaranteed a contiguous 32 Kbyte region (inside the 64 Kbyte external window), without having to worry about crossing over 64 Kbyte boundaries.

For example, to write 2 Kbytes to location 0FF00h in memory using traditional mapping schemes, you would first set the window to page 0 (see DRAM between 00h and 0FFFFh), then write 0100h bytes out of the 2 Kbytes to 0FF00h through 0FFFFh. Next, set the window page to 1 (see DRAM between 010000h and 01FFFFh), and write the remainder to 010000h through 0106FFh.

Using the SP186's 32 Kbyte windows, simply set the window to 08000h (see DRAM between 08000h and 017FFFh), and write the 2K block in one operation. This is particularly critical when using a DMA device to load the SP186 directly.

To set the internal window to 0 (see DRAM between 00h and 0FFFFh), write a 01111h to the SB(x)* bits in the PORTL* port. To set the window to 08000h (see DRAM between 08000h and 017FFFh), write a 01110h to the SB(x)* bits.

The following table shows which DRAM address on the SP186 is visible to the host within the 64-Kbyte window by setting the SB(x)* bits.

Table 6: DRAM Addressing

SB3*	SB2*	SB1*	SB0*	DRAM on 512K board
1	1	1	1	00h to 0FFFFh
1	1	1	0	08000h to 017FFFh
1	1	0	1	010000h to 01FFFFh
1	1	0	0	018000h to 027FFFh
1	0	1	1	020000h to 02FFFFh
.				
.				
.				
0	0	1	0	068000h to 077FFFh
0	0	0	1	070000h to 07FFFFh
0	0	0	0	078000h to 07FFFh

Example: Write a 055h to location 013200h. The 64 Kbyte window on the host is from 0F0000h to 0FFFFFh (S2 = 0Fh).

Host enables the board with a write to the SLVSEL port.
Host writes a 01101h to the SBx* bits of the PORTL port.
Host writes the 055h to 0F3200h to complete transaction.

Care must be taken when DMA devices with real-time requirements access the SP186, because the 80186 isn't 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. This ensures that no host software manipulations will cause a loss of refresh on the SP186.

The DUART Input and Output Bits

The 2681 DUART handles:

- The serial receive/transmit function
- Several output bits necessary on the board
- Several input bits
- The timer/counter.

The 2861 can assert the INT1 interrupt on the 80186 using its INTRN output. The 16 ports of the DUART control these functions and are defined in the reprint of the 2681 Data sheet in Appendix A.

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 explain what this means to the circuitry.

Table 7: Output Bits of the 2681

Bit	Function	Description
OP7	CLRPERR	Enables and Disables DRAM parity checking. (Parity checking disabled on RESET*)
OP6	BANK*	Enables SP186s to swap the upper 256 Kbytes of memory with the lower. (Normal linear mapping on RESET*)
OP5	nc	
OP4	nc	
OP3	CLRINT86*	Asserts CLR to the flip-flop that controls INT0 to the 80186. (Allows INT0 to be asserted; INT0 could be asserted at power-on)
OP2	SETINT100	Provides the transition to cause a Vectored Interrupt to S-100 bus. (Initially high; needs to be set low, then high to cause interrupt)
OP1	RTS out B	Request to Send output B. If the SP186 is set to be DCE as shipped, this bit goes to RS-232 CTS pin 5 for channel B. If the SP186 is DTE, then this bit provides DTR pin 20. (Forces DTR or CTS to spacing, inhibiting transmission)
OP0	RTS out A	Request to Send output A. If the SP186 is set to be DCE as shipped, this bit goes to RS-232 CTS pin 5 for channel A. If the SP186 is DTE, then this bit provides DTR pin 20. (Forces DTR or CTS to spacing, inhibiting transmission)

The following table defines the input bits of the 2681:

Table 8: Input Bits of the 2681

Bit	Function	Description
IP6	nc	
IP5	nc	
IP4	nc	
IP3	DCD in B	Data Carrier Detect input B. This bit comes from the RS-232 Carrier Detect pin 8 channel B. This bit is normally only used when the SP186 is DTE.
IP2	DCD in A	Data Carrier Detect input A. This bit comes from the RS-232 Carrier Detect pin 8 channel A. This bit is normally only used when the SP186 is DTE.
IP1	CTS in B	Clear to Send input B. If the SP186 is set to be DCE as shipped, the bit comes from RS-232 DTR pin 20 for channel B. If the SP186 is DTE, this bit monitors the CTS pin 5 for channel B.
IP0	CTS in A	Clear to Send input A. If the SP186 is set to be DCE as shipped, the bit comes from RS-232 DTR pin 20 for channel A. If the SP186 is DTE, this bit monitors the CTS pin 5 for channel A.

DRAM Parity Checking

DRAM parity generation and checking is done in U41 and U42 (74LS280) and latched in U54 (74LS74). When a parity error is detected on the SP186, NMI* (non-maskable interrupt) is sent to the 80186 or an S-100 Vectored Interrupt can be asserted. Normally, this type of error causes the host to abort the processes in the particular slave due to potentially unreliable results within the slave memory. A memory check is then run to make sure that the error is soft (e.g., alpha particle), and new processes are loaded. When the system is RESET*, parity is disabled. If parity checking is not desired on the board, no action is necessary except to make sure that bit 7 in the 2681 OPR is never set.

To activate parity, the software must first initialize the parity check DRAM chips U20 and U29 by writing to every location. A loop that reads each location (wordwide, from 0h to 0FFFEh) and writes the same location back is effective in initializing the DRAM. Not until every byte has been initialized (written) can parity be activated. Set D7 in the 2681 OPR high to allow parity checking to start. Parity will be checked on every 80186 memory read (including opcode fetches) and on reads of DRAM from the host. Parity is generated on every write to the DRAM, whether from the 80186 or from the host system.

Multi-tasking

A unique banking feature allows the SP186 to swap the upper and lower 256 Kbyte bank in its memory map. This allows two processes, one running in the physically lower memory and one in the physically higher memory to think that they are both executing in lower memory with their own interrupt vector table. To conceptualize this, think of the total 1 Mbyte address range of the 80186 as four 256K regions, labeled A0, A1, A2, and A3. Next, think of the 512 Kbytes of physical memory as two 256 Kbyte regions, labeled M0 and M1. On 86RESET* bit 6 in the 2681 OPR is set low, negating BANK*. This results in the following memory map:

Table 9: 80186 Memory Map with BANK* Negated

80186 Memory	-->	Physical Memory
A0 (00h to 03FFFFh)		M0 (bank 0)
A1 (040000h to 07FFFFh)		M1 (bank 1)
A2 (080000h to 0BFFFFh)		M0 (bank 0)
A3 (0C0000h to 0FFFFFh)		M1 (bank 1)

NOTE: The 512 Kbyte memory appears in both the upper and lower halves of the 1 Mbyte range of the 80186 address space. This allows the 80186 DMA channel to refresh the 512 Kbytes even though it's scanning through all of the 1-Mbyte space. Memory also appears at the 80186's boot location 0FFFF0h without having to put separate memory there.

By setting bit 6 in the 2681 OPR, BANK* is asserted. This changes the memory map to the following:

Table 10: 80186 Memory Map with BANK* Asserted

80186 Memory	-->	Physical Memory
A0 (0h to 3FFFFh)		M1 (bank 1)
A1 (40000h to 7FFFFh)		M1 (bank 1)
A2 (80000h to BFFFFh)		M1 (bank 1)
A3 (C0000h to FFFFFh)		M1 (bank 1)

Physical memory bank 0 disappears and physical memory bank 1 appears in every 256 Kbyte 80186 memory bank. Refresh by the 80186 DMA controller continues to function properly.

Dual Ported Memory Arbitration

S-100 bus access to the resources of the SP186 is controlled by PAL U53 (part p368). It controls the necessary status and strobes to run an internal bus cycle. The PAL generates pRDY when the proper status is met, and does not end pRDY until ENDWAIT is generated by counter U4. PAL U53 also controls the assertion of PHANTOM* and SIXTN*, thus allowing the SP186 to overlap system memory and to respond and transfer a full 16 bits at a time. The chain of events for a single memory transfer is:

1. Host or DMA device requests memory (correct status and address).
2. SP186 puts host or DMA device in wait state and asserts 86HOLD to the 80186.
3. 80186 issues 86HLDA*, which puts data (on write) and address on the internal bus.
4. SYNCHLDA* is asserted, which starts counter U4, and causes a RAS* to start through U32 (74F00) and U5 (74AS74).
5. DRAM read or write cycle is performed depending on MEMW.
6. ENDWAIT is asserted, which terminates pRDY and drives data onto the S-100 bus for a read.

Half of counter U40 (74LS393) controls the hold off, which keeps the 80186 in hold by asserting 86HLDA* for a certain number of strobes after the last access. This counter also ensures that the host does not keep the 80186 in hold too long. If this happened, refresh of the DRAM could be lost. Therefore, the host or DMA device can't run more than eight cycles on the internal bus before the SP186 forces the host or DMA device to wait and run a cycle of its own. Refresh is maintained in this way.

SP186 Internal Timing

The 2681 interface is straightforward. When the 80186 asserts PCS0* and either 86WR* or 86RD*, the 2681 either accepts data or presents data to the bus AD7 through AD0. The address LA4-LA1 controls which section of the 2681 is accessed. Since the 80186 has a 16-bit bus and the 2681 has an 8-bit bus, all ports accessing the 2681 are even, and double what is shown in the 2681 manual (shifted to the left one bit). The 16 ports of the 2681 then appear at 00h, 02h, 04h, ... 01Ch, 01Eh.

The port map for the 2681 is in the reprint of its 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 80186 is reset as the 80186 RESET* output goes to the 2681. The INTRN output is able to assert INT1 to the 80186 through inverter U1.

The Dynamic RAM circuitry provides liberal timing to the RAM chips, while maintaining high speed.

- ALE's trailing (falling) edge provides the clock to U5. This starts the DRAM cycle because all addresses are valid at that edge.
- The delay introduced by L1 and C1 provides the RAS/address hold time, and U6 and PAL U7 provide the address/CAS setup time after the addresses have been multiplexed through U10, U11, and U8.
- On 80186 accesses, CAS is delayed until either 86RD* or 86DWR* is asserted.
- On a read, CAS is held until the trailing edge of 86RD*, so that data is properly latched into the 80186.
- On a write, data flows from the data lines into the DRAM when 86DWR* is asserted, and RAS and CAS are held until the end of 86DWR*.

When setting up the internal DMA counter and timer to cause a refresh, make sure that 256 consecutive words are refreshed over a total time of no more than every 4 milliseconds. If evenly spaced, this gives about 15 microseconds between word refreshes.

Programming for the SP186

Introduction

If you are running the SP186 under a CompuPro operating system, refer to the System Installation Guide for standard switch and jumper settings and for installation of the software to run 16-bit programs on the 186 slave.

If you are trying to bring up the SP186 in some other environment, study the code in Appendix B before you write any of your own code. Appendix B is a sample CP/M 86 program that loads some initialization code into the 80186 and starts it executing. No representation is made that this is the best way to program the SP186; it simply illuminates some of the possible pitfalls in getting the SP186 up and running.

The values loaded by the sample program into the internal registers of the 80186 and the 2681 DUART are typical values only, and may not be applicable to every system. Please consult the data sheet for the 80186 MPU and 2681 DUART if custom programming for the SP186 is done.

SP186 Hardware Initialization

The SP186 is designed so that up to 16 SP186 boards can all reside in the same 64K memory page and use the same two I/O ports. The S-100 host CPU must choose which SP186 it wants to talk to. If there is only one, it should have a select address of 00h, and thus the system should write a 00h to the select port (SELsp86 in the sample listing). The select value can be between 00h and 0Fh depending on which one of up to 16 SP186 boards you want to access. Even if the SP186 is the only board that resides in a particular memory page and I/O port, it must still be selected before the host CPU can access it.

When the board is powered up, the 80186 is in a reset state and it can not arbitrate for it's internal bus with the host processor. Any access to the SP186's memory would cause the S-100 bus to hang forever in a wait state, waiting for the 80186 to give the host processor the internal bus. It is imperative that the 80186 not be in a reset state when the host attempts to access the SP186 internal DRAM. Writing a 041h to the SP186 information port (INFsp86 in the sample listing) releases reset to the 80186, and sets external host window to the top 64K page of internal DRAM.

The 80186 is now running whatever random code happens to be at 0FFFF0h (the 80186 restart vector). The host CPU should then move a HALT instruction (opcode 0F4h) to location 0FFFF0h of the 80186, and then issue a reset back to the 80186.

After waiting to make sure that the 80186 gets reset in even the fastest systems (1 uSec is enough), let the 80186 run again, at which time it should pick up the 0F4h opcode and enter a HALT state. It is important for the host CPU to check that the 0F4h is still there, as it could possibly have been corrupted by the 80186 itself when the 80186 was running the arbitrary code. If 0F4h is not found, the process should be repeated until it is, guaranteeing the 80186 is now in a known state.

Refresh

Once the 80186 is in a HALT, the host can safely load code into the 80186 DRAM, including the code that initializes the DUART, the internal 80186 port map, and the code that causes the DMA counters to refresh (it should cause a refresh of a word at least every 15 microseconds). It is important that during this time the host maintains refresh on the DRAM by accessing any consecutive 256 words at least once every 4 milliseconds. Simply loading code to at least 256 words in a row can provide this refresh.

80186 Sample Listing

The sample loader in Appendix B initializes the 80186 as above by writing a 0 to the select port and calling a subroutine called "START_REFRESH".

The START_REFRESH routine first puts the 80186 in a halt and verifies that it got in a halt by checking that the HALT opcode is intact at location 0F000:FFF0H. It then loads the code at the label "DDRAM" to location 0F000:F000 in the 80186 memory map. The DDRAM code initializes the internal ports and DMA channels so that DMA channel 0 and TIMER 2 can take care of refreshing the DRAM.

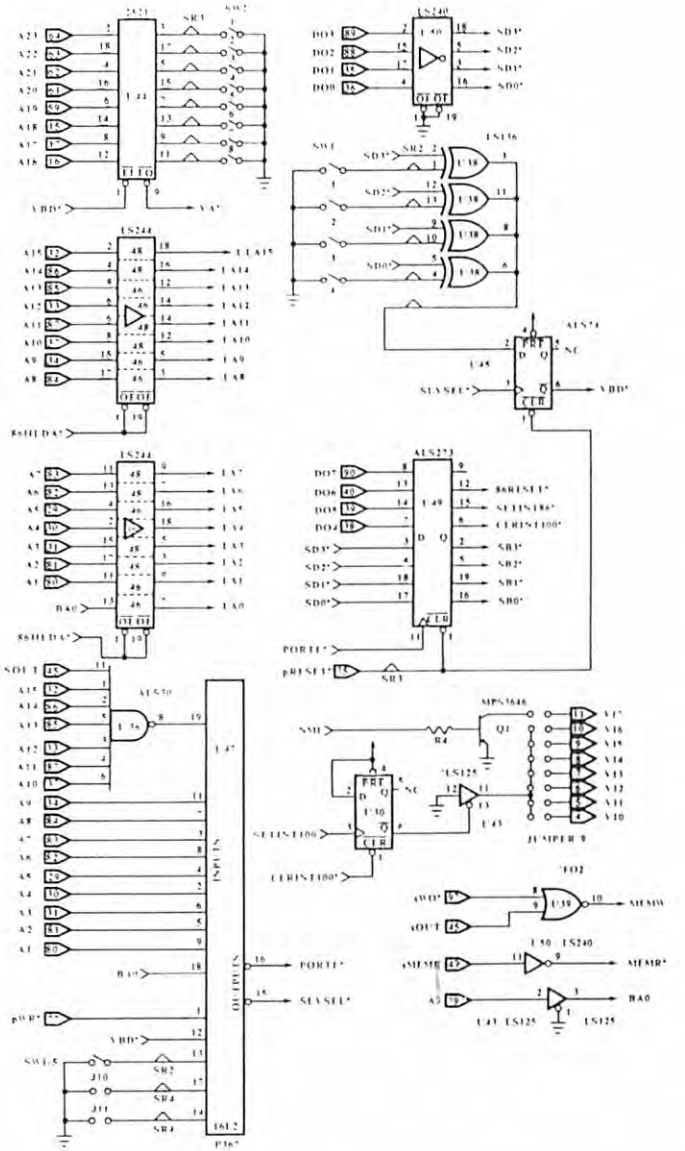
A far jump is then put at the 80186 restart vector (0F000:FFF0H) and the reset line is raised and lowered, causing the 80186 to execute the initialization code. The host processor will then wait for the 80186 to change the byte at F000:FFF0h to 0, signaling that the 80186 finished executing the initialization code. Once the 80186 has set

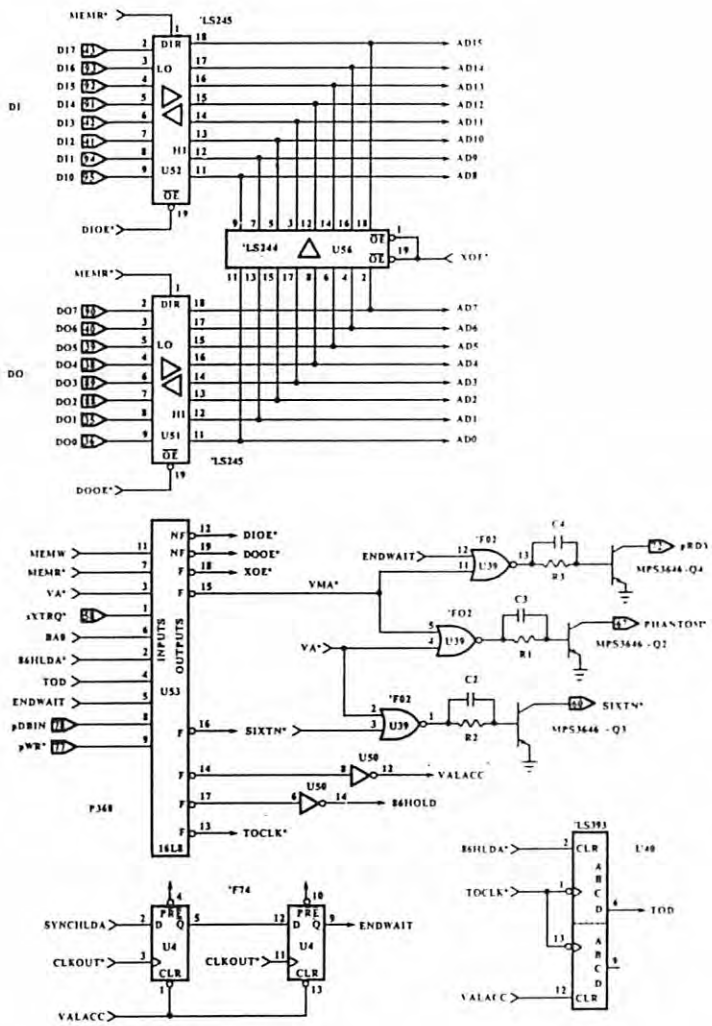
the flag to 0, signaling that it is finished with the initialization code, it reads that byte. When the 80186 reads 0FFh, it jumps to the location pointed to at 0F000:FFF1h.

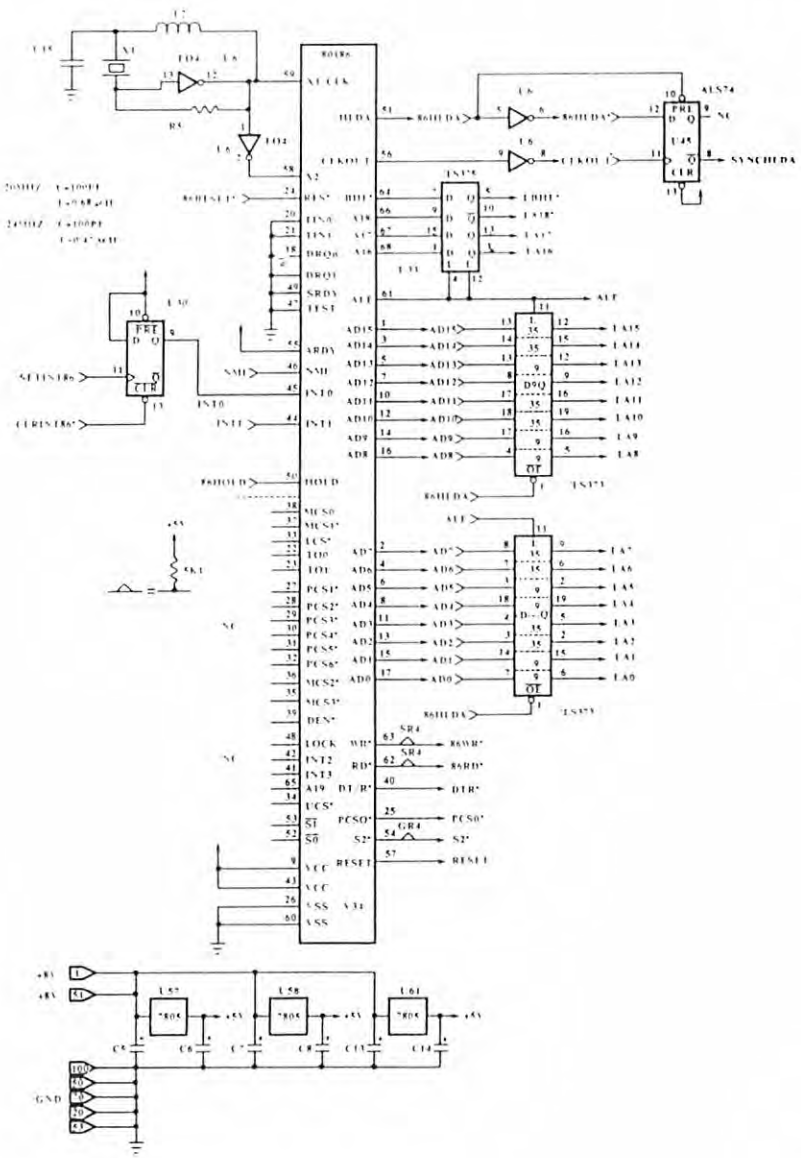
The host processor finally loads the runtime code SP186_CODE (this would normally be whatever runtime code was to be executed) to 40:80h in the SP186 RAM, fills in the jump vector, and signals the 80186 to execute.

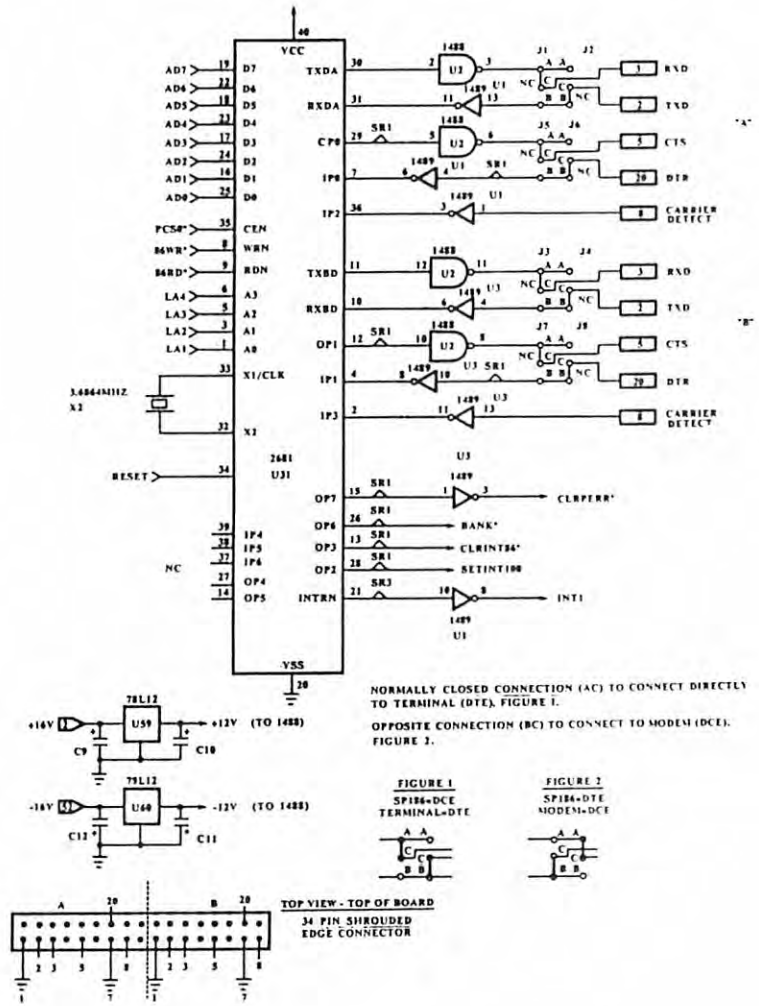
The 80186 jumps to 40:80h, the routine to set up the 80186 interrupt vectors, initialize the 2681 DUART and the 80186 interrupt controller, and print a message to a terminal hooked up to channel A at 19.2k baud. The 80186 then sits in a simple idle loop.

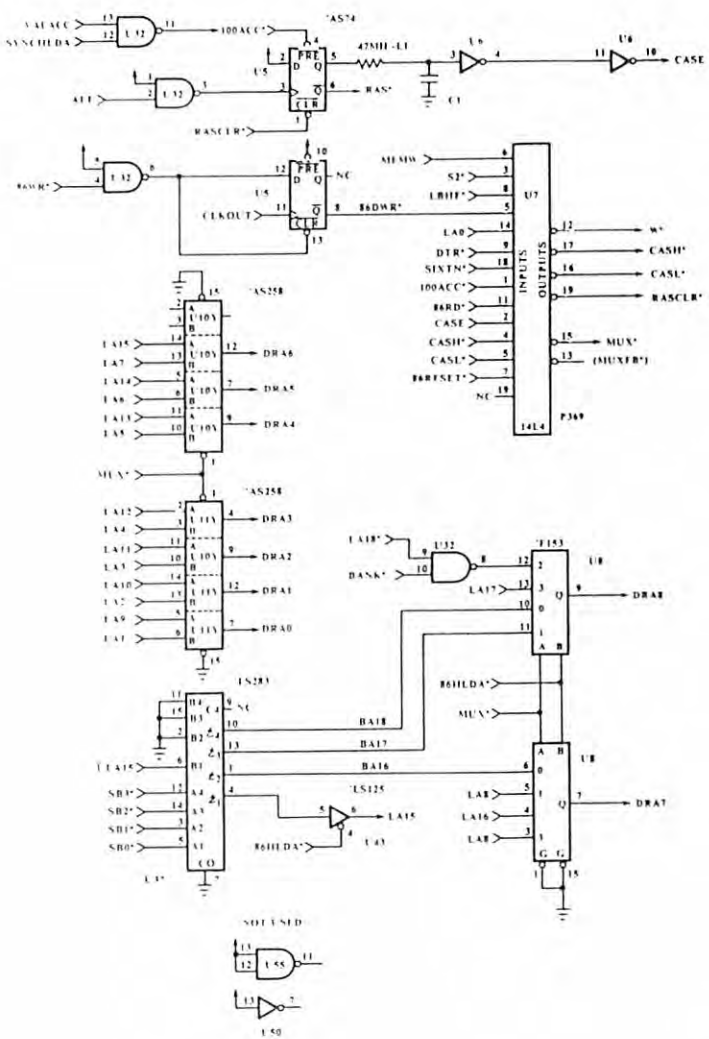
The HOST and DUART interrupt vectors are set to go to routines that will print a message to the terminal on channel A, send the 80186 EOI and return. All of the rest of the interrupts are set to go to a trap routine that will print the offending interrupt number on the terminal on channel A.

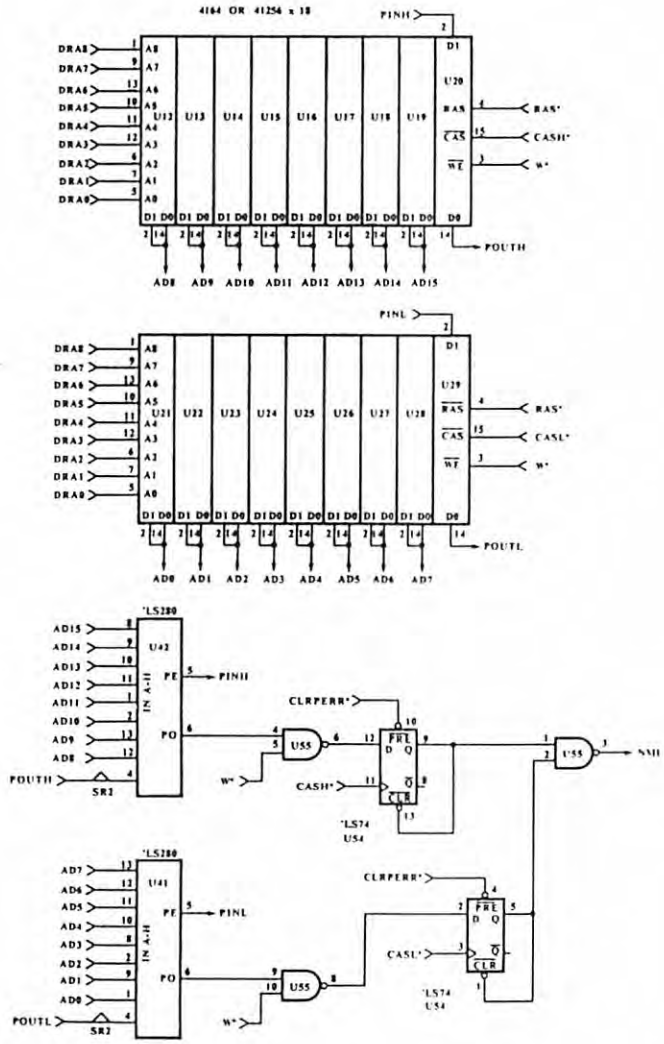







NORMALLY CLOSED CONNECTION (AC) TO CONNECT DIRECTLY TO TERMINAL (DTE). FIGURE 1.
OPPOSITE CONNECTION (BC) TO CONNECT TO MODEM (DCE). FIGURE 2.
FIGURE 1
SP186=DCE
TERMINAL=DTE
FIGURE 2
SP186=DTE
MODEM=DCE
TOP VIEW - TOP OF BOARD
34 PIN SHROUDED
EDGE CONNECTOR



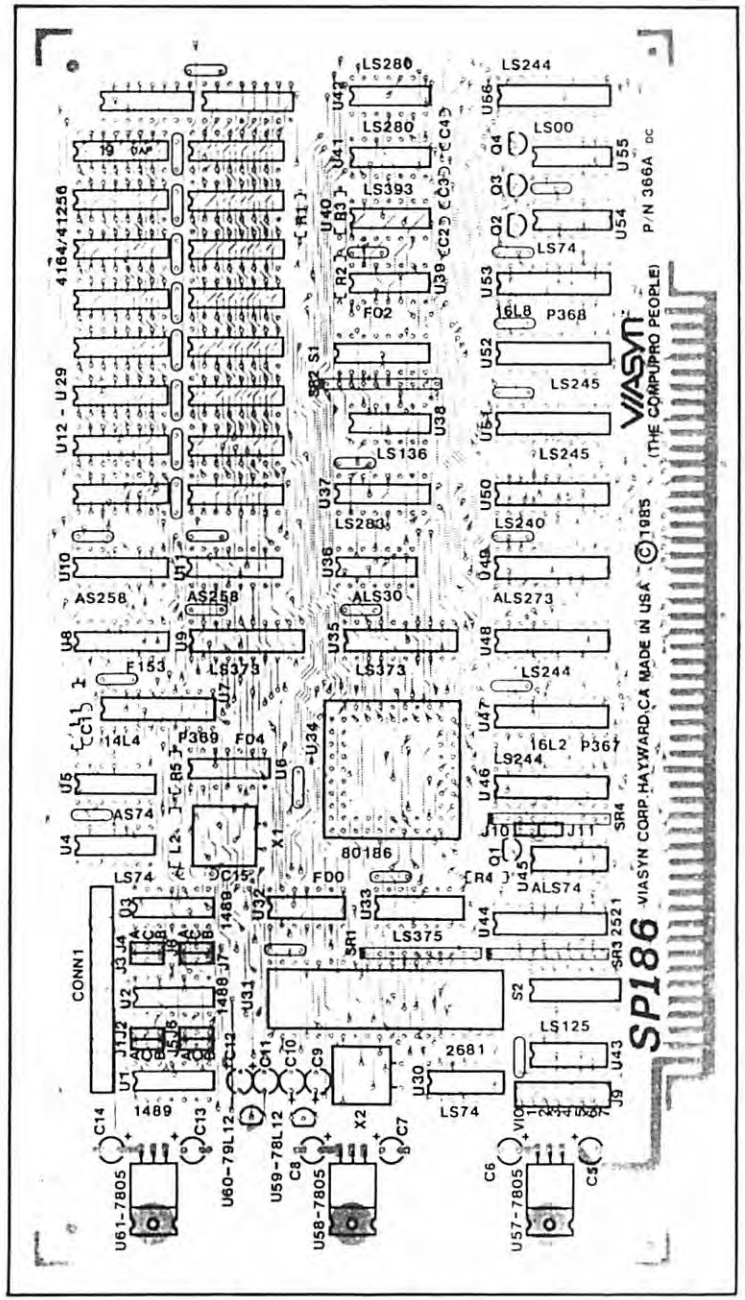


This page intentionally blank

Parts List

Name	CompuPro PN	Quantity	Location
Integrated Circuits			
74F00	3160-0000	1	U32
74LS00	3140-0000	1	U55
74F04	3160-0004	1	U6
74F02	3160-0002	1	U39
74ALS30	3150-0030	1	U36
74LS74	3140-0074	3	U4,U30,U54
74AS74	3145-0074	1	U5
74ALS74	3150-0074	1	U45
74LS125	3140-0125	1	U43
74LS136	3140-0136	1	U38
74F153	3160-0153	1	U8
74AS258	3145-0258	2	U10,U11
74LS240	3140-0240	1	U50
74LS244	3140-0244	3	U46,U48,U56
74LS245	3140-0245	2	U51,U52
74ALS273	3150-0273	1	U49
74LS280	3140-0280	2	U41,U42
74LS283	3140-0283	1	U37
74LS373	3140-0373	2	U9,U35
74LS375	3140-0375	1	U33
74LS393	3140-0393	1	U40
25LS2521	3140-1000	1	U44
1488	3186-0000	1	U2
1489	3186-0010	2	U1,U3
7805	3197-0020	3	U57,U58,U61
78L12	3197-0060	1	U59
79L12	3197-0070	1	U60
80186	3190-0025	1	U34
16L8	3135-0050	1	U53
16L2	3135-0040	1	U47
14L4	3135-0020	1	U7
4164/256	3172-4164/0000	18	U12-U29
2681	3186-2681	1	U31
Transistors			
MPS3646	4850-0040	4	Q1,Q2,Q3,Q4
Resistors			
5.1KSIP	4730-0210	4	SR1,SR2,SR3,SR4
1K5	4710-0460	3	R1,R2,R3
2K2	4710-0480	1	R4
1K0	4710-0440	1	R5

NAME	CompuPro PN	Quantity	Location
Capacitors			
Ceramic Bypass		27	all unmarked
Radial TANTALUM		10	C5-C14
47pF cer	1550-0010	3	C2,C3,C4
100pFmica	1530-0100	2	C1,C15
Inductors			
1.0uH	1810-0009	1	L2
10uH	1810-0016	1	L1
Crystals			
3.6864 MHz Fundamental		1	X2
20 MHz XTAL		1	X1
Miscellaneous			
TANDEM DIPSHUNT		1	FOR J9
8-POS DIPSWITCH		2	S1,S2



Component Layout

Appendix A
The DUART

MICROPROCESSOR DIVISION

DUAL ASYNCHRONOUS RECEIVER/TRANSMITTER (DUART) SC2681 SERIES

Preview

DESCRIPTION

The Signetics SC2681 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.

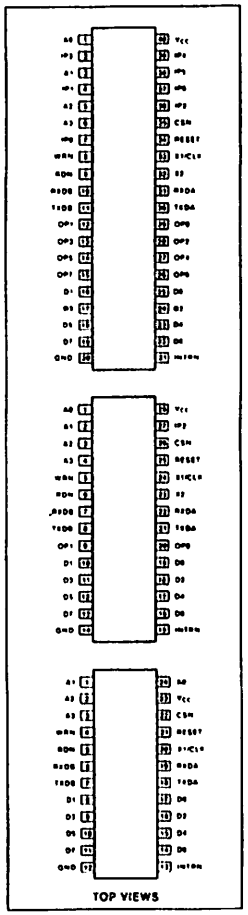
Also provided on the SC2681 are a multipurpose 7-bit input port and a multipurpose 8-bit output port. These can be used as general purpose I/O ports or can be assigned specific functions (such as clock inputs or status/interrupt outputs) under program control.

The SC2681 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/transmitter
- 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 transmitter 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	SC2681CSI24	SC2681CSI28	SC2681CSI40
Plastic DIP	SC2681CSN24	SC2681CSN28	SC2681CSN40

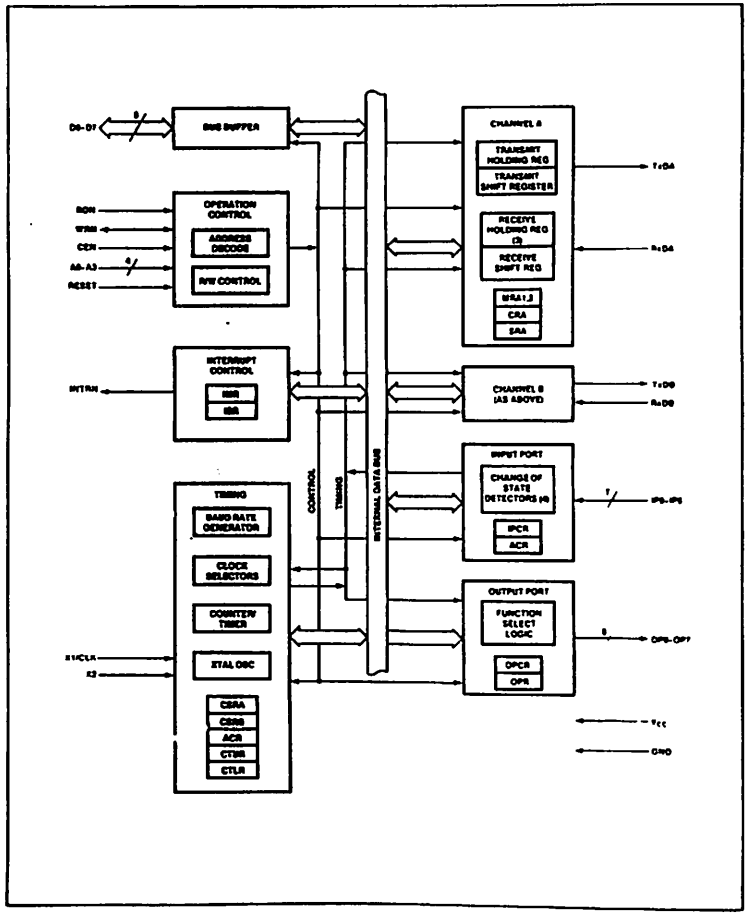
[1] 600 mil wide DIP
[2] 600 mil wide DIP

MICROPROCESSOR DIVISION

DUAL ASYNCHRONOUS RECEIVER/TRANSMITTER (DUART) SC2681 SERIES

Preview

BLOCK DIAGRAM



Preview

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.
X2	X	X		I	**Crystal 2:** Connection for other side of the crystal. Should be open if crystal is not used.
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	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.

Preview

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 (RxCA). 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 (DUART). 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. 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 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.

Preview

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 significant 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

Preview

Table 2. REGISTER BIT FORMATS

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

	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
MR2A MR2B	CHANNEL MODE 00 = Normal 01 = Auto echo 10 = Local loop 11 = Remote loop		Tx RTS CONTROL 0 = no 1 = yes	CTS ENABLE Tx 0 = no 1 = yes	STOP BIT LENGTH* 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

* For 5 bit codes 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 See text				TRANSMITTER CLOCK SELECT See text			

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

	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
SRA SRB	RECEIVED BREAK 0 = no 1 = yes *	FRAMING ERROR 0 = no 1 = yes *	PARITY ERROR 0 = no 1 = yes *	OVERRUN ERROR 0 = no 1 = yes	TxEMT 0 = no 1 = yes	TxRDY 0 = no 1 = yes	FFULL 0 = no 1 = yes	RxRDY 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 (character pointed to by FIFO pointer). 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
OPCR	OP7 0 = OPR[7] 1 = TxRDYB	OP6 0 = OPR[6] 1 = TxRDYA	OP5 0 = OPR[5] 1 = RxRDY/FFULLB	OP4 0 = OPR[4] 1 = RxRDY/FFULLA	OP3 00 = OPR[3] 01 = C/T OUTPUT 10 = TxCB (1X) 11 = RxCB (1X)		OP2 00 = OPR[2] 01 = TxCA (16X) 10 = TxCA (1X) 11 = RxCA (1X)	

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

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

Table 2. REGISTER BIT FORMATS (continued)

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

	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
IMR	IN. PORT CHANGE INT 0 = off 1 = on	DELTA BREAK B INT 0 = off 1 = on	RxRDY/FFULLB INT 0 = off 1 = on	TxRDYB INT 0 = off 1 = on	COUNTER READY INT 0 = off 1 = on	DELTA BREAK A INT 0 = off 1 = on	RxRDY/FFULLA INT 0 = off 1 = on	TxRDYA INT 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 reclocked and retransmitted on the TxDA output.
2. The receive clock is used for the transmitter.

Preview

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 deselection 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:

CSRA[7:4]	Baud Rate ACR[7] = 0	Baud Rate 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:

CSRA[3:0]	Baud Rate ACR[7] = 0	Baud Rate 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 — Access Type: Write Only

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:

CSRB[7:4]	Baud Rate ACR[7] = 0	Baud Rate 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:

CSRB[3:0]	Baud Rate ACR[7] = 0	Baud Rate 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.

Preview

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.

Preview

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 following:

- 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%.

Preview

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, IP0 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].

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

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.

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 MR2B[5]. 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 (SR[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 MR2A[5]. 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.

Preview

CTUR and CTLR — Counter/Timer Registers

The CTUR and CTLR hold the eight MSB s and eight LSB s respectively of the value to be used by the counter/timer in either the counter or timer modes of operation.

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 = 1111) 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 = 1110). 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 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.

Appendix B
SP186 Sample Code

```
;*****************************************************
;       SP186 Sample Initialization Code
;
;       Created: 12/18/85
;*****************************************************
;
nolist

=           include SW186.INC     ;SP186 board & 2681 DUART equates
=
=           ;
=           ;  Sp186 equate file
=           ;
= 0080    OS_OFFSET       EQU   80H       ;Leave room for stack
= 0040    OS_SEG          EQU   40H       ;Leave room for interrupt vectors
= F000    DDRAM_CS        EQU   0F000h    ;Address of 186 initialization code
=                                         ; CS: for DDRAM is F000: page 15
=           ;-------------------------------------------------------
=           ;
=           ;  sp86 I/O equates
=           ;
= 0004    sp86delay       EQU     4
= F000    sp86PG  EQU     0F000H
= FC42    SELsp86 EQU     0FC42h          ;Lower four bits are binary number of
=                                         ;desired sp86 board
= FC43    INFsp86 EQU     SELSP86+1       ;PortL. See spec.
= 00BF    RSTsp86 EQU     10111111b
=
= 00FE    PG0sp86 EQU     11111110b       ;Turn bit(s) low to select which 64k window
= 00FD    PG1sp86 EQU     11111101b
= 00FC    PG2sp86 EQU     11111100b
= 00FB    PG3sp86 EQU     11111011b
= FFFA    PG4sp86 EQU     not 05h
= FFF9    PG5sp86 EQU     not 06h
= FFF8    PG6sp86 EQU     not 07h
= FFF7    PG7sp86 EQU     not 08h
=           ;
=           ;
=           ; I/O Address of 2681 UART
= 0000    DUART   EQU     0000h           ;PCS0 Select (from 186 only) 0000-7Fh.
=           ; 2615 uart register offset equates
=           ;
= 0000    MRA     EQU     0               ;Mode register A (r/w)
= 0002    SRA     EQU     2               ;Status register A (r)
= 0002    CSRA    EQU     2               ;Clock select register A (w)
= 0004    CRA     EQU     4               ;Command register A (w)
= 0006    RHRA    EQU     6               ;Rx holding register A (r)
= 0006    THRA    EQU     6               ;Tx holding register A (w)
```



```
= 0008    IPCR    EQU     8               ;Input Port change register (r)
= 0008    ACR     EQU     8               ;Auxilary Control Register (w)
= 000A    ISR     EQU     10              ;Interrupt Status Register (r)
= 000A    IMR     EQU     10              ;Interrupt Mask Register (w)
= 000C    CTU     EQU     12              ;Counter/timer Upper (r)
= 000E    CTLOW   EQU     14              ;Counter/Timer Lower (r)
= 000C    CTUR    EQU     12              ;C/T Upper Register (r)
= 000E    CTLOWR  EQU     14              ;C/T Lower Register (w)
= 0010    MRB     EQU     16              ;Mode register B (r/w)
= 0012    SRB     EQU     18              ;Status register B (r)
= 0012    CSRB    EQU     18              ;Clock select register B (w)
= 0014    CRB     EQU     20              ;Command register B (w)
= 0016    RHRB    EQU     22              ;Rx holding register B (r)
= 0016    THRB    EQU     22              ;Tx holding register B (w)
= 001A    INPRT   EQU     26              ;Input Port (r)
= 001A    OPCR    EQU     26              ;Output Port (w)
= 001C    SCC     EQU     28              ;Start counter command (r)
= 001C    SETOUT  EQU     28              ;Set output port bits command (w)
= 001E    STCC    EQU     30              ;Stop counter command (r)
= 001E    RSETOUT EQU     30              ;Reset output port bits command (w)
=           ;
=           ; OUTPUT Port Bits
= 0080    CLRPERR EQU 10000000b
= 0040    BANK    EQU 01000000b
=
= 0080    ZINT    EQU 10000000b
= 0040    VINT    EQU 01000000b
=           ;  Bit D3 is fed back to input 3
=           ;  Bit D2 is fed back to input 2
= 0002    CTSB    EQU 00000010b     ;Channel B CTS
= 0001    CTSA    EQU 00000001b     ;Channel A CTS
=           ;
=           ;  INPUT Port Bits
= 0020    SB1     EQU 00100000b     ;Reads the currently selected host page
= 0010    SB0     EQU 00010000b     ;(00=page 0, 01=page 1, 10=page 2,11=page 3)
=           ; Bit D3 is fed output from output D3
=           ; Bit D2 is fed output from output D2
= 0002    DTRB    EQU 00000010b     ;DTR status for channel B
= 0002    DTRA    EQU 00000010b     ;DTR status for channel A
*           ;
*           ;
*           ;-------------------------------------------------------
*           ;
*           ; Some 86 instruction op-codes
* 00F4    HLT86   EQU     0F4h        ;Hlt
  00EA    JmpF86  EQU     0EAh        ;JmpF
          ;
          ;-----------------------------------
          ;
          ; SYSTEM Support 1 equates
  0040    INTBASE EQU     40H
  0050    SS1MP0  EQU     50H
```

```
= 0051    SS1MP1  EQU  51H
= 0052    SS1SP0  EQU  52H
= 0053    SS1SP1  EQU  53H
= 0060    SEOI    EQU  60H
=         ;
=         ;----------------

=           include 188.INC      ;80188 internal register equates
=         ;
=         ;
=         ; <80188 EQUATES>
=         ;
= FF00    i_CTL_BLK     EQU  0FF00h              ;power-up CNTL Block base address
= FFA0    i_UMCS        EQU  i_CTL_BLK + 0A0h    ;Upper Memory Chip Select
= FFA2    i_LMCS        EQU  i_CTL_BLK + 0A2h    ;Lower Memory Chip Select
= FFA4    i_PACS        EQU  i_CTL_BLK + 0A4h    ;Peripheral Chip Select
= FFA8    i_MPCS        EQU  i_CTL_BLK + 0A8h    ;Mid-Range Chip Select
=
= FFC0    i_DMA0SL      EQU  i_CTL_BLK + 0C0h    ;DMA#0 Source  low half
= FFC2    i_DMA0SH      EQU  i_CTL_BLK + 0C2h    ;DMA#0 Source high half
= FFC4    i_DMA0DL      EQU  i_CTL_BLK + 0C4h    ;DMA#0 destination  low half
= FFC6    i_DMA0DH      EQU  i_CTL_BLK + 0C6h    ;DMA#0       "      high half
= FFC8    i_DMA0TC      EQU  i_CTL_BLK + 0C8h    ;DMA#0 Txfer count reg.
= FFCA    i_DMA0CW      EQU  i_CTL_BLK + 0CAh    ;DMA#0 CNTL work reg.
=
= FFD0    i_DMA1SL      EQU  i_CTL_BLK + 0D0h    ;DMA#1 Source  low half
= FFD2    i_DMA1SH      EQU  i_CTL_BLK + 0D2h    ;DMA#1 Source high half
= FFD4    i_DMA1DL      EQU  i_CTL_BLK + 0D4h    ;DMA#1 destination  low half
= FFD6    i_DMA1DH      LQU  i_CTL_BLK + 0D6h    ;DMA#1       "      high half
= FFD8    i_DMA1TC      EQU  i_CTL_BLK + 0D8h    ;DMA#1 Txfer count reg.
= FFDA    i_DMA1CW      EQU  i_CTL_BLK + 0DAh    ;DMA#1 CNTL work reg.
=
= FF50    i_T0CNT       EQU  i_CTL_BLK + 50h     ;Timer#0 Count reg.
= FF52    i_T0MAXA      EQU  i_CTL_BLK + 52h     ;Timer#0 Max Count A reg.
= FF54    i_T0MAXB      EQU  i_CTL_BLK + 54h     ;Timer#0 Max Count B reg.
= FF56    i_T0M_CTL     EQU  i_CTL_BLK + 56h     ;Timer#0 Mode/CoNTroL reg.
=
= FF58    i_T1CNT       EQU  i_CTL_BLK + 58h     ;Timer#1 Count reg.
= FF5A    i_T1MAXA      EQU  i_CTL_BLK + 5Ah     ;Timer#1 Max Count A reg.
= FF5C    i_T1MAXB      EQU  i_CTL_BLK + 5Ch     ;Timer#1 Max Count B reg.
= FF5E    i_T1M_CTL     EQU  i_CTL_BLK + 5Eh     ;Timer#1 Mode/CoNTroL reg.
=
= FF60    i_T2CNT       EQU  i_CTL_BLK + 60h     ;Timer#2 Count reg.
= FF62    i_T2MAX       EQU  i_CTL_BLK + 62h     ;Timer#2 Max Count REG.
= FF66    i_T2M_CTL     EQU  i_CTL_BLK + 66h     ;Timer#2 Mode/CoNTroL reg.
=
= FF22    i_iEOI        EQU  i_CTL_BLK + 22h     ;End-Of-Interrupt
= FF24    i_iPOLL       EQU  i_CTL_BLK + 24h
= FF26    i_iPOLLSTAT   EQU  i_CTL_BLK + 26h
= FF28    i_iMASK       EQU  i_CTL_BLK + 28h
= FF2A    i_iPRIOMASK   EQU  i_CTL_BLK + 2Ah
= FF2C    i_iINSER      EQU  i_CTL_BLK + 2Ch     ;"IN_SERVICE" reg
```



```
= FF2E    i_iREQ        EQU  i_CTL_BLK + 2Eh     ;Interrupt REQuest
= FF30    i_iSTAT       EQU  i_CTL_BLK + 30h     ;Interrupt Controller Status
= FF32    i_iTIMCTL     EQU  i_CTL_BLK + 32h     ;Timer Control reg
= FF34    i_iDMA0CTL    EQU  i_CTL_BLK + 34h     ;DMA Controls
= FF36    i_iDMA1CTL    EQU  i_CTL_BLK + 36h
= FF38    i_int0CTL     EQU  i_CTL_BLK + 38h
= FF3A    i_int1CTL     EQU  i_CTL_BLK + 3Ah     ;INT Chnnel Controls
= FF3C    i_int2CTL     EQU  i_CTL_BLK + 3Ch
= FF3E    i_int3CTL     EQU  i_CTL_BLK + 3Eh
=
= 0000    ICW1          EQU  0h                  ;8259a port offsets
= 0001    ICW2          EQU  1h
= 0001    ICW3          EQU  1h
= 0001    ICW4          EQU  1h
= 0001    OCW1          EQU  1h
= 0000    OCW2          EQU  0h
= 0000    OCW3          EQU  0h
= 0000    OCW4          EQU  0h
=
=         ; 188 specific instructions
=         ;
=         Codemacro   PUSHA          ;Saves Regs: AX,CX,DX,BX,SP,BP,SI,DI.
=                 DB 01100000b
=         EndM
=
=         Codemacro   POPA                    ;Restore in reverse order as PUSHa
=                 DB 01100001b                ;SP popped but discarded.
=         EndM
=
=         Codemacro   PUSHW Wordd:Dw          ;Push Immediate Word to Stack
=                 DB 01101000b
=                 DW Wordd
=         EndM
=
=         Codemacro   PUSHb BByte:Db          ;Push Immediate Byte to Stack
=                 DB 01101000b
=                 DW BByte
=         EndM
=
=         ; SHIFT IMMediate w/REGISTER ONLY FOR NOW
=         ;
=         Codemacro   SHLRw Reg:R, Count:Db   ;Shift Word Register Logical Left
=                 DB 11010001b                ;w/immediate byte
=                 MODRM 4,Reg
=                 DB count
=         EndM
=
=         Codemacro   SHLRb Reg:R, Count:Db   ;Same as SHLRw but now w/Byte.
=                 DB 11010000b
=                 MODRM 4,Reg
=                 DB count
=         EndM
```

```
                Codemacro  SHRrw Reg:R, Count:Db ;Same as SHLrw but now SHIFT Right
                        DB 11010001b
                        MODRM 4,Reg
                        DB count
                EndM

                Codemacro       SHRrb Reg:R, count:Db   ;Same as SHRrw but w/Byte
                        DB 11010000b
                        MODRM 4,Reg
                        DB count
                EndM

                ;
                CODEMacro CSSEG
                        DB      2EH
                EndM
                        CSEG
                INIT_SP186:
0000 90                 NOP
                ;
0001 B80000             MOV     AX,0            ;Get current board number
                START_186:
0004 BA42FC             MOV     DX,SELSP86      ;Get SP186 select port number
0007 EE                 OUT     DX,AL           ;Select the SP186
0008 42                 INC     DX              ;Point to SP186 information port
0009 B041               MOV     AL,41H          ;Release reset, point to last
                                                ;possible 64K
000B EE                 OUT     DX,AL           ;
000C B800F08EC0         MOV AX,SP86PG ! MOV ES,AX ;Point to SP186 memory page

0011 E85D00     0071    CALL    START_REFRESH   ;Start 186 DMA0 refreshing the DRAM

0014 BA43FC             MOV     DX,INFSP86      ;Point to SP186 information port
0017 B04F               MOV     AL,4FH          ;Point to bottom 64K page
0019 EE                 OUT     DX,AL

001A 1E                 PUSH DS
001B 0E1F               PUSH CS ! POP DS
001D BF8004             MOV DI,OS_OFFSET+(OS_SEG shl 4) ;Point to destination
                                                        ;of 186 code
0020 BE2701     R       MOV  SI,offset SP186_CODE  ;Point to start of 186 code
                                                   ;locally

0023 B90801             MOV CX,(SP186_CODE_SIZE/2) ;Get length of 186 code
0026 F3A5               REP     MOVSW           ;Move 186 code in place
0028 1F                 POP     DS
0029 26C706000400       MOV ES:word ptr .400H,0 ;Wait until 186 changes this to non 0
     00
0030 BA43FC             MOV     DX,INFSP86      ;Point to SP186 information port
0033 B041               MOV     AL,41H          ;Point to top 64K page
0035 EE                 OUT     DX,AL
```



```
0036 1E                 PUSH    DS
0037 0E1F               PUSH CS ! POP DS
0039 BECB00   R         MOV SI,offset JMPF_BEGIN+4 ;Point to far jump code
003C BFF4FF             MOV     DI,0FFF4H   ;Point to end of vector
003F FD                 STD             ;Do backwards move so flag is moved last
0040 B90500             MOV     CX,5        ;Move flag,offset and segment
0043 F3A4               REP     MOVSB   ;Move new offset, segment and execute
                                        ;flag in place,
0045 FC                 CLD                     ;Reset move direction
0046 1F                 POP     DS
0047 BA43FC             MOV     DX,INFSP86  ;Point to SP186 information port
004A B04F               MOV     AL,4FH      ;Point to back to bottom 64K page
004C EE                 OUT     DX,AL
                ;
004D 5058       WAITE:  PUSH AX ! POP AX
004F B840F08EC0         MOV  AX,SP86PG+40H ! MOV ES,AX  ;SP186 segment to ES
0054 26833E000000       CMP ES:word ptr .0,0    ;Wait for SP186 to finish
005A 74F1       004D    JE      WAITE           ;initialization
                ;
                ;
005C B04F               MOV     AL,4FH
005E BA43FC             MOV     DX,INFSP86
0061 EE                 OUT     DX,AL           ;Clear any pending VI from 186
                ;*      MOV     AL,5FH
                ;*      OUT     DX,AL           ;Enable VIs from this SP186

0062 B109               MOV     CL,S_PRINT      ;CCP/M print string function
0064 BA0001   R         MOV  DX,offset SP186MSG ;Point to signon message
0067 CDE0               INT     BDOS            ;Tell world we started 186
                ;
0069 B100               MOV     CL,0
006B BA0000             MOV     DX,0
006E CDE0               INT     BDOS
                ;
0070 C3                 RET
                ;
                        DSEG
                        ORG     100H
0100 535031383620SP186MSG DB    'SP186 has been initialized',CR,LF,EOS
     686173206265
     656E20696E69
     7469616C697A
     65640D0A24
                ;
                ;**********************************
                ;
                        CSEG
                ;
                ; Move the code needed for the 186 to start it's DMA0 channel
                ; to do refresh for the DRAM and start the 186 executing it
                ; RAM window is already at page 0f0000h
```

```
                        ;
                        START_REFRESH:
0071 BBF0FF             MOV     BX,0FFF0H       ;Point to 186 reset vector
0074 26C607F4  HALT86:  MOV ES:Byte Ptr [BX],HLT86 ;Put 186 halt instruction
                                                ;at reset vector
0078 E83700    00B2     CALL    START_86        ;Release reset to start 186
007B 2680F4F4           CMP ES:Byte Ptr [BX],HLT86 ;Verify that halt instruction
                                                ;is still there
007F 75F3      0074     JNE     HALT86          ;If 186 munched halt instruction,
                                                ;try again
                        ;
0081 1E        RFSH:    PUSH    DS
0082 0E1F               PUSH CS ! POP DS
0084 8BFB               MOV     DI,BX
0086 BEC100  R          MOV SI,offset JMPF_RFSH ;Point to refresh init Jump
0089 B90300             MOV     CX,JMPF_SIZE/2
008C F3A5               REP     MOVSW           ;Move jump to refresh code inplace
008E BECC00  R          MOV SI,offset DDRAM
0091 BF00F0             MOV     DI,DDRAM_CS     ;Point to location within the 186
                                                ;for refresh code
0094 B94200             MOV     CX,128-(DDRAM_SIZE/2)
0097 F3A5               REP     MOVSW
0099 1F                 POP     DS
009A E81500    00B2     CALL    START_86        ;Start 186 executing refresh code
                        ;
009D 505850581 RFSH1:   PUSH AX ! POP AX ! PUSH AX ! POP AX ;Waste some time
00A1 505850581          PUSH AX ! POP AX ! PUSH AX ! POP AX ;Waste some time

00A5 2680FFEA           CMP ES:Byte Ptr [BX],JMPF86 ;See if JMPF still there
00A9 74F2      009D     JE      RFSH1           ;Just wait if so

00AB 2680FF00           CMP ES:Byte Ptr [BX],0  ;See if 186 finished yet
00AF 75D0      0081     JNE     RFSH            ;Start over if not JMPF or 00
00B1 C3                 RET                     ;Else 186 is waiting for us
                        ;

                        ;***********************************************
                        ;
                        ; Twiddle RESET to the 186 to start/restart it executing
                        ; Entry: DX = SP186 information port
                        ;        186 Reset vector at 0F000:FFF0H
                        ; Exit: 186 executing  at reset vector
                        ;
                        START_86:
00B2 B001               MOV     AL,01H          ;Drop reset to the 186
00B4 EE                 OUT     DX,AL
00B5 5058               PUSH AX ! POP AX        ;Waste some time for things to settle
00B7 5058               PUSH AX ! POP AX
00B9 B041               MOV     AL,41H          ;Release reset to the 186
00BB EE                 OUT     DX,AL
00BC 5058               PUSH AX ! POP AX        ;Waste some time for things to settle
00BE 5058               PUSH AX ! POP AX
00C0 C3                 RET
                        ;
                        ;
                        ;=======================================
```



```
                        ; All code below here is run by the SP186
                        ;=======================================
                        JMPF_RFSH:
00C1 EA                 DB      JMPF86          ;186 far jump op code
00C2 00F0               DW      DDRAM_CS        ;Offset for DRAM init code
00C4 00F0               DW      0F000H          ;Segment of DRAM init code
00C6 90                 DB      90H             ;NOP (to make even words)
  0006          JMPF_SIZE EQU   offset $ - offset JMPF_RFSH ; Length of far
                                                ; jump in bytes
                        JMPF_BEGIN:
00C7 FF                 DB      0FFH            ;Begin execution flag
00C8 8000               DW  ((BEGIN_IP+OS_OFFSET)-SP186_CODE)
00CA 4000               DW      OS_SEG
                        ;
                        ; DRAM refresh code - DMA channel 0 is set up to
                        ;  continuly cycle to do refresh on all of the DRAM
                        ;  We will also set up many other things in the 186 here
  F000          SP186OF EQU     0F000H ;Offset for SP186 initialization code
                        ;
00CC 8CC88ED8  DDRAM:   MOV AX,CS ! MOV DS,AX   ;Set code and data same for nov
00D0 8ED0               MOV     SS,AX
00D2 BC00F0             MOV     SP,SP186OF      ;Set up the stack
00D5 BB2DF0             MOV BX,(offset I86_TAB - offset DDRAM) + SP186OF
00D8 8B17               MOV     DX,[BX]         ;Get first port to use
                        INITRFSH:
00DA 8B4702             MOV     AX,2[BX]        ;Pick up data value
00DD EF                 OUT     DX,AX
00DE 83C304             ADD     BX,4            ;Bump to next port/
                                                ;data combination
00E1 8B17               MOV     DX,[BX]         ;Pick up next port to use
00E3 83FAFF             CMP     DX,0FFFFH       ;See if reached end of table
00E6 75F2      00DA     JNZ     INITRFSH        ;Send entire table
00E8 909090             NOP ! NOP ! NOP
00EB BBF0FF             MOV     BX,0FFF0H       ;Point to 186 restart vector
00EE C60700             MOV Byte Ptr [BX],0     ;Tell S100 bus we finished
                                                ;init code
                        WAIT_S100:
00F1 803FFF             CMP Byte Ptr [BX],0FFH  ;Wait until S100 bus starts
                                                ;us running
00F4 75FB      00F1     JNE     WAIT_S100
00F6 FF6F01             JMPF DWord Ptr 1[BX]    ;Jump into new code segment

                        ;
                        ;-------------------------------------------------------
                        ; 80186 INTERNAL INITIALIZATION
                        ;
                        ; TIMER #2, DMA1 for DRAM refresh
                        ; Destination SYNC will allow a complete CPU cylcle to
                        ; be run between DMA refresh transfer
                        ;
 74B7 DMA1W     EQU     74B7H
                        ;
                        ; PCS6 FIXED              NO  NO    DEST    TIM#2     YEH     WORD
                        ;       ADDRESS MEM   INC. STOP INT   SYNC     CTRL      RUN!   TXFER
                        ; ------------------------------+-----------------------------
                        ; | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |
```

```
                ;       ---------------------------------+-------------------------
                ;   [ Destination  +  Source ]                    ^
                ;                                                 HIGH PRIORITY
                ;-------------------------
                I86_TAB:
00F9 A4FF           DW      I_PACS
00FB 3E00           DW      003EH           ;Duart at port 0, 2 I/O wait states
00FD A8FF           DW      I_MPCS          ;Mid range selects are don't care,
                                            ;0 wait states
00FF BC81           DW      81BCH           ;Peripherals mapped to I/O, A1 & A2
                                            ;provided
0101 60FF           DW      I_T2CNT
0103 0000           DW      0000            ;Current timer count value starts at 0
0105 62FF           DW      I_T2MAX
0107 1800           DW      18H             ;Max timer count for 16 us refresh count
0109 66FF           DW      I_T2M_CTL
010B 01C0           DW      0C001H          ;ENable, Inhibit high, Continuous bit set
010D C0FF           DW      I_DMA0SL
010F 0000           DW      0000            ;Set low word of DMA source to 0
0111 C2FF           DW      I_DMA0SH
0113 0000           DW      0000            ;Set high word of DMA source to 0
0115 C4FF           DW      I_DMA0DL
0117 C800           DW      0200            ;Destination: PCS6 port # 300h
0119 C6FF           DW      I_DMA0DH        ;Use this as DMA/RFSH ACK !
011B 0000           DW      0000H           ;DMA channel #0 control word
011D C8FF           DW      I_DMA0TC        ;Transfer count register
011F 0000           DW      0000            ;Set up for max transfer
0121 CAFF           DW      I_DMA0CW        ;DMA control work register
0123 9774           DW      7497H           ;Dest is I/O,Dest pointer constant,
                                            ;Source is mem
                                            ;Source pointer incs,dest sync,DMA tmr2,
                ;                           ;start,word
0125 FFFF           DW      0FFFFH          ;Mark end of table
  005D      DDRAM_SIZE  EQU     (offset $ - OFFSET ddram + 2)
                ;
                ;-----------------------------------------
                ;*******************************************************
                SP186_CODE:
                ; This is the actual runtime code that the 186 will execute
                BEGIN_IP:
0127 8CC88ED8       MOV AX,CS ! MOV DS,AX ;Set up for 8080 model for now
012B 8EC0           MOV     ES,AX           ;So all registers are the same
012D 8ED0           MOV     SS,AX
012F BCF200         MOV     SP,TOS          ;Set up stack pointer
0132 BFD602         MOV     DI,INTWORK
0135 B8CCCC         MOV     AX,0CCCCH       ;86 INT 3 instruction
0138 B98000         MOV     CX,256/2
013B F3AB           REP     STOSW           ;Fill uninitalized interrupt table
                                            ;with INT 3
                ;
013D 33C0           XOR     AX,AX
013F 8BF8           MOV     DI,AX
0141 8EC0           MOV     ES,AX
0143 BBD602         MOV     BX,INTWORK      ;Point to interrupt vector table
0146 B90001         MOV     CX,256          ;Fill all 186 interrupt vectors
0149 8CC8           MOV     AX,CS
                FILLINTS:
```



```
014B 93             XCHG    AX,BX           ;INTWORK pointer to AX,save CS in BX
014C AB                     STOSW           ;Put INTWORK offset in int vector
014D 93             XCHG    AX,BX           ;Get INTWORK segment back to AX,save
                                            ;INTWORK offset
014E AB                     STOSW           ;Put CS segment in int vector
014F 43             INC     BX              ;Bump to next INTWORK spot
0150 E2F9   014B    LOOP    FILLINTS        ;Fill all 256 interrupt vectors
                ;
0152 BB0C00         MOV     BX,3*4          ;Point to INT3 vector
0155 26C7076601     MOV ES:Word Ptr [BX],(offset TRAP_ROUTINE - SP186_CODE)+OS_OFFSET
015A BB3000         MOV     BX,12*4         ;Point to INT12 (186 INT0)
015D 26C707C801     MOV ES:Word Ptr [BX],(offset HOST_INT - SP186_CODE)+OS_OFFSET
0162 BB3400         MOV     BX,13*4         ;Point to INT13 (186 INT1)
0165 26C7079101     MOV ES:Word Ptr [BX],(offset DUART_INT - SP186_CODE)+OS_OFFSET
                ;
                INIT2681:
016A BBF700         MOV     BX,I2681_TAB
016D 8B17           MOV     DX,[BX]         ;Get first port to use
                IO_INIT:
016F 8A4702         MOV     AL,2[BX]        ;Pick up data value
0172 EE             OUT     DX,AL
0173 83C303         ADD     BX,3            ;Bump to next port/data combination
0176 8B17           MOV     DX,[BX]         ;Pick up next port to use
0178 83FAFF         CMP     DX,0FFFFH       ;See if reached end of table
017B 75F2   016F    JNZ     IO_INIT
                ;
017D BB2601         MOV     BX,INT_TAB      ;Point to DUART/186 initialization table
0180 8B17           MOV     DX,[BX]         ;Get first port to use
                INITINTS:
0182 8B4702         MOV     AX,2[BX]        ;Pick up data value
0185 EF             OUT     DX,AX
0186 83C304         ADD     BX,4            ;Bump to next port/data combination
0189 8B17           MOV     DX,[BX]         ;Pick up next port to use
018B 83FAFF         CMP     DX,0FFFFH       ;See if reached end of table
018E 75F2   0182    JNZ     INITINTS
                ;
0190 8C0ED402       MOV     .LOCSEG,CS      ;Save our local code segment for ints
0194 8CD88EC0       MOV AX,DS ! MOV ES,AX   ;Set up for 8080 model
                ;
                    if (((offset $-SP186_CODE)+OS_OFFSET)  MOD 2) eq 1
0198 90             NOP                     ;Force word boundry
                    endif
  00F2      TOS     EQU     (offset $ - SP186_CODE) + OS_OFFSET
0199 9090           NOP ! NOP               ;Stack can start here
                ;
019B FB             STI                     ;Start up the interrupt system
019C EB3D   01DB    JMPS    IDLE            ;Jump around i/o initialization table
                ;
                ; ---------------------------------
  00F7          I2681_TAB EQU (offset $-SP186_CODE)+OS_OFFSET
019E 0800               DW      ACR
  00F9          ACRREG EQU (offset $ - SP186_CODE)+OS_OFFSET
01A0 B0                 DB      0B0H            ;Baud rate set 1
```

```
  00FA          CHNLA_INIT EQU (offset $ - SP186_CODE)+OS_OFFSET
01A1 0000               DW      MRA             ;Mode register 1 for channel A
01A3 13                 DB      13H             ;8 bit, no parity
01A4 0000               DW      MRA             ;Mode register 2 for channel A
01A6 0F                 DB      0FH             ;Normal, RTS/CTS off, 2 stop bits
01A7 0200               DW      CSRA            ;Command register A
01A9 CC                 DB      0CCH            ;Tx/Rcv = 19.2K baud
01AA 0400               DW      CRA
01AC 15                 DB      15H             ;Reset to MRA, enable Txmitter/Recv
  0106          CHNLB_INIT EQU (offset $ - SP186_CODE)+OS_OFFSET
01AD 1000               DW      MRB             ;Mode register 1 for channel B
01AF 13                 DB      13H             ;8 bit, no parity
01B0 1000               DW      MRB             ;Mode register 2 for channel B
01B2 0F                 DB      0FH             ;Normal, RTS/CTS off, 2 stop bits
01B3 1200               DW      CSRB            ;Command register A
01B5 CC                 DB      0CCH            ;Tx/Rcv = 19.2K baud
01B6 1400               DW      CRB
01B8 15                 DB      15H             ;Reset to MRA, enable Txmitter/Recv
  05C0          TICKCNT EQU     (SP86DELAY*16)*(1000/43) ; 4.3 us clock rate to
                                                         ; get number 16 ms
01B9 0C00               DW      CTUR
01BB 00                 DB  TICKCNT shl 8
01BC 0E00               DW      CTLOWR
01BE C0                 DB  TICKCNT and 0ffh
01BF 0A00               DW      IMR             ;Counter's ISR triggers INTR pin
  011A          INTSTATE  EQU (offset $ - SP186_CODE)+OS_OFFSET
01C1 22                 DB      00100010b       ;Enable only RX interrupts
01C2 1A00               DW      OPCR            ;OP configuration
01C4 00                 DB      00              ;All output bits normal
01C5 1C00               DW      SETOUT
01C7 0B                 DB      00001011b       ;No parity; Linear addressing;
01C8 1E00               DW      RSETOUT         ;Allow INT; CTS
01CA 0C                 DB      00001100b       ;Clear INTO (to host)
01CB FFFF               DW      0FFFFH          ;END OF TABLE
                ;
  0126          INT_TAB EQU (offset $-SP186_CODE)+OS_OFFSET
01CD 3AFF               DW      I_INT1CTL
01CF 1800               DW      18H             ;Highest Priority,Level triggered,
                                                ;direct
01D1 38FF               DW      I_INT0CTL
01D3 1900               DW      19H             ;Priority 1,Level triggered,direct
01D5 28FF               DW      I_IMASK
01D7 CF00               DW      0CFH            ;Unmask only host interrupt
                                                ;INT0 and DUART INT1
01D9 FFFF               DW      0FFFFH          ;MArk end of table
                ;---------------------------------
                ;
01DB BB1402     IDLE:   MOV     BX,HELLO186     ;Tell world we exist
                                                ;(channel 1 of DUART)
01DE E81000 01F1        CALL    PMSG
01E1 C7060000FFFF       MOV Word Ptr .0,0ffffh;Set word at 40:00 to non zero to sign
01E7 5058       IDLELP: PUSH AX ! POP AX        ;Just waste some time
01E9 EBFC   01E7        JMPS    IDLELP
                ;
                ;-----------------------------------------------------
                ;
```



```
                ; UTILITY ROUTINES
                ;
                ; Print a message to CONSOLE 0 (for now)
                ;  Entry BX = pointer to message
01EB 53         PMSGLP: PUSH    BX              ;Save message pointer
01EC E80F00 01FE        CALL    CONOUT0         ;Call output routine for channel A
01EF 5B                 POP     BX              ;Recover message pointer
01F0 43                 INC     BX              ;Bump to next character of message
                PMSG:
01F1 8A0F               MOV     CL,[BX]         ;Pick up a message characer
01F3 80F924             CMP     CL,EOS          ;See if it is end of string
01F6 75F3       01EB    JNE     PMSGLP          ;Print it if it is any other character
01F8 C3                 RET
                ;
                ; Send a character to channel B
01F9 BA1200     CONOUT1:MOV     DX,SRB          ;Point DX at channel B status register
01FC EB03       0201    JMPS    CONOUT          ;Jump to general output routine
                ;
                ; Send a character to channel A
01FE BA0200     CONOUT0:MOV     DX,SRA          ;Point DX at channel A status register
                ;
0201 EC         CONOUT: IN      AL,DX           ;Get DUART status for this channel
0202 A804               TEST    AL,4            ;See if transmit ready
0204 74FB       0201    JZ      CONOUT          ;Wait until transmit ready
                ;Should check for DTR and/or XON/XOFF
0206 83C204             ADD     DX,4            ;Bump to data register
0209 8AC1               MOV     AL,CL           ;Put character in AL for output
020B EE                 OUT     DX,AL           ;Send character to UART
020C C3                 RET
                ;
                ;
                ; Unitializated Interrupt routine
                ;  In case of an unititialized interrupt, execution will end up here
                ;  Address of entry into the INTWORK table is on stack, INT 3 in
                ;  INTWORK table sent us here
                ;
                TRAP_ROUTINE:
020D 5850               POP AX ! PUSH AX        ;Get address of the INT 3
020F 1E                 PUSH    DS              ;Save current data segment
0210 0E1F               PUSH CS ! POP DS        ;Set up for 8080 model
0212 2DD702             SUB  AX,INTWORK+1
0215 BF8802             MOV  DI,TRP_CODE
0218 E80B00 0226        CALL    CNVTBYTE        ;Convert INT number to ascii
021B BB5D02             MOV     BX,INT_TRP      ;Point to trap message
021E E8D0FF 01F1        CALL    PMSG            ;Print trap message to channel A
0221 1F                 POP     DS
0222 83C406             ADD     SP,6            ;Move stack past INT3 stuff
0225 CF                 IRET                    ;Return back to bad interrupter

                ;
                ; Convert byte in AL to 2 ascii digits and save at [DI]
                ;
```

```
                CNVTBYTE:
0226 50         PUSH    AX              ;Save number to be translated
0227 BB0401     MOV     BX,XLATTBL      ;Point to hex to ascii translate table
022A D0E004     SHRrb   AL,4            ;Shift upper nibble to lower half
022D D7         XLAT    BX              ;Translate nibble to ascii
022E 8805       MOV     [DI],AL         ;Save ascii digit in user memory
0230 58         POP     AX              ;Recover original lower nibble
0231 240F       AND     AL,0FH          ;Mask off upper nibble
0233 D7         XLAT    BX              ;Translate nibble to ascii
0234 884501     MOV     1[DI],AL        ;And save it in user memory
0237 C3         RET
                ;
                ;****************************************
                ; INTERUPT ROUTINES
                ;****************************************
                ;
                ;  Interrupt from the DUART
                ;
                DUART_INT:
0238 1E         PUSH    DS
0239 2E8E1ED402 CSSEG ! MOV DS,Word Ptr .LOCSEG
023E 8C16D002   MOV Word Ptr .SS_SAVE,SS
0242 8926D202   MOV Word Ptr .SP_SAVE,SP
0246 8E16D402   MOV SS,Word Ptr .LOCSEG
024A BCCE02     MOV     SP,INTSTCK
024D 6006       PUSHA ! PUSH ES
                ;
024F BB4602     MOV     BX,DUARTMSG     ;Point to DUART message
0252 E89CFF C1F1 CALL   PMSG            ;Print it on channel A
                ;
0255 BA22FF     MOV     DX,I_IEOI       ;Point to 186 EOI register
0258 B80D80     MOV     AX,8000H+13     ;Specific EOI for INT1
025B EF         OUT     DX,AX           ;Send EOI to 186
                ;
025C 0761       POP ES ! POPA           ;Recover all registers
025E 8E16D002   MOV SS,Word Ptr .SS_SAVE ;Swap back to original stack
0262 8B26D202   MOV SP,Word Ptr .SP_SAVE
0266 1F         POP     DS              ;Recover Data segment
0267 CF         IRET
                ;
                ;
                ; Send an interrupt to s100 bus
                SETINT100:
0268 B004       MOV     AL,4
026A E61C       OUT     SETOUT,AL       ;Set the int bit high
026C E61E       OUT     RSETOUT,AL      ; then low to cause an interrupt
026E C3         RET                     ; on the VI line
                ;
                ;----------------------------------------
                ; Handle an interrupt from the HOST processor
                ;
                HOST_INT:
```



```
026F 1E         PUSH    DS
0270 2E8E1ED402 CSSEG ! MOV DS,Word Ptr .LOCSEG
0275 8C16D002   MOV Word Ptr .SS_SAVE,SS
0279 8926D202   MOV Word Ptr .SP_SAVE,SP
027D 8E16D402   MOV SS,Word Ptr .LOCSEG
0281 BCCE02     MOV     SP,INTSTCK
0284 6006       PUSHA ! PUSH ES
0286 8CC88ED8   MOV AX,CS ! MOV DS,AX   ;Using 8080 memory model for n
028A 8EC0       MOV     ES,AX
                ;
028C B008       MOV     AL,8            ;Clear interrupt from HOST
028E E61C       OUT     SETOUT,AL       ;Remove the interrupt source
0290 E61E       OUT     RSETOUT,AL      ;Reenable hostints
                ;
0292 BB3002     MOV     BX,HOSTMSG
0295 E859FF 01F1 CALL   PMSG
                ;
0298 BA22FF     MOV     DX,I_IEOI       ;Send End Of Interrupt to 186
029B B80C80     MOV     AX,8000H+12     ;Specific end of interrupt
029E EF         OUT     DX,AX           ;Send EOI to 186
                ;
029F 0761       POP ES ! POPA
02A1 8E16D002   MOV SS,Word Ptr .SS_SAVE
02A5 8B26D202   MOV SP,Word Ptr .SP_SAVE
02A9 1F         POP     DS
02AA CF         IRET
                ;
                ;
                ;************************************************************
                ;
                ;       DSEG ;For resident 186 code
  0204          XLATTBL EQU (offset $ - SP186_CODE)+OS_OFFSET
02AB 303132333435       DB      '0123456789ABCDEF'      ;Used for quick binary
     363738394142                                       ;to ascii
     43444546
                ;
  0214          HELLO186 EQU (offset $ - SP186_CODE)+OS_OFFSET
02BB 0D0A53503138       DB      CR,LF,'SP186 has been started.',CR,LF,EOS
     3620686173320
     6265656E2073
     746172746564
     2E0D0A24
                ;
  0230          HOSTMSG EQU (offset $ - SP186_CODE)+OS_OFFSET
02D7 496E74657272       DB      'Interrupt from HOST',CR,LF,EOS
     757074206672
     6F6D20484F53
     540D0A24
  0246          DUARTMSG EQU (offset $ - SP186_CODE)+OS_OFFSET
02ED 496E74657272       DB      'Interrupt from DUART',CR,LF,EOS
     757074206672
     6F6D20445541
```

```
     52540D0A24
                    :
  025D              INT_TRP EQU (offset $ - SP186_CODE)+OS_OFFSET
0304 0D0A20535031           DB      CR,LF,' SP186: Uninitialized interrupt,
     38363A20556E                   code = 0'
     696E69746961
     6C697A656420
     696E74657272
     757074742C2063
     6F6465203D20
     30
  0288              TRP_CODE EQU (offset $ - SP186_CODE)+OS_OFFSET
032F 7878480D0A24           DB      'xxH',CR,LF,EOS
                    :
  0210              SP186_CODE_SIZE EQU     offset $ - offset SP186_CODE + 2
                    ;  End of SP186 initialized data area
                    :
                    :
                    ;  SP186 uninitializated data area
  0000              STACKRM EQU (offset $ - SP186_CODE) MOD 2
  02CE              INTSTCK EQU (offset $ - SP186_CODE)  + OS_OFFSET +
                    STACKRM + 40H
                    :
  02D0              SS_SAVE EQU INTSTCK + 2
  02D2              SP_SAVE EQU SS_SAVE + 2
  02D4              LOCSEG EQU SP_SAVE +2
                    :
  02D6              INTWORK EQU     LOCSEG + 2

  03D6              ENDDATA EQU     INTWORK + 256


END OF ASSEMBLY.  NUMBER OF ERRORS:   0.  USE FACTOR: 10%
```


User Notes

LIMITED WARRANTY

Viasyn Corporation warrants this computer product to be in good working order for a period of ninety days 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 item 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 unauthorised 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 or dealer, please call VIASYN at (415) 786-0909 to obtain a Return Material Authorisation (RMA) number, or write to VIASYN at 26538 Danti Court, Hayward, CA, 94545-3999, Att: RMA. Be sure to include a copy of the original bill of sale to establish a 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. Be sure to mark the RMA number on the outside of the shipping container or delivery may be refused. Contact your System Center/dealer or write to VIASYN at the above address for further information.

All expressed and implied warranties for this product, 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, either 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 shall 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 or write the VIASYN RMA department to obtain a quotation for factory service. If this product was sold as a system by VIASYN, it may eligible and you may elect to purchase on site/depot maintenance from SPERRY. Contact your System Center/Dealer, your nearest SPERRY office or VIASYN for more details.

If you have purchased a SPERRY service and maintenance agreement, the following two paragraphs also apply:

If VIASYN or its service contractor fails after repeated attempts to perform any of its obligations set forth in this agreement, VIASYN's or its service contractor's entire liability and VIASYN's customer's sole and exclusive remedy for claims related to or arising out of this agreement for any cause and regardless of the form of action, whether in contract or tort, including negligence and strict liability, shall be Viasyn's customer's actual, direct damages such as would be provable in a court of law, but not to exceed the cost of the item of equipment involved.

In no event shall VIASYN or its service contractor be liable for any incidental, indirect, special or consequential damages, including but not limited to loss of use, revenue or profit, even if VIASYN or its service contractor has been advised, knew or should have known of the possibility of such damages; or damages caused by VIASYN's customer's failure to perform its obligations under this agreement; or claims, demands or actions against VIASYN's customer by any other party.

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

EFFECTIVE 9/1/86. This warranty supersedes all previous warranties. All previous editions are obsolete.

9020-0048