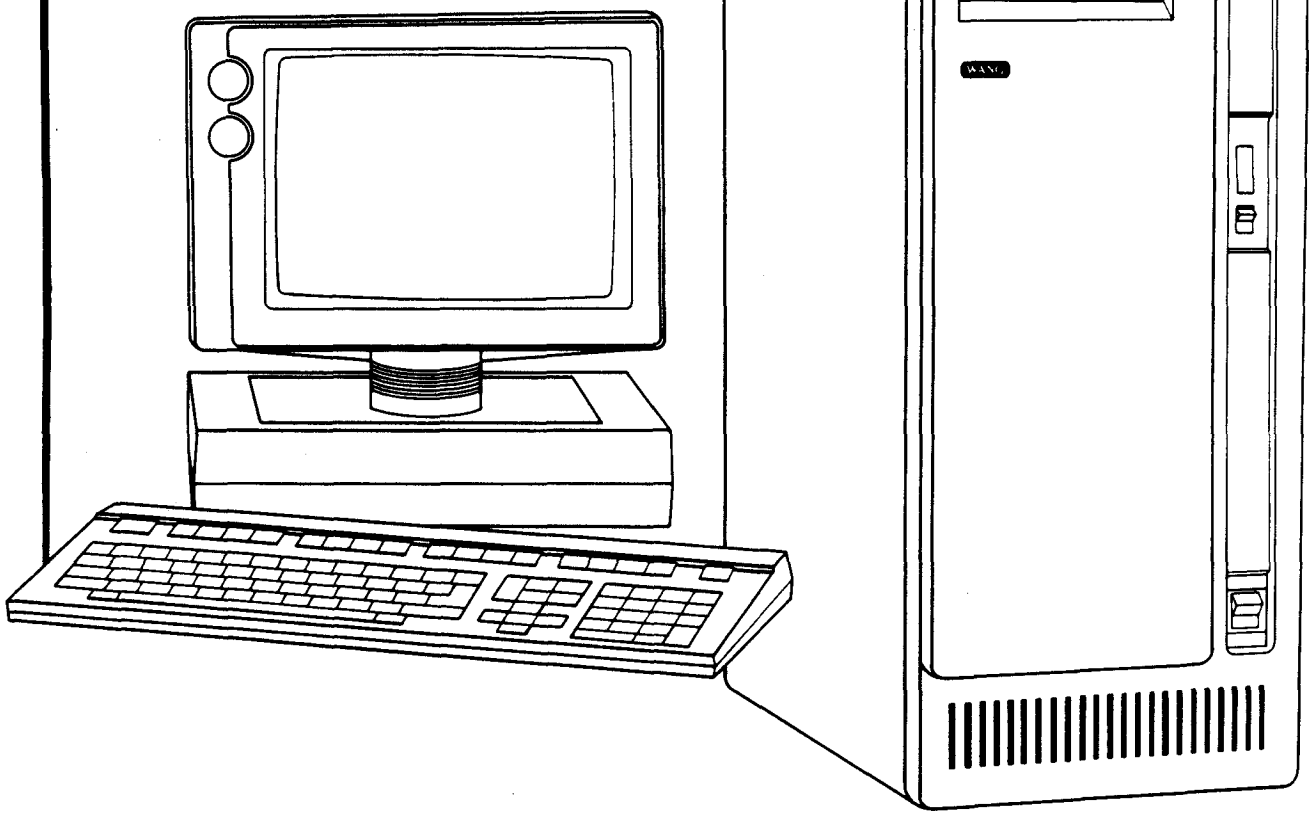
WANG

1403

OIS 40/50/60

BOARD REPAIR WORKBOOK VOLUME 2



CUSTOMER ENGINEERING
TRAINING AND DOCUMENTATION

741-9041

WANG

CUSTOMER ENGINEERING TRAINING CENTER

OIS 40/50/60
BOARD REPAIR WORKBOOK VOLUME 2

PREFACE

This document is intended to be used for TRAINING PURPOSES only. The material contained in this document, while accurate during the developement of this workbook, may not reflect the latest developments or changes to the OIS 40/50/60 system.

TECHNICAL SUPPORT DOCUMENTS

OIS 50 INTERNAL PRINTER CONTROLLER HARDWARE SPECIFICATIONS HM-60
OIS 40/50 RESOURCE MANAGEMENT UNIT THEORY OF OPERATION 751-0902
OIS 40/50 RESOURCE CONTROL UNIT THEORY OF OPERATION 751-0911
OIS 40/50 IWS FULL MATRIX CONTROLLER SPECIFICATION HM-67
WL-2630 OIS/VS COLLECTIVE GATE ARRAY SPECIFICATION HM-37
OIS-50 INTERNAL WISE SPECIFICATION REVISION 3 HM-85
OFFICE INFORMATION SYSTEMS OIS 40/50/60 741-1267
OIS SYSTEM ADMINISTRATION GUIDE 700-5562E

First Edition - December, 1985

OUTLINE OF WORKBOOK

TABLE OF CONTENTS VOL I

TABLE OF CONTENTS (cont.)

TABLE OF CONTENTS (cont.)

TABLE OF CONTENTS VOL II

CHAPTER Page

SECTION 6
Internal Printer Controller (IPC)

SECTION 7
Internal WISE Controller (IWISE)

TABLE OF CONTENTS (cont.)

CHAPTER Page

SECTION 7 (cont.)

SECTION 8
DIAGNOSTICS

CHAPTER Page

SECTION 9
APPENDICIES

LIST OF ILLISTRATIONS

LIST OF ILLISTRATIONS (cont.)

SECTION 3
Resource Control Unit (RCU)

LIST OF ILLISTRATIONS (cont.)

SECTION 4
Internal Workstation Controller (IWS)

LIST OF ILLISTRATIONS (cont.)

LIST OF TABLES

LIST OF TABLES (cont.)

SECTION 4
Internal Workstation Controller (IWS)

LIST OF TABLES (cont.)

SECTION 6
INTERNAL PRINTER CONTROLLER (IPC)

SECTION 6
Internal Printer Controller (IPC)

6.1 INTRODUCTION

The Internal Printer Controller, commonaly called the IPC board is designed to control one printer through a modified RS232C serial interface. This board is comprised of four sections.

1. 50BUS Interface
2. Z80A Microprocessor
3. 64K Bytes of Memory
4. RS-232C Interface

The 50BUS interface provides the communication link between the RCU and the Z80A of the IPC. Through this bus, commands and data are transfered to the IPC. This interface is similiar to the 50BUS interface of the other boards in the OIS 40/50/60 system.

The onboard Z80A of the IPC receives instructions and data from the RCU and then goes about transfering the data to the printer through the RS-232C interface. One of the Z80A's responsablities is to monitor the performance of the printer while transferring data to the interface when requested. If a malfuntion should happen the Z80A is informed and it in turn notifies the system of the problem.

In order to increase the speed of the system and reduce the number of transfer operations, the IPC contains 64K of Dynamic RAM. This memory holds the operating codes for the onboard Z80A along with the data to be printed.

The heart of the IPC is the RS-232C interface. This interface consises of a SC2661C Enhanced Programmable Communication Interface (EPCI). The SC2661C is a universal sychronous/asychronous data communication controller chip that provides all the timing and control signals for transfers of data over the RS-232C connector. The onboard Z80A programs the SC2661C upon power up for asynchronous transmition with the proper start/stop and data bits, along with the selected Baud rate. The SC2661C then performs the parallel to serial conversion and inserts the proper number of start and stop bits. At the complection of each byte transfer the SC2661C will inform the onboard Z80A that the next byte can be loaded and status of the current operation can be read through the internal status register of the EPCI.

A block diagram of the IPC showing the four major components, there interconnections with the printer and the OIS 40/50/60 system, is shown in FIGURE 6.1-1. Each of the four section will be described in detail in the following pages.

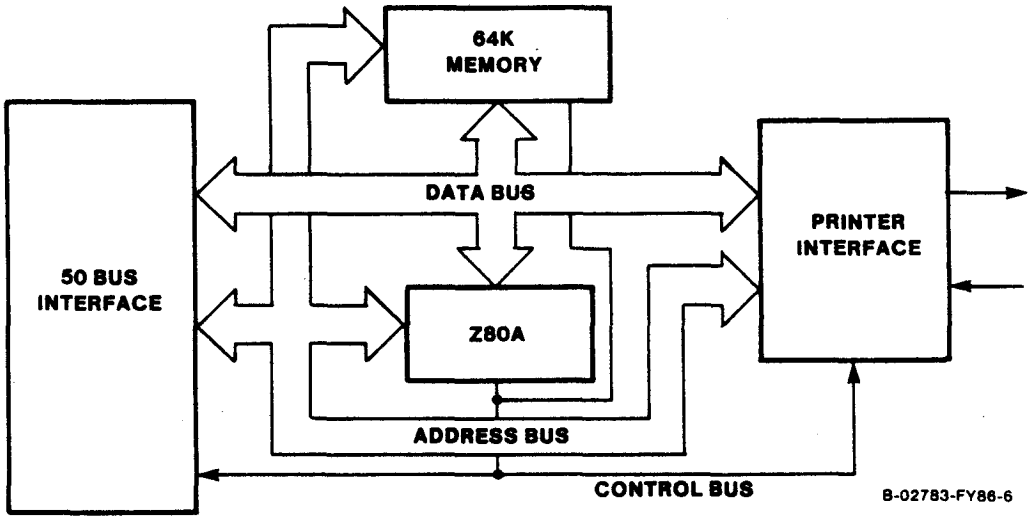


FIGURE 6.1-1
Internal Printer Controller (IPC) Block Diagram

6.2 50BUS Interface

The 50BUS provides a means of communication between the onboard Z80A CPU and the RCU board of the system. When the RCU desires to communicate with the IPC it first generates a BUS REQUEST. Upon receiving this request the onboard Z80A deactivates its busses after the complection of the current instruction, and allows the RCU access to the onboard memory and status registers. This bus request is generated by a combination of the &50BSLCT and &50BUSRQ signal. Both signals are feed through buffer L99 (1A10) to OR gate L74. When both signals are active L74 produces &IPCBSRQ to the Z80A. The Z80A will acknowledge this request by activating pin 23 &BUSAK, thus driving its Data, Address, and Control busses to a tristated condition. &BUSAK is also used to select the 50BUS Address Bus buffers, hence alowing the RCU access. The RCU will then place the appropiate address on the bus and perform the desired transfer operation. The information that the RCU transfers to or from the IPC could be in the form of either data or instructions for the IPC or it could be status information from the IPC. A detailed description of the 50BUS signals and there fuction is given in Sections 3.4.3 and 3.4.4 of Volume 1.

6.3 Z80A Microprocessor

The Z80A of the IPC board has two major tasks to perform. First, upon power up of the system it must receives program instructions from the RCU and then configure the SC2661C chip. Once this has been done the Z80A is responisable for transfering data to the SC2661C for transmition to the printer, and monitoring the status of the printer. The basic structure of the suport chips for the Z80A is the same as the rest of the circuit boards in the OIS 40/50/60 system. The Z80A is driven by a 4Mhz clock supplied by a crystal and associated circuits. If you drsire further information on the structure of the Z80A suport circuits refer to the appropiate section of either the RMU or the IWS controller.

6.3.1 NOP Generation

When the system is first powered on, each of the controller boards must be IPLed. Since each board receives different code and the system can not load all of the boards at the same time each board must be placed in a hold type state until it can be IPLed. This hold state is a function of the NOP generator. When the power is first applied to the system the RCU generates a master reset signal to all boards. This master reset signal known as &MRC is buffered through L99 (1A10) where it is places on the clock input of flipflop L37 (1A8). The "D" input of L37 has +5VDC applied to it via R1. On the trailing edge of &MRC L37 will set causing &NGON to be generated. &NGON will enable data buffer L52 (1B7). The "A" inputs of L52 are all tied to +5VDC, and once L52 is enabled the "Y" outputs will all go low. As you can see from the schematics these outputs are tied diresctly to the Z80A's data bus.

When the &MRC signal was sent to all the boards, the Z80A of the IPC board was placed in a reset state, which caused it to enter into a M1 cycle. During a M1 cycle the Z80A fetches an instruction from the data bus, this instruction is HØØ generated by the NOP generator. A NOP instruction causes the Z80A to perform continuous M1 cycles. When the Z80A is perfroming a NOP instruction the program counter incerments one step for each NOP perfromed. But the Data bus is held at all zero so the Z80A continues to perfrom NOPs. When &NGON was generated it was also sent to L106 the status register. Here it asserted the NOT RUNNING bit. The RCU then works its way through the boards on the system and determines which ones need IPLing. When the RCU finds a NOT RUNNING bit set, it will first generate a restart command to the device to clear the NOT RUNNING bit and disable the NOP generator. Then it will down load a jump to location zero instruction at location zero. This keeps the Z80A in a tight loop, thus keeping the PC register from advancing. Then when the RCU find time it will down load the operation instruction to the board, followed by another restart command. At this time the IPC's Z80A can then go about performing the instructions that it was given.

There are two other times that the NOP generator could be involked. The first is if a parity error is detected during a transfer to or from main memory. If this happens the signal PARERR (Parity Error) is generated and sent to L28 (1B10). It passes through L28 and on to the perset input of L37 the NOP flipflop. This PARERR signal is also sent to the status register to inform the RCU that a problem has occured. The second incident that will invoke the NOP generator is if the pwoer on the printer is turned off and the CTS (Clear to Send) signal is lost. If this happens the &POWERON signal will deactivate, this high level signal will then be places on pin 5 of L28 where it will be inverted and sent to the preset input of L37 the NOP generator flipflop. This &POWERON signal is also sent to the status register for the RCU to read. One final note on the NOP generator operation. If the &POWERON signal is not being generated, either by the printer being turned off or the cable being disconnected, the IPC board will not IPL. This problem can be resolved by placing J4 between pins 2 & 3.

NOTE

If you move J4 from pins 1 & 2 so that you can test the board without a printer connected. Then remember to move it back when you are finished with the test.

6.3.2 I/O Operations

The Z80A of the IPC board communicates to the other devices on the board through memory mapped I/O locations. These operation can be divided into two areas;

1. Device Status and Diagnostic Operations
2. SC2661C Operations

Tables 6.3-1 and 6.3-2 provide a brief explanation of each I/O operation.

TYPE	LOCATION	FUNCTION
IN	07	Device Switch 2
IN	08	Device Switch 1
IN	09	Diagnostic Mode Indicator
IN	0A	Resets the SC2661C
IN	0B	Toggles Diagnostic Mode *
IN	0C	Clears Parity Error Flipflop
IN	0D	Toggles Parity Flipflop *

Table 6.3-1
Device and Diagnostic I/O Operations

* Each read from these locations will cause the state of the flipflop to change states. Further explaination of their operation can be found later in this Section.

The folowing table list the operation associated with the I/O locations that corrisponds to the SC2661C (FPCI) chip. The decoding of these locations are within the SC2661C once the chip has been selected. The SC2661C is selected any time that bit 4 of the address bus is high during an I/O operation.

TYPE	LOCATION	FUNCTION
IN	10	Reads Receive Data Holding Register
OUT	11	Writes to Transmit Holding Register
IN	12	Reads the Status Register
OUT	13	Writes to SYN1/SYN2/DLE Registers *
IN	14	Reads Mode Registers MR1/MR2 *
OUT	15	Writes to Mode Registers MR1/MR2 *
IN	16	Reads Command Register
OUT	17	Writes to Command Register

Table 6.3-2
SC2661C I/O Operations

* Each of these I/O locations have multiple registers assigned to them. Internal circuits in the SC2661C keep track of wich register is to be accessed each time a read or write operation occurs for that location. (i.e. If you perform an OUT 13, you will write to the SYN1 register. Then if you perform another OUT 13, you will write to the SYN2 register. When the maximim number of registers for a given location has been reached the internal circuits will recycle to point to the first register.) The register pointers are reset to their start position when either the reset input of the SC2661C is activated or by reading the Command register. A detailed explaination of each of the registers within the SC2661C can be found in Section 6.5.2.

All I/O operation for the IPC board are selected through three LS138 decoders, L30, 31, and 32 located on sheet 1 of the schematics. Each of these decoders need three enabling signals to alow them to operate, G2A, G2B and G1. The G1 and G2B inputs of each are driven from &BM1 and either &BRD or &BIORQ signals respectivaly. The G1 input is an active high input and will be enabled whenever the Z80A is NOT in a M1 machine cycle. Remembering that the M1 machine cycle is an instruction fetch activity. The G2B input of L31 and L30 are driven by the active low signal of &BRD (Buffered Read). When the Z80A is performing a read operation this signal will be driven low, thus activating the G2B input of these decoders. The G2B input of L32 is driven from the &BIORQ (Buffered I/O request) signal of the Z80A. This signal is activated whenever the Z80A performs an I/O instruction. Before I go any futher let's look at FIGURE 6.3-1 to see how these three decoders are wired.

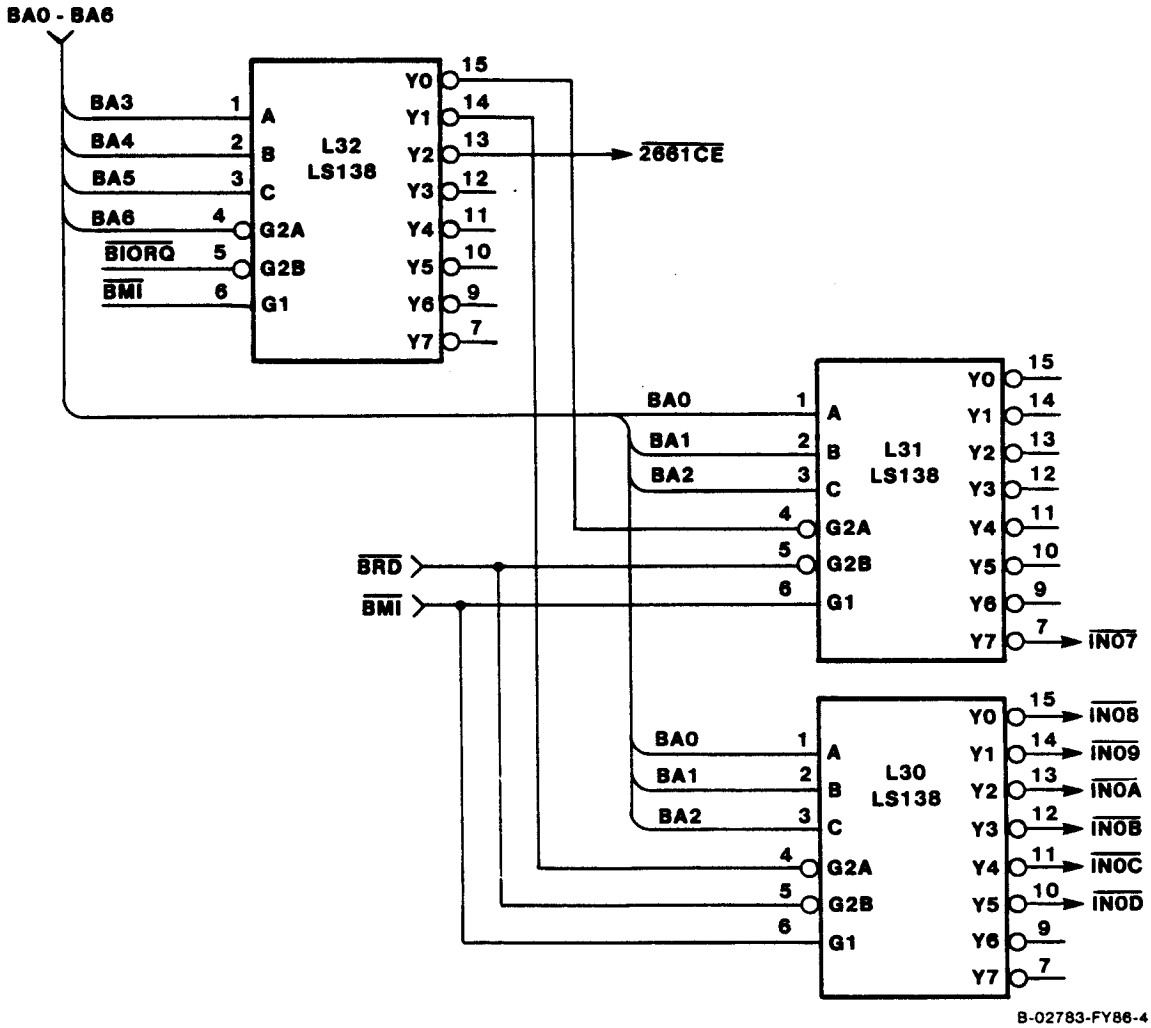


FIGURE 6.3-1
I/O Decoders

The third enable input of both L30 and L31 are driven by one of the outputs of L32. This tells us that L32 must be selected before either L30 or L31 can be enabled. L32 receives it's G2A enable from the BA6 bit of the Address Bus. Whenever this bit is low and the other two enable inputs are activated L32 will decode address bits BA3 - BA5. Only the first three outputs of L32 are used by the IPC board. If BA3 through BA5 are all low then the Y0 output will be driven low. This low out will then enable L31, allowing it to decode BA0 through BA2 of the Address Bus.

The Y1 output of L32, when activated will enable L30, thus allowing it to decode the lower three bits of the Address Bus. Finally when the Y2 output of L32 is active we will select the SC2661. The SC2661 will then decode the lower three bits of the Address Bus.

Let's now take a look at some of the I/O operation and see what functions they perform. The IN07, 08 and 09 operations allow either the system or the onboard Z80A to read the setting of the device switches and the condiction of the Diagnostic Mode Indicator. These three operations are quite simple in that they enable the appropiate buffer, thus placing the contence of the switches or registers on the Data Bus.

The next I/O operation is the IN0A. When the IN0A instruction is performed by the Z80A, L30's Y2 output will be driven low. This high to low transion is inverted and presented to the clock input of flipflop L59 (2B11). The flipflop will set causing the &Q output to go low, generating &WAIT and 2661RST via L58. At the same time the clear input of L57 an LS163 counter is removed. L57 then starts counting. When the QD output goes high on the 8^{th} count flipflop L59 will reset via L67 and L90. This delay is needed to allow the SC2661 time to reset it's internal registers. The &WAIT signal that was generated during this operation was presented to the Z80A placing WAIT states within the instruction cycle.

The IN0B operation allows the system to switch between Normal mode and Diagnostic mode. Each time the IN0B instruction is performed flipflop L35 (1B1) is toggled. When the Normal mode of operation is selected read and write operation to memory occur with even parity. But when the Diagnostic mode of operation is selected read and write operation to memory occur with odd parity. The reason for having odd parity during these transfers is so that the software can check the memory circuits for parity errors.

When a parity error occurs flipflop L59 (3B2) sets, and causes one of two parity error signals to be activated depending on the selected mode of operation. To return this flipflop to the reset condition an IN0C instruction must be preformed. Decoder L30 (1C2) generates IN0C which is gated through AND gate L90 (3B3) to clear flipflop L59.

When the INOD instruction is performed flipflop L89 (2C8) will toggle between Normal parity error responce, and MNI responce. During Normal parity responce, if a parity error occurs the NOP generator is turned on. This causes the Z80A to run in a very tight loop until the problem is recified. For futher information on the NOP generater refer to Section 6.3.1. If the MNI type of resopnce is selected, and a parity error occurs a None Maskable Interrupt will be generated. This type of responce can be used for diagnostic purposes by informing the software that a problem has occured without locking up the micro.

The rest of the I/O operation are all decoded within the SC2661 chip. These operations will be used extencivelly by the Z80A to communicate with the SC2661. The key to these operations is the selection of the SC2661. This is accomplished when BA4 is high and BA3, 5 and 6 are low, and of course the Z80A is performing an I/O operation. When this happens L32 (1F3) generates SC2661CE (2661 chip select). Then the SC2661 decodes the lower three bits of the Address Bus to determine the operation. As I stated before, some of these operations when performed, will selecte different registers within the SC2661 for the same I/O command. It is important to keep track of the number of times a given I/O operation is performed in order to know which register is being accessed. Normally this is done by the software but if during testing the operator tries to read or write to these locations he or she must be aware of the fact that each access will be to a different register. A complete breakdown of the SC2661 operations and commands are descussed in the Section 6.5.

6.4 IPC Memory

The memory of the IPC board consists of nine HM4864-3 64Kx1 memory chips. Eight of these are used for main memory and one for parity. Since these chips are dynamitic they require a 256-row refresh every 2 to 4 Msec. The Z80A provides a 7 bit refresh address at the end of each M1 machine cycle, yielding a total of 128 row addressess. To provide for 256-row addressing a synthetic refresh address bit is multiplexed into the A7R bit of the Z80A Address Bus. The circuits involved in this operation are similar to the ones in the RMU boards.

6.4.1 REFRESH

In a 256-row refresh address operation, the Z80A issues A6 low for the first 128-row refresh and high for the second 128-row refresh. By using this A6 bit as a control input to a "D" type flipflop we can generate a synthetic 256-row refresh address. Referring to FIGURE 6.4-1 and FIGURE 6.4-2 we see the four flipflops that provide this operation. Flipflop L60-1 and L60-2 are used to toggle the synthetic refresh bit while flipflop L51 switches between the synthetic bit and the normal A7 bit, and L43-2 signals the start of refresh.

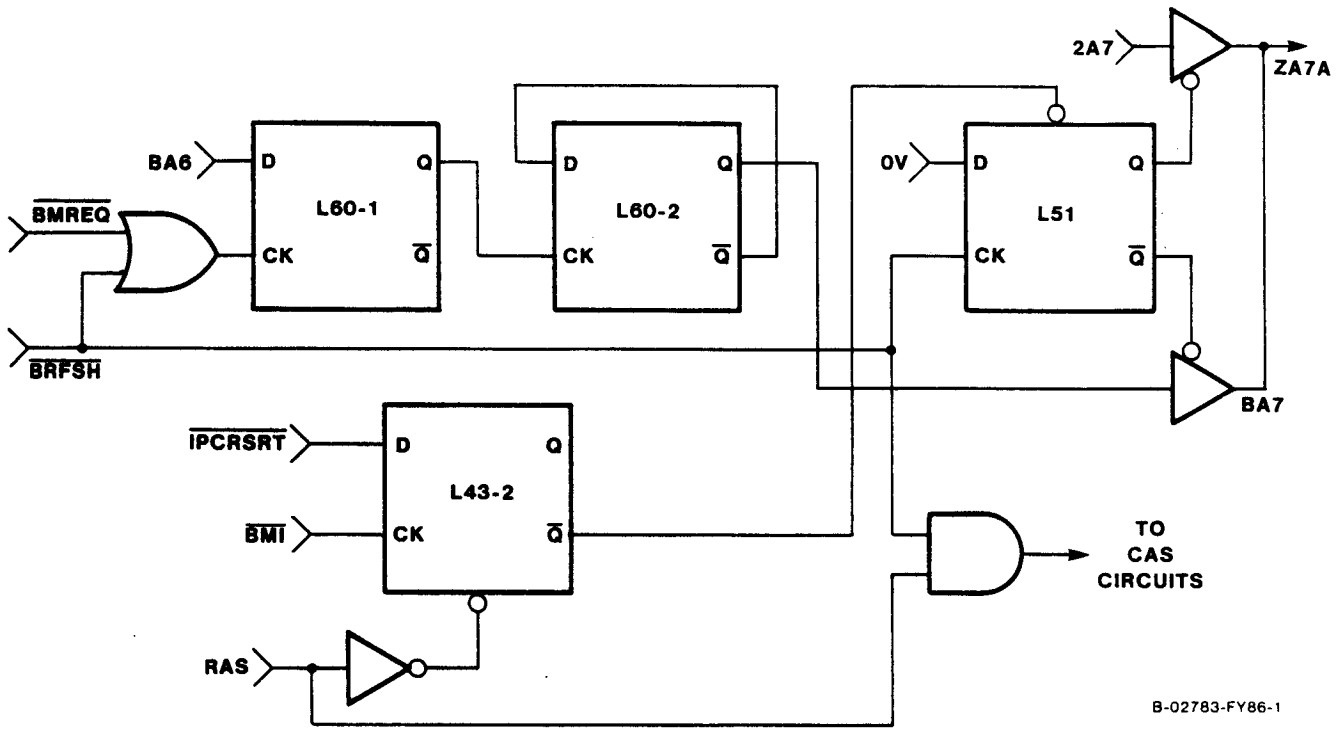


FIGURE 6.4-1
REFRESH LOGIC DIAGRAM

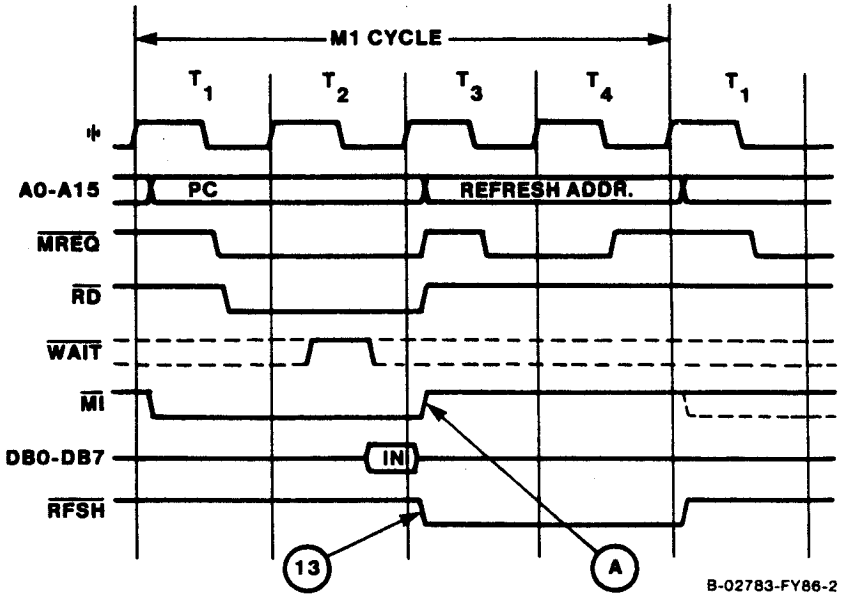


FIGURE 6.4-2
M1 Timing Diagram

Flipflop L51 is normally in a reset state, this enables driver L68 to pass the ZA7 bit from the Z80A to the Address Buss buffer L62. Halfway through the M1 machine cycle &BM1 will deactivate (Refer to FIGURE 6.4-2 Point A) Causing flipflop L43 to set, thus generating Early Refresh (ERFSH) from the "&Q" output. Early Refresh is placed on the preset input of L51, causing it to set and switching the ZA7A output from ZA7 to BA7. The "Q" output of L43 is passed to the "D" input of L94 where on the next 1/2 phi clock pulse L94 will set starting the RAS cycle.

At the same time that &BM1 went high, &BMREQ went high and &BRFSH went low, signaling the start of the refresh cycle (Refer to FIGURE 6.4-2 Point B). In the middle of T3 time &BMREQ became active again, setting up L60-1. During this time (T3 & T4) the Z80A places the refresh address on the Address Bus. Then sometime during T4 state of M1, &BMREQ deactivates again, clocking flipflop L60-1. The state of the BA6 bit of the Address Bus, which is feeding the "D" input of L60-1, will be transfered through to L60-2. Each time the Z80A presents a refresh address on the bus, it (the Z80A) will toggle the state of BA6. When BA6 is in the high state L60-2 will clock and change the state of the synthetic BA7 bit. Using this techeque, a minium of two M1 machine cycle are required to refresh the full 64K bytes of memory. At the end of the refresh cycle &BRFSH will deactivate, this low to high action clocks L51-1 into a reset state, switching the synthetic BA7 bit out of the circuit.

Remember that during a refresh cycle we only need a RAS cycle. While we are in a refresh cycle, &BRFSH signal is low, hence, AND gate L69 is disqualified, blocking the RAS signal from proceeding.

6.4.2 RAS/CAS

There are three different ways that the RAS/CAS cycle can be started. 1.) Normal read or write request, 2.) Instruction fetch, 3.) Refresh cycle. Of these three the Refresh action was decussed in the last section so we will consintrate on the first two.

During a normal read or write operation, this exculdes instruction fetch even though it is in reality a read operation, the Z80A will generate &BMREQ (Buffered Memory Request). This signal will be inverted by L41 (3B11) and passed through gates L50 to generate EPDB (Enable Parity and Data Bus) refer to FIGURE 6.4-3. This signal enables the parity circuits and selects the Memory out buffer. Also the transision of this signal, clocks flipflop L51-2 into a set state causing the "Q" output to go high, hence driving the "D" input of flipflop L94-1 high through OR gate L76. On the next low to high shift of 1/2 phi Z clock L94 will set causing &RAS to generated, signaling

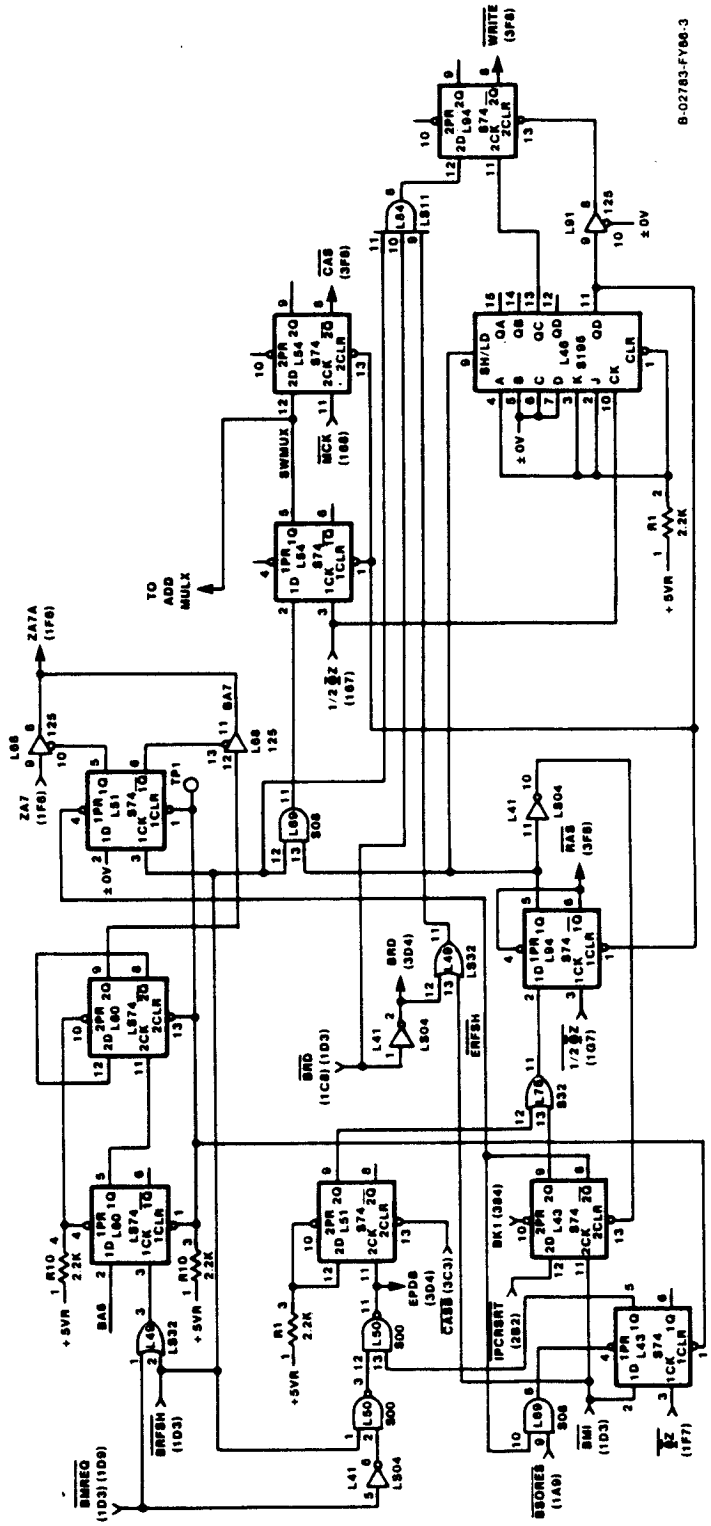


FIGURE 6.4-3
RAS/CAS LOGIC DIAGRAM

the start of the RAS/CAS cycle. The remainder of the RAS/CAS cycle will be descussed later. At the complection of the RAS/CAS cycle L51-2 will be placed in the reset condition by &CASB on its reset input. This will treminate the cycle on the next &1/2 phi z clock.

The RAS/CAS cycle is also started when the Z80A fetches an instruction. When the Z80A begins an instruction fetch cycle it generates &BM1 (Machine 1). This signal is directed to the "D" input of flipflop L43-1. On the next low to high transision of phi Z clock L43 will reset causing the "Q" output to go low. This low is inverted through NAND gate L50 to generate EPDB which then starts the RAS/CAS cycle, just as it did in the last paragraph. Flipflop L43-1 will go back to the set state at T3 time of the M1 cycle, when &BM1 deactivates. The Z80A also generates &BMREQ during the instruction fetch cycle, but it comes at a much later time in the cycle. If the circuits relied on &BMREQ to activate the RAS/CAS cycle during an instruction fetch the address that the PC register of the Z80A placed on the Address Bus would not be stable long enough to generate a reliable address.

To complete the RAS/CAS cycle two more operation must be perfromed. First the address multiplexors must be switched and second the CAS signal must be asserted. Once the RAS flipflop L94 has been set the "Q" output is sent to two locations. The first is the Shift/Load input of shift register L46. This places the register in a shift mode of operation. This register has a two fold operation. One, it determines the complection of the RAS/CAS cycle, and two, it signals the proper time for the write enable signal during a write operation. The second place that the "Q" output of L94-1 is sent is the "D" input of flipflop L54-1, via AND gate L69. Remember, if we are in a refresh cycle AND gate L69 will be disqualified, thus blocking this path. Once the high is placed on the "D" input of L54 and the next low to high transision of 1/2 phi z clock happens L54-1 will set. This generates SWMUX (Swicth Multiplexor) which causes the two address selectors to select the high order bits of the Address Bus. Going back the "Q" output of L54-1 where SWMUX was generated we see that it is also placed on the "D" input of L54-2. On the next low to high transision of &MCK L54-2 will set, generating &CAS from the &"Q" output. At this time the address operation is complete and we are able to perfrom either a read or write.

The only real difference between a read operation and a write operation is that during a write cycle, the Z80A will wait until T4 time before it transfers the data from it's register to the memory. The write enable signal should not be activated until after both RAS and CAS have happened. The timing of this signal is controlled by counter L46. If you recall back when RAS was generated we placed L46 into a shift mode of operation. Then while we were setting up SWMUX and CAS L46 was shifting the single one it had placed in it. When this one

finally reaches the QC output we will clock the write enable flipflop L94-2. This flipflop will set if and only if all the inputs to AND gate L84 are active high. When L94-2 does go set we generate &WRITE which allows us to write to the memory. On the next transision of 1/2 phi z clock, shift register L46 will terminate the RAS/CAS cycle by clearing the write enable flipflop, the RAS flipflop, the SWMUX flipflop and the CAS flipflop.

6.5 RS-232C Interface

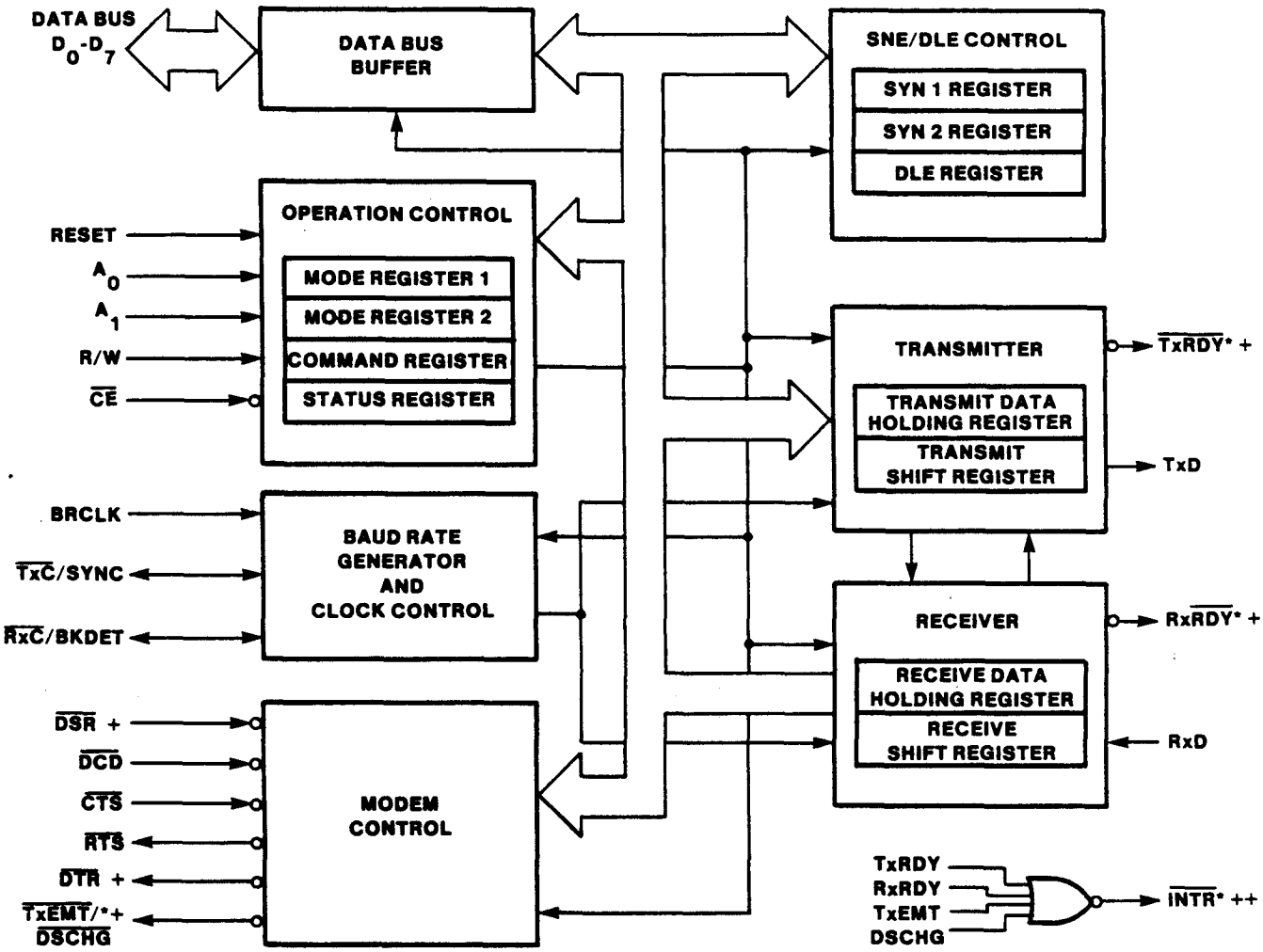
The RS-232C interface is our link with the printer. This interface provides us with the means of communication with the printer and provides the translation of digital data to serial anlog data. The heart of this interface is the SC2661 EPCI (Enhanced Programmable Communications Interface). By using the SC2661 the amount of circuits needed to interface with the printer is reduced greatly. In the following paragraphs I will describe to you the workings of the SC2661 and how they are used in the Internal Printer Controller (IPC) board.

6.5.1 SC2661 EPCI Functional Description

The SC2661 EPCI is a universal synchronous/asynchronous data communications controller chip. It can interface easily with an 8 bit or 16 bit microprocessor and may be used in either a polled or interrupt driven enviroment. The SAC2661 accepts program instructions from a microprocessor, while supporting many serial data communication displines. Both synchronous and asynchronous communication in either half or full duplex modes are available to the user.

The SC2661 receives parallel data from the microprocessor and converts it into serial data for transmision. The apporpiate number of start and stop bits are assigned along with the choosen parity. At the same time the SC2661 is capable of receiving serial data from an external device and converting it into parallel data for use by the microprocessor. On the serial receive side the SC2661 detects any errors such as parity, overrun and framing, then the SC2661 informs the controlling microprocessor as to the status of the received data.

The Baud rate of the transmision of the data can be programed to except either and external of internal clock. One of sixteen different baud rates can be selected when operating under the internal clock circuits through program control. Also through program control the operator can select the number of stop bits (1, 1 1/2, or 2) along with the number of data bits (5 to 8) and of course the type of parity (odd or even).



NOTE
* OPEN DRAIN OUTPUT PIN.
+ 28-PIN ONLY
++ 24-PIN ONLY

B-02783-FY86-5

FIGURE 6.5-1
SC2661C Block Diagram

Some other features of the SC2661 are, doubled buffered transmitter and receive registers for faster operation, TTL compatable inputs and oputputs, full or half duplex operation. For further information on the capabilities of the SC2661 refer to Signetics MOS Microprocessor Data Manual 1983.

In order to understand the operation of this chip and to help the technician troubleshoot the circuits that are connected to it the following description of the operation and logic involved within the SC2661 are very important. By refer to the block diagram in FIGURE 6.5-1 we see that the SC2661 consists of six major sections. These are the transmitter, receiver, timing, operation control, modem control, and SYN/DLE control. These section communicate with each other through an internal data bus and an internal control bus. The internal data bus interfaces with the outside microprocessor data bus via a data bus buffer.

Operation Control

This functional block stores configeration and operation commands from the microprocessor and generates the appropiate signals to various internal section to control the operation of the device. Read and write circuits are contained within the device in order to communicate with the CPU via the data bus.

Timing

The EPCI contains a baud rate generator which is programmable to accept external transmit and receive clocks or to divide an external clock to perfrom data communications. The unit can generate 16 commonly used baud rates, any one of which can be selected for full or half duplex operation.

Receiver

The receiver accepts serial data on the RxD pin, converts the serial data into parallel format, checks for bits or characters that are unique to the communication technique and sends an "assembled" character to the CPU.

Transmitter

The transmitter accepts parallel data from the CPU, converts it to a serial bit streem, inserts the appropiate characters or bits (based on the communication technique) and outputs a composite serial stream of data on the TxD output pin.

Modem Control

The modem control section provides an interface for three input signals and three output signals for handshaking operations and status indication between the CPU and modem

SYN/DLE Control

This section contains control circuitry and three 8-bit registers storing the SYN1, SYN2 and DLE characters provides by the CPU. These registers are used for synchronous mode of operation to provide the character required for synchronization, idle fill and data transparency.

Register Access

Several register within the SC2661 are accessable to the microprocessor to faciliate the communication between the two devices. These registers help the microprocessor select the desired mode of operation and determine the status of both the transmit and reveice operations.

The access to these registers is through four external pin of the SC2661 along with the data bus. By applying the appropiate signals to the &CE, &&R/W, A1 and A0 pins these registers can be either read from or written to. The conditions necessary to access these registers are shown in Table 6.5-1. Some of the registers have mulitple levels that can be access by multiple accesses with the same conditions. Each time one of these multiple level registers are accessed the internal circuits will determine which level of the register will be either read from or written to. The internal pointers that determine which level of the multiple level registers are accessed can be reset to their starting point by either reading the Command register or by applying the appropiate level signal to the RESET input.

$\overline{CE}$	A1	A0	$\overline{R}$/W	FUNCTION
1	X	X	X	Tri-state data bus
0	0	0	0	Read Receive Holding Register
0	0	0	1	Write Transmit Holding Register
0	0	1	0	Read Status Register
0	0	1	1	Write SYN1/SYN2/DLE Registers
0	1	0	0	Read Mode Registers 1/2
0	1	0	1	Write Mode Registers 1/2
0	1	1	0	Read Command Register
0	1	1	1	Write Command Register

Table 6.5-1
2661 Register Addressing

6.5.2 SC2661 Registers

In the following paragraphs I will describe the different registers within the SC2661 and provide you with a brief explaination of their operation. If you would like a more detailed explaination you should refer to Signitcs MOS Microprocessor Data Manual.

There are two Mode Registers in the SC2661. As I stated earlier these registers are accessed by placing the appropiate level signals on the four addressing pins (refer to Table 6.5-1). Remember that the first access to these registers will select Mode Register 1 and the second access will select Mode Register 2.

Mode Register 1 controls the type of transmition and the baud rate multiplier along with establishing certian criteria for the transmition and receiption of the data. Through this mode register, the operator can select the character length, the type of parity and whether parity is to be used or not. Also the operator can select the number of stop bits that are to placed at the end of each word transmitted. If the operator selects synchronous transmision then he or she may also select the number of sync characters and the type of sync. A complete breakdown of the indivual bits and their function can be found in the reference manual refered to above.

Mode Register 2 deals with the selection of the baud rate and the selection of either the internal baud rate generator or the external clock inputs. Through this register the operator can select whether the internal BRD (Baud Ragte Generator) or the external clock pins are to be used, for your purpose the external pins will be used. Along with this selection the operator can select one of 16 different baud rates, if the internal BRG is selected, or he/she can select the function of the two external pins. The operator has many different combination that he or she may choose from. As an example the operator can select an external clock for transmit and the internal BRG for receive, or vasious combination thereof. Keep in mind that for the purpose of this workbook, further explaination is not really necessary. The IPC board will alway be set up to use an external clock that is running at 5.06 Mhz., and that pins 9 and 25, the programable external pins, are not used.

The command register is just that, a command register. It controls when the SC2661 will begin transmitting or reveiving. Out of the 8 bit that are located within this register thare are seven (7) commands that it may perform. Certial bit within this register will cause some of the output pins to go to a high state and stay there until these bits are changed. This is important to know if you are missing signals or are not receiving the proper responce from the signal you are sending the circuit.

The first two bits that you should know about are bits CR0 and CR2 (the CR stands for Control Register bit). CR0 controls the transmit operation. When this bit is high the SC2661 will begin to transmit. If this bit is brought low the transisiom will terminate at the end of the current word, and the TxD output pin (19) will go high and remain there. The second of these two bit is CR2 which controls the receive operation. When this bit is brought high the SC2661 will begin seaching for a start bit from the connected device. If CR2 is brought low the receive operation will terminate immediately, and any character that was being assembled will be abborted.

The next two bits to be decussed are CR1 and CR5. These two bits control the Data Terminal Ready and Request To Send signals. Each time the SC2661 prepares to transmit a word it will generate a Request To Send signal. The receiving end will then generate a clear to Sent signal informing the SC2661 it is OK to transmit. If CR5 is set the SC2661 will generate this Request To Send signal all the time, and if this bit is brought from a high to a low the SC2661 will then force this output high at the complection of the current transmition where it will remain.

CR1 on the other hand controls the condition of the Data Terminal Ready signal. This signal is used for the receiving of data from another communication device. This signal would be connected to the Clear to Send input of the other device, indicating to it that its Request To Send was acknowedged. If this bit (CR1) is low then &DTR will be forced high, indicating a not ready condition. If CR1 is high then pin 24 (&DTR) will indicate a ready condition by going low.

The remaining bit of the command register have no effect on the external pins of the chip. One final point concerning the command register, each time that the microprocessor reads the data of the command register the pointers of the mulit-level registers are reset.

The status and sync registers have no direct effect on the output or input pins. Through the status register the micro can obtain the status of the current operation, and through the sync registers the micro can dictate the format for synchronous operation. Further explaination of these registers and infromation that they provide can be found in the fore-mentioned reference.

6.6 Troublshooting the IPC board

To aid the technician in the repair of the IPC circuit board a repair aid has been developed by the ATE department. This repair aid is designed to be used by any technician, even if the tech does not have prior knowledge of the Z80A micro-code. To provide this type of repair aid the Millenium Microsystem Analyzer (uSA) is used. the technicain should be familier whith the operation of the Millenium before using this aid. Fault isolation is accomplished by combining in-circuit emulation and signature analysis provide by the Millenium. A complete listing of the test involved and proceedure steps are covered in Section 8 of this manual.

SECTION 6 QUIZ

1) What are the four sections of the Internal Printer Controller (IPC)?

2) What function does the NOP generator provide?

3) If you wish to test the IPC board without a printer attached, what jumper would you change?

4) What decodes the SC2661C I/O operations?

5) What action will cause the reset of the register pointers within the SC2661C?

6) When the IPC board is in the diagnostic mode what type of parity is used?

7) What four pins on the SC2661C are used to access the internal registers?

8) What functions does Mode Register 1 control?

SECTION 7
INTERNAL WISE CONTROLLER

SECTION 7
Internal WISE Controller (IWISE)

7.1 INTRODUCTION

This section will provide the reader with all the information necessary to allow him or her to troubleshoot the 210-8270 PCB, more commonly referred to as IWISE. The reader will first be presented with an overview of the IWISE system and it's capabilities. A detailed description of the IWISE protocol will follow. The reader should study this information very carefully in order to comprehend the timing and logic involved in the transfer of data from one system to another. The remaining sub sections will provide you with a detailed description of the logic circuits of the IWISE PCB.

7.2 Overview

The Internal WISE Controller, more commonly call IWISE, provides the user with a high speed communication link from one OIS system to any other system that uses the Wang standard dual coax communication link (928 standard interface). Through IWISE the operator of one system can manipulate files, documents and peripherals of another system, thus increasing the power and flexibility of both systems.

The IWISE board, which is mounted in channel (5) of the OIS 40/50/60 system contains all the logic necessary to provide this data link. The software needed for running the wise system must be resident in both systems, and both system must be configured properly. The host or master system is the only system that is require to have an IWISE board mounted, unless several systems are to be connected together. You should refer to the appropriate section(s) of the maintenance manual to insure that your system is set up correctly.

There are six function that the IWISE board can perform, these are;

1. Status Read
2. One Byte Read
3. One Byte Write
4. 256 Byte Read (Block Read)
5. 256 Byte Write (Block Write)
6. Restart

These functions are transparent to the operator, but with these six function the operator can as stated before transfer data between the two systems. A detailed description of each of these functions will be given in subsequent sub sections.

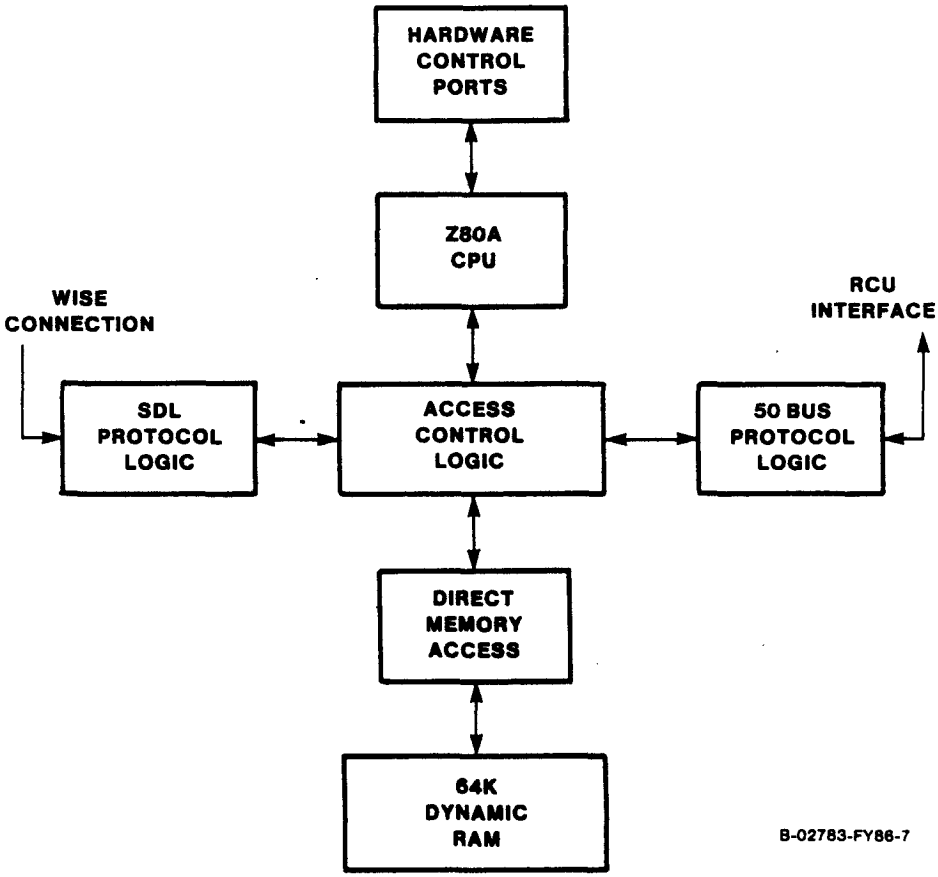


FIGURE 7.2-1
IWISE Block Diagram

The IWISE board is dived into seven section. FIGURE 7.2-1 shows a block diagram of the IWISE board. These seven section provide the means of communication between the host system and the remote system. A brief description of each section is presented in the following paragraphs.

7.2.1 Central Processing Unit

The CPU of the IWISE board contains a Z80A microprocessor, a Z80A Counter Timer Circuit and support logic, all of which are operated by a 4Mhz clock. The Z80A performs instructions that are present to it by the host system and provides refresh for the 64K Bytes of on board memory. It also has control of some of the hardware on the IWISE board. Of the three devices that can access the onboard memory the CPU has the lowest priority. A complete breakdown of memory priority can be found in the section on Access Control Logic.

7.2.2 Serial Data Link Protocol logic

The serial data link protocol logic provides the communication link between the IWISE memory and the remote master via the serial coax data link. The SDL is the second priority device for access to the IWISE memory.

7.2.3 50BUS Protocol Logic

The 50BUS Protocol logic provides the communication link from the OIS 40/50/60 master to the IWISE via the 50BUS. The 50BUS has the highest priority for access to the IWISE memory. The 50BUS protocol logic provides intermediate storage for an address and data accepted from the 50BUS as it is tranferred to the IWISE memory through the access logic.

7.2.4 Access Control

The access logic provides priority control for the three independent memory users on the IWISE board. The three priority levels are;

1. 50 BUS Protocol Logic
2. SDL Protocol Logic
3. CPU Logic

7.2.5 Direct Memory Access Logic

The direct memory access logic provides a read/write operation into the IWISE memory for the 50BUS, the Serial Data Link, and the CPU under control of the Access Control Logic. This logic ensures that access from either the 50BUS or the SDL are not interrupted and provides the proper wait state logic for the CPU during it's access.

7.2.6 IWISE Memory

The IWISE memory consists of 64K by 9 bits of dynamic RAM. This memory can be accessed by either the 50BUS Protocol logic, the SDL Protocol Logic, or the CPU. Refresh is provided by the CPU when it has access. A parity bit is generated and stored in the ninth bit each time data is written to memory, and then checked on all reads from memory.

7.2.7 Hardware Control Ports

The IWISE uses the Z80A input/output commands to provide IWISE hardware control. Through these ports the system can place the IWISE board in a diagnostic mode, select or deselect the Data Link and control the CTC. A detailed description of these ports will be discussed in Section 7.10

7.3 IWISE Protocol

A protocol is a set of rules and regulations that governs the transmission and reception of data via the serial data link. These rules and regulation provide a logical sequence for the transfer of data from one system to another.

The logic circuits that control the transfer of the data are designed around this protocol and expect that the data it receives will follow these rules. If the rules are not followed, then incorrect transfers of data, or even worse no transfers of data can be accomplished. As for the technician that is assigned the task of repairing the circuits involved in this process, he or she must understand the sequence of operation that are taking place in order to effectively troubleshoot and diagnose any problem.

The protocol is arranged in two phases. The first phase is called the command phase. In this phase the SDL logic receives a command from either the master or the remote system. This command will consist of from one to three bytes. The second phase of the protocol is the data phase. In this phase the SDL logic will either transmit or receive the

requested data. If the command was received from the remote system and the command requested that data be transmitted to the remote, then the SDL logic must turn around the Data Link. Remember that the Data Link is a serial coaxial line and that data can only travel in one direction at a time. This turn around time is dictated by the protocol to be 8 usec. Let's take a look at an example of what will take place for a typical transfer.

For this example the remote will be requesting a one byte read. The SDL logic is normally in the receive mode of operation. During this time the logic is looking for a start bit. This start bit will signal to the SDL that the remote is sending information. The SDL receives the command and decodes it. The decoder logic determines that the command is a one byte read operation. It then awaits for the address bytes to be sent by the remote unit. The higher order byte of the address is sent first followed by the lower order byte. Once the entire address is received the SDL will turn the Data Link around. As I stated before this takes 8 usec. During the turnaround time the requested byte of data is retrieved from the IWISE memory and places in a holding register waiting for transmission. At the end of the turnaround time the SDL will transmit the request data to the remote unit. After the SDL logic transmits the data it will then turnaround the Data Link to await another command. This second turnaround time is dictated by the protocol to be no longer than 7.5 usec.

The above sequence is followed for each of the six commands that the IWISE board can perform. The only difference that might take place is if the command was for a write operation. During a write operation there would be no turnaround time involved, because the remote unit is the one doing all the transmitting.

7.3.1 Serial Transmit Operation

Now that the guide lines of transferring information from one unit to another have been explained, we can turn our attention to a more detailed view of this transfer of information. When moving information from one unit to another through a serial Data Link we use a method called Asynchronous transmission.

Whether the byte of information to be transmitted is a command or it is data the rules that apply are the same. Each word that is transmitted (we call the transmitted information a word) consist of one start bit, eight bits of data, one bit of parity, and one stop bit. This gives us a total of (11) bits. Each of these bits lasts for a given period of time. The receive clock that keep the receive circuits in sync with the transmitted data determines the length of each bit. In the IWISE system this clock period, or what we call the bit time is 233.9 nsec. So with a total of (11) bits in a word we get a word time of 2.573 usec.

The start bit, which is always the first bit of any transmission is a signal to the received circuits to start. Remember that the SDL logic is normally in the receive mode. While in this mode of operation and there is not activity on the Data Link, the line contains all zeros, indicating to the receiver that nothing is happening. When the receiver detects the start bit, which is a logic one, it turns on the rest of the receive circuits and starts looking for data bits. As each data bit is received they are shifted into a serial-to-parallel register. After eight bits are received the SDL logic then looks for a parity bit, which it uses to check the validity of the transmission. After the parity bit is received the SDL looks for the stop bit. This stop bit is always a zero and indicates to the receiver that the transmission is complete. After the transmission is complete the SDL logic circuits then acts upon the received byte and determines whether is was a command byte or a data byte.

Wang uses a system of transmission called Non-Return-To-Zero. What this means is that each bit of the transmitted word does not have to return to a zero reference point before the next bit is transmitted. When using this type of transmission each bit is timed in order to determine where the logical break is between bits. The bit time as I stated before is 233.9 nsec. At the end of this 233.9 nsec the shift register that collects the transmitted word will switch to the next bit. If you refer to FIGURE 7.3-1 you will see a typical word transmission.

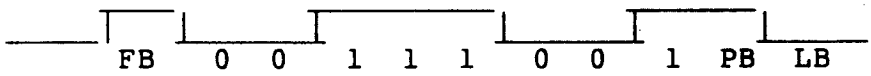


FIGURE 7.3-1
Typical Transmitted Word

FB = First bit (Start Bit)
LB = Last Bit (Stop Bit)
0 = Logic level zero
PB = Parity Bit (odd parity)
1 = Logic level one

Each time the shift register switches to the next bit a counter is incremented. When this counter reaches ten the SDL logic looks for the parity bit. Then on the eleventh count, the logic circuits look for the stop bit. If on the eleventh count there is not a zero the circuits will indicate an error. This error is called a framing error, meaning that the transmission has lasted to long, or ended to early, and that the data received is not valid. To help you visualize this type of error think of how a move projector works. As the film moves through the lens assembly the shutter is opened and closed allowing the light to pass through the film. If the film is not centered in the shutter when it is open then the viewer will see a black bar, either at the top or at the bottom of the picture. This bar represents the logical break between pictures. You didn't see the whole picture so part of the information was lost. The same thing happens in the transmission of data from one unit to another if the clock rates of the two units are not synchronized. This synchrnoization is accomplished by the start bit. The leading edge of the start bit switches on the clock circuits of the receiver, thus providing a starting or synchronizing point.

Now let's look at the parity bit. When the transmitting unit assembled the data to be sent, it generated a parity bit. In the Wang system we use odd parity. This means that an odd number of one are always transmitted. Referring back to FIGURE 7.3-1 we see that the data bits consisted of four zeros and four ones. Since there are four ones in the data byte the parity circuits will generate an extra (1) so that there will be a total of five (1)s or an odd number of (1)s transmitted. When the receive circuits receive the data bits they also count the number of (1)s. They then determine the number of (1)s in the byte and generate their own parity bit. This parity bit is then compared with the parity bit that was generated by the transmitter. If they are the same, then the transmission was successful, otherwise an error occurred. If an error occurs the SDL logic circuits informs the CPU of the error so it can take appropriate steps to correct it.

Before we leave this topic we have one more item to discuss, the order of the data bits. When the transmitter assembles the data to be transmitted it selects the MSB (Most Significant Bit) of the byte first, and then works its way towards the LBS (Least Significant Bit).

To rap this topic up let's take one last look at the structure of the transmitted word. The first bit received is the Start Bit, this bit is always a one. Then the eight data bits follow with the MSB first. After the data bits, comes the parity bit. This bit will be either a one or a zero depending on the number of ones in the data bits. Finally the stop bit is received, this is always a zero, which if you noticed returns the Data Link back to its none-busy state so that the receive circuits can except the next word. In the next section you will learn about the different commands and how they are structured.

7.4 Command Structure

In this section you will learn the different commands that are acceptable to the IWISE system. You will also learn how each command is organized and what functions the IWISE system must perform for these commands.

We learned back in Section 7.3 that there are five commands that the IWISE system can except. I will review them for you here.

	COMMAND	HEX CODE
1.	Status Read	B0
2.	One Byte Read	A2
3.	One Byte Write	A3
4.	256 Byte Read (Block Read)	A4
5.	256 Byte Write (Block Write)	A5
6.	Restart	A8

The hex code that is next to each of the command is the code that the system will look for when it received the command word. If one of these codes are not received during the command phase of the transfer then the logic circuits will generate an error to the CPU. On the following pages each of the commands will be discussed in detail.

7.4.1 Status Read

The Status read command code is B0(H). When this command is received and detected, the IWISE Serial Data Link Protocol Logic (SDLPL) must switch itself into the transmission mode, this requires a line turnaround, and transmit the IWISE status information to the master which sent the request.

The command structure for the Status read command look like this:

COMMAND	TURNAROUND TIME	DATA TRANSMITTED
FB,C7,C6,C5,C4,C3,C2,C1,C0,PB,LB	.....	FB,0,1,1,1;0,CE,ME,1,PB,LB

Where: FB = first bit, C(x) = command bits, PB = parity bit, CE = channel error, ME = memory error, LB = last bit

The bit assignments for the Status read command is listed in Table 7.4-1 on the next page.

BIT DESIGNATION	SIGNAL NAME	SIGNAL DESIGNATION	BIT VALUE
D0	NONE	NONE	"0"
D1	DATA LINK ERROR	DLERR CE	"0" = 'OK' "1" = 'DL ERROR'
D2	MEM PARITY ERROR	MPE ME	"0" = 'OK' "1" = 'Mem ERROR'
D3	NONE	NONE	"1"
D(4-7)	DEV TYPE NUMBER	DT #	"0"(L)

Table 7.4-1
STATUS BYTE ASSIGNMENTS

D0 This bit does not contain any information it is always a zero

D1 This bit is the Data Link Error bit. If this bit is a one, it means that the IWISE memory is disabled for write operations from the Serial Data Link because one of the Data Link fault condition has occurred:

1. Data Link Parity Error (DPLE). This means the the Data Link Protocol Logic detected a parity error in the information received from a remote master.
2. Data Link Command Error (COMERR). This means that the first byte that was accepted by the Serial Data Link Protocol Logic was not a COMMAND.
3. Data Link Sequence Error (SEQERR). This means that an interval of more than 8 usec. without any information transmitted from a remote master, was detected during a command sequence.

The D1 error bit is cleared when the hardware status byte is read by the remote master or by the restart command issued by the 50 BUS.

D2 Memory Parity Error bit. If this bit is a one, it means that a parity error occurred in the IWISE memory. If the IWISE board was in a normal mode of operation then the IPL bit would be set and the IWISE board would suspend operation until the OIS 50 re-IPLed the board. If the IWISE board was running in the diagnostic mode of operation, then a Non-Maskable interrupt to location 0066 or 0166 will occur. The location of the Non-Maskable interrupt depends on the state of the PAGESEL (Page Select) flip flop.

D3 This bit does not contain any information and is always a one.

D4-7 These bits identify the IWISE board to the system. They will always read as follows:

D4 = "1"
D5 = "1"
D6 = "1"
D7 = "0"

7.4.2 One Byte Read Sequence

The One Byte Read Sequence consists of; the One Byte Read Command (A2) followed by two bytes of the desired IWISE memory address (high order byte of address first), which is transmitted on the Data Link by the remote master to the IWISE. The line is then turned around and the requested data is transmitted to the remote unit from the IWISE board. The command sequence look like this:

COMMAND

FB,C7,C6,C5,C4,C3,C2,C1,C0,PB,LB

ADDRESS HIGH ORDER

FB,A7,A6,A5,A4,A3,A2,A1,A0,PB,LB

ADDRESS LOW ORDER

FB,A7,A6,A5,A4,A3,A2,A1,A0,PB,LB

TURNAROUND TIME

8 usec.

DATA TRANSMITTED

FB,D7,D6,D5,D4,D3,D2,D1,D0,PB,LB

7.4.3 Block Read Sequence

This sequence transmits a block of 256 bytes of data from the IWISE board to the remote master. The Block Read sequence consists of: the Block Read Command (A4) followed by two bytes of the desired starting memory address (higher order byte first). Then the Data Link is turned around and the 256 byte block is transmitted to the remote master. The command sequence look like this;

COMMAND

FB,C7,C6,C5,C4,C3,C2,C1,C0,PB,LB

ADDRESS HIGH ORDER

FB,A7,A6,A5,A4,A3,A2,A1,A0,PB,LB

ADDRESS LOW ORDER

FB,A7,A6,A5,A4,A3,A2,A1,A0,PB,LB

TURNAROUND TIME

8 usec.

First Byte of DATA TRANSMITTED

FB,D7,D6,D5,D4,D3,D2,D1,D0,PB,LB

Second Byte of DATA TRANSMITTED

FB,D7,D6,D5,D4,D3,D2,D1,D0,PB,LB

.....

.....

256th Byte of DATA TRANSMITTED

FB,D7,D6,D5,D4,D3,D2,D1,D0,PB,LB

NOTE: The least significant byte of the start memory address for the Block Read Command must always be zero.

7.4.4 One Byte Write Sequence

The One Byte Write sequence consists of the One byte Write Command (A3) followed by two bytes of the desired memory address (higher order first) and one byte of data to be written into the IWISE memory. There is no turnaround time require for this operation because the remote unit is doing all the transmitting. The command sequence is as follows:

COMMAND

FB,C7,C6,C5,C4,C3,C2,C1,C0,PB,LB

ADDRESS HIGH ORDER

FB,A7,A6,A5,A4,A3,A2,A1,A0,PB,LB

ADDRESS LOW ORDER

FB,A7,A6,A5,A4,A3,A2,A1,A0,PB,LB

DATA TRANSMITTED

FB,D7,D6,D5,D4,D3,D2,D1,D0,PB,LB

7.4.5 Block Write Sequence

This sequence transmits a block of 256 bytes of data from the remote master to the IWISE memory. The Block Write sequence consists of the Block Write Command (A5) follow by two bytes of the desired memory address in the IWISE board (higher order byte first) then the 256 data bytes are transmitted. The command sequence look like this:

COMMAND

FB,C7,C6,C5,C4,C3,C2,C1,C0,PB,LB

ADDRESS HIGH ORDER

FB,A7,A6,A5,A4,A3,A2,A1,A0,PB,LB

ADDRESS LOW ORDER

FB,A7,A6,A5,A4,A3,A2,A1,A0,PB,LB

7.4.5 Block Write (cont.)

First Byte of DATA TRANSMITTED

FB,D7,D6,D5,D4,D3,D2,D1,D0,PB,LB

Second Byte of DATA TRANSMITTED

FB,D7,D6,D5,D4,D3,D2,D1,D0,PB,LB

.....

.....

256th Byte of DATA TRANSMITTED

FB,D7,D6,D5,D4,D3,D2,D1,D0,PB,LB

NOTE: The least significant byte of the start memory address for the Block Read Command must always be zero.

7.4.6 Restart Command sequence

This command does not make the IWISE restart because the remote master never performs the IPL function with IWISE. When this command is decoded, the Data Link will be terminated, and the reception of a new command by the IWISE will be suspended until at least 8 usec. of silence occurs on the serial Data Link. The command sequence look like this:

Command

FB,C7,C6,C5,C4,C3,C2,C1,C0,PB,LB

8 usec of silence

Next command may now be received

This concludes the command structures of the IWISE system. In the follow pages I will discuss in detail each of the seven parts of the IWISE board.

7.5 Detailed Theory of Operation

In this section I will present to you a detailed description of each of the seven parts of the IWISE board. You will be referred to the schematics and drawings from time to time. There is one item that I would like to point out. For reasons unknown to this developer the engineer that designed this board decided to use negative logic. Now this is ok but it does make things confusing sometimes. If you are not sure of what I mean by negative logic just look at one of the schematics. You will see that the AND gates and the OR gates have bubbles on both there inputs and outputs. What this does is the diagram looks like a AND gate but in reality it is an OR gate. So please try not to get confused as I explain a circuits and I say that two signals are ANDed together and you look at the circuit and see a OR gate. It's not my fault!

7.5.1 Central Processing Unit

The CPU of the IWISE board consists of a Z80A microprocessor running at 4 Mhz in conjunction with a Z80A-CTC for prioritizing and controlling the interrupts received from the IWISE system.

7.5.1.1 Clock

The Z80A system clock is generated by crystal Y1 which runs at 16 Mhz. This 16 Meg signal is then divided by two flipflops L61-1 and L61-2 (1G10). The "Q" output of L61-1 produces an eight (8) Mhz clock used for the priority control logic (to be described later). The 4 Mhz system clock produced by the "Q" output of L61-2 is distributed throughout the circuits for timing operation. The only other conditioning that this signal receives is through transistor Q1 (1E10) and inverter L44. This circuits ensures that the 4 Mhz clock going to the Z80A has good clean transitions, to promote more reliable operations.

7.5.1.2 Buffers (Data & Address)

Both the data and address line of the Z80A are buffers prior to connecting to the memory of the IWISE board. L47 serves as a data buffer while L30 and L14 buffer the address bus. When the 50BUS or the Data Link request access to the IWISE memory these buffers are tri-stated, thus detaching the Z80A from the internal busses. L14 and L30 receive their enable signals from the BUSAK (Bus Acknowledge) signal of the Z80A, while L30 the data buffer uses a different technique. The data buffer L30 receives it's enable signal from a combination of &MREQ and &IORQ. During a normal read or write operation the Z80A will generate &MREQ (Memory Request). This signal,

after being buffered by L45 (1C7) is placed on pin 2 of OR gate L28A (1G8). Here is the first occurrence of negative logic, L28A is drawn as a NAND gate with it's inputs bubbled. Remember that it works as (and really is) an OR gate. The other input to L28A on pin 1 comes form the IPL bit. As long as the system has not been just powered up, or a parity error occurred this signal will be LOW. Since L28A is an OR gate and both inputs are low it will output a low form pin 3, placing this on pin 1 of AND gate L79. As you know any low into an AND gate will cause a low out. This low out of pin 3 of L79 will enable Data Bus Buffer L47. One thing I would like to point out here is that pin 19 of L47 is shown as an active high input. In reality this enable input is active low. This is just an error on the draftsman part.

There is one other time that the Z80A needs access to the data bus, this is when an I/O operation is being performed. The major difference between an I/O operation and normal read operation is that the Z80A does not produce a &MREQ signal for the I/O operation. So to enable the data buffer another means must be used. Some I/O operation do not use the data bus or do not need the data buffer enabled. For example the IN XX07 instruction will cause the NOP generator to start running. The other time that the data bus buffer is not needed is when communicating with the CTC chip, because the CTC is tied directly to the Z80A data bus. This communication is through I/O operations with addresses of XX10 or above.

Now with these exceptions in mind let's see how the I/O operation are handled. The &IORQ signal is generated from the Z80A and passed through buffer L45 where it is placed on pin 13 of OR gate L56. Pin 12 of L56 receives its signal from bit 4 of the address bus (if bit four is high then the address is XX10 or above). As long as this bit is low then L56 will output a low to pin 2 of OR gate L21. Pin 1 of L21 receives its signal from and inverted IN XX07 signal (this signal is decode by the I/O decoder L60 that I will discuss later). As long as we are not performing an IN XX07 instruction this signal pin 1 of L21 will be low. Now with both pins 1 and 2 low pin 3 will output a low to AND gate L79 (weren't we here before). This low will then be transferred to the enable input of the data buffer L47.

7.5.1.3 NOP Generator

Attached directly to the Z80A data bus pins are the NOP generator and the CTC. These two devices do not use the data buffer. The NOP generator places a NOP code on the data bus of the Z80A whenever the system is powered on, or when a parity error is encountered or if an IN XX07 instruction is performed. We have see this type of circuit in the other board of the OIS 40/50/60 system so a detailed explanation is not necessary. The CTC on the other hand does need a little explanation.

7.5.1.4 CTC

The CTC operation as you know is to control the different interrupts from the IWISE system. In order to do this operation the CTC must have its internal counter programed by the Z80A, and when a interrupt occurs the CTC places the address of the interrupt routine on the data bus for the Z80A to use as a vector. These two operations do not need access to the memory of the IWISE board. To program the CTC counters the Z80A must perform an OUT XX10 or above (a listing of the addresses for the CTC can be found in Table 7.5-2). When the Z80A does this OUT instruction the &IORQ signal is generated along with bit 4 of the address bus being brought high. Bit 4 of the address bus, which we call MADD4 (Memory Address 4) is inverted through L46 (1E7) and placed on the &CE (Chip Enable) input of the CTC. The CTC then decodes it's &RD, &IORQ and the CS0 and CS1 inputs to determine which register the data is to be placed. Remember that during this OUT instruction the Z80A will place the appropriate data on the D0-D7 pins. If you noticed I said that the CTC decodes the &RD input along with the others. Well when the Z80A performs an OUT instruction this signal is not active, but when the Z80A desires to read the status of the down counter in the CTC, this signal is active.

7.5.1.5 I/O

The Z80A communicates to different hardware devices through it's input/output instructions. When the Z80A performs either an IN or an OUT instruction it produces a signal called &IORQ. These operation are decoded by two LS154 decoders, L59 and L60 (1D-G2). The &IORQ signal along with either the &RD for IN operation, or the &WR for OUT operation determines which of the decoders are selected. There are 32 I/O operations, 16 IN operation and 16 OUT operations, of course not all of these operation are used. Table 7.5-1 lists each of these operations. A more detailed description of what they control can be found in the section on Hardware Control Ports.

Table 7.5-1
I/O Port Allocations

Port Number	Operation	Function
XX01	OUT	IWISE Diagnostic Mode Switch
	IN	Read Diagnostic Mode Switch
XX02	OUT	Data Link Mode Switch
	IN	Read Data Link Mode Switch

Table 7.5-1 (cont.)
I/O Port Allocations

Port ADDRESS	Operation	Function
XX03	OUT	Parity Select Switch
	IN	Read Parity Select Switch
XX04	OUT	Data Link Test Mode Clock Switch
	IN	Read Data Link Test Mode Clock Switch
XX05	OUT	Clears Detect Flipflops
	IN	Reads Status of Detect Flipflops
XX06	OUT	Sets the Test Transmission Serial Data Flipflop
	IN	Reads the Test Transmission Serial Data Flipflop
XX07	OUT	Data Link Disable Switch
	IN	Invokes NOP Generator
XX08	OUT	Restart Disable
	IN	IWISE Status
XX09	OUT	Page Select Switch
	IN	Read Page Select Switch
XX0A	OUT	NOT USED
	IN	Code Request
XX0B	OUT	Clear MPE bit of Status Register
	IN	Restart IWISE
XX0C	OUT	Load Indicators 0,1
	IN	NOT USED
XX0D	OUT	Load Indicators 2,3
	IN	NOT USED
XX0E	OUT	Load Indicators, 4,5
	IN	NOT USED
XX0F	OUT	NOT USED
	IN	Read Serial Data Transmitted by IWISE

Input/output address from XX10 through XX13 are use to control the CTC timer operations. These address are decoded by the CTC chip from the MADD0, MADD1, and MADD4 bits of the address bus. Their allocations are presented in Table 7.5-2.

Table 7.5-2
CTC I/O Allocations

Port ADDRESS	Operation	Function
XX10	OUT	Counter/Timer 0 (High Priority)
XX11	OUT	Counter/Timer 1
XX12	OUT	Counter/Timer 2
XX13	OUT	Counter/Timer 3 (Low Priority)

7.5.1.6 Interrupts

While we are here talking about interrupts we should look at the different types of interrupts that the Z80A can handle. First the Maskable Interrupt. This interrupt is controlled by the CTC and will force the Z80A to run an interrupt routine at a predetermined location depending on the cause of the interrupt. One important point about Maskable Interrupts, if the &BUSRQ (Bus Request) input of the Z80A is active a Maskable Interrupt will not be serviced. The second type of interrupt that the Z80A can handle is the Non-Maskable Interrupt.

When the IWISE board is in a diagnostic mode of operation (this can be done be performing an OUT XX01 with D0=1) and the parity circuits detects a parity error a Non-Maskable Interrupt will be generated. When the Z80A receives this interrupt it will complete the cycle that it is currently in and proceed directly to location 0066(H), where it will find instructions on what to do for the interrupt. This type of interrupt will terminate a Bus request activity by deactivating the &BUSAK signal.

Looking at sheet 1 of the schematics at location A11 we see the signals &MPERR (Memory Parity Error) and IWDM (IWISE Diagnostic Mode). These two signals are feed into OR gate L92, IWDM is inverted first. IWDM will be a logic one when active, so after it is inverted by L95, both pins 4 & 5 of L92 will have lows on them. This means that pin 6 will output a low and cause a Non-Maskable Interrupt to the Z80A. More information on memory parity error can be found in the section on the parity circuits.

There are just two more parts of the CPU section left to discuss, so hang in there we are almost done. What does remain is the WAIT state generator and the RESET circuits. First the WAIT state generator just to get it out of the way.

7.5.1.7 WAIT

During each M1 cycle of the Z80A WAIT states are inserted. These WAIT states are used to allow the memory circuits time to place the data on the bus before the Z80A reads it into its registers. These WAIT states are inserted by two flipflops L30A-1 and -2. When the Z80A starts a M1 machine cycle it generates the signal M1 from pin 27, where it is buffered through L45. From pin 5 of L45 (1C7) we can follow M1 over to pin 2 of L30A-1(1D9), the "D" input. On the next rising edge of the 4 Mhz clock L30A-i will reset. The "Q" output of L30A-1 is connected to the WAIT input of the Z80A. This low causes the Z80A to insert a WAIT state in the present cycle. On the next rising edge of the 4 Mhz clock L30A-2 will reset, causing L30A-1 to set, thus removing the WAIT input to the Z80A.

7.5.1.8 RESET

Now the last circuit of the CPU section, the RESET circuits. The reset operation is divided into two parts. When the OIS 40/50/60 system is first powered up, the RCU board generates a MRESET (Master Reset) signal to all the internal boards. This MRESET signal enters on sheet 1 of the schematics at location A11, here it generates &RES RESTART, which resets the Z80A of the IWISE board. It also generates &IWMRESET (IWISE Master Reset). &IWMRESET is distributed throughout the board. It first sets the IPL bit which activates the NOP generator and Disables the Data Link Receive circuits. Next &IWMRESET activates the Data Link disable flipflop, clears the Data Link Command Decoder, clears the Receive Buffer, and activates the 50 Restart Control flipflop. At this time the board is basically in a reset state, but there is no operation code on board and the NOP generator is forcing the Z80A to perform constant M1 machine cycles.

The second part of the reset operation is for the RCU to IPL the board. First the RCU will down load a jump to zero instruction to location zero of memory. Then the RCU will generate a 50 Select and a 50 Restart command to the IWISE board. This will clear the restart flipflop L77 (1C11), which in turn triggers L110 a Monostable multivibrator. The "Q" output of L110 clears the memory parity error flip, while the &"Q" output clears the CTC, reset the Z80A (again), and generates &IWRESTART (IWISE Restart). This signal then clears several status bits and flipflops, refer to Table 7.5-3.

Table 7.5-3
&IWRESTART RESET TABLE

1.	IPL Bit	4.	MPE Bit
2.	CLERR Bit	5.	PAREVEN F/F
3.	DLTM F/F	6.	TTMSD F/F
7.	Page Sel F/F		

7.6 Serial Data Link Protocol Logic

Before the Serial Data Link logic can be discussed we must first determine who and what can control it. We will start by looking at the IWISE memory and how it is accessed.

There are three devices that can access the IWISE memory, the 50BUS, the Z80A, and the SDL logic. The 50BUS and the SDL logic have the highest priority for access, while the Z80A the lowest. The onboard Z80A has very little effect on the transfer of data through the SDL so we will ignore it for now.

The IWISE memory can be thought of as being divided into two major areas. This is not a physical division but a software division, and both the local master and the remote master can access these areas. For the sake of simplicity we will call these areas LOCAL and REMOTE. When the local master desires to communicate with the remote it will write the appropriate code into the remotes area of memory. The remote will then read this code and determine what information is requested. The remote on the other hand will communicate with the local master by writing commands in the local masters part of the IWISE memory. Each of the two masters consider the IWISE memory as their slave, and have no idea that the other master is also controlling it.

The local master has a direct route to the IWISE memory through the 50BUS, while the remote master must communicate to the memory through the SDL logic. The local master can not access the SDL logic circuits. If the local master requires data to be transferred via the SDL it must generate a request. This request consist of writing a request command to the remote's area of the IWISE memory and then waiting for the remote to transfer the data. When the remote has finished the transfer it will write the status of the operation into the local masters area of memory.

Let's now turn our attention to the SDL circuits. These logic circuits are control entirely by the remote master. The remote master will send commands through the serial link, and the SDL logic circuits will decode these commands and act upon them. As I discuss the circuits of the SDL think of them as a slave to the remote master. This slave can handle six different commands, but only one at a time. With these six commands our slave (SDL) can either read or write data to the IWISE memory, receive or transmit data on the serial link, or restart the SDL logic. One final point, when the SDL receives a read command it will transfer data from the IWISE memory to the remote master on the serial link. And when the SDL receives a write operation it will take the incoming data from the serial link and write it to the IWISE memory. Now with that all straight, let's proceed to the logic circuits.

7.6.1 SDL Receive

Before I get into the logic circuits of the SDL, let's look at the sequence of events that take place during a command operation. The command is divided into either two or three phases depending on the type of command. The first phase is always the same, this is the command phase. In the command phase the receives circuits receive the command an transfer it to the command decoder logic. Here it is checked for validity and parity. The second phase is determined by the type of command that was received in the command phase. If the command received was a Read Status, then the second phase will be to turnaround the serial link and transmit the status register information to the remote master. If the command received was a restart command then the second phase will be restart the SDL logic. These two command will end after two phases.

If the command received was that of either a read or write operation then the second phase will consist of receiving the address of the read or write operation. The SDL logic will then enter into the third phase, the data phase. During the third phase the SDL will either receive or transmit the requested data.

7.6.2 Data Acquisition

The acquisition of data from the serial link is the same for each of the phases. Serial data from the data link is presented to the inputs of L150 (5C10). This 75107 takes the high and low information on the data link line and converts it into TTL levels. The serial received data generated by L150 is then transferred to the start bit detector. The detection of a start bit is by clocking the received data through three flipflops. If after three clock pulses of the 32.4 Mhz clock all three "Q" output of L134 (5B9) are high then NAND gate

L117 will output a zero. This zero call &START BIT set flipflop L116-1 and flipflop L69-1. Also &START BIT clears flipflop L69-2 and the serial to parallel shift register. When &START BIT sets L116-1 the &"Q" output deactivates the start bit circuits. The "Q" output of L116-1 enabled clock divider L163 and placed it into a shift mode of operation. This divides the 34.2 Mhz clock by 8 to provide us with a BIT STROBE of 233.9 nsec.. BIT STROBE is use to clock the serial data into the serial to parallel shift register.

The serial data from L150 is then passed through two flipflops, L69-1 and L69-2 on its way to L86 the shift register. After 9 BIT STROBE clock pulses the "Q_H" output of L86 will go high. This high is placed on the "D" input to L70-1. Then on the 10^{th} BIT STROBE L70 is clocked, signaling that the data byte has been received. When L70-1 sets, the "Q" output enables L70-2, L85, and L68. A sequence of six steps will now take place. These six steps are described in Table 7.6-1.

Table 7.6-1
BYTE ACQUISITION AND CHECK TIMING

L85 Count	Function
000_2	Reset Start Bit flipflop.
001_2	Load data into buffer, check parity, and clear delay.
100_2	Enable parity error circuits.
101_2	Transfer data to command decoder, enable address and data receive circuits.
110_2	Check for command errors.
111_2	End byte acquisition and clock command sequencer.

L70-2, L85, and L68 make up the Byte Acquisition and Check circuit. This circuit insures that the received byte is valid. L85 a simple four bit counter will provide a sequental output to decoder L68. As L85 advances in its count L68 will draw one of its outputs low, thus generating a sub command. When L70-1 was set by the ninth BIT STROBE L68 brought the "Y_0" output low. This high to low transition is inverted and sent to the clock input of L116-1 the Start Bit Detector flipflop. L116-1 will then go reset causing the BIT STROBE clock circuits to stop, and re-enabling the start bit detector for the next byte.

Our 34.2 Mhz clock is still running, so L70-1 divides this by two, and clocks L85, causing it to advance by one. Referring to Table 7.6-1 we see that at a count of 001_2 the "Y_1" output goes low. This low

is inverted and sent to the clock input of L87, the Receive Data Buffer. The received data is then transferred to this buffer and placed on the Receive Data Bus. Also at this time the parity bit is clocked into the parity checker.

Along with the above operation this output of L68 also causes the Receive Delay Timer to clear. The Receive Delay Timer insures that the receive circuits are not enabled until 8 usec after the reception of the last byte. This circuit will be described in detail later.

Going back to L68, we see that the next two output are not connected to anything. When the count in L85 reaches 100_2 the "Y_4" output will be driven low. This signal enables the parity error circuits and allows any errors to be reported to the system. If an error is detected here the remaining part of this sequence will be terminated and the status registers will be loaded with the error code.

On the next count of L85 we will generate a low from "Y_5" of L68. If we have gotten this far then we have a valid byte, but we do not know if it is a valid command. The "Y_5" output of L68 generates RCVBRDY (Receive Byte Ready). This signal transfers the data from the Receive Data Bus to the Command Decoders, or if this was the second or third phase of a command, the transfer would be into the address or data buffers.

L85 will now advance one more count, causing L68 to produce a low at output "Y_6". This generates the sub command &COM CHECK, which enable the command check circuits.

Finally L85 will advance to a count of 111_2, causing L68 to activate output "Y_7". Two function happen here, one, L70-1 will be cleared causing this sequence to end. Two the command NEXT ST is generated. NEXT ST is used to clock the command sequencer. This sequencer keeps track of the number of phases within a command.

This ends the byte acquisition segment. Each byte that is received whether it is a command, address, or data byte will follow this sequence.

7.6.2 Command Decoder

The command decoder determines what command was received and generates any sub commands that are needed for that command. These circuits are located on sheet 6 in the upper right hand corner. L37, L171 and L153 decode the received data on the Receive Data Bus. From here the decoded command is transferred to the "D" inputs of Command Buffer L34. If you recall that the data was transfer to the Receive Data Bus when the Acquisition Sequencer was at a count of 001_2. Then

at a count of 101_2 the signal &RCVBRDY (Receive Byte Ready) was generated by the Acquisition Sequencer. This signal is applied to the clock input of L34 through NOR gate L65. NOR gate L65's pin three input is tied to the "4Q" output of L34. This output is call DL COMM READY (Data Link Command Ready), normally this output is low until a command is received. The &RCVBRDY signal clocks L34 and causes "4Q" output to go high if a valid command was received. This high on "4Q" output disables NOR gate L65, thus insuring that the next byte received is not interpreted as a command.

L34, as I stated before, generates sub commands for the different operation requested by the remote master, Table 7.6-2 show these sub commands and there functions.

Table 7.6-2
Command Decoder Sub Command

Sub Command	Function
DL COMM READY	This sub command indicates that a command has been received. It enables the command action circuits.
DL RD MEM	This is the Data Link Read Memory command.
DL WR MEM	This is the Data Link Write Memory command.
DL BLOCK OPER	This sub command indicates that a block operation is to take place. This block operation could be either a Read or Write type of operation.
DL RD STAT	This is the Data Link Read Status command.
DL RESTART	This is the Data Link Restart command.
DL TRANSMRQ	Data Link Transmit request is generated whenever a Read Command is active. Once again this could be either a block or a byte read operation, or a Status read.

The action that each of these sub commands perform will be discussed as we proceed through the receive and transmit operations.

7.6.3 Command Sequencer

At the end of each byte acquisition segment the signal NEXT ST is generated by L68 (5C3). This signal is used to advance the Command Sequencer. Each of the commands, as I stated before, has a number of corresponding phases to it. The Command Sequencer determines which phase is in progress at any give time. This sequencer consist of two flipflops and a decoder located on sheet 6 of the schematics. L74-1 and L74-2 work as a two bit counter. It is important that you have the latest revision of the schematics for this circuits, Rev 7 or above should do.

We will start the discussion of this circuit by assuming that the receive circuits have just been reset. Both halves of L74 will then be in the set state. This condition is caused by the fact the command decoder was cleared and the "4Q" output (DL COMM READY) was deactivated. This zero forced both halves of L74 to set. Remember that when a command is received this output of L34 will go high allowing both halves of L74 to start counting when they receive a clock pulse. With both flipflops in the set state the inputs of L18, the Command Sequence Decoder, are zero. At this time the "1Y" output of L18 is low. There is nothing connected to this output so no action takes place. But at the end of the Byte Acquisition segment that received the command, the signal NEXT ST is generated. NEXT ST will then pass through AND gate L33 and clock flipflop L74-1, causing it to increment the count by one. At this time decoder L18 will activate the signal DL LD ADDH (Data Link Load Address High) signal. I will get back to this signal in a minute. Each time that the NEXT ST signal is generated by the byte acquisition logic this sequencer will advance by one, until the signal DL DATA is generated. When this happens AND gate L33 (6E4) will be disqualified and stop any further clocking of the sequencer.

Three signals are generated by the command sequencer; &DL LD ADDH, &DL LD ADDL, and &DL DATA. These three signals represent the different phases of the command received. The DL LD ADDH and ADDL signals are used by the receive circuits to load the address into the address buffers. The DL DATA signal is use to indicate that we are in the data phase of the operation.

When a command requires that an address be received from the remote master the signal DL LD ADDH and ADDL will be generated at the appropriate time, DL LD ADDH being the first one generated. This signal, in conjunction with the RCVBRDY signal, clocks the high byte of

the address into its buffer. The DL LD ADDL signal is then generated. This signal along with RCVBRDY clocks the low order byte of the address into the low address counters.

Counters are used for the low byte of the address because during a block operation these counters are advanced for each byte of the transfer. When doing a block operation the low byte of the address must be all zeros, this represent the beginning of the block. When these two counter reach their max count the RCO output will generated a &Last Byte signal, indicating the end of the block operation.

During the DATA phase of a command there are two possible actions that could take place. If the command was a read operation then the transmit circuits must be activated and the data Link turned around. This is followed by the data being transferred from the IWISE memory to the data link transmitters.

If the command was a write operation, then during the DATA phase of the operation each of the following bytes, or byte, acquired by the receive circuits would be transferred to the IWISE memory though the data buffer.

The Command Sequencer can be short counted. This would take place if the command received was READ STATUS. When this command is received the Command Decoder generates DL RS STAT (pin 12 of L34). This signal will be inverted and placed on the clear inputs of both halves of flipflops L70. This places each of these flipflop in the reset state, forcing the Command Sequence Decoder L18 to generate &DLDATA. Remember that during a READ STATUS no address in necessary.

One more command to talk about, this is the RESTART command. When the receive circuits detect a RESTART command they generate DL RESTART form the command decoder. This signal is sent to NAND gate L119 on sheet 5 in the upper left hand corner. There is a problem with the schematics here. L119 is drawn with a bubble on the output pin. This is WRONG, there should not be a bubble here. DL RESTART passes through this gate after being inverted by L103. From here it is place on the preset input of L116-2 the Data Link Terminator flipflop. Setting this flipflop causes the whole world to reset. Well maybe not the whole world, but most of it. By going back to NAND gate L119, the one that was drawn wrong, we can see that there are several signal which will cause flipflop L116-2 to set. Each of these signals will be generated during either the receive or transmission of data. Once this flipflop is set it will stay set until the Receive Delay Timer reset it, see I told you we would get back to this circuit.

Remember that the Receive Delay Timer insures that the receive circuits are not activated until 8 usec. after the reception of the

latest byte. Near the beginning of the byte acquisition cycle this timer was cleared by the CLR DELAY 2 signal from the Byte Acquisition Sequencer. After being cleared, this timer started counting from zero. 128 clock pulses later, L115 will output a high on pin 8. "Oh by the way, the Receive Delay Timer is comprised of counter L115 and flipflop L131-2 located at (5F8)." When pin 8 of L115 goes high it will clock the Data Link Terminator flipflop, L116-2 causing DL TERM to deactivate.

7.6.4 Transmit Circuits

The transmit circuits are activated whenever any one of the Read Commands are received. There are two steps to transferring data through the serial data link to the remote master. The first step is to move the data from the IWISE memory or Status register to the Transmit Buffer. The second step is the converting of this parallel data in to serial data and inserting start, stop, and parity bits.

Whenever any one of the read commands are received, the sub command DL TRANSMRQ (Transmit Request) is generated. This sub command will activate the transmit circuits. For now we will leave this signal alone, I will get back to it later.

If the receive circuits detected a READ STATUS command, the Command Decoder will generate the sub command DL RD STAT.(Data Link Read Status). This signal has several paths to follow. First, from pin 12 of L34 (6F2), the Command Decoder Buffer, we see that DL RD STAT is inverted to generate &RDSTAT. &RDSTAT is then sent to the preclear inputs of the Command Sequencer Counter flipflops L19-1 & -2. This places the Command Sequencer in the &DLDATA phase. Remember that a status read operation does not need an address so we can skip that phase. Now going back to pin 12 of L34, the Command Decoder Buffer, and follow DL RD STAT over to sheet 2. At location C5 we see that DLRDSTAT clocks flipflop L109-1. This flipflop is used by the Z80A to determine if the remote master is online with the SDL. There is one more place the the signal DLRDSTAT goes, that is to location D10. Here it enable the outputs of Status Register L89. The outputs of this register are connected directly to the Transmit Buffer.

If the Command Decoder detected a Byte Read command then it will generate DL RD MEM along with the DL RD READY and DL TRANSMRQ. For this command the address phase of the command sequence will take place, thus placing the address of the desired data in the Address Buffers. The signal DL RD MEM is then sent to sheet 4 location C7. Here we have two places to go. First, DLRDMEM is inverted and placed on the output control pin of the Memory to Transmit Buffer, hence allowing the data to be moved from the data bus to the Transmit Buffer. Second, DLRDMEM

places a logic one on the "D" input of flipflop L153-1. This is the Read Memory Access Request flipflop. The clock input to this flipflop is driven by two signals, &RCVBDRY and &NEWBYTERQ. These two signals insures that flipflop L153-1 is clocked at the proper time. The signals &NEWBYTERQ is generated during the transmit cycle and will be discussed later.

If the Command Decoder detects a Block Read operation it will then generate the signal DL BLOCK OPER. This signal is used to extend the Byte Read cycle. During a Block Read operation the transmit operation is repeated 256 times, or until the address counters on sheet 4 generate the signal &LAST BYTE.

Now let's go to the second part of the transmit operation, the converting of parallel data to serial data and transmitting it out the Serial Data Link.

As I discuss these operation keep in mind the format of the serial data that is transmitted out (START BIT, 8 DATA BITS, PARITY BIT, STOP BIT). For this discussion we will assume that the data has been transferred from the IWISE memory and is sitting on the inputs of the Transmit Buffer.

Whenever the Command Decoder detects a read operation it will generate the signals DL TRANSMRQ. This signal initiates the transmit operation. By itself DL TRANSMRQ performs the following jobs;

1. Place the bit length counter in the count mode.
2. Enables the End Transmission flipflop.
3. Enables the Transmit Data flipflop.
4. Enables the parity flipflop.

The bit length counter, L83 (6C10) determines the length of each bit that is transmitted. It divides the 34.2 Mhz clock by 8 to generate a pulse width of 233.9 nsec. When DL TRANSMRQ goes high pin 9, the Load/Count input is placed in the count mode. L83 will not start counting yet, because pins 7, and 10 are not enabled. But when it does start counting the "Qc" output will generate SHIFT CLOCK.

The End transmission flipflop, L49-1 (6B9), determines when the last byte of data has been transmitted. DL TRANSMRQ enables this flipflop by placing a high level on pin 9 of AND gate L33. When the output of L33 is high the End Transmission flipflop will be allowed to clock and terminate the transmission.

The enabling of the Transmit Data flipflop is through AND L33 pins 11, 12, and 13. DL TRANSMRQ is placed on pin 12 of L33 (6B8), pin 13 receives its input from the End Transmission flipflop. The output, which will be high at this time will deactivate the preclear input of L98-2, the Serial Data Out flipflop.

To enable the Parity flipflop DL TRANSMRQ is feed through AND gate L100 (6C7), to deactivate the preclear input of L99-2, the Parity flipflop. This flipflop counts the number of ones (1's) in the serial data as it is transferred out.

When DL TRANSMRQ is combined with &DLDATA the following function are performed:

1. Wise Link read access request is sent to the priority control circuits.
2. Starts the Turnaround timer.
3. Places the receive circuits on hold.
4. Enable the transmit output circuits.

To activate the Wise Link read access request to the priority control circuits, DL TRANSMRQ is ANDed with &DLDATA by L33 pins 1, 2, and 3 (6E10). The output of L33 clocks flipflop L63-1. The "D" input of L63-1 will be high for all read operation, except for one, Status Read. With a high on the "D" input, the "&Q" output will go low when clocked, this generates &FIRSTBYTERD (First Byte Read). The priority control circuits use this signal to request access to the memory circuits. After the access request has been processed the priority control circuits will then clear flipflop L63-1 by generating &CLEARFIRSTBYTERQ.

When DL TRANSMRQ and &DLDATA were ANDed together by L33 pins 1, 2, and 3, L51 inverted the output and removed the clear condition of the Turnaround Timer L66. This timer will the start counting the 4 Mhz clock pulses. At a count of 64, the "2Qb" output will go high indicating the end of the turn around time.

The same signal that started the Turnaround timer also generated &HOLD RECEIVE and TRANSM. &HOLD RECEIVE places the receive circuits on hold until the transmit operation is complete. The TRANSM signals activates the transmit output buffer.

Now that all the transmit circuits are enabled and the Turnaround timer is started, we are ready to start transmitting. When the Turn Around timer indicates that the turnaround time is over, it will place a logic 1 on the "D" input of flipflop L98-1 (the Transmit Start flipflop). This 1 is also placed on NOR gate L65 (6E10) to terminate the counting operation of the Turnaround Timer.

The Transmit Start flipflop will set on the arrival of the next rising edge of the 34.2 Mhz clock. The "&Q" output of L99-1 will disable the receive input circuit, and the "Q" output will enable the bit length counter L83. L83 will divide the 34.2 Mhz clock by 8, thus generating SHIFT CLOCK. This signal will then be use to transfer the data from the transmit buffer to the transmit output circuits. The SHIFT CLOCK is used to transfer the data from the Parallel-to-Serial Shift Register through the Serial Data Out flipflop, and to increment the DL Transmit Bit Counter L82.

As the data is transferred to the Data Link the transmit circuits must insert a start bit, the data bits, a parity bit, and a stop bit. To insure the proper placing of these bits L82 keeps track of the number of SHIFT CLOCKs that are generated by L83. As L82 counts, L81 decodes this number and generates the appropriate signals to insert the control bits at the proper time. Table 7.6-3 lists the outputs of L81 and the action that they provide.

Table 7.6-3
L81 Action Table

2Y0	2Y1	2Y2	2Y3	1Y0	1Y1	1Y2	1Y3	L82 Output	ACTION
L	H	H	H	H	H	H	H	0000_2	Start bit load.
H	L	H	H	H	H	H	H	0001_2	1st bit of data shifted.
H	H	L	H	H	H	H	H	0010_2	2nd bit of data shifted.
H	H	H	L	H	H	H	H	0011_2	3rd bit of data shifted.
H	H	H	H	H	H	H	H	0100_2	4th bit of data shifted.
H	H	H	H	H	H	H	H	0101_2	5th bit of data shifted.
H	H	H	H	H	H	H	H	0110_2	6th bit of data shifted.
H	H	H	H	H	H	H	H	0111_2	7th bit of data shifted.
H	H	H	H	L	H	H	H	1000_2	8th bit of data shifted.
H	H	H	H	H	L	H	H	1001_2	Insert Parity.
H	H	H	H	H	H	L	H	1010_2	Insert Stop bit.
H	H	H	H	H	H	H	L	1011_2	Resets L82, Resets Parity F/F

In the following paragraphs I will discuss the actions that takes place during the shifting of the data from the Parallel-to-Serial Shift register to the Data Link. You should refer to Table 7.6-3 as this discussion is presented.

Starting out the operation L82 contains a count of 0000_2. This causes L81 to output the sequence indicated in Table 7.6-3. This condition exist even before the first SHIFT CLOCK is generated. The 2Y0 output of L81 is low, and this low is placed on the "D" input of the LOAD SHIFT flipflop L98-1. Which is now in the set state, but when the SHIFT CLOCK transits from a zero to a one, L98-1 will reset. The "Q" output will then go low. This low is then directed to the LOAD/SHIFT input of L106 (the Parallel-to-Serial Shift Register). A low on pin 1 switches L106 from the shift mode to the load mode, loading the data from the transmit bus. When SHIFT CLOCK change state again, this time from a one to a zero, L98-1 (the LOAD/SHIFT F/F) will set again, this action is caused by the feed back of the inverted "Q" output and the zero condition of SHIFT CLOCK. Setting L98-1 causes L106 to switch back to the shift mode of operation. Our data to be transmitted is now in the Parallel-to-Serial shift register and ready for conversion.

The 2Y0 output of L81 is also placed on pin 3 of NAND gate L64 (6G8). L64 inverts this zero and places a one on the "D" input of L98-2, the Serial Data Out F/F. When SHIFT CLOCK caused the LOAD/SHIFT F/F to change stated it also caused L98-2 to transfer this one on its "D" input to its "Q" output. This one represents the start bit of our transmission. Now as the SHIFT CLOCK pulses are generated, by L83, the data bits in the shift register are shifted out one at a time.

After the 8^{th} bit is shifted out, L81 signals that it is time for the parity bit. As in all of Wang equipment ODD parity is used. While the data bits were being shifted out, flipflop L99 kept track of the number of ones (1's) that were shifted. This was done by toggling L99 each time a one was shifted out of L106. Then, at parity insertion time, the state of L99 indicates the state of the parity bit.

I said that L81 signaled that it was time for the parity bit to be inserted. This action is indicated by the 1Y1 output going low. Following this output we see that it is placed on the inputs of two gates, L50 and L32 both at location (6G8). By examining these two gate we see that placing a low on one of their input will cause L50 to enable and L32 to disable. The data bits were passing through L32 but now they are blocked. The state of flipflop L99-2 can now be passes through to the Serial Data Out flipflop.

Effectively the transmission is complete at this time, except that the Stop bit must be generated. The Stop bit is always a zero and the normal condition of the Data Link is also a zero. So to generated the Stop bit all we have to do is insure that the Serial Data Out flipflop is in the reset state. This is accomplished quite simply. As the data was shifted out of the Parallel-to-Serial Shift register, zeros were shifted in. Then after the parity bit is shifter out, gates L50 and L32 are returned to their normal state. The Parallel-to-Serial Shift register contains all zeros at this time so the next bit out is a zero, thus causing a zero to be placed on the Data Link.

The last thing that must be done is to prepare for the next byte of transmission. When the 1Y3 output of L81 is low we will clear the parity flipflop and reset L82 back to a count of 0000_2.

7.6.5 Termination of Transmission.

The termination of the transmit operation is controlled by L49-1 (6B9). This flipflop will be clocked to the reset state when the transmission sequence is complete. The question here is when is the transmit operation complete. There are two way to terminate this operation, one, at the end of a single byte transmission, and two, at the end of a block transmission.

Let's look at the latter termination first. When a block operation is generated the Command Decoder will generate the signal DL BLOCK OPER. This signal will be placed on pin 13 of L64 (6B11). Pin 2 of L64 will have a high from the signal &LAST BYTE. &LAST BYTE is generated from the address counters when they have counted 256 address. So this signal is normally high until that time. The last input to L64 is pin 1, this input is also high at this time because the Turnaround Counter has completed its count. With all three of the input to L64 high we will generate a Low out to pin 10 of L100 (6B10). This low will stay here until the last byte of data is transferred from the memory to the Parellal-to-Serial shift register. Pin 9 of L100 receives its input from the 2Y1 output of L81, the Bit Number Decoder. Referring to Table 7.6-3 we see that the 2Y1 output will go low when L81 has a count of 0001_2. At this time the first bit of the byte is being shifted out of the Parellal-to-Serial shift register, and the signal &NEW BYTE RQ (New Byte Request) is generated. &NEW BYTE RQ informs the memory priority control circuits that another byte is to be transferred from memory. &NEW BYTE RQ also forces flipflop L49-1 into the set condition. As long as L49 stays in the set state the transmission operation will continue. The first time that the 2Y1 output goes low, and if pin 9 of L100 is not low, we will then disable the generation of &NEW BYTE RQ. The transition of 2Y0 from a low to a high will cause flipflop L49-1 to reset. When a byte read operation is in effect the signal &NEW BYTE RQ is never generated and flipflop L49-1 will reset right away. The setting of L49-1 generates the signal &TRANSMITTERM, causing the transmit circuits to reset.

The signal; &TRANSMTERM will not be generated right away. This is because pin 12 of gate L100 (6B8) is held high until after the stop bit is shifted out, and L82 is reset, Refer to Table 7.6-3. Remember that when L82 resets, its outputs are all zero, thus causing L81 to output a low on 2Y0 output. This low qualifies L100 and allows &TRANSMTERM to be generated.

7.7 50BUS Protocol Logic

The 50BUS Protocol logic provides a link between the IWISE memory and the system master. The system master can transfer data either to or from the IWISE memory through these logic circuits. The information transferred could be either data or instruction for the IWISE board. The 50BUS is made up of, as you know, 40 signals. These signals either by themselves or in combination will select and direct the transfer operation. Table 7.7-1 show each of the 50BUS signals and their function.

Table 7.7-1
50BUS Signals

Signal Name	Designation	Logic Type	Signal Source	Wire Count
Master Reset	M R C	Negative	50BUS	1
50BUS Restart (Reset)	50BRESTART	Negative	50BUS	1
50BUS SELECT Dev 7	50BSLT7	Negative	50BUS	1
50BUS Request	50BUSREQ	Negative	50BUS	1
50BUS Acknowledge	50BUASAK	Negative	IWISE	1
50BUS Request/Strobe	50BREQ	Negative	50BUS	1
50BUS Status	50STAT	Negative	50BUS	1
50BUS Read/Write	50BR/W	Neg for Read Pos for Write	50BUS	1
50BUS Address	50BA(0-15)	Positive	50BUS	16
50BUS Data	50BD(0-7)	Positive	Bidir	8

There are five steps in the transfer of data between the system master and the IWISE memory. These five steps are valid for both read and write operation. The only times that these five steps are not all performed is during either a Status read or a 50BUS reset operation. Of course if the master reset signal is generated from the system master these five steps are not performed.

The first step in the transfer of data is for the system master to generate a 50BUS request. This signal informs the IWISE Z80A that the system master wants control of the IWISE bus system. When the Z80A of the IWISE board receives this signal it will complete the current operation and then tri-state its busses. The tri-stating of the IWISE busses generates a bus acknowledge. This acknowledge constitutes the second step of the transfer. The system master receives this acknowledge signal and prepares for the transfer.

The third step is the generating of the address information. This address is the location that the system master desires to either read or write the data. If the system master desires to only read the status of the IWISE board then no address is required. During the status read operation the system master generates &50STAT.

The fourth and fifth step can be combined into one. During this step the system master generates two signals, &50BSTROBE and &50B&R/W. These two signals are combined with &50BSLCT7, which is always generated when the system master communicates with the IWISE board. The &50B&R/W signal is generated slightly ahead of &50BSTROBE. This allows the memory priority circuits opportunity to gain access to the IWISE memory. Remember that the 50BUS has the highest priority when it comes to the IWISE. Shortly after &50B&R/W is generated the system master will generate &50BSTROBE. This signal will clock both the address and the data from the 50BUS into buffers on the IWISE board.

If the system master is performing a read operation it will then wait for the IWISE board to place the data on the 50BUS. And if a write operation is performed then the IWISE board will transfer the data from the 50BUS buffer to the memory when the priority circuits indicate that access is granted.

The circuits used for interfacing the 50BUS with the IWISE board are located on sheet 4 of the schematics. Review these circuits and if needed reread the above explanation to help you understand there operation. These circuits are very basic gating circuits so no further explanation is needed.

7.8 Access Control

The access control logic determines who shall have access to the IWISE memory. As you know there are three devices that require access to the IWISE memory, the Z80A, the 50BUS and the Data Link (sometimes called the Wise Link). Not all of these devices can access the memory at the same time, so priorities must be set. The highest priority is given to the device that can wait the least amount of time, this goes to the 50BUS. The next level of access goes to the Data Link, and finally the last level goes to the Z80A.

To gain access to the IWISE memory each device must have control of the Memory Data Bus ($ZMDATA_0$ - $ZMDATA_7$), and the Memory Address Bus ($MADD_0$ - $MADD_{15}$). Once the device has control of these two Busses it may transfer the data.

Access to the Data and Address Bus is a two tier operation. The first tier separates the lowest level priority from the two higher levels of priority. The Z80A, which has the lowest level of priority, has a direct connection to the two busses. When the Z80A requires access to the memory it generates the appropriate signals to activate the memory circuits through this direct connection. When either the 50BUS or the Data Link requires access to the memory they will request access through the Z80A. This request is in the form of a DMA operation. The requesting device will generate the signal ZBUSREQ (Z Bus Request). Upon receiving this request the Z80A will complete the current operation and then detach itself from the Data and Address bus, thus allowing the other device access.

The second tier of this operation separates the two higher priorities. Two sets of three flipflops and an AND gate perform this operation. The idea behind this circuits is that if both the 50BUS and the Data Link request access to the Data and Address bus at the same time, the 50BUS request will be granted, forcing the Data Link to wait. The only time that the 50BUS would have to wait is if the Data Link had access to the busses when the 50BUS made its request. This wait period would be short because the Data Link must release the Data and Address busses each time a byte of data is transferred. When the Data Link releases the busses the 50BUS can then take control.

Now that we have an idea of what happens, and how the priorities are set, we can take a closer look at the circuits that accomplish this task. Let's first take a look at the generation of the ZBUSREQ (Z Bus Request) signal. Moving over to the lower center of sheet 6 of the schematics we will find three flipflops, L78 (6B6), L146 (6B5) and L49 (6B4). The center flipflop L146, the Z Bus Request flipflop, is the focal point of this circuit. The setting of this flipflop is on a first come first served bases, no priority is needed at this point.

When the 50BUS requires use of the Z Bus it will generate the signal &50ZBUSREQ. This low is placed on the "D" input to flipflop L49, then on the next low to high transition of the 4Mhz clock L49 will reset. The "&Q" output will go high, placing a one on the "D" input of L78, through OR gate L130 (6B6). At the next low to high transition of the 4Mhz clock L78 will set.

When the Data Link requires use of the Z Bus it will generate the signal WLZBUSREQ. When the Data Link starts receiving the high order byte of the address it will force this request signal high. WLZBUSREQ is sent to AND/OR INVERTER gate L91 (6B7). Here the signal is inverted and passed to the clock input of flipflop L78. Nothing happens yet, but when the Data Link advances to the next stage (Receive low byte of address) WLZBUSREQ will return to a low state. This transition will cause flipflop L78 to reset, thus placing a one on the "D" input of L146. Once again when the 4Mhz clock goes from a low to a high state L146 will set. This completes the first tier of the access operation, the Z80A has been instructed to release its hold on the Z Bus system.

The second tier of this operation, as you know, is to determine who requested the Z Bus insuring that if the request came from two location at the same time that the 50BUS has priority. The six flipflops that are involved in this operation are arranged in two group of three.

The first pair of flipflops that I will talk about are the Access Request flipflops, L112-1 and L112-2 (6D/C5). Both the 50BUS and the Data Link have an Access Request flipflop. When either of these two devices requires access to the IWISE memory they will clock the appropriate request flipflop. When clocked each of these two flipflop will go to the reset state, thus producing a zero on the "Q" output. This "Q" output is tied directly to the "D" input of the Priority Control flipflops L111-1 and L129-1 (6D/C4). The zero on the "D" inputs to these two flipflop will pass through to the "Q" outputs on the next transition of the 8Mhz clock. The only thing that would stop this transfer is if the Z Bus was in service already. If this was the case the clock pulse would be held up until the Bus was cleared. Now comes the time when a decision must be made. If both devices requested access at the same time everything that I have just talked about will happen simultaneously, both L111-1 and L129-1 will be in the reset state. The "Q" output of L111-1 is then fed back to the preset input of L129-1 through AND gate L113 (6C5). This will cause L129-1 to return to the set state, thus stopping the forward progress of the Data Link request. Of course if there was no request for the Z Bus by the 50BUS then L129-1 would remain in the reset state.

Now that one of the two Priority Control flipflops are reset we can continue. The "Q" outputs of both Priority Control flipflops are connected to the "D" inputs of Request Activate flipflops L111-2 and L129-2 (6D/C3). On the next low to high transition of the 8Mhz clock either L111-2 or L129-2 will be reset. The two "&Q" outputs of these flipflops are sent to two NAND gates L162 pin 9, and L162 pin 13 (6D/C3). Here our signals will wait until the Z80A has acknowledged the Z Bus Request discussed earlier. When the Z80A does acknowledge the Z Bus request it will set flipflop L146-2 (6B2). The "Q" output of this flipflop will enable both section of NAND gate L162. Whichever section has a logic one on its other input will generates the appropriate Active signal, either 50ACTIVE or WLACTIVE.

To insure that a second request for access is not processed until the current request has been completed, NAND gate L96 (6C3) and flipflop L78-2 have been installed. If either of the Priority Control flipflops or the Request Activate flipflops are in the reset state L96 will output a one. This one will be clocked into flipflop L78-2, causing the "Q" output to disqualify OR gate L23 (6D5). By placing a one on this gate, L23, will block the 8Mhz clock from transferring the data from the Access Request flipflops to the Priority Control flipflops. At the completion of the BUS activity the memory circuits will generate &CLEARPC to NOR gate L92 (6D7). This signal when combined with ZBUSACK will force both sets of Access request flipflops and Priority Control flipflops to the set state. Then on the next transition of the 8Mhz clock, the Request Activate flipflops will be set, thus returning the circuit to its normal condition.

One last condition must be looked at before we leave this topic. Let's consider what would happen if an access request was generated by either device and the other device was already in control. First we know that this requesting device must wait until the current operation is complete. I stated in the above paragraph that at the end of the memory cycle the memory circuits will generate &CLEARPC to clear the priority control circuits. Well if another request had been generated during the current operation we do not want to erase this request. To prevent this from happening we must selectively clear the Priority Control circuits. Looking at the Request Activate flipflops we see that the "Q" outputs are tied back to pins 5 and 2 of OR gates L94 (6D/C6). Now only one of these flipflops will be in the reset state at a time. This means that the other flipflop is in the set state. The "Q" output of this latter flipflop will have a one on it, and thus cause OR gate L94 to be disqualified, blocking the CLEARPC signal from clearing that part of the circuit. The circuit that had control will then be force to give up that control to the other device. If you think about this last paragraph you will see that it will apply also if both devices request access at the same time. True the 50BUS request will be granted first, but the Data Link will not have to regenerate its request.

7.9 IWISE Memory

The memory of the IWISE board contains 64K of dynamic RAM. This memory is no different that the memories of the other boards in the OIS 40/50/60 system except that it can be accessed by three different devices, the 50BUS, the Data Link, and the CPU (Z80A of the IWISE board). Refresh is provided only when the CPU has access.

Referring to sheet 3 of the schematics we see the memory circuits. The RAS/CAS timing is provided by a series of five flipflops located slightly right and below of center. At the beginning of a memory cycle each of these flipflop will be in the reset state. Then as each one is set the appropriate signal will be generated. Starting from the left, the first flipflop (L40-1) is the RAS F/F. When this flipflop is set, the RAS inputs of the memory chips will be activated. On the next clock pulse the second flipflop (L26-1) will set. This flipflop will switch the address multiplexer from the lower byte of the address bus to the upper byte. Then on the next clock pulse after L26-1 was set flipflop L26-2 will set. This flipflop is the CAS F/F. When it is set the CAS inputs of the memory chips will be activated. The next flipflop is this series (L24-1) acts as a delay of one clock cycle. After this delay of one clock cycle, flipflop L24-2 will set. This flipflop is the Write Enable F/F. If the memory operation was of the write type then the Write Enable inputs of the memory chips would be activated. The setting of this last flipflop will cause the entire chain of flipflops to be returned to there starting state.

The memory circuits signal to the three memory master when it has completed a memory cycle. This signaling is controlled through two flipflops L25-1 and L25-2 (3A6/7). At the beginning of the memory cycle L25-1 will be in a reset state and L25-2 will be in a set state. They will remain in this condition until the Switch Mux flipflop has been reset and the CAS flipflop is in the set state. This condition only happens after the Write Enable flipflop has started to reset the five memory cycle flipflops. When the above two flipflops are in the states indicated then NAND gate L32 (3A4) will output a low, causing both L25-1 and L25-2 to reset, this generates &ENDMEM, ENDMEM, and &CLEARPC.

To start a memory cycle one of following six signals must be generated:

1.	&50WRREQ	50BUS Write Request
2.	&ZMWRRQ	Z80A Write Request
3.	&WLWRREQ	Wise Link Write Request
4.	&50RDREQ	50BUS Read Request
5.	&WLRDREQ	Wise Link Read Request
6.	&ZMRDRQ	Z80A Read Request

When one of the above signals are generated, Memory Cycle flipflop L40-1 will be clocked to the reset state, starting the memory cycle. At the completion of the memory cycle the memory circuits will signal the end of the operation as we saw earlier. This signaling will cause the signal that start the operation to be removed. This action causes the Memory Cycle flipflop to return to the set state.

If during a Data Link write operation an error was detected, the write enable inputs to the memory chips would be disabled. This error handling operation is performed by NAND gate L3 (3C10) and L54 (3F9).

7.9.1 Zero Page Access

The last section of the memory circuits that we must talk about is the Zero Page Select logic. The Zero page of memory for a Z80A is a very important page of memory. When the Z80A is placed in a reset state it will look at the first location in the zero page to find instruction. Also page zero is used for the storing of interrupt routines. Now since there is more than one Z80A that uses the IWISE memory and we do not want one of these Z80A's to over write or read instruction on the zero page of the other, then we must supply each with its own zero page. Of course each Z80A thinks that it is using the zero page locations, but its not.

For those of you that are not familiar with paging let's take a quick look at the operation. If you know how paging works then you may skip this paragraph. A page of memory is described as being a 256 byte block of memory. For a Z80A based system there are 64 pages, or blocks, of memory. The upper eight bits of the Address Bus, $BADD_8$ - $BADD_{15}$, are used to select the different pages. The lower eight bits of the Address Bus, $BADD_0$ - $BADD_7$, select locations within the given page.

When the Data Link or the 50BUS tries to use the zero page of the IWISE memory, it is directed to one of the first 256 location within the IWISE memory. In other words, either if these two devices are truly using zero page location. The Z80A of the IWISE board has the option of choosing either the true zero page or the alternate zero page. This alternate zero page is in reality page one. The only problem that could arise if if the Z80A of the IWISE board was directed to the alternate page zero and one of the other devices had written code in this page. This is possible because whenever any other page is written to or read from the address is not modified. This shouldn't bother us to much, as board repair technicians, but if the software that you are using has a bug in you could be getting erroneous problems.

The circuits that allow you to select either the true or alternate zero page are located on sheet 2 of the schematics. L74-1 (2B4) will do the selecting. Gates L43, L42, and L72, all located around 2C2, will perform the conversion. Assuming that flipflop L74-2 is set, indicating redirection, and all the inputs of all three section of L43 are zero, the $MADD_8$ bit of the Address Bus will be a one. Hence redirecting the zero page address from the Z80A to page one. If flipflop L74-2 is placed in the reset state then this redirection circuit is disabled. The selection of having redirection or not is performed by the OUT 09 instruction. If the OUT 09 instruction is performed with data bit zero a logic one then redirection is activated. The Z80A can read the setting of this flipflop by performing an IN 09 instruction.

7.10 Hardware Control Ports

The Hardware Control Ports of the IWISE board allows portions of the IWISE board to be switched from a normal mode of operation to a diagnostic mode. Control of these Ports is through the IN and OUT instruction of the Z80A. A complete listing of the IN and Out instruction and a brief description of their operation is provided in Table 7.5-1 (I/O Port Allocations). In the following paragraphs I will discuss in a more detail manner these port and the effect they have on the operation of the IWISE board. Each port will be presented with the I/O location and the appropriate Data bits that are needed to control them.

I/O Port XX01 (IWISE Diagnostic Mode Flipflop)

An OUT instruction to this port will cause flipflop L127-1 (2G6) to select either the diagnostic mode or normal mode of operation. In the Diagnostic mode, represented by having the flipflop in the set state, memory parity errors will generate a nonmaskable interrupt. This type of interrupt will force the Z80A to fetch an the next instruction from memory location 0066(H).

If this flipflop, L127-1, is in the reset state then the IWISE board is in a normal mode of operation. In this mode, memory parity errors are recognized as fatal errors, the IPL bit is set, and the Z80A is force to perform NOP instructions until the master re-IPLs the board.

I/O Port XX02 (Data Link Test Mode Flipflop)

This port in conjunction with two other ports, XX04, and XX06 is used to test the receive section of the Serial Data Link. When the Z80A performs an OUT 02 instruction with DB_0 equal to a one, flipflop L127-2 (2E6) will be placed in the set state. With this flipflop in the set state, the 34.2 Mhz clock used for the Serial Data Link circuits will be deselected, and flipflop L160-2 will be use as a clock. L160-2 can be toggled through the OUT XX04 instruction. This allows the diagnostic the ability to single step through a receive operation.

I/O Port XX03 (Parity Switch Flipflop)

The parity switch flipflop is used to select either even or odd parity generation. Normally odd parity is generated when writing data to the memory, but during diagnostics even parity can be selected in order to check selected memory locations. To switch the Parity Switch flipflop into the even parity mode, the Z80A must perform an OUT XX03 instruction with BD_0 a one.

I/O Port XX04 (Data Link Test Mode Clock Flipflop)

This flipflop, L160-2, is intended to provide an artificial clock pulse for the IWISE Data Link in the Data Link Test Mode (See I/O Port XX02), in conjunction with the Test Transmission Serial Data Flipflop (See I/O Port XX06). Together these three flipflops provide a means of testing the Serial Data Link circuits in a single step mode of operation. This flipflop, L160-2, can be toggled by the Z80A performing several OUT 02 instruction with the BD_0 bit in a different states for each OUT instruction.

I/O Port XX05 (Detect Latch)

The detect Latch is used by the software to determine if any of the masters are on line with the IWISE memory. When one of the two masters go on line with the IWISE memory, one of two flipflops will be placed in a reset state. L163-1 (2D5) is use to detect if the 50BUS is on line and L109-1 (2D4) is use by the Data LInk. The software reads the condition of these two flipflops by performing and IN 05 instruction. Reading these two flipflops does not change there condition. When reading these two flipflops the BD_0 bit indicates status for L163-1, and the BD_1 bit indicates status for L109-1. Once these flipflops have been placed in the reset state by the appropriate controlling signal they must be returned to the set state by the Z80A. This action can be accomplished by performing an OUT 5 instruction, the content of the data bus does not matter.

I/O Port XX06 (Test Transmission Serial Data Flipflop)

This flipflop, L144-2 (2E4) is used in the Data Link Test Mode to replace the real Data Link input of the receive circuits. By changing the state of this flipflop an artificial data stream can be injected into the receive circuits for testing. When the Z80A performs an OUT 06 instruction the content of the BD_0 bit of the data bus will be transferred to the Data Link receive circuits. This is used conjunction with both the Data Link Test Mode Clock flipflop (I/O port XX05) and the Data Link Test Mode flipflop (I/O port XX04).

I/O Port XX07 (Data Link Disable Flipflop)

The Data Link Disable flipflop gives the programmer the ability to enable/disable the data link channel which connects IWISE to an external master.

The data input to this flipflop is also connected to the BD_1 bit of the Z80A data bus. To enable the IWISE serial Data Link, the Z80A must perform and OUT 07 instruction with data bit BD_1 equal to a zero. The Data Link can also be enabled by performing an OUT 08 instruction, this way of enabling the Data Link does not require any preset data on the data bus. To disable the Data Link, the Z80A must perform an OUT 07 instruction with the BD_1 bit a one.

Note that data read from this port always returns zero and does not reflect the status of the Data Link Disable flipflop.

I/O Port XX08 (IWISE Status Register)

This port provides the diagnostic program with the ability to read the IWISE hardware status register. Table 7.4-1 shows each of the bits in the status register and designation.

A write operation to this port, data not needed, will cause the Data Link Disable flipflop to set, thus enabling the Data Link, and will cause the 50BUS Restart signal (flipflop L63-2 (4D7)) to be disabled.

I/O Port XX09 (Page Select Flipflop)

This flipflop is intended to control the IWISE memory organization when it communicates with the Z80A. There are two possible memory organizations: mapped and unmapped. To select the unmapped memory organization, the I/O address XX09 must be loaded with Data:00. To change to a mapped organization, this address must be loaded with DATA:01. For more information on the operation of this flipflop refer to Section 7.9.1 (Zero Page Access).

I/O Port XX0A (Code Request Switch)

This input port give the programmer the ability to read the current position of switch 1 on the IWISE board. This switch is use to inform the programmer whether the IWISE board should be loaded with diagnostic program or IWISE software. The operator must manually change the position of this switch. When read, the BD_6 of the data bus is the only bit that will reflect the condition of this switch.

I/O Port XX0B (Z80RESTART and CLRMEMERRBIT Port)

This port is used for diagnostic purposes only. A read of this address will force the IWISE to RESTART. Writing to this port will cause the memory parity error bit in the IWISE status register to be cleared. No data is either read from or written to this address.

I/O Port XX0C, XX0D, XX0E (Indicators)

These three ports can be used for diagnostic purposes to display up to three bytes of data. This data is loaded into the indicators by writing to I/O location XX0C through XXE. Location XX0C is intended to be the most significant byte, while location XX0E represents the least significant byte of the display.

The output from these three port are connected to J1, a 16 pin dip socket located in about the center of the board. To be able to utilize this connector, one would have to build an indicator board with three LED displays and some gating circuits. The OUT XX0C through XX0E signals could be used to enable the different displays. FIGURE 7.10-1 shows the J1 connector and its pin assignments.

J1

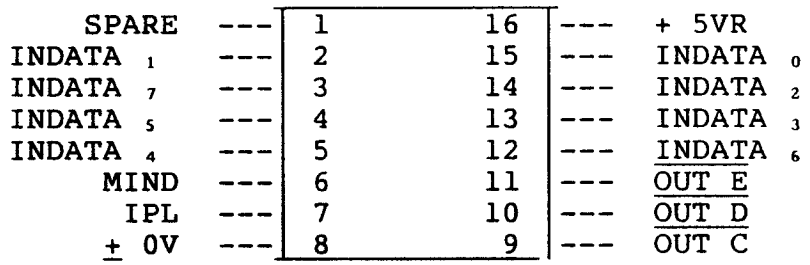


FIGURE 7.10-1
J1 Pin Assignments

I/O Port XX0F (Data Link Transmission Serial Data)

This port is intended to be use in the data link test mode only. It provides the programmer with the ability to read the current level of the serial data link signal generated by the transmission circuits. This level can be read by performing an IN 0F instruction, the BD_0 bit of the data bus will indicate the level of the Serial Data Link. All other bits of the data bus will be zero.

This concludes the theory of operation section for the IWISE board of the OIS 40/50/60 system.

SECTION 7 QUIZ

1) What are the seven section of the IWISE board?

2) What function does the Access control logic provide?

3) When accessing the IWISE memory what device has the lowest priority?

4) What functions are performed during the second phase of the IWISE protocol?

5) When data is transmitted between systems what portion of the data byte is transmitted first?

6) What logic level is the start bit?

7) During the command phase of the IWISE operation, in what order is the address of the data received?

8) Between the address phase and the data phase of the IWISE operation what happens?

9) At the end of a read operation a turnaround timer is started, how long does this turnaround time last?

10) What type of information can be determined from the IWISE status register?

11) During a block operation, what must the lower byte of the address be set to?

12) What are the six signals that will start a memory cycle?

13) What function does the zero page access circuits provide?

14) Through what I/O port can the Z80A change the state of the page flipflop?

15) When the IWISE board is placed in a diagnostic mode what type of parity is used?

SECTION 8
DIAGNOSTICS

SECTION 8
DIAGNOSTICS

8.1 INTRODUCTION

In this section I will provide you with a brief description of the diagnostics that are available for the five PCA's of the OIS 40/50/60 system. Some of the diagnostics are designed to provide help in troubleshooting system problems while others are designed for individual boards. The use of these diagnostics will aid you in isolating problems that you may encounter. Some of these diagnostics are very thorough, while others are not. By applying the knowledge you have gained by reading this book and your own understanding of electronic circuits you should be able to solve most problems.

The documentation that goes with the diagnostics will provide you with the information needed to run and determine what each test is trying to perform. The error codes and the description of the test will help you in determining what test and what part of the circuit is at fault.

The best way to troubleshoot any of the five boards in the OIS 40/50/60 system is to have a thorough understanding of the operation behind it. If you know how a circuit is suppose to work, then you will be able to tell if that board is acting improperly. Good Luck in you troubleshooting efforts.

8.2 System Diagnostics

System diagnostics are diagnostics that will help you determine what part of the OIS 40/50/60 system has a failure. These diagnostics usually check the interaction of two different boards or the interaction between a board and a I/O device. This does not mean that these diagnostics can not be used to isolate a problem on a given board. If you run one of these diagnostics and it determines that a problem has occurred within a board, you can run that part of the diagnostic that failed over and over again while you are troubleshooting. Most of the system diagnostics provide you with scope loops and routine loop for this purpose.

There are four system diagnostics available for the OIS 40/50/60 system. Three of these provide the user with scope, and routine loops, as described above. The other, provides the user the ability to select which device they want to test. Table 8.2-1 shows the four diagnostics and whether they provide looping operations.

Diagnostics Name	Looping
MASTER DTOS	YES
ONLINE DTOS	YES
B.I.T.	YES
SYSEX	NO

Table 8.2-1

8.2.1 MASTER DTOS

Master DTOS or sometimes called Master Monitor, provides the user with the means of testing most of the circuits contained on the RMU and RCU boards, along with tests for the IWISE board. There is no provision in this diagnostic for testing the Internal Printer Controller (IPC) or the Internal Workstation Controller (IWS) controllers, these two controllers can be tested with either the ONLINE DTOS or the two non-system diagnostics. Each of the test on the Master DTOS floppy have several sub-test within, these sub-tests can be used to isolate a problem within a board.

The five tests contained on the Master DTOS floppy are:

1. Z80A Instruction Set
2. Winchester Disk Test
3. SLDK Diagnostic
4. RMU/RCU Interface Test
5. IWISE Diagnostic

The Master DTOS diagnostics provides the user with a log of the errors encountered. This error log is placed on the floppy, so do not write protect this floppy. You are asked if you wish to clear the log each time that you IPL the disk, if there are enteries in the log. It is advisable to have a spare copy of this disk on hand, just in case the system you are testing has a problem in writing data to the floppy.

The OIS 40/50/60 Master DTOS information can be ordered as a package containing both the diagnostic diskette and the associated documentation or individually in which case either the diskette or the documentation is specified.

The package is part number 195-2673-g
The diskette part number is 732-0035
The documentation part numbers are;

760-1042A	Z80A Instruction Test
760-1163	Winchester Dist Test
760-1158	RMU/RCU Interface Test
760-1157	Slave Data Link Test

8.2.1.1 Z80A Instruction Set

This test exercises all of the instruction that the Z80A of the RMU board is capable of executing. By running this test the user can determine if a problem exist in the Z80A or the data and address bus. If the operator is familiar with the instruction set of the Z80A and how they are performed then he/she should be able to determine if the problem is a fault of the Z80A or the external circuits that provide the Z80A with the information it requires. An example of this would be if the operator encountered a error code of 05 while running this test. By looking at the list of error code, found in the documentation of the Master DTOS, we see that this code indicated that either the Rotate Right or Rotate Left instruction did not work. If this is the only instruction that failed then the operator could assume that the problem resides in the Z80A. But if other instructions also failed then there might be a problem in either the data or address bus network. As you can see the operator must take into account not only the test that failed but the tests that did not fail when determining what part to replace. In doing this type of troubleshooting the technician saves time and money, by not replacing components needlessly.

It is possible with this test to loop on a single instruction. The only problem that you may encounter in doing so, is that these tests are very short and sometimes it is difficult to catch the one you want. By looping on one of these tests you may be able to isolate the problem even further.

8.2.1.2 Winchester Disk Test

The Winchester Disk Test checks read, write, and seek operations of the Winchester drive. All write operation are done on the diagnostic cylinder of the Winchester, so any data on the Winchester is not destroyed. **It is advisable though, that any data on the Winchester be removed prior to running this test, if it is suspected that there is a problem with the seek or write circuits of the drive.**

When running this test, keep in mind that there is a lot of interaction between the RMU and the RCU boards for each of the commands. An example of this interaction is if a seek to track zero test is being performed. The Z80A, of the RMU, must provide the 8X305, of the RCU, with the proper code for seeking to track zero. The 8X305 then must generate the correct sequence of commands to the Winchester controller circuits. Once again the experinced operator will not only look at the test that failed, but will look at tests that did not fail to determine if simular function worked correctly.

8.2.1.3 SLDK Diagnostic

The SLDK (Slave Data Link Exerciser) tests the Serial Data Link circuit located on the RMU board. Even though these circuits are on the RMU board, they are controlled from the RCU. Once again caution must be used before replacing suspected failed components, because of the interaction of the RMU and the RCU.

When performing this test the master unit must be connected to an external slave with at least 16K of memory. It is also important that the slave unit is in proper working order and that the cables are secure. If the slave unit or the cables are not in proper repair, then erroneous errors may appear. Also the slave unit should be a Z80A based unit, and not an 8080 based one. 8080 based units may not execute instruction quick enough for the tolerances of this diagnostic.

The error codes that are given for this diagnostic can not be used by themselves to determine what component has failed. The operator must also take into consideration the test that is being performed at the time of the error. There are sixteen different tests within this diagnostic, each one checks a different part of the Serial Data Link circuits. So if you receive an error code of 02, which indicates a Bad Status returned on Slave Restart, you will have to also check which test is being performed before you can start looking for the failed component.

8.2.1.4 RMU/RCU Interface Test

This diagnostic tests the interface circuits between the RMU and the RCU. If a failure occurs while running this diagnostic, the RCU board is the most likely failed unit. There are three test in this diagnostic, the first test checks the SRF (Status Register File) and the PRF (Parameter Register File). The second test checks the 8X305's scratch pad RAM, and the third test checks the 4K buffer.

8.2.1.5 IWISE Diagnostic

There are two versions to this diagnostic, short and long. The short version runs nine times faster than the long one, this is because addressing and counter increments are performed in increments of nine instead of one. When selecting the shorter version of this test the test will not check all of the IWISE memory location. The operator can select which version he/she wishes to run. The Data Link Cables to the IWISE must be removed when running this diagnostic. Failure to remove these cables could result in random error occurring during the serial data test.

Using this diagnostic the technician should be able to check most of the Serial Data Link transmit and receive circuits on the IWISE board.

8.2.2 ONLINE DTOS

ONLINE DTOS Device 4 allow the operator to select, load, control, and monitor one or more diagnostic programs. This diagnostic is intended to run on an operating system, though you may run it on your test rig as long as it meets the minimum requirements specified by the diagnostic.

There are 16 different diagnostic routines that the operator can select from. Some of these diagnostic are repeats of the ones located in the Master DTOS package, others can only be used on the particular external workstation or I/O device. There are some additional diagnostics that were not found in the Master DTOS package so the technician should be familiar with these.

The complete OIS ONLINE DTOS Device 4 package, including software diskette and documentation can be ordered with part number 195-2171-0F. This package contains a detailed explanation of the operating procedures and a complete explanation of each of the tests, including test procedures and error codes.

8.2.3 B.I.T (Built in Test)

The power-up diagnostic, call the B.I.T., resides in PROM on the RMU PCA and checks the system integrity each time the system is powered up or reset.

This diagnostic takes approximately 20 seconds to complete. The primary use of the B.I.T. diagnostic at the field level is for isolation of board failures within the OIS 40/50/60 Master Unit. Fault isolation is accomplished through the use of error codes displayed on the Master Units front panel LED's. These error codes are termed, "non-recoverable", indicating faulty operating conditions, or "fatal", indicating faulty circuit boards. If the B.I.T. diagnostic passes successfully, without encountering any errors, the letters "UP" will be displayed on the LED's for approximately one second. Control is then transferred to the bootstrap code. The Internal WISE Controller (IWISE) is not tested by the B.I.T.

The board repair technician can also use this diagnostic to aid him or her in troubleshooting the RMU or RCU circuits board. The technician can use this diagnostic by selecting the Diagnostic Option Switch, located on the RMU. This switch (SW3) is a bank of 8 switches that allows the operator to select certain operating modes for the B.I.T..

To select the diagnostic mode of operation the operator must place switch 8 of SW3 in the on position. When this is done the lower four switches (1-4) are enabled, this does not mean that if switch 8 is in the off position that the B.I.T. will not operate. There are four options that the operator may select, either separately or in combination.

Switch 4 (Continue on Error)

If switch 4 is placed in the ON position then the diagnostic will continue even if an error is encountered. The B.I.T. will not continue, however, if a non-recoverable error is encountered.

Switch 3 (Loop on Error)

If this switch is placed in the on position, then the diagnostic will loop on the routine that caused the error until either the error is no longer detected or switch 3 is turned off.

Switch 2 (Stop on Error)

With this switch on the B.I.T. will stop when it encounters any type of error.

Switch 1 (Loop on B.I.T.)

In this mode the B.I.T. will execute repeatedly until switch 1 is turned off.

A complete listing of the error codes can be found in appendix B of the OIS 40/50/60 Product Maintenance Manual, part number 741-1267.

8.2.4 SYSEX (System Exerciser)

The system exerciser is a diagnostic that will check system integrity. It is normally used prior to installing a new system or after a system has had major changes to it. What the system exerciser does is to check different communication paths between either of the two disk drives and another device. It is a good diagnostic for detecting intermittent errors. There are four SYSEX diagnostics, two for the OIS 50, and two for the OIS 60. Of these four diagnostics there are two versions, the first version will work on a system with an internal workstation connected to channel 1, while the second works on systems with an external workstation connected to channel 6. The first version of each of these diagnostics is labeled SYSEX nn, where nn indicates the OIS model number; i.e. SYSEX 60 is for the OIS 60 system using an internal workstation on channel 1. The second version of these diagnostics is labeled SYSEX nnX, where nn indicates the OIS model number and X indicates, use an external workstation on channel 6. So in review we have the following SYSEX Diagnostics:

2.	SYSEX 50	Internal Workstation
3.	SYSEX 50X	External Workstation
4.	SYSEX 60	Internal Workstation
5.	SYSEX 60X	External Workstation

There are several hardware and software requirements that the operator must adhere to. These requirements insure that important data on the Winchester or Floppy does not get destroyed. The hardware requirements are dictated by the SYSEX Diagnostic that you are using, see the above paragraph.

The software requirements are the same for all of these diagnostics. No matter what system you are working on the Winchester must be initialized with any Volume name and a volume password of BSL. To use the floppy drive as part of the test you must have a spare floppy that is initialized with a volume name of BSL and a volume password of BSL.

WARNING

ALL DATA ON BOTH THE WINCHESTER AND THE FLOPPY WILL BE DESTROYED

As I stated earlier these diagnostics are very good at detecting intermittent problems. If you have a board that has an intermittent problem you may want to run this diagnostic over night. Then when you come to work the next day you can check the error log and see if the board failed.

With this diagnostic you can have up to six types of transfer operations to one board, thus giving you maximun activity in the shortest period of time. Remember though that the Winchester and the spare floppy that you use will have all of its files and data destroyed.

The SYSEX diagnostics can be ordered with the following part numbers:

	SYSEX NAME	PACKAGE #	DOCUMENTATION #	SOFTWARE #
2.	SYSEX 50	195-2690-9	760-1165	732-0034
3.	SYSEX 50X	195-4772-9	760-1408	732-0058
4.	SYSEX 60	195-2925-9	760-1254	732-0048
5.	SYSEX 60X	195-4755-9	760-1403	732-8042

The above part numbers may change from time to time due to new releases, so please check with software distribution before ordering.

8.3 Individual Board Diagnostics

There are currently two diagnostics that are designed for individual boards, one for the 210-8274 board and one for the 210-8280 board. Both of these diagnostics require the use of the Millennium Microsystem Analyzer (uSA) for running the tests. Each has a Module Repair Guide (MRG) explaining in detail, operating procedures and required equipment. MRG # 110 is for the 210-8280 Internal Printer Controller (IPC) board, and MRG # 111 is for the 210-8274 Internal Workstation Controller (IWS) board. Part numbers for these two MRG's are below;

1.	MRG #110	Internal Printer Controller (IPC)	729-1591
2.	MRG #111	Internal Workstation Controller (IWS)	729-1595

Each of these MRG's provide the user with two ways of testing the board. The first way is through a series of diagnostic exercises an error detection. The second way is through signature analysis. A complete listing of the test descriptions and signatures are provided with both MRG's.

SECTION 9
APPENDICIES

APPENDIX

F

APPENDIX F:
PART 1
8X305 COMMAND SEQUENCES FOR SDL OPERATIONS

To communicate with a Serial Data Link port on the RMU board, the RCU issues seven commands (TRANSMIT, RECEIVE, CHANNEL SELECT, TRANSMIT DATA, RECEIVE DATA, STATUS READ, and DIAGNOSTIC) in various sequences. The RMU decodes these commands from the 50BUS address lines when device 1 is selected (ie, /50BSLCT1 is active) and 50BREQ is strobed. The 8X305 generates the codes for the above commands in predefined sequences to perform several different operations. The Serial Data Link Restart, Read (1- and 256-byte), Write (1- and 256-byte), and Status Read operational sequences are outlined below.

Restart Sequence

NO.	FUNCTION	R/W	PORT	DATA
1	Enable 8X305 Data Bus to 50BUS	W	44H	01H
2	Select Device 1	W	40H	02H
3	Select Channel Command	W	41H	02H
4	Select Channel Number	W	43H	00-03H
5	Activate Request	W	47H	Irrelevant
6	Deactivate Request	W	47H	Irrelevant
7	Select Transmit Data Command	W	41H	03H
8	Load Restart Word	W	43H	A8H
9	Activate Request	W	47H	Irrelevant
10	Deactivate Request	W	47H	Irrelevant
11	Select Transmit Command	W	41H	00H

PART 1
8X305 COMMAND SEQUENCES FOR SDL OPERATIONS

NO.	FUNCTION	R/W	PORT	DATA
12	Load Transmit Enable Bit	W	43H	01H
13	Activate Request	W	47H	Irrelevant
14	Deactivate Request	W	47H	Irrelevant
15	Disable 8X305 Data from 50BUS	W	44H	00H
16	Select Status Command	W	41H	05H
17	Activate Request	W	47H	Irrelevant
18	Read Status. Continue reading until 50BD2 = 1. (Look for a Byte Request)	R	40H	
19	Deactivate Request	W	47H	Irrelevant
20	Select Transmit Command	W	41H	00H
21	Enable 8X305 Data Bus to 50BUS	W	44H	01H
22	Load Transmit Disable Bit	W	43H	00H
23	Delay 1 us	N/A	N/A	N/A
24	Activate Request	W	47H	Irrelevant
25	Deactivate Request	W	47H	Irrelevant

End of Restart Write Sequence

PART 1
8X305 COMMAND SEQUENCES FOR SDL OPERATIONS

1-Byte Read Sequence

NO.	FUNCTION	R/W	PORT	DATA
1	Enable 8X305 Data Bus to 50BUS	W	44H	01H
2	Select Device 1	W	40H	02H
3	Select Channel Command	W	41H	02H
4	Select Channel Number	W	43H	00-03H
5	Activate Request	W	47H	Irrelevant
6	Deactivate Request	W	47H	Irrelevant
7	Select Transmit Data Command	W	41H	03H
8	Load 1-Byte Read Instruction	W	43H	A2H
9	Activate Request	W	47H	Irrelevant
10	Deactivate Request	W	47H	Irrelevant
11	Select Transmit Command	W	41H	00H
12	Load Transmit Enable Bit	W	43H	01H
13	Activate Request	W	47H	Irrelevant
14	Deactivate Request	W	47H	Irrelevant
15	Select Transmit Data Command	W	41H	03H
16	Load High-Order Address of Slave Memory	W	43H	HO Address
17	Activate Request	W	47H	Irrelevant
18	Deactivate Request	W	47H	Irrelevant
19	Enable Halt Circuit	W	56H	Irrelevant

PART 1
8X305 COMMAND SEQUENCES FOR SDL OPERATIONS

1-Byte Read Sequence (cont.)

NO.	FUNCTION	R/W	PORT	DATA
20	Select Transmit Data Command	W	41H	03H
21	Load Low-Order Address of Slave Memory	W	43H	LO Address
22	Activate Request	W	47H	Irrelevant
23	Disable Halt Circuit	W	56H	Irrelevant
24	Deactivate Request	W	47H	Irrelevant
25	Disable 8X305 Data Bus from 50BUS	W	44H	00H
26	Select Status Command	W	41H	05H
27	Activate Request	W	47H	Irrelevant
28	Read Status. Continue Reading Until 50BD2 = 1. (Look for a Byte Request)	R	40H	
29	Deactivate Request	W	47H	Irrelevant
30	Select Transmit Command	W	41H	00H
31	Load Transmit Disable Bit	W	43H	00H
32	Enable 8X305 Data Bus to 50BUS	W	44H	01H
33	Activate Request	W	47H	Irrelevant
34	Deactivate Request	W	47H	Irrelevant
35	Select Receive Command	W	41H	01H
36	Load Receive Enable Bit	W	43H	01H
37	Delay 60 us	N/A	N/A	N/A

PART 1
8X305 COMMAND SEQUENCES FOR SDL OPERATIONS

1-Byte Read Sequence (cont.)

NO.	FUNCTION	R/W	PORT	DATA
38	Activate Request	W	47H	Irrelevant
39	Deactivate Request	W	47H	Irrelevant
40	Disable 8X305 Data Bus from 50BUS	W	44H	00H
41	Select Status Command	W	41H	05H
42	Activate Request	W	47H	Irrelevant
43	Read Status. Check 50BD0. If 50BD0 = 1 then a byte of data has been received. If 50BD3 = 1, then a No Data Timeout has occurred and the operation will be terminated.		R	40H
44	Deactivate Request	W	47H	Irrelevant
45	Select Receive Data Command	W	41H	04H
46	Activate Request	W	47H	Irrelevant
47	Read Received Data	R	40H	
48	Deactivate Request	W	47H	Irrelevant
49	Select Status Command	W	41H	05H
50	Activate Request	W	47H	Irrelevant
51	Read Status. Check 50BD1. If 50BD1 = 1, then a parity error has been detected.	R	40H	
52	Deactivate Request	W	47H	Irrelevant
53	Select Receive Command	W	41H	01H

PART 1
8X305 COMMAND SEQUENCES FOR SDL OPERATIONS

1-Byte Read Sequence (cont.)

NO.	FUNCTION	R/W	PORT	DATA
54	Load Receiver Disable Bit	W	43H	00H
55	Enable 8X305 Data Bus to 50BUS	W	44H	01H
56	Activate Request	W	47H	Irrelevant
57	Deactivate Request	W	47H	Irrelevant

End of 1-Byte Read Sequence

256-Byte Read Sequence

NO.	FUNCTION	R/W	PORT	DATA
1	Load Data Buffer Low-Order Address (eight bits)	W	30H	LO Address
2	Load Data Buffer High-Order Address (four bits, low-order nibble of byte)	W	31H	HO Address
3	Enable 50BUS Data to Data Buffer	W	57H	40H
4	Enable 8X305 Data Bus to 50BUS	W	44H	01H
5	Select Device 1	W	40H	02H
6	Select Channel Command	W	41H	02H
7	Select Channel Number	W	43H	00-03H

PART 1
8X305 COMMAND SEQUENCES FOR SDL OPERATIONS

1-Byte Read Sequence (cont.)

NO.	FUNCTION	R/W	PORT	DATA
8	Activate Request	W	47H	Irrelevant
9	Deactivate Request	W	47H	Irrelevant
10	Select Transmit Data Command	W	41H	03H
11	Load 256-Byte Read Instruction	W	43H	A4H
12	Activate Request	W	47H	Irrelevant
13	Deactivate Request	W	47H	Irrelevant
14	Select Transmit Command	W	41H	00H
15	Load Transmit Enable Bit	W	43H	01H
16	Activate Request	W	47H	Irrelevant
17	Deactivate Request	W	47H	Irrelevant
18	Select Transmit Data Command	W	41H	03H
19	Load High-Order Address of Slave Memory	W	43H	HO Address
20	Activate Request	W	47H	Irrelevant
21	Deactivate Request	W	47H	Irrelevant
22	Enable Halt Circuit	W	56H	Irrelevant
23	Select Transmit Data Command	W	41H	03H
24	Load Low-Order Address of Slave Memory	W	43H	LO Address
25	Activate Request	W	47H	Irrelevant
26	Deactivate Request	W	47H	Irrelevant

PART 1
8X305 COMMAND SEQUENCES FOR SDL OPERATIONS

1-Byte Read Sequence (cont.)

NO.	FUNCTION	R/W	PORT	DATA
27	Disable 8X305 Data Bus from 50BUS	W	44H	00H
28	Disable Halt Circuit	W	56H	Irrelevant
29	Select Status Command	W	41H	05H
30	Activate Request	W	47H	Irrelevant
31	Read Status. Continue reading until 50BD2 = 1. (Look for a Byte Request)	R	40H	
32	Deactivate Request	W	47H	Irrelevant
33	Select Transmit Command	W	41H	00H
34	Load Transmit Disable Bit	W	43H	00H
35	Enable 8X305 Data Bus to 50BUS	W	44H	01H
36	Activate Request	W	47H	Irrelevant
37	Deactivate Request	W	47H	Irrelevant
38	Select Receive Command	W	41H	01H
39	Load Receive Enable Bit	W	43H	01H
40	Delay 60 us	N/A	N/A	N/A
41	Activate Request	W	47H	Irrelevant
42	Deactivate Request	W	47H	Irrelevant
43	Disable 8X305 Data Bus from 50BUS	W	44H	00H
44	Enable Buffer to the 50BUS	W	57H	04H

PART 1
8X305 COMMAND SEQUENCES FOR SDL OPERATIONS

1-Byte Read Sequence (cont.)

NO.	FUNCTION	R/W	PORT	DATA
45	Enable Halt Circuit	W	56H	Irrelevant
46	Select Receive Data Command	W	41H	04H
47	Write Data to 50BUS	W	43H	Irrelevant
48	Activate Request	W	47H	Irrelevant
49	Write Data to Buffer (Turn On Strobe)	W	33H	Irrelevant
50	Turn Off Write Strobe	W	33H	Irrelevant
51	Deactivate Request	W	47H	Irrelevant
52	Test for 256 bytes received. If not, jump to 47. If done, fall through to 53.			
53	Disable Halt Circuit	W	56H	Irrelevant
54	Disable Buffer from 50BUS	W	57H	00H
55	Select Status Command	W	41H	05H
56	Activate Request	W	47H	Irrelevant
57	Read Status. Check 50BD1. If 50BD1 = 1, a parity error has been detected. If 50BD3 = 1, a No Data Timeout has occurred and the operation will be terminated.	R	40H	
58	Deactivate Request	W	47H	Irrelevant
59	Select Receive Command	W	41H	01H
60	Load Receiver Disable Bit	W	43H	00H

PART 1
8X305 COMMAND SEQUENCES FOR SDL OPERATIONS

1-Byte Read Sequence (cont.)

NO.	FUNCTION	R/W	PORT	DATA
61	Enable 8X305 Data Bus to 50BUS	W	44H	01H
62	Activate Request	W	47H	Irrelevant
63	Deactivate Request	W	47H	Irrelevant

End of 256-Byte Read Sequence

1-Byte Write Sequence

NO.	FUNCTION	R/W	PORT	DATA
1	Enable 8X305 Data Bus to 50BUS	W	44H	01H
2	Select Device 1	W	40H	02H
3	Select Channel Command	W	41H	02H
4	Select Channel Number	W	43H	00-03H
5	Activate Request	W	47H	Irrelevant
6	Deactivate Request	W	47H	Irrelevant
7	Select Transmit Data Command	W	41H	03H
8	Load 1-Byte Instruction	W	43H	A3H
9	Activate Request	W	47H	Irrelevant
10	Deactivate Request	W	47H	Irrelevant

PART 1
8X305 COMMAND SEQUENCES FOR SDL OPERATIONS

1-Byte Write Sequence (cont.)

NO.	FUNCTION	R/W	PORT	DATA
11	Select Transmit Command	W	41H	00H
12	Load Transmit Enable Bit	W	43H	01H
13	Activate Request	W	47H	Irrelevant
14	Deactivate Request	W	47H	Irrelevant
15	Select Transmit Data Command	W	41H	03H
16	Load High-Order Address of Slave Memory	W	43H	HO Address
17	Activate Request	W	47H	Irrelevant
18	Deactivate Request	W	47H	Irrelevant
19	Enable Halt Circuit	W	56H	Irrelevant
20	Select Transmit Enable	W	41H	03H
21	Load Low-Order Address of Slave Memory	W	43H	LO Address
22	Activate Request	W	47H	Irrelevant
23	Deactivate Request	W	47H	Irrelevant
24	Select Transmit Data Command	W	41H	03H
24	Load Write Data Going to Slave Memory	W	43H	Write Data
26	Activate Request	W	47H	Irrelevant
27	Deactivate Request	W	47H	Irrelevant
28	Disable Halt Circuit	W	56H	Irrelevant

PART 1
8X305 COMMAND SEQUENCES FOR SDL OPERATIONS

1-Byte Write Sequence (cont.)

NO.	FUNCTION	R/W	PORT	DATA
29	Disable 8X305 Data from 50BUS	W	44H	00H
30	Select Status Command	W	41H	05H
31	Activate Request	W	47H	Irrelevant
32	Read Status. Continue reading until 50BD2 = 1. (Look for a Byte Request)	R	40H	
33	Deactivate Request	W	47H	Irrelevant
34	Enable 8X305 Data to 50BUS	W	44H	01H
35	Select Transmit Command	W	41H	00H
36	Load Transmit Disable Bit	W	43H	00H
37	Delay 1 us	N/A	N/A	N/A
38	Activate Request	W	47H	Irrelevant
39	Deactivate Request	W	47H	Irrelevant

End of 1-Byte Write Sequence

256-Byte Write Sequence

NO.	FUNCTION	R/W	PORT	DATA
1	Load Data Buffer Low-Order Address Bits (eight bits)	W	30H	LO Address

PART 1
8X305 COMMAND SEQUENCES FOR SDL OPERATIONS

256-Byte Write Sequence (cont.)

NO.	FUNCTION	R/W	PORT	DATA
2	Load Data Buffer High-Order Address Bits (four bits, low-order nibble of byte)	W	31H	HO Address
3	Disable All Write Buffers to Data Buffer	W	57H	00H
4	Enable 8X305 Data Bus to 50BUS	W	44H	01H
5	Select Device 1	W	40H	02H
6	Select Channel Command	W	41H	02H
7	Select Channel Number	W	43H	00-03H
8	Activate Request	W	47H	Irrelevant
9	Deactivate Request	W	47H	Irrelevant
10	Select Transmit Data Command	W	41H	03H
11	Load 256-Byte Instruction	W	43H	A5H
12	Activate Request	W	47H	Irrelevant
13	Deactivate Request	W	47H	Irrelevant
14	Select Transmit Command	W	41H	00H
15	Load Transmit Enable Bit	W	43H	01H
16	Activate Request	W	47H	Irrelevant
17	Deactivate Request	W	47H	Irrelevant
18	Select Transmit Data Command	W	41H	03H
19	Load High-Order Address of Slave Memory	W	43H	HO Address

PART 1
8X305 COMMAND SEQUENCES FOR SDL OPERATIONS

256-Byte Write Sequence (cont.)

NO.	FUNCTION	R/W	PORT	DATA
20	Activate Request	W	47H	Irrelevant
21	Deactivate Request	W	47H	Irrelevant
22	Enable Halt Circuit	W	56H	Irrelevant
23	Select Transmission Data Command	W	41H	03H
24	Load Low-Order Address of Slave Memory	W	43H	LO Address
25	Activate Request	W	47H	Irrelevant
26	Deactivate Request	W	47H	Irrelevant
27	Disable 8X305 Data From 50BUS. Enable Buffer Data to 50BUS.	W	44H	02H
28	Select Transmit Data Command	W	41H	03H
29	Read Data From Buffer (Turn On Strobe)	W	32H	Irrelevant
30	Turn Off Read Strobe	W	32H	Irrelevant
31	Read 50BUS	R	40H	Ignore
32	Activate Request	W	47H	Irrelevant
33	Deactivate Request	W	47H	Irrelevant
34	Test for 256 bytes transferred. If not, jump to 29. If done, fall to 35.			
35	Deactivate Halt Circuit	W	56H	Irrelevant

PART 1
8X305 COMMAND SEQUENCES FOR SDL OPERATIONS

256-Byte Write Sequence (cont.)

NO.	FUNCTION	R/W	PORT	DATA
36	Disable Buffer Data to 50BUS	W	44H	00H
37	Select Status Command	W	41H	05H
38	Activate Request	W	47H	Irrelevant
39	Read Status. Continue reading until 50BD2 = 1. (Look for a Byte Request)	R	40H	
40	Deactivate Request	W	47H	Irrelevant
41	Enable 8X305 Data to 50BUS	W	44H	01H
42	Select Transmit Command	W	41H	00H
43	Load Transmit Disable Bit	W	43H	00H
44	Delay 1 us	N/A	N/A	N/A
45	Activate Request	W	47H	Irrelevant
46	Deactivate Request	W	47H	Irrelevant

End of 256-Byte Write Sequence

Status Read

NO.	FUNCTION	R/W	PORT	DATA
1	Enable 8X305 Data Bus to 50BUS	W	44H	01H
2	Select Device 1	W	40H	02H
3	Select Channel Command	W	41H	02H

PART 1
8X305 COMMAND SEQUENCES FOR SDL OPERATIONS

Status Read (cont.)

NO.	FUNCTION	R/W	PORT	DATA
4	Select Channel Number	W	43H	00-03H
5	Activate Request	W	47H	Irrelevant
6	Deactivate Request	W	47H	Irrelevant
7	Select Transmit Data Command	W	41H	03H
8	Load Status Read Instruction	W	43H	BO
9	Activate Request	W	47H	Irrelevant
10	Deactivate Request	W	47H	Irrelevant
11	Select Transmit Command	W	41H	00H
12	Load Transmit Enable Bit	W	43H	01H
13	Activate Request	W	47H	Irrelevant
14	Deactivate Request	W	47H	Irrelevant
15	Disable 8X305 Data Bus from 50BUS	W	44H	00H
16	Select Status Command	W	41H	05H
17	Activate Request	W	47H	Irrelevant
18	Read Status. Continue reading until 50BD2 = 1. (Look for a Byte Request)	R	40H	
19	Deactivate Request	W	47H	Irrelevant
20	Select Transmit Command	W	41H	00H
21	Load Transmit Disable Bit	W	43H	00H

PART 1
8X305 COMMAND SEQUENCES FOR SDL OPERATIONS

Status Read (cont.)

NO.	FUNCTION	R/W	PORT	DATA
22	Enable 8X305 Data Bus to 50BUS	W	44H	01H
23	Activate Request	W	47H	Irrelevant
24	Deactivate Request	W	47H	Irrelevant
25	Select Receive Command	W	41H	01H
26	Load Receive Enable Bit	W	43H	01H
27	Delay 60 us	N/A	N/A	N/A
28	Activate Request	W	47H	Irrelevant
29	Deactivate Request	W	47H	Irrelevant
30	Disable 8X305 Data Bus from 50BUS	W	44H	00H
31	Select Status Command	W	41H	05H
32	Activate Request	W	47H	Irrelevant
33	Read Status. Check 50BD0. If 50BD0 = 1, a byte of data has been received. Check 50BD3. If 50BD3 = 1, a No Data Timeout has occurred and the operation will be terminated.	R	40H	
34	Deactivate Request	W	47H	Irrelevant
35	Select Receive Data Command	W	41H	04H
36	Activate Request	W	47H	Irrelevant
37	Read Received Data	R	40H	

PART 1
8X305 COMMAND SEQUENCES FOR SDL OPERATIONS

Status Read (cont.)

NO.	FUNCTION	R/W	PORT	DATA
38	Deactivate Request	W	47H	Irrelevant
39	Select Status Command	W	41H	05H
40	Activate Request	W	47H	Irrelevant
41	Read Status. Check 50BD1. If 50BD1 = 1, a parity error has been detected.	R	40H	
42	Deactivate Request	W	47H	Irrelevant
43	Select Receive Command	W	41H	01H
44	Load Receiver Disable Bit	W	43H	00H
45	Enable 8X305 Data Bus to 50BUS	W	44H	01H
46	Activate Request	W	47H	Irrelevant
47	Deactivate Request	W	47H	Irrelevant

End of Status Read Sequence

PART 2
8X305 MICROCONTROLLER COMMAND SEQUENCES

To communicate with the floppy disk drive and several internal devices, the RCU issues seven commands (TRANSMIT, RECEIVE, CHANNEL SELECT, TRANSMIT DATA, RECEIVE DATA, STATUS READ, and DIAGNOSTIC) in various sequences. The RMU decodes these commands from the 50BUS address lines when device 1 is selected (ie, /50BSLCT1 is active) and 50BREQ is strobed. The Microcontroller generates the codes for the above commands in predefined sequences to perform several different operations. The floppy disk read, floppy disk write, and internal device operational sequences are outlined below.

FLOPPY DISK READ OPERATION/50BUS INTERFACE

No.	Function	R/W	Port	Data
1	Load data buffer low-order address bits (four bits, high-order nibble of byte)	W	30H	LO address
2	Load data buffer high-order address bits (four bits, low-order nibble of byte)	W	31H	HO address
3	Select device 2	W	40H	04H
4	Enable 50BUS data to data buffer	W	57H	04H
5	Disable 8X305 data bus from 50BUS	W	44H	00H
6	Generate 50BUS read signal	W	45H	40H
7	Read status. Check bit 1. If bit 1 = 1, a byte is ready to be put in data buffer	R	41H	
8	Delay 1.6 us	N/A	N/A	N/A

PART 2
8X305 MICROCONTROLLER COMMAND SEQUENCES

FLOPPY DISK READ OPERATION/50BUS INTERFACE

No.	Function	R/W	Port	Data
9	Activate request	W	47H	Irrelevant
10	Write data to buffer (turn on strobe)	W	33H	Irrelevant
11	Write data to buffer (turn off strobe)	W	33H	Irrelevant
12	Deactivate request	W	47H	Irrelevant
	Repeat steps 7-12 until a sector or sectors of data have been transferred (the 8X305 Microcontroller must keep track of the number of bytes received to detect the end of the DMA operation). When all data has been received, proceed to step 13.			
13	Issue Terminal Count to floppy disk controller (turn TC on)	W	46H	04H
14	Turn Terminal Count off	W	46H	00H
15	Disable 50BUS data to data buffer	W	57H	00H

End of Floppy Disk Read Operation

FLOPPY DISK WRITE OPERATION/50BUS INTERFACE

No.	Function	R/W	Port	Data
1	Load data buffer low-order address bits (eight bits)	W	30H	LO address

PART 2
8X305 MICROCONTROLLER COMMAND SEQUENCES

FLOPPY DISK WRITE OPERATION/50BUS INTERFACE (cont.)

No.	Function	R/W	Port	Data
2	Load data buffer high-order address bits (four bits, low-order nibble of byte)	W	31H	HO address
3	Select device 2	W	40H	04H
4	Enable buffer data to 50BUS	W	44H	02H
5	Generate 50BUS write signal	W	45H	00H
6	Read status. Check bit 1. If bit 1 = 1, a byte is needed to be placed on the disk	R	41H	
7	Delay 1.6 us	N/A	N/A	N/A
8	Read data from buffer (turn on strobe)	W	32H	Irrelevant
9	Read data from buffer (turn off strobe)	W	32H	Irrelevant
10	Activate request	W	47H	Irrelevant
11	Deactivate request	W	47H	Irrelevant
	Repeat steps 6-11 until a sector or sectors of data have been transferred (the 8X305 Microcontroller must keep track of the number of bytes sent to detect the end of the DMA operation). When all data has been sent, proceed to step 12.			
12	Issue Terminal Count to floppy disk controller (turn TC on)	W	46H	04H

PART 2
8X305 MICROCONTROLLER COMMAND SEQUENCES

FLOPPY DISK WRITE OPERATION/50BUS INTERFACE (cont.)

No.	Function	R/W	Port	Data
13	Turn Terminal Count off	W	46H	00H
14	Disable buffer data to 50BUS	W	44H	00H

End of Floppy Disk Write Operation

TESTING BUFFER MEMORY USING 8X305

No.	Function	R/W	Port	Data
1	Load low-order buffer address (eight bits)	W	30H	LO address
2	Load high-order buffer address (four bits, low-order nibble of byte)	W	31H	HO address
3	Enable 8X305 data to 50BUS	W	44H	01H
4	Enable 50BUS data to buffer	W	57H	04H
5	Write test data to 50BUS	W	43H	Test data
6	Turn on buffer write strobe	W	33H	Irrelevant
7	Turn off buffer write strobe	W	33H	Irrelevant

PART 2
8X305 MICROCONTROLLER COMMAND SEQUENCES

TESTING BUFFER MEMORY USING 8X305 (cont.)

No.	Function	R/W	Port	Data
8	If test data is the same for all locations, jump to step 6; if test data is different, jump to step 5. If the 8X305 Microcontroller has finished writing all of buffer memory, fall through to step 9.			
9	Load low-order buffer address (eight bits)	W	30H	LO address
10	Load high-order buffer address (four bits, low-order nibble of byte)	W	31H	HO address
11	Disable 8X305 data to 50BUS and enable buffer data to 50BUS.	W	44H	02H
12	Disable 50BUS data to buffer.	W	57H	00
13	Turn on buffer read strobe	W	32H	Irrelevant
14	Turn off buffer read strobe	W	32H	Irrelevant
15	Read 50BUS data	R	40H	Test data
16	Check data and jump back to step 13. Continue until all locations are checked.			

End of Buffer Memory Test Using 8X305

PART 2
8X305 MICROCONTROLLER COMMAND SEQUENCES

BLOCK READ SEQUENCE FROM IWS OR IWISE

No.	Function	R/W	Port	Data
1	Disable 8X305 data bus to 50BUS	W	44H	00H
2	Enable 50BUS data to buffer	W	57H	04H
3	Load data buffer low-order address (eight bits)	W	30H	LO address
4	Load data buffer high-order address (four bits, low-order nibble of byte)	W	31H	HO address
5	Select device	W	40H	03-07H
6	Load high-order slave memory address	W	42H	HO address
7	Load low-order slave memory address	W	41H	LO address
8	Generate bus request to slave and turn on Read signal	W	45H	60H
9	Look for Bus Acknowledge signal from slave	R	41H	LSB = 1
10	Activate request	W	47H	Irrelevant
11	Delay 600 ns	N/A	N/A	N/A
12	Write data to data buffer (turn on strobe)	W	33H	Irrelevant
13	Turn off buffer write strobe	W	33H	Irrelevant
14	Deactivate request	W	47H	Irrelevant

PART 2
8X305 MICROCONTROLLER COMMAND SEQUENCES

BLOCK READ SEQUENCE FROM IWS OR IWISE (cont.)

No.	Function	R/W	Port	Data
15	Increment low-order slave memory address	W	41H	LO address
	Repeat steps 10-15 until a page or pages of data have been transferred. (The 8X305 Microcontroller must keep track of the number of bytes received to detect the end of the DMA operation.) When all data has been received, proceed to step 16.			
16	Turn off slave bus request and Read signal	W	45H	40H
17	Disable 50BUS data to buffer	W	57H	00H

End of Block Read Sequence from IWS or IWISE

1-BYTE READ SEQUENCE FROM IWS OR IWISE

No.	Function	R/W	Port	Data
1	Select device	W	40H	03-07H
2	Load high-order slave memory address	W	42H	HO address
3	Load low-order slave memory address	W	41H	LO address
4	Disable 8X305 data bus to 50BUS data bus	W	44H	00H

PART 2
8X305 MICROCONTROLLER COMMAND SEQUENCES

1-BYTE READ SEQUENCE FROM IWS OR IWISE (cont.)

No.	Function	R/W	Port	Data
5	Generate bus request to slave and turn on Read signal	W	45H	60H
6	Look for Bus Acknowledge signal from slave	R	41H	LSB = 1
7	Activate request	W	47H	Irrelevant
8	Delay 600 ns	N/A	N/A	N/A
9	Read data	R	40H	Don't care
10	Deactivate request	W	47H	Irrelevant
11	Turn off slave bus request and Write signal	W	45H	40H

End of 1-Byte Read Sequence from IWS or IWISE

BLOCK WRITE SEQUENCE TO IWS OR IWISE

No.	Function	R/W	Port	Data
1	Disable 8X305 data bus to 50BUS and enable buffer data to 50BUS	W	44H	02H
2	Load data buffer low-order address (eight bits)	W	30H	LO address
3	Load data buffer high-order address (four bits-low-order nibble of byte)	W	31H	HO address

PART 2
8X305 MICROCONTROLLER COMMAND SEQUENCES

BLOCK WRITE SEQUENCE TO IWS OR IWISE (cont.)

No.	Function	R/W	Port	Data
4	Select device	W	40H	03-07H
5	Load high-order slave memory address	W	42H	HO address
6	Load low-order slave memory address	W	41H	LO address
7	Generate bus request to slave and turn on Write signal	W	45H	20H
8	Look for Bus Acknowledge signal from slave	R	41H	LSB = 1
9	Read data from data buffer (turn on strobe)	W	32H	Irrelevant
10	Turn off buffer read strobe	W	32H	Irrelevant
11	Activate request	W	47H	Irrelevant
12	Delay 600 ns	N/A	N/A	N/A
13	Deactivate request	W	47H	Irrelevant
14	Increment low-order slave memory address	W	41H	LO address

Repeat steps 9-14 until a page or pages of data have been transferred. (The 8X305 Microcontroller must keep track of number of bytes sent to detect the end of the DMA operation.) When all data has been sent, proceed to step 15.

PART 2
8X305 MICROCONTROLLER COMMAND SEQUENCES

BLOCK WRITE SEQUENCE TO IWS OR IWISE (cont.)

No.	Function	R/W	Port	Data
13	Turn off slave bus request and Write signal	W	45H	40H
14	Disable data buffer from 50BUS	W	44H	00H

End of Block Write Sequence to IWS or IWISE

1-BYTE WRITE SEQUENCE TO IWS OR IWISE

No.	Function	R/W	Port	Data
1	Select device	W	40H	03-07H
2	Load high-order slave memory address	W	42H	HO address
3	Load low-order slave memory address	W	41H	LO address
4	Enable 8X305 data bus to 50BUS data bus	W	44H	01H
5	Load data to be written to slave	W	43H	Data
6	Generate bus request to slave and turn on Write signal	W	45H	20H
7	Look for Bus Acknowledge signal from slave	R	41H	LSB = 1
8	Activate request	W	47H	Irrelevant

PART 2
8X305 MICROCONTROLLER COMMAND SEQUENCES

1-BYTE WRITE SEQUENCE TO IWS OR IWISE (cont.)

No.	Function	R/W	Port	Data
19	Delay 600 ns	N/A	N/A	N/A
10	Deactivate request	W	47H	Irrelevant
11	Turn off slave bus request and Write signal	W	45H	40H

End of 1-Byte Write Sequence to IWS or IWISE

STATUS SEQUENCE FROM IWS OR IWISE

No.	Function	R/W	Port	Data
1	Select device	W	40H	03-07H
2	Disable 8X305 data bus to 50BUS	W	44H	00H
3	Turn on device status	W	46H	08H
4	Read device status	R	40H	Status
5	Turn off device status	W	46H	00H

End of Status Sequence from IWS or IWISE

PART 2
8X305 MICROCONTROLLER COMMAND SEQUENCES

RESTART SEQUENCE TO IWS OR IWISE

No.	Function	R/W	Port	Data
1	Select device	W	40H	03-07H
2	Turn on 50BUS reset	W	46H	10H
3	Delay 6 us	N/A	N/A	N/A
4	Turn off 50BUS reset	W	46H	00H

End of Restart Sequence of IWS or IWISE

BLOCK READ SEQUENCE FROM MASTER MEMORY

No.	Function	R/W	Port	Data
1	Disable refresh circuit	W	60H	Irrelevant
2	Disable 8X305 data bus to 50BUS	W	44H	00H
3	Enable 50BUS data to buffer	W	57H	04H
4	Load data buffer low-order address (eight bits)	W	30H	LO address
5	Load data buffer high-order address (four bits-low-order nibble of byte)	W	31H	HO address
6	Select device	W	40H	01H
7	Load high-order slave memory address	W	42H	HO address

PART 2
8X305 MICROCONTROLLER COMMAND SEQUENCES

BLOCK READ SEQUENCE FROM MASTER MEMORY (cont.)

No.	Function	R/W	Port	Data
8	Load low-order slave memory address	W	41H	LO address
9	Generate bus request to slave and turn on Read signal	W	45H	60H
10	Look for Bus Acknowledge signal from slave	R	41H	LSB = 1
11	Activate request	W	47H	Irrelevant
12	Delay 600 ns	N/A	N/A	N/A
13	Write data to data buffer (turn on strobe)	W	33H	Irrelevant
14	Turn off buffer write strobe	W	33H	Irrelevant
15	Deactivate request	W	47H	Irrelevant
16	Increment low-order slave memory address	W	41H	LO address
	Repeat steps 11-16 until a page or pages of data have been transferred. (The 8X305 Microcontroller must keep track of the number of bytes received to detect the end of the DMA operation.) When all data has been received, proceed to step 17.			
17	Turn off slave bus request and Read signal	W	45H	40H
18	Disable 50BUS data to buffer	W	57H	00H

End of Block Read Sequence from Master Memory

PART 2
8X305 MICROCONTROLLER COMMAND SEQUENCES

1-BYTE READ SEQUENCE FROM MASTER MEMORY

No.	Function	R/W	Port	Data
1	Disable refresh circuitry	W	60H	Irrelevant
2	Select device	W	40H	01H
3	Load high-order slave memory address	W	42H	HO address
4	Load low-order slave memory address	W	41H	LO address
5	No operation	N/A	N/A	N/A
6	Disable 8X305 data bus to 50BUS data bus	W	44H	00H
7	Generate bus request to slave and turn on Read signal	W	45H	60H
8	Look for Bus Acknowledge signal from slave	R	41H	LSB = 1
9	Activate request	W	47H	Irrelevant
10	Delay 600 ns	N/A	N/A	N/A
11	Read data	R	40H	Don't care
12	Deactivate request	W	47H	Irrelevant
13	Turn off slave bus request and Write signal	W	45H	40H

End of 1-Byte Read Sequence from Master Memory

PART 2
8X305 MICROCONTROLLER COMMAND SEQUENCES

BLOCK WRITE SEQUENCE TO MASTER MEMORY

No.	Function	R/W	Port	Data
1	Disable refresh circuitry	W	60H	Irrelevant
2	Disable 8X305 data bus to 50BUS and enable buffer data to 50BUS	W	44H	02H
3	Load data buffer low-order address (eight bits)	W	30H	LO address
4	Load data buffer high-order address (four bits, low-order nibble of byte)	W	31H	HO address
5	Select device	W	40H	01H
6	Load high-order slave memory address	W	42H	HO address
7	Load low-order slave memory address	W	41H	LO address
8	Generate bus request to slave and turn on Write signal	W	45H	20H
9	Look for Bus Acknowledge signal from slave	R	41H	LSB = 1
10	Read data from data buffer (turn on strobe)	W	32H	Irrelevant
11	Turn off buffer read strobe	W	32H	Irrelevant
12	Activate request	W	47H	Irrelevant
13	Delay 600 ns	N/A	N/A	N/A

PART 2
8X305 MICROCONTROLLER COMMAND SEQUENCES

BLOCK WRITE SEQUENCE TO MASTER MEMORY (cont.)

No.	Function	R/W	Port	Data
14	Deactivate request	W	47H	Irrelevant
15	Increment low-order slave memory address	W	41H	LO address
	Repeat steps 10-15 until a page or pages of data have been transferred. (The 8X305 Microcontroller must keep track of the number of bytes sent to detect the end of the DMA operation.) When all data has been sent, proceed to step 16.			
16	Turn off slave bus request and Write signal	W	45H	40H
17	Disable data buffer from 50BUS	W	44H	00H

End of Block Write Sequence to Master Memory

1-BYTE WRITE SEQUENCE TO MASTER MEMORY

No.	Function	R/W	Port	Data
1	Disable refresh circuitry	W	60H	Irrelevant
2	Select device	W	40H	01H
3	Load high-order slave memory address	W	42H	HO address
4	Load low-order slave memory address	W	41H	LO address

PART 2
8X305 MICROCONTROLLER COMMAND SEQUENCES

1-BYTE WRITE SEQUENCE TO MASTER MEMORY (cont.)

No.	Function	R/W	Port	Data
5	Enable 8X305 data bus to 50BUS data bus	W	44H	01H
6	Load Data to be written to slave	W	43H	Data
7	Generate bus request to slave and turn on Write signal	W	45H	20H
8	Look for Bus Acknowledge signal from slave	R	41H	LSB = 1
9	Activate request	W	47H	Irrelevant
10	Delay 600 ns	N/A	N/A	N/A
11	Deactivate request	W	47H	Irrelevant
12	Turn off slave bus request and Write signal	W	45H	40H

End of 1-Byte Write Sequence to Master Memory

APPENDIX

G

APPENDIX G:

RMU-RCU COMMUNICATION PROTOCOL

Introduction

When the RMU performs a function involving the RCU, the Z80A loads the command code and necessary data into the Parameter Register File (PRF). The RCU reads the RMU-generated command from the PRF and then executes the command. If the command is an immediate command, the RCU must execute the command immediately, because the Z80A is waiting for command completion. If the command is a non-immediate command, the RCU executes the command after completing its immediate operation. After the RCU has executed the command (whether immediate or non-immediate), it returns the required data to either the Parameter Register File or the Status Register File (SRF). The RMU accesses this data by reading the appropriate register file.

Upon receiving an immediate command, the RCU validates the command code, copies the PRF to its scratchpad RAM, performs an operational sequence to complete the requested operation, and, finally, releases the RMU. Upon receiving a non-immediate command, the RCU validates the command code, copies the PRF to its scratchpad RAM, sets RCU Busy status in the SRF, and releases the RMU. The RCU then performs an operational sequence to complete the requested operation and issues either an Interrupt Request or a Clear Busy Status signal to inform the RMU that the operation has been completed.

The following are short summaries of the exchanges that take place when the RMU issues specific commands to the RCU. Brief descriptions of the commands are followed by lists that define the items required and returned for each operation. Items required are those bytes sent by the RMU to the RCU via the PRF. Items returned are the RCU-generated responses to the RMU commands. Items can be returned to the RMU via either the PRF or the SRF.

Both the RMU and the RCU can read from or write to the PRF; both boards cannot access the PRF simultaneously, however. FF20-FF2FH are the Parameter Register File bytes. Only the RCU can write to the SRF, and only the RMU can read from the SRF. FF10-FF1BH are the Status Register File bytes. The contents of SRF bytes are listed near the end of this Appendix.

ABORT FLOPPY

Abort Floppy commands the RCU to discontinue the FDD data transfer. This command normally is made when a read or write operation is terminated prematurely. Abort Floppy is an immediate command, and the RMU is bus-requested until the RCU completes the command.

Items Required:

Byte	Contents
FF20	Request Code - 01H
FF22	Specify FDC Abort - 01H

Items Returned:

Byte	Contents
F21	Command Acknowledge 00 Command Not Accepted 01 Invalid Command 80 Command Accepted and Completed
FF10	RCU Status

SET SLAVE LIST

The Set Slave List is an immediate command that command provides a logical to physical mapping of the Slave Unit Numbers used in Data Link operations. Mapping allows the system to be configured for development and/or diagnostics. For example, if the master requires the IWISE to be Logical Unit 01, FF22 contains 05H when the command is made. Then, whenever a Data Link command uses unit 01, the RCU performs the command on the IWISE. Any combination of logical to physical mapping is allowed except Logical Unit 00, which is always Master Memory. A programming error has occurred if Physical Unit 00H is listed. After Set Slave List is issued, the RCU PROM Release Number is available through FF2B-2F.

Items Required:

Byte	Contents
FF20	Request Code - 02H
FF22	Physical Unit Number for Logical Unit 01
FF23	Physical Unit Number for Logical Unit 02
FF24	Physical Unit Number for Logical Unit 03
FF25	Physical Unit Number for Logical Unit 04
FF26	Physical Unit Number for Logical Unit 05
FF27	Physical Unit Number for Logical Unit 06
FF28	Physical Unit Number for Logical Unit 07
FF29	Physical Unit Number for Logical Unit 08
FF2A	Physical Unit Number for Logical Unit 09

Items Returned:

Byte	Contents
FF21	Command Acknowledge 00 Command Not Accepted 01 Invalid Command 80 Command Accepted and Completed
FF10	RCU Status
FF2B-2F	RCU Prom Release Number. (Can use up to five bytes. Note this is the PRF, not the SRF.)

Physical Unit Numbers are defined as follows:

01 - IWS 1	06 - Serial Port Channel 1
02 - IWS 2	07 - Serial Port Channel 2
03 - IWS 3	08 - Serial Port Channel 3
04 - IWS 4	09 - Serial Port Channel 4
05 - IWISE	

1-BYTE RESET

The 1-Byte Reset command resets both internal and Serial Port Slaves, which forces the slave processor to execute instructions at slave memory location zero. This command is an immediate command, similar to the 928 Data Link reset function.

The returned 1-Byte Transfer Status Byte is valid when Command Acknowledge is 80H and invalid when Command Acknowledge is 00H or 01H. 1-Byte Slave Status may not be valid if 1-Byte Transfer Status indicates a Parity Error or No Data Timeout.

Any 1-Byte command involving Master Memory (Unit 00) is an invalid command. Following the invalid command, Command Acknowledge is 01H and Transfer and Slave Status are set to 00H.

Items Required:

Byte	Contents
FF20	Request Code - 03H
FF22	Specify Reset - 01H
FF23	Unit Select

Items Returned:

Byte	Contents
FF21	Command Acknowledge 00 Command Not Accepted 01 Invalid Command 80 Command Accepted and Completed
FF10	RCU Status
FF13	1-Byte Transfer Status
FF14	1-Byte Slave Status

1-BYTE STATUS

The 1-Byte Status command is an immediate command that is similar to the 928 Data Link slave status function. The returned 1-Byte Transfer Status is valid when Command Acknowledge is 80H. 1-Byte Slave Status may not be valid if 1-Byte Transfer Status indicates a Parity Error or No Data Timeout. Any 1-Byte Command involving Master Memory (Unit 00) is an invalid command.

Items Required:

Byte	Contents
FF20	Request Code 03H
FF22	Specify Status - 02H
FF23	Unit Select

Items Returned:

Byte	Contents
FF21	Command Acknowledge 00 Command Not Accepted 01 Invalid Command 80 Command Accepted and Completed
FF10	RCU Status
FF13	1-Byte Transfer Status
FF14	1-Byte Slave Status

1-BYTE READ

The 1-Byte Read command is an immediate command that is similar to the 928 Data Link read data function. The returned 1-Byte Transfer Status is valid when Command Acknowledge is 80H. 1-Byte Slave Status may not be valid if the 1-Byte Transfer Status indicates a Parity Error or No Data Timeout. Data may not be valid if Slave Status indicates an error. Any 1-Byte command involving Master Memory (Unit 00) is an invalid command.

Items Required:

Byte	Contents
FF20	Request Code - 03H
FF22	Specify Read - 03H
FF23	Unit Select
FF24	Memory Address Low
FF25	Memory Address High

Items Returned:

Byte	Contents
FF21	Command Acknowledge 00 Command Not Accepted 01 Invalid Command 80 Command Accepted and Completed
FF10	RCU Status
FF13	1-Byte Transfer Status
FF14	1-byte Slave Status
FF15	Data Read from Slave (One byte)

1-BYTE WRITE

The 1-Byte Write command is an immediate command that is similar to the 928 Data Link write data function. The returned 1-Byte Transfer Status is valid when Command Acknowledge is 80H. 1-Byte Slave Status may not be valid if 1-Byte Transfer Status indicates a Parity Error or No Data Timeout. Any 1-Byte command involving Master Memory (Unit 00) is an invalid command.

Items Required:

Byte	Contents
FF20	Request Code - 03H
FF22	Specify Write - 04H
FF23	Unit Select
FF24	Memory Address Low
FF25	Memory Address High
FF26	Data to Write (One byte)

Items Returned:

Byte	Contents
FF21	Command Acknowledge 00 Command Not Accepted 01 Invalid Command 80 Command Accepted and Completed
FF10	RCU Status
FF13	1-Byte Transfer Status
FF14	1-Byte Slave Status