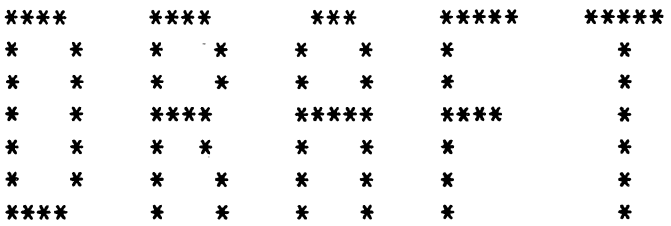
HONEYWELL INFORMATION SYSTEMS ITALIA		SPEC. NO.	SHEET	REVISION
		A78xxxxxx	1/62	DRAFT1

Prepared by Date	Designation
P. GONELLA 86.05.07	PRODUCT DESIGN DESCRIPTION
Approved by Date	S. G. M. 2
	L I N E P R O C E S S O R 0

REVISION RECORD

REV	AUTHORITY	DATE	APPROVED BY	SHEETS AFFECTED
				- ALL -



INDEX

1. GENERAL DESCRIPTION

The Line Processor 0 is an intelligent communication processor with SGM2-VME bus interface, which supports up to 6 full duplex channels with RS-232C and RS-422A interfaces, for serial communications and one Centronics/IBM parallel printer interface.

In VME enviroment the Line Processor 0 is a slave controller fully compatible which responds to 32-bit addressing and 8 or 16 bit data transfers. Communications between the system CPUs and the LP0 can take place in three ways:

-- from host to LP0 or vice versa by message interchange via a shared-RAM allocated on the LP0;

-- from host to LP0 by writing an 1-bit attention register to interrupt the LP0;

-- from LP0 to host by an interrupter that generates interrupts to the VME bus on any of the seven levels and supplies an 8-bit vector during interrupt acknowledge cycle. The request level and the vector are programmable by the local processor.

The memory of the LP consists of 32Kbytes EPROM area, 512Kbytes of Local Dinamic Memory, 64Kbytes of Shared Sram Memory and 16Kbytes of I/O Sram Memory.

1.1 HARDWARE DESCRIPTION

The Major Block Diagram of the LINE Processor 0 board is described in details in Fig. 1.1 where it is possible to see the following functional areas:

- the local area;
- the I/O area;
- the shared area;

The main functional blocks are:

- MICROPROCESSORS AND DMA EMULATOR

  the microprocessors used are the 16 bit Motorola MC68000. They operate at 12.5 Mhz clock and they have an addressing capability up to 16 Mbytes (see para. 1.1.1); in this board one is master, the other is slave and it emulates a DMA.

- MASTER AND SLAVE INT & INTA LOGIC

  this logic permits to handle the interrupt lines (see para. 1.1.2.);

  NOTIFY INTERRUPTS.

  by this logic the slave processor can interrupt the master processor and viceversa. (see para. 1.1.2);

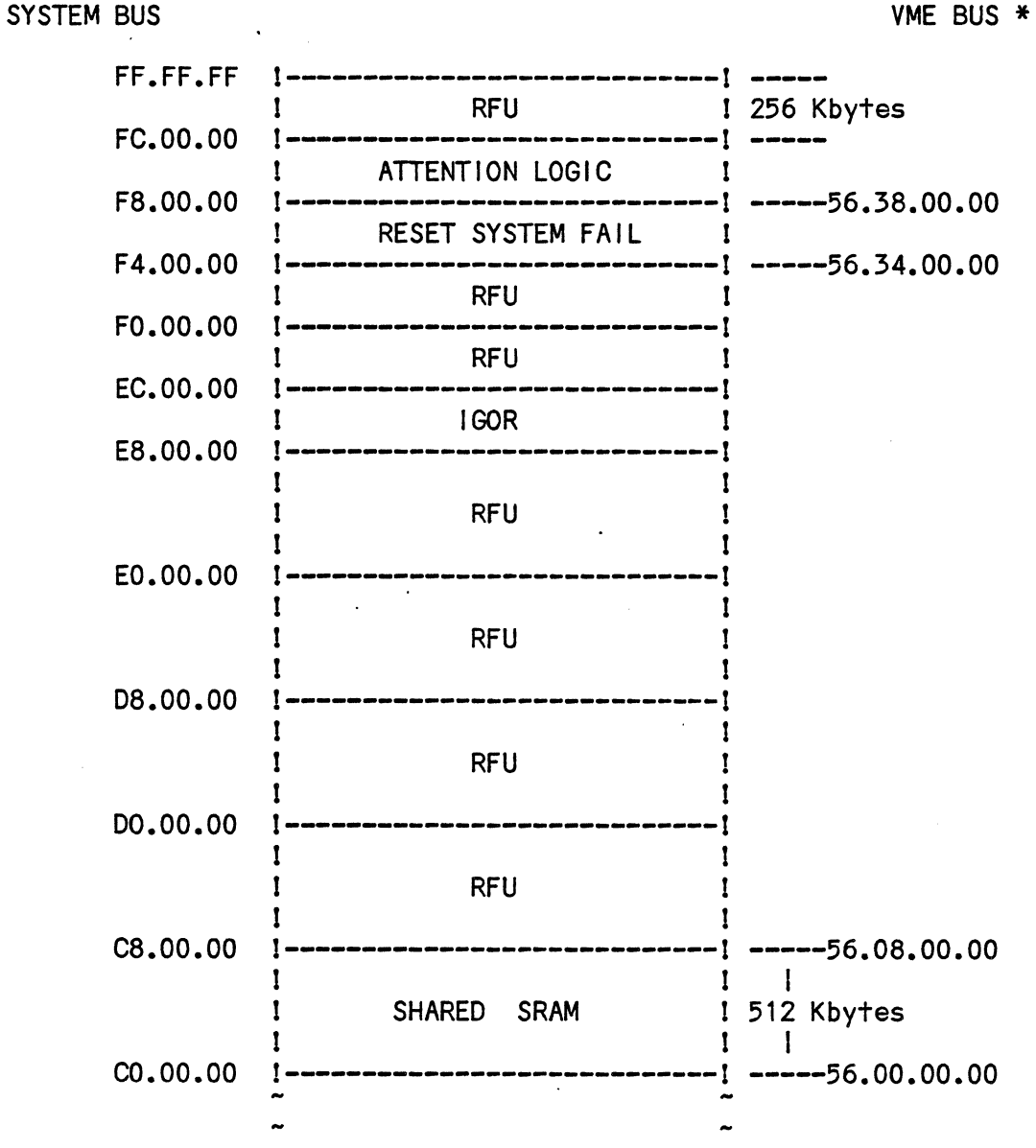
- CHIP SELECT AND CONTROL LOGIC FOR I/O, LOCAL AND SHARED AREA

  the main functions carried out by this logic are the following:

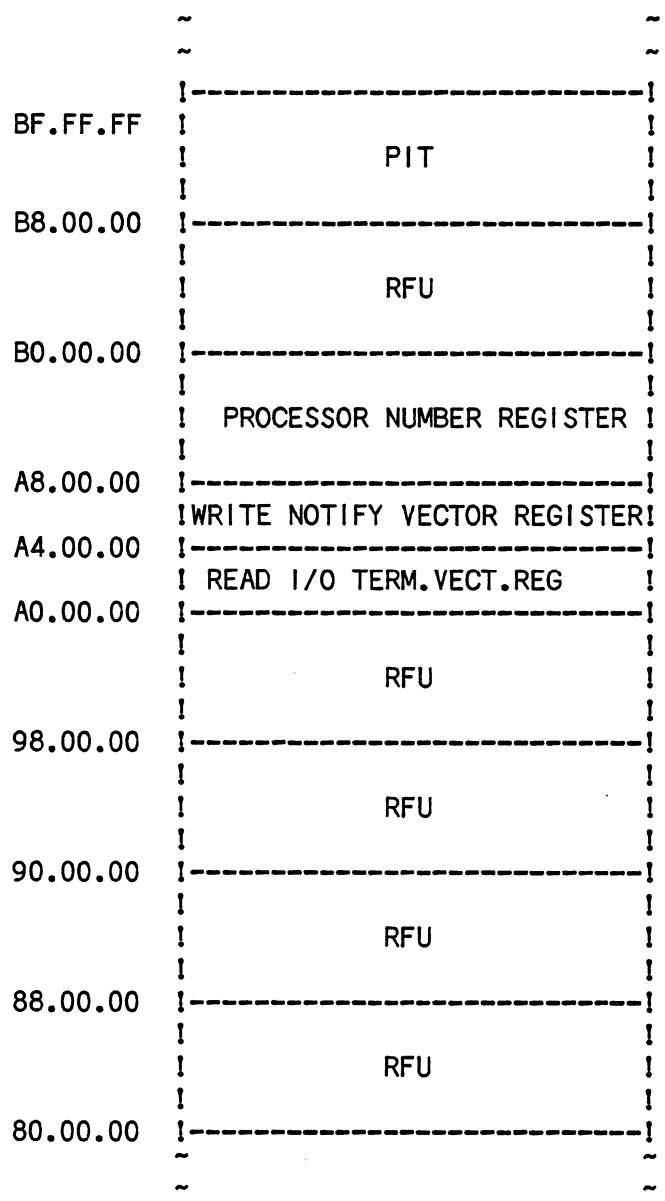
  - generation of the chip-selects;
  - generation of control signals (for example, the READ and WRITE clocks of the memory, etc);
  - generation of DATA TRANSFER ACKNOWLEDGE signal towards MC68000;
  - generation of a time-out signal (BERR line) every 16us if on-board peripheral don't return DTACK signal within this time;

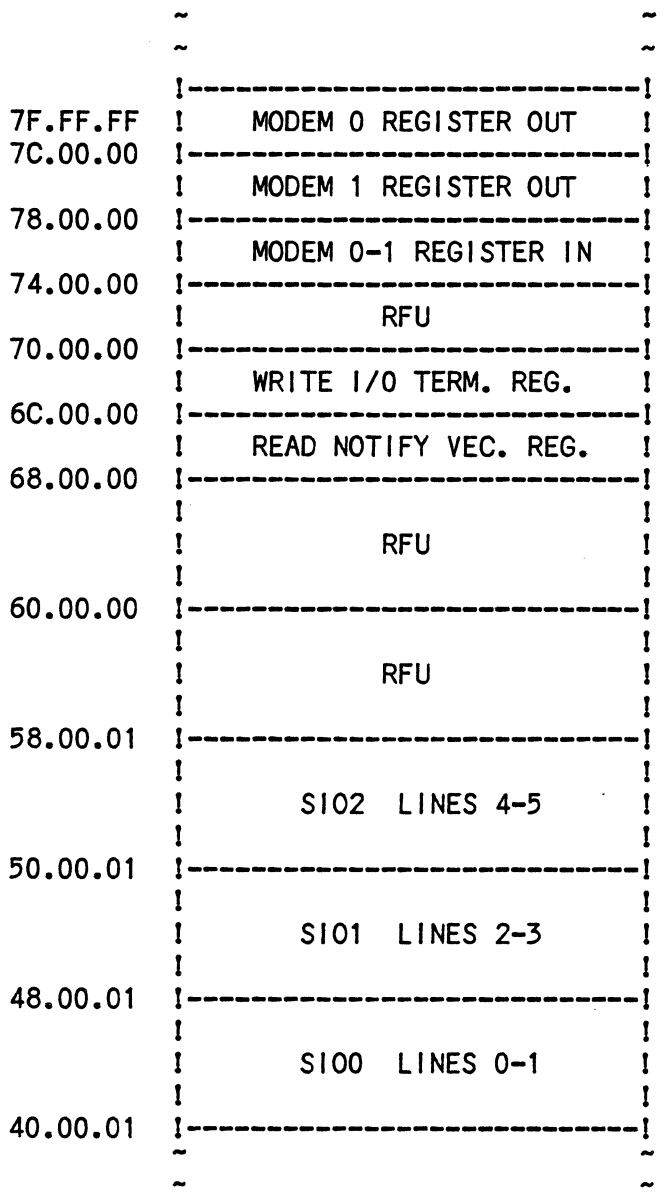
The 16 Mbyte addressing spaces of the processors are subdivided as shown below:

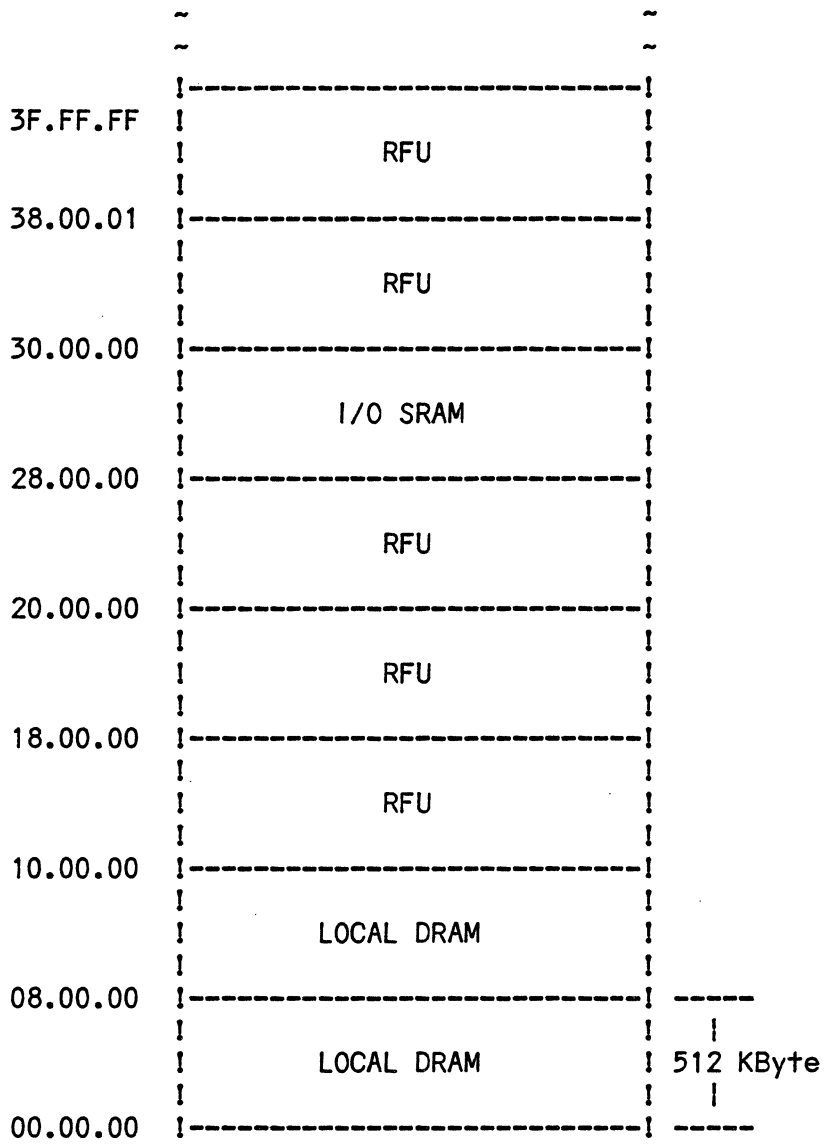
LINE PROCESSOR 0 - MASTER PROCESSOR MEMORY MAP



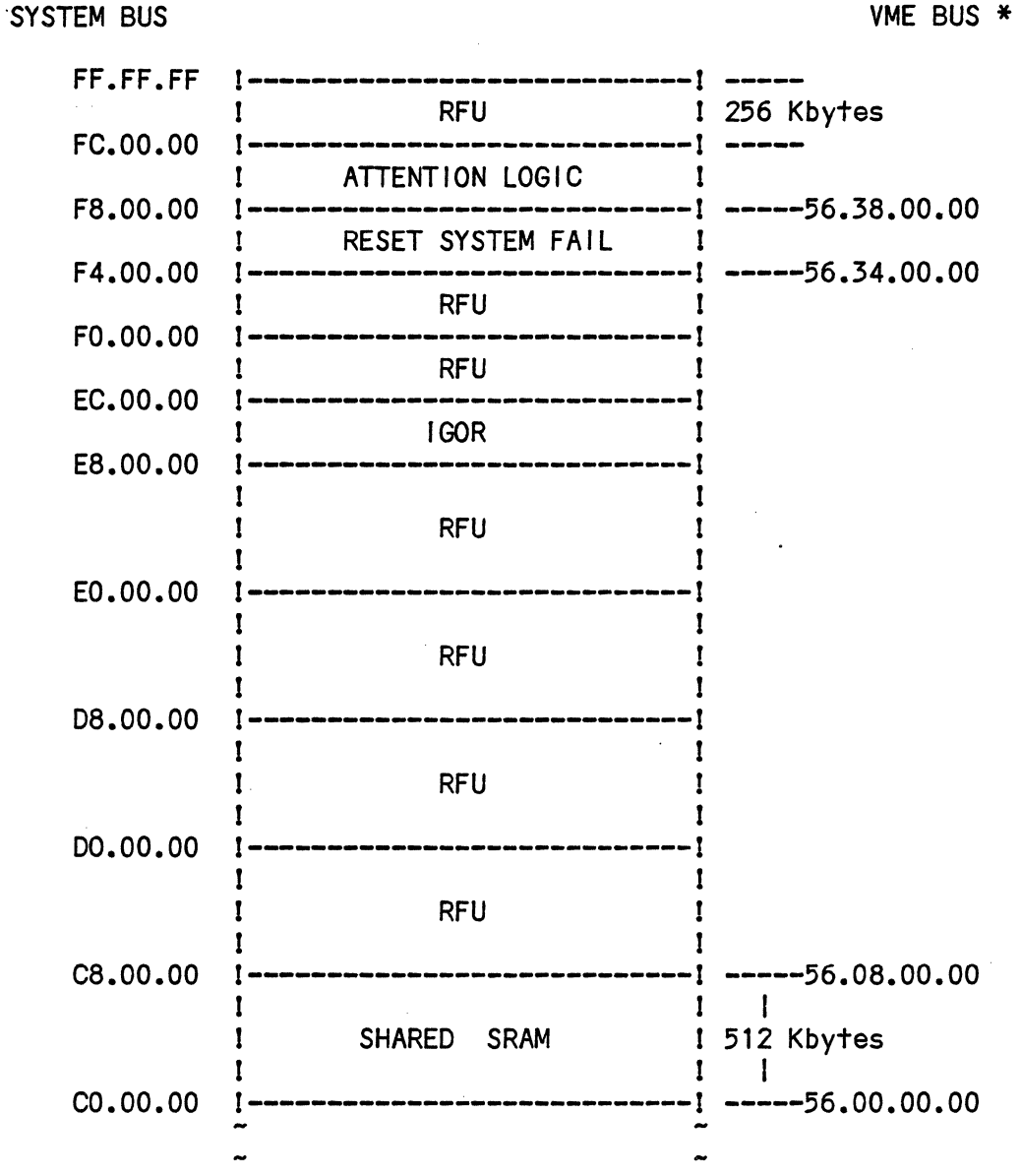
* Map for the first Line Processor with the Processor Number 0. For the others see para. 1.1.7 processor number and board type detection.


BF.FF.FF
PIT
B8.00.00
RFU
B0.00.00
PROCESSOR NUMBER REGISTER
A8.00.00
WRITE NOTIFY VECTOR REGISTER
A4.00.00
READ I/O TERM.VECT.REG
A0.00.00
RFU
98.00.00
RFU
90.00.00
RFU
88.00.00
RFU
80.00.00

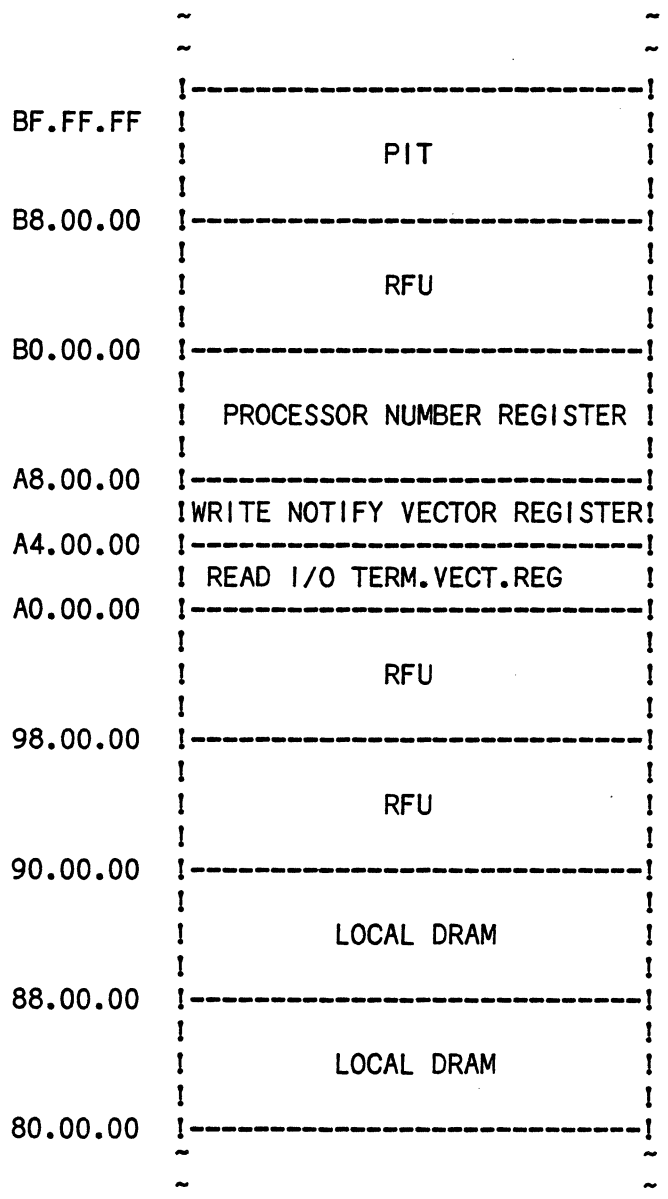

7F.FF.FF
MODEM 0 REGISTER OUT
7C.00.00
MODEM 1 REGISTER OUT
78.00.00
MODEM 0-1 REGISTER IN
74.00.00
RFU
70.00.00
WRITE I/O TERM. REG.
6C.00.00
READ NOTIFY VEC. REG.
68.00.00
RFU
60.00.00
RFU
58.00.01
SIO2 LINES 4-5
50.00.01
SIO1 LINES 2-3
48.00.01
SIO0 LINES 0-1
40.00.01

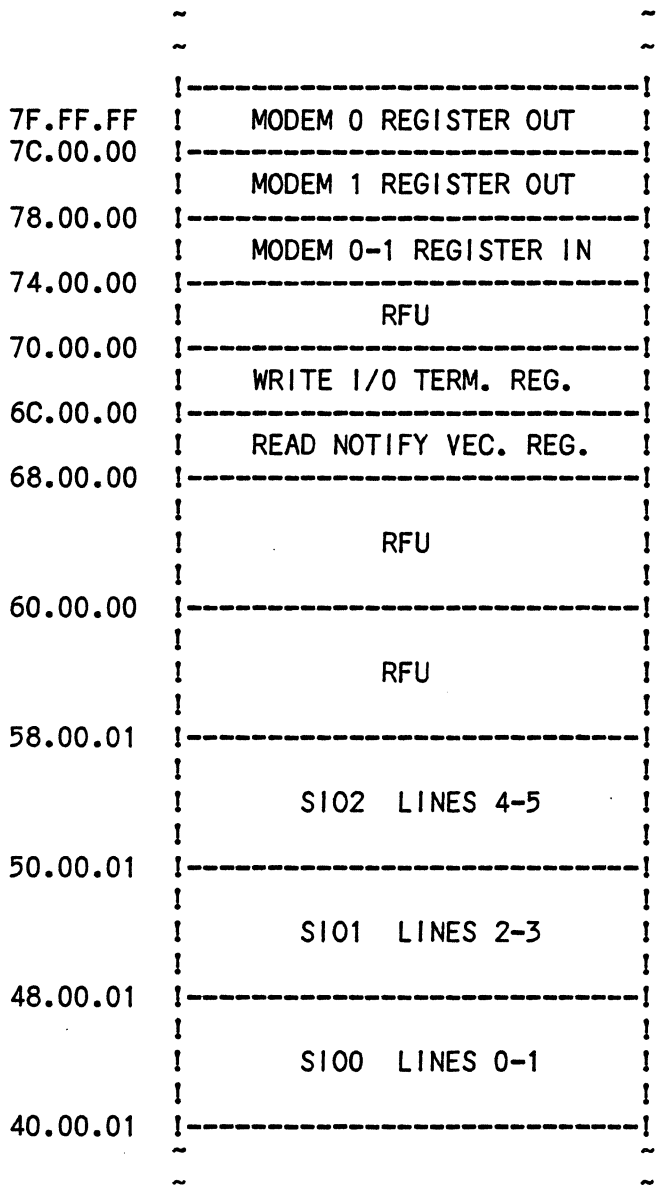

3F.FF.FF
RFU
38.00.01
RFU
30.00.00
I/O SRAM
28.00.00
RFU
20.00.00
RFU
18.00.00
RFU
10.00.00
LOCAL DRAM
08.00.00
LOCAL DRAM
512 KByte
00.00.00

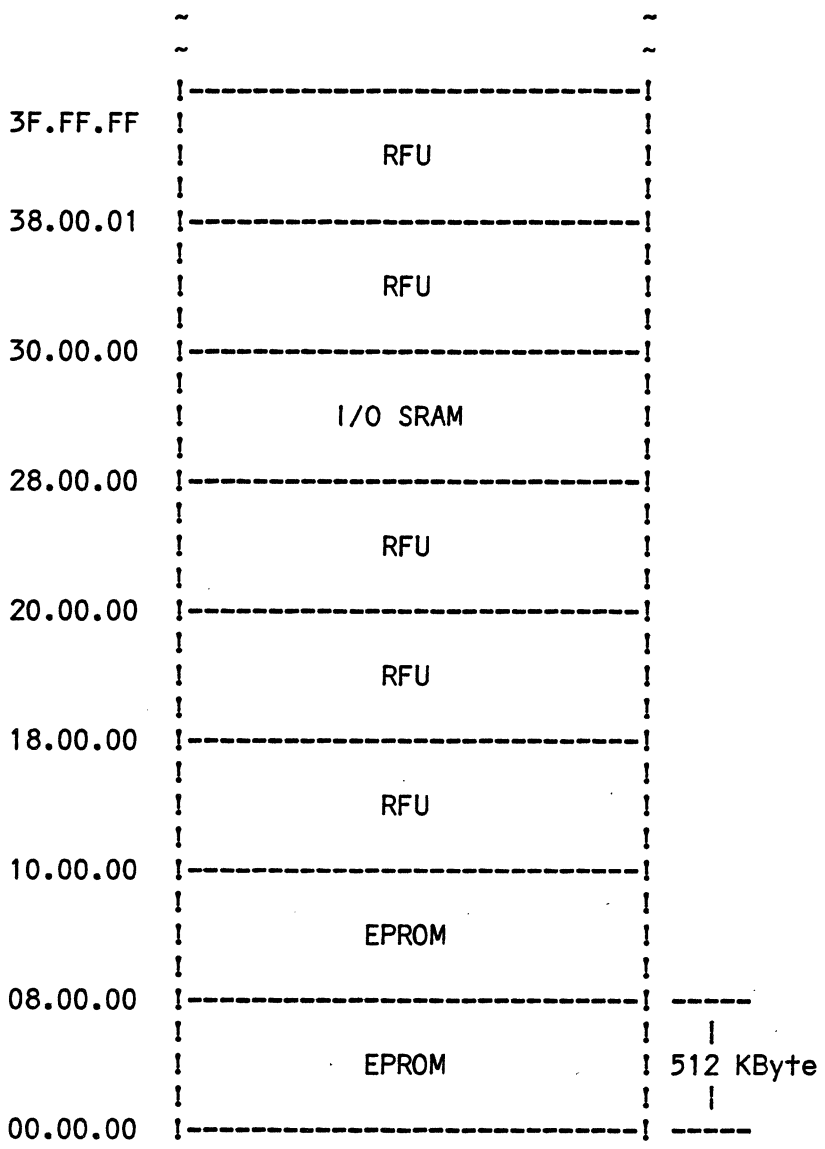
LINE PROCESSOR 0 - SLAVE PROCESSOR MEMORY MAP



* Map for the first Line Processor with the Processor Number 0. For the others see para. 1.1.7 processor number and board type detection.


BF.FF.FF
PIT
B8.00.00
RFU
B0.00.00
PROCESSOR NUMBER REGISTER
A8.00.00
WRITE NOTIFY VECTOR REGISTER
A4.00.00
READ I/O TERM.VECT.REG
A0.00.00
RFU
98.00.00
RFU
90.00.00
LOCAL DRAM
88.00.00
LOCAL DRAM
80.00.00


7F.FF.FF
MODEM 0 REGISTER OUT
7C.00.00
MODEM 1 REGISTER OUT
78.00.00
MODEM 0-1 REGISTER IN
74.00.00
RFU
70.00.00
WRITE I/O TERM. REG.
6C.00.00
READ NOTIFY VEC. REG.
68.00.00
RFU
60.00.00
RFU
58.00.01
SIO2 LINES 4-5
50.00.01
SIO1 LINES 2-3
48.00.01
SIO0 LINES 0-1
40.00.01


3F.FF.FF
RFU
38.00.01
RFU
30.00.00
I/O SRAM
28.00.00
RFU
20.00.00
RFU
18.00.00
RFU
10.00.00
EPROM
08.00.00
EPROM
00.00.00
512 KByte

- ONBOARD/VME DECODE

this logic carries out the shared-bus requests to the BUSCON when either the local processor or the system CPUs want to transfer data into shared memory, or attention register, or BIL;

- I/O MEMORY

The I/O memory consists of 2 8x8Kbytes static ram connected in such a way that it constitutes one 8Kword bank.
It is mapped in the following address range :

- 08 Kwords bank: 28.00.00 Hex --> 28.3F.FF Hex.

- EPROM

In the I/O area we find the EPROMS, two 16 Kbytes packages (Eprom 27128) have been used. These two packages are connected in such a way that they constitute one 16 Kwords bank; the required access time is 200 ns which allows the dialogues to be carried out with one wait cycle. The 8Kwords/16Kwords EPROM bank is mapped in the following address range of the slave 68000:

- 16 Kwords bank: 00.00.00 Hex --> 00.7F.FF Hex.

The Eprom code carries out the following functions:

- Exception Vector Table;
- Resident Diagnostic Routines;
- H/W initialization of the whole LP0 board;

- LOCAL MEMORY

the local memory consists of 16 256K x1 DRAM packages plus two 256K x1 DRAM packages utilized like check bit. The 256K x1 packages are connected in such that they constitute one 256 Kwords bank. This memory can be accessed at byte length (right byte and left byte) and at word length with zero or maximum one wait cycles.

The LOCAL MEMORY bank is mapped, for the slave processor, in the following address range:

80.00.00 Hex --> 87.FF.FF Hex;

On the other hand, the master processor sees the local memory in the following address range:

00.00.00 Hex --> 07.FF.FF

When an error occurs in the Local Memory, the check logic sets the 68000 in the HALT state and asserts the VME SYSFAIL signal. SYSFAIL condition can be removed under Operating System control writing or reading (byte or word length) the RESET SYSTEM FAIL flip/flop located at the following address:

F4.00.00 Hex for the system bus

56.34.00.00 Hex for the VME bus (first processor number)

The diagnostics can generate an inverted memory check bit to test the error detection circuit. For this feature the PC4 Port C bit of the Parallel Interface Timer has been used.
Writing a 0 into the PC4 bit of the PIT the inversion of the check bit will be generated. For The Port C Addressing space, refer to figure 1.1.4.3. (MC68230 Register Model).

- SERIAL INPUT OUTPUT

three USART (SIO) have been used. Each of these controllers can handle two Serial Ports (see para 1.1.3);

- PARALLEL INTERFACE TIMER (PIT)

It is a logic which permits to send single or periodic programmable interrupts to the MC68000 microprocessor and to connect the printers having an IBM and CENTRONICS type parallel interface (see para. 1.1.4). PIT is also utilized for diagnostics purpouses, notify interrupts and it is located in the local area;

- VME BUS CONTROLLER (BUSCON)

It is an interface device that assures VMEbus compatibility, allowing either the LPO microprocessor and the system CPUs to dialogue with the shared devices;

- SHARED MEMORY

the shared memory consists of eight 8Kx8 SRAM packages plus two 64Kx1 SRAM packages utilized like check bit. The 64 Kbytes packages are connected in such a way to constitute one 32 Kwords bank. This memory can be accessed at byte length (right byte and left byte) and at word length with at least four wait cycles.

The SHARED MEMORY bank is mapped in the following address range:

C0.00.00 Hex --> C0.FF.FF Hex;

When an error occurs in the Shared Memory, the check logig asserts the BUS ERROR signal to the processor that is reading the memory.

The diagnostics can generate an inverted memory check bit to test the error detection circuit. For this feature the PC4 Port C bit of the Parallel Interface Timer has been used.
Writing a 0 into the PC4 bit of the PIT the inversion of the check bit will be generated. For The Port C Addressing space, refer to figure 1.1.4.3. (MC68230 Register Model).

- VME BUS INTERRUPTER LOGIC
this logic acts as an interrupt requester on VMEbus (see para.1.1.5);

- ATTENTION LOGIC AND SYSTEM FAIL DETECTION

this logic permits the system CPUs to interrupt the LP and, in case of LP0 system fail, to recognize the system fail condition of the board (see para. 1.1.6);

- PROCESSOR NUMBER AND BOARD TYPE DETECTION

this logic allows to recognize in which slot of the VME bus the board was inserted to configure the lines.
It is also possible to know which kind of communication board was inserted (SP0 or LP0). (see para. 1.1.7).

SGM2 - LINE COMMUNICATION PROCESSOR LPØ
MAJOR BLOCK DIAGRAM

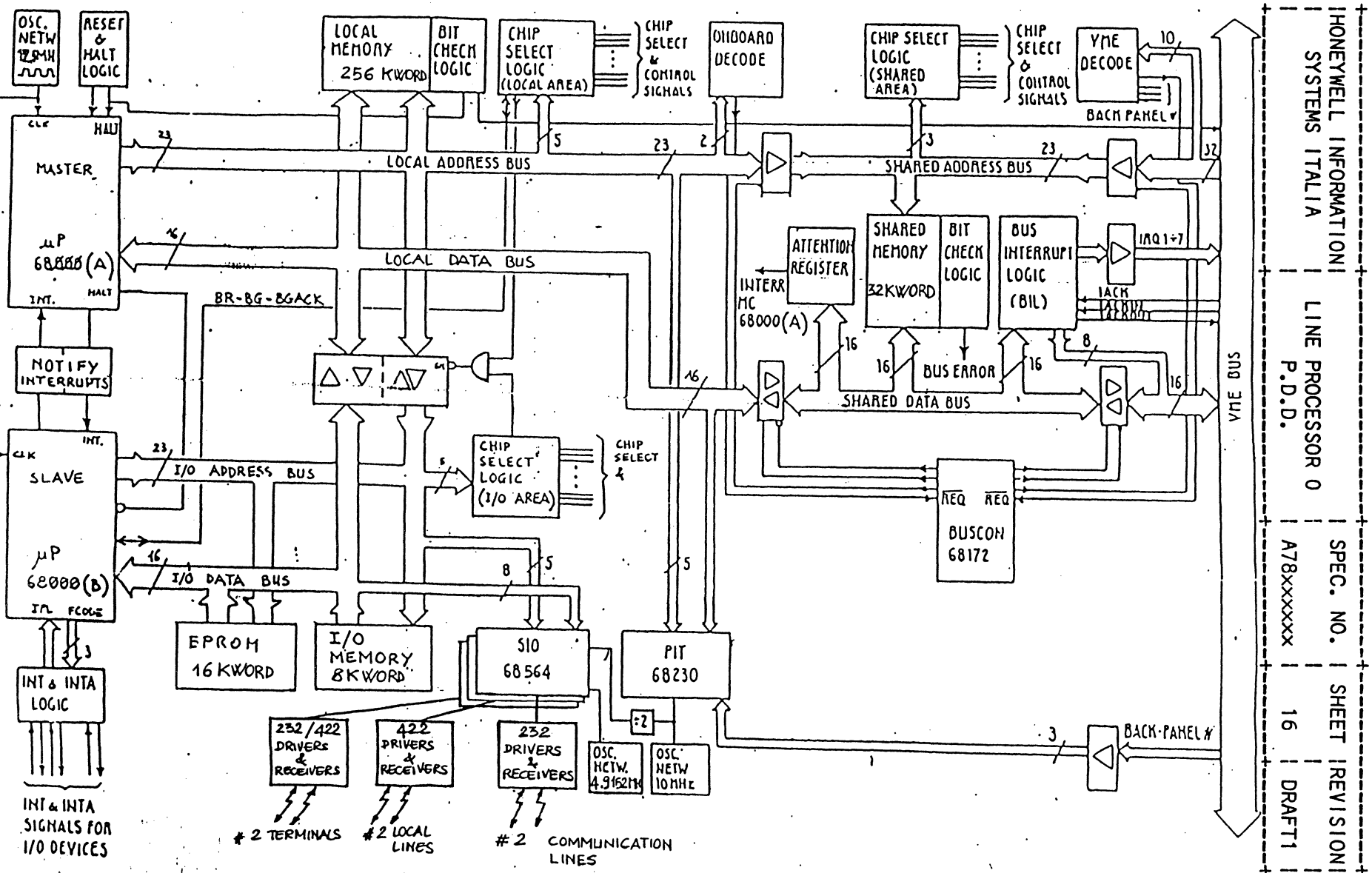


Fig. 1.1 - LPO Major Block Diagram

1.1.1 MICROPROCESSORS AND DMA EMULATOR

The LP0 uses two Motorola "MC68000 16-BIT MICROPROCESSOR" with a 12.5 Mhz frequency clock.

The main characteristics of this microprocessor are the following:

- 64 pins Microprocessor;
- 16 bits Data Bus;
- 24 bits Address Bus;
- 16 Megabytes which are directly addressable;
- I/O mapped in memory;
- 17 internal 32 bits registers;
- 32 bits Program Counter;
- 16 bits Status Register;
- 56 different types of instructions;
- operation at BITs, DIGITs, BYTEs (8 bytes) WORDs (16 bits) LONG WORDs (32 bits) length;
- two priviliged statuses: SUPERVISOR status and USER status.

The two microprocessors share the same bus and memories and can communicate each other by the NOTIFY INTERRUPT and the TERMINATION I/O INTERRUPT (see para. 1.1.2).
At the initial time the master is in HALT state until the slave initializes the LP0 board. When the slave processor finishes its job, (it prepares exception tables, it down-loads programs in the local and I/O memory from shared memory, etc.), it wakes up the master writing a logic zero in the PC2 bit of the PIT Port C. (see para.1.1.4).
Thus the master leaves its halt state and starts its routine while the slave keeps on waiting for commands from the master.
It is to be noted that, when needed, both microprocessors can have access to the all shared resources, but the access is much faster to the HW resources connected to its bus. For instance:

Master 68000 access to	LOCAL MEMORY	in 400 ns.
	I/O MEMORY	in 560 ns.
	SHARED MEMORY	in 640 ns.
Slave 68000 access to	LOCAL MEMORY	in 560 ns.
	I/O MEMORY	in 320 ns.
	SHARED MEMORY	in 800 ns.

If the MASTER processor wants to read or write the I/O memory and at the same time, the SLAVE processor wants to have an access to the LOCAL MEMORY, the MASTER processor re-runs the last instruction after the end of the slave access in the LOCAL MEMORY.

In case of TAS instruction on the I/O memory by the MASTER processor during the same condition upon described, the master will not re-run the instruction, but a BUS ERROR will happen. A software treatment of this BUS ERROR must be done.

For further details refer to "16 BIT MICROPROCESSOR DATA MANUAL" Revision June 1983-B012B of MOTOROLA INC..

1.1.2 MASTER AND SLAVE INT & INTA LOGIC

The INT & INTA LOGIC carries out the following functions:

- It receives the interrupt signals;
- It handles the interrupt priorities;
- It notifies to the MC68000 the interrupts with the highest priority by coding the three lines, IPL0, IPL1, IPL2 (Interrupt Control);
- It acknowledges and handles the Interrupt Acknowledge cycle using FC0, FC1, FC2 lines (Processor Status) and the A1, A2, A3 address signals.

In the LP0 board have been realized two interrupt & interrupt acknowledgment logics: one for the master processor, the other for the slave.

The master handles the following interrupts:

Level		Interrupt		
+ with the highest priority				
7 (NMI)		NOT USED		
6	SIOs	SIO0	SIO1 LINE#2	(Special RX Cond. (RX Data Request < TX Data Request (STS CHG Request
			LINE#3	(Special RX Cond. (RX Data Request (TX Data Request (STS CHG Request
			LINE#0	(Special RX Cond. (RX Data Request < TX Data Request (STS CHG Request
			SIO2 LINE#1	(Special RX Cond. (RX Data Request (TX Data Request (STS CHG Request
			LINE#4	(Special RX Cond. (RX Data Request < TX Data Request (STS CHG Request
			LINE#5	(Special RX Cond. (RX Data Request (TX Data Request (STS CHG Request
5		NOT USED		
4		INTERRUPT FROM DMA EMULATOR (SLAVE PROCESSOR)		
3		TIMER		
2		ATTENTION REGISTER		
1		PARALLEL PRINTER		
0		NO INTERRUPTS		
- with the lowest priority				

All interrupt are vectored.

When the SLAVE needs to communicate with the MASTER processor it must write the interrupt vector in the I/O TERMINATION REGISTER located at:

Hex 6C.00.01 for the write cycle

and

Hex A0.00.01 for the read cycle

Then writing a zero in the PC0 bit of the Pit Port C, it gives interrupt to the master at fourth level.

The right flow of the SLAVE operations is:

1) Test until the PC0 bit is set to one from the MASTER processor (NO INTERRUPT PENDING).

2) When the PC0 is one, write the interrupt vector on the I/O TERMINATION REGISTER.

3) Write a zero in the PC0 bit of the Pit Port C (INTERRUPT TO THE MASTER).

On the contrary, the slave processor handles the following interrrupts:

Level	Interrupt
+ with the highest priority	
7 (NMI)	NOT USED
6	NOT USED
5	SIOs
4	NOT USED
3	NOT USED
2	NOT USED
1	NOTIFY INTERRUPT

SIO1 / SIO0:
- LINE#2: (RXRDY DMA signal / (TXRDY DMA signal
- LINE#3: (RXRDY DMA signal / (TXRDY DMA signal
- LINE#0: (RXRDY DMA signal / (TXRDY DMA signal
- LINE#1: (RXRDY DMA signal / (TXRDY DMA signal

All the interrupts are vectored

The Sios interrupt the slave processor using the TXRDY and RXRDY pins connected to a priority encoder that gives the following priority to each line (from highest to lowest):

Line 2 Receive Ready

Line 2 Transmit Ready

Line 3 Receive Ready

Line 3 Transmit Ready

Line 0 Receive Ready

Line 0 Transmit Ready

Line 1 Receive Ready

Line 1 Transmit Ready

The interrupt vector is automatically generated by a logic that provides different interrupt vectors:

Hex 40 for Line 2 Receive Ready Interrupt

Hex 41 for Line 2 Transmit Ready Interrupt

Hex 42 for Line 3 Receive Ready Interrupt

Hex 43 for Line 3 Transmit Ready Interrupt

Hex 44 for Line 0 Receive Ready Interrupt

Hex 45 for Line 0 Transmit Ready Interrupt

Hex 46 for Line 1 Receive Ready Interrupt

Hex 47 for Line 1 Transmit Ready Interrupt

The master processor to interrupt the slave, must write the interrupt vector in the NOTIFY REGISTER located at:

Hex A4.00.01 for the write cycle

and

Hex 68.00.01 for the read cycle

Then writing a zero in the PC1 bit of the PIT Port C it interrupts the slave processor.

The right flow of the MASTER operations is:

1) Test until the PC1 bit is set to one from the SLAVE processor (NO INTERRUPT PENDING).

2) When the PC1 is one, write the interrupt vector on the NOTIFY REGISTER.

3) Write a zero in the PC1 bit of the Pit Port C (INTERRUPT TO THE SLAVE).

When an interrupt is acknowledged by the MC68000 microprocessor an "INTERRUPT ACKNOWLEDGE" cycle is performed and the INT & INTA LOGIC activates the interrupting device which responds by sending one VECTOR (byte) on the Lower Data Bus.

This vector is then latched and used by the MC68000 to select one of the 256 possible pointers of the Exception Vector Table located in Eprom for the slave microprocessor and in local DRAM for the master.

The MC68000 Microprocessor can be set at an "Interrupt Priority Level" so that the interrupts having a lower or equal priority will not serviced.

1.1.3 SERIAL INPUT OUTPUT

The LP0 serial ports allow Local and Remote Asynchronous Communications and Local and Remote Synchronous communication by means of RS-232C (V24/V28) and RS-422A (V11) electrical interface.

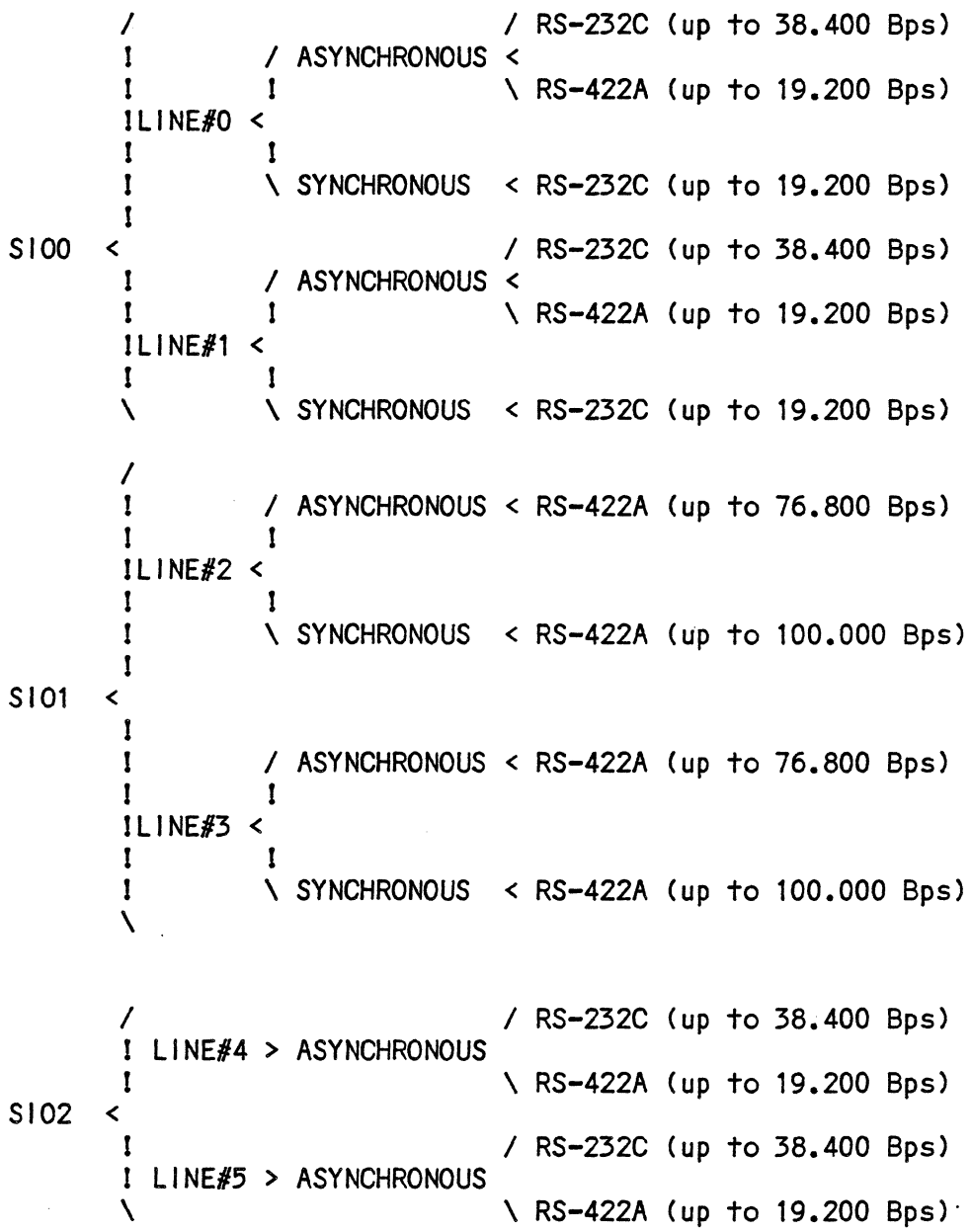
The handling of the serial ports are made by the Serial Input Output (SIO) 68564 which are LSI chips with 48 pins dual in line packages operating at 5 Mhz clock.

Their main characteristics are the following:

- compatible with MC68000;
- two independent full-duplex channels;
- directly addressable registers (all control register are read/write);
- receive data registers are quadruply buffered, transmit registers are doubly buffered;;
- Self-test capability;
- Dma pins;
- daisy chain priority interrupt logic provides automatic interrupt vectoring without external logic;
- Asynchronous features:
  * 5,6,7 or 8 bits/character
  * 1,1/2 or 2 stop bits
  * even, odd or no parity
  * x1, x16, x32 and x64 clock modes
  * break generation and detection
  * parity, overrun and framing error detection
- Byte Synchronous features:
  * internal or external character synchronization;
  * one or two sync characters in separate registers;
  * automatic sync characters insertion;
  * CRC 16 or CRC-CCITT block check generation and checking;
- Bit synchronous features:;
  * abort sequence generation and detection;
  * automatic zero insertion and detection;
  * automatic flag insertion between messages;
  * address field recognition;
  * I-field residue handling;
  * valid receive messages protected from overrun;
  * CRC-CCITT block check generation and checking;

1.1.3.1 CONFIGURATION

The configuration consists of 6 Serial Ports . These ports or lines are indicated as follows:



To the LINE#0 is assigned the role of CONSOLE.

The maximum speeds reachable by each line are:

	RS-232C ASYNC	RS-232C SYNC	RS-422A ASYNC	RS-422A SYNC
LINE 0	38.400	19.200	19.200	
LINE 1	38.400	19.200	19.200	
LINE 2			76.800	100.000
LINE 3			76.800	100.000
LINE 4	38.400		19.200	
LINE 5	38.400		19.200	

The maximum distance reachable with the RS-232C interface is 15 meters.

According to the DSA-46 that defines the standard for DTE to DCE direct connection via balanced voltage digital interface circuits, the maximum distance reachable with the RS-422A is:

BIT RATE	DISTANCE
up to 20KBps.	1200 meters
up to 40KBps.	600 meters
up to 80KBps.	300 meters
up to 100KBps.	240 meters

The choice between the electrical interfaces RS-232C and RS-422A is made using different cables for lines 0/1/4/5. The lines 2/3 have only the RS-422A interface.

The SIOs Address Summary Table is given below (all the SIOs accesses must be performed on the LOWER DATA BUS):

ADDRESS (Hex)	ABBREVIATION	# LINE	REGISTER NAME	ACCESS READ/WRITE	ACCESS READ ONLY
40.00.01	CMDREG	0	Command Register	X	
40.00.03	MODECTL	0	Mode Control Register	X	
40.00.05	INTCTL	0	Interr. Control Reg.	X	
40.00.07	SYNC 1	0	Sync Word Register 1	X	
40.00.09	SYNC 2	0	Sync Word Register 2	X	
40.00.0B	RCVCTL	0	Receiver Control Reg.	X	
40.00.0D	XMTCTL	0	Transmitter Contr.Reg	X	
40.00.0F	STAT 0	0	Status Register 0		X
40.00.11	STAT 1	0	Status Register 1		X
40.00.13	DATARG	0	Data Register	X	
40.00.15	TCREG	0	Time Constant Reg. .	X	
40.00.17	BRGCTL	0	Baud Rate Gen.Cnt.Reg	X	
40.00.19	VECTRG	0/1	Int.Vect.Reg.(Note 2)	X	
40.00.1B		0	(Note 1) .	X	
40.00.1D		0	(Note 1)	X	
40.00.1F		0	(Note 1)	X	
40.00.21	CMDREG	1	Command Register	X	
40.00.23	MODECTL	1	Mode Control Register	X	
40.00.25	INTCTL	1	Interr. Control Reg.	X	
40.00.27	SYNC 1	1	Sync Word Register 1	X	
40.00.29	SYNC 2	1	Sync Word Register 2	X	
40.00.2B	RCVCTL	1	Receiver Control Reg.	X	
40.00.2D	XMTCTL	1	Transmitter Contr.Reg	X	
40.00.2F	STAT 0	1	Status Register 0		X
40.00.31	STAT 1	1	Status Register 1		X
40.00.33	DATARG	1	Data Register	X	
40.00.35	TCREG	1	Time Constant Reg. .	X	
40.00.37	BRGCTL	1	Baud Rate Gen.Cnt.Reg	X	
40.00.39	VECTRG	0/1	Int.Vect.Reg.(Note 2)	X	
40.00.3B		1	(Note 1) .	X	
40.00.3D		1	(Note 1)	X	
40 00 3F		1	(Note 1)	X	

Notes:
1 - Not used, read as "FFH".
2 - Only one Vector Register, accessible through either channel.

ADDRESS (Hex)	ABBREVIATION	# LINE	REGISTER NAME	ACCESS READ/WRITE	ACCESS READ ONLY
48.00.01	CMDREG	2	Command Register	X	
48.00.03	MODECTL	2	Mode Control Register	X	
48.00.05	INTCTL	2	Interr. Control Reg.	X	
48.00.07	SYNC 1	2	Sync Word Register 1	X	
48.00.09	SYNC 2	2	Sync Word Register 2	X	
48.00.0B	RCVCTL	2	Receiver Control Reg.	X	
48.00.0D	XMTCTL	2	Transmitter Contr.Reg	X	
48.00.0F	STAT 0	2	Status Register 0		X
48.00.11	STAT 1	2	Status Register 1		X
48.00.13	DATARG	2	Data Register	X	
48.00.15	TCREG	2	Time Constant Reg. .	X	
48.00.17	BRGCTL	2	Baud Rate Gen.Cnt.Reg	X	
48.00.19	VECTRG	2/3	Int.Vect.Reg.(Note 2)	X	
48.00.1B		2	(Note 1) .	X	
48.00.1D		2	(Note 1)	X	
48.00.1F		2	(Note 1)	X	
48.00.21	CMDREG	3	Command Register	X	
48.00.23	MODECTL	3	Mode Control Register	X	
48.00.25	INTCTL	3	Interr. Control Reg.	X	
48.00.27	SYNC 1	3	Sync Word Register 1	X	
48.00.29	SYNC 2	3	Sync Word Register 2	X	
48.00.2B	RCVCTL	3	Receiver Control Reg.	X	
48.00.2D	XMTCTL	3	Transmitter Contr.Reg	X	
48.00.2F	STAT 0	3	Status Register 0		X
48.00.31	STAT 1	3	Status Register 1		X
48.00.33	DATARG	3	Data Register	X	
48.00.35	TCREG	3	Time Constant Reg. .	X	
48.00.37	BRGCTL	3	Baud Rate Gen.Cnt.Reg	X	
48.00.39	VECTRG	2/3	Int.Vect.Reg.(Note 2)	X	
48.00.3B		3	(Note 1) .	X	
48.00.3D		3	(Note 1)	X	
48 00 3F		3	(Note 1)	X	

Notes:
1 - Not used, read as "FFH".
2 - Only one Vector Register, accessible through either channel.

ADDRESS (Hex)	ABBREVIATION	# LINE	REGISTER NAME	ACCESS READ/WRITE	ACCESS READ ONLY
50.00.01	CMDREG	4	Command Register	X	
50.00.03	MODECTL	4	Mode Control Register	X	
50.00.05	INTCTL	4	Interr. Control Reg.	X	
50.00.07	SYNC 1	4	Sync Word Register 1	X	
50.00.09	SYNC 2	4	Sync Word Register 2	X	
50.00.0B	RCVCTL	4	Receiver Control Reg.	X	
50.00.0D	XMTCTL	4	Transmitter Contr.Reg	X	
50.00.0F	STAT 0	4	Status Register 0		X
50.00.11	STAT 1	4	Status Register 1		X
50.00.13	DATARG	4	Data Register	X	
50.00.15	TCREG	4	Time Constant Reg. .	X	
50.00.17	BRGCTL	4	Baud Rate Gen.Cnt.Reg	X	
50.00.19	VECTRG	4/5	Int.Vect.Reg.(Note 2)	X	
50.00.1B		4	(Note 1) .	X	
50.00.1D		4	(Note 1)	X	
50.00.1F		4	(Note 1)	X	
50.00.21	CMDREG	5	Command Register	X	
50.00.23	MODECTL	5	Mode Control Register	X	
50.00.25	INTCTL	5	Interr. Control Reg.	X	
50.00.27	SYNC 1	5	Sync Word Register 1	X	
50.00.29	SYNC 2	5	Sync Word Register 2	X	
50.00.2B	RCVCTL	5	Receiver Control Reg.	X	
50.00.2D	XMTCTL	5	Transmitter Contr.Reg	X	
50.00.2F	STAT 0	5	Status Register 0		X
50.00.31	STAT 1	5	Status Register 1		X
50.00.33	DATARG	5	Data Register	X	
50.00.35	TCREG	5	Time Constant Reg. .	X	
50.00.37	BRGCTL	5	Baud Rate Gen.Cnt.Reg	X	
50.00.39	VECTRG	4/5	Int.Vect.Reg.(Note 2)	X	
50.00.3B		5	(Note 1) .	X	
50.00.3D		5	(Note 1)	X	
50 00 3F		5	(Note 1)	X	

Notes:
1 - Not used, read as "FFH".
2 - Only one Vector Register, accessible through either channel.

The line Bit Rate must be specified during the SIO initialization sequence by loading an 8 bit Time Constant for every Bit Rate Generator.

The following table supplies the Time Constant values for the most frequently line speeds:

		Time con Decimal (X1 CK)	Time con Hex. (X1 CK)	Divided by (X1 CK)	Time con Decimal (X16 CK)	Time con Hex. (X16 CK)	Divided by (X16 CK)
	100000	12	0C	4	N. A.	N. A.	N. A.
	76800	16	10	4	1	1	4
BIT	38400	32	20	4	2	2	4
	19200	64	40	4	4	4	4
	9600	128	80	4	8	8	4
RATE	4800	16	10	64	16	10	4
	2400	32	20	64	32	20	4
	1200	64	40	64	4	4	64
	600	128	80	64	8	8	64
	300	255	FF	64	16	10	64

Note: N.A. means Not Applicable.

The Time Constant value given above, have been calculated in the following mode:

$$\text{OUTPUT FREQUENCY (*)} = \frac{\text{INPUT FREQUENCY}}{(\text{divided by selected}) \times (\text{Time Constant value in decimal})}$$

(*) Output Frequency of the Bit Rate Generator. Pay attention to the clock rate!

The Input Frequency is 4.9152 and the Clock Rate (x1, x16, x32, X64), is settable in the bit 6 and 7 of the MODE CONTROL REGISTER.

For further details refer to 68564 data sheet.

1.1.3.2 ELECTRICAL INTERFACE AND MODEM SIGNALS.

The Serial Ports use as interface three kinds of connectors:

LINE 0 25 pins connector with female pins;

LINE 1 25 pins connector with female pins;

LINE 2 15 pins connector with male pins;

LINE 3 15 pins connector with male pins;

LINE 4 9 pins connector with female pins;

LINE 5 9 pins connector with female pins;

The 25 pins connector includes both RS-232C and RS-422A interface signals. The table below shows the pin assignement:

Connec. Pin N.	Interface Circuit Name	Note
1	Cable Shield Ground	* $
2	Transmitted Data	*
3	Received Data	*
4	Request to Send	*
5	Clear to Send	*
6	Data Set Ready	*
7	Signal Ground	* $
8	Data Carrier Detect	*
9	Make Busy Out	*
10	Not Used	*
11	New Sync	*
12	Enable Back Static	*
13	Transmitted Data A	$
14	Transmitted Data B	$
15	Transmitted Clock	*
16	Received Data A	$
17	Received Clock	*
18	Local Loop Back	*
19	Received Data B	$
20	Data Terminal Ready	*
21	Remote Loop Back	*
22	Ring Indicator	*
23	Data Rate Selector	*
24	Transmitted Internal Clock	*
25	Test Indicator	*

* RS-232C Interface
$ RS-422A Interface

The ENABLE BACK STATIC signal, when in the low level, allows to receive the external clock.

THe 15 pins connector includes only the RS-422A interface signals:

Connec. Pin N.	Interface Circuit Name	Note
1	Cable Shield Ground	$
2	Transmitted Data A	$
3	Not Used	$
4	Received Data A	$
5	Not Used	$
6	Received Clock A	$
7	Transmitted Clock A	$
8	Signal Ground	$
9	Transmitted Data B	$
10	Not Used	$
11	Received Data B	$
12	Not Used	$
13	Received Clock B	$
14	Transmitted Clock B	$
15	Not Used	$

The 9 pins connector includes both the RS-232C and RS-422A interface signals. The table given below shows the pin assignment:

Connec. Pin N.	Interface Circuit Name	Note
1	Cable Shield Ground	* $
2	Transmitted Data	*
3	Received Data	*
4	Receive Data A	$
5	Receive Data B	$
6	Not Used	*
7	Signal Ground	* $
8	Transmission Data A	$
9	Transmission Data B	$

* RS-232C Interface
$ RS-422A Interface

The interface signals are driven and received using the following standard packages:

RS-232C		RS-422A	
+ 1488	(Driver)	+ 3487	(Driver)
+ 75154	(Receiver)	+ 3486	(Receiver)

On the RS-232C and RS-422A interfaces there is a network which forces the SPACE condition on the Receive Data signal of the SIO when the cable is disconnected or the terminal is off.

1.1.3.3 MODEM SIGNALS

Many modem signals are provided by the SIO, but not all. Thus there are three registers, two for the output, one for the input, which allow the connection of the two RS-232 lines.

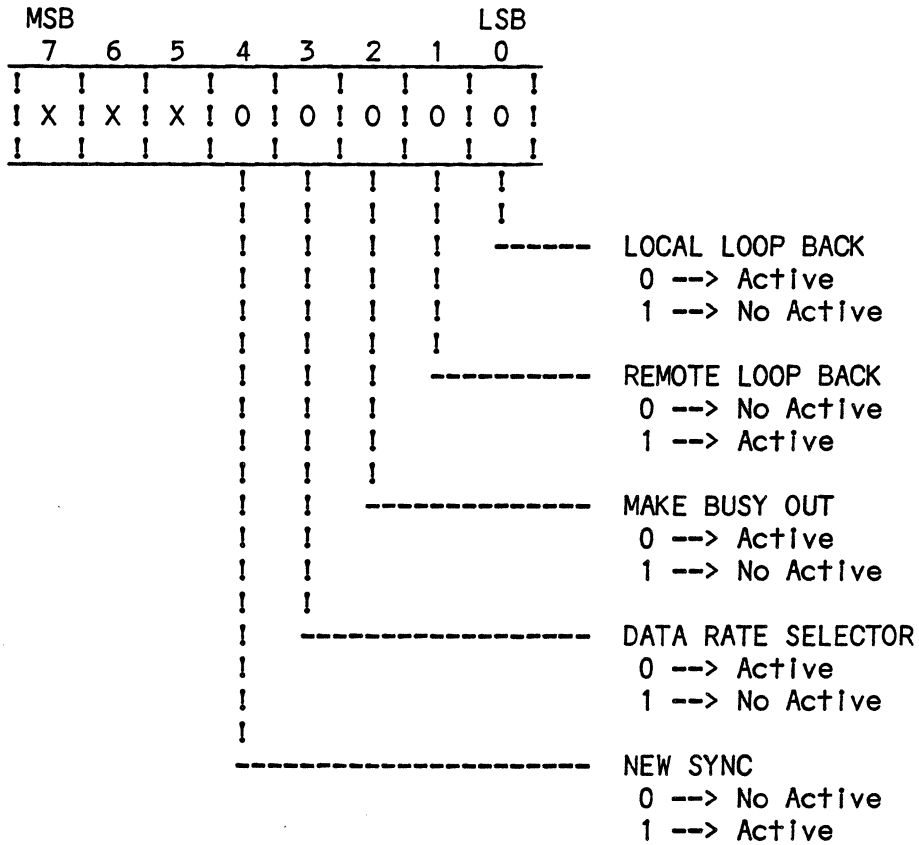
MODEM REGISTER LINE 0 OUTPUT is located at:

Hex 7C.00.01

MODEM REGISTER LINE 1 OUTPUT is located at:

Hex 78.00.01

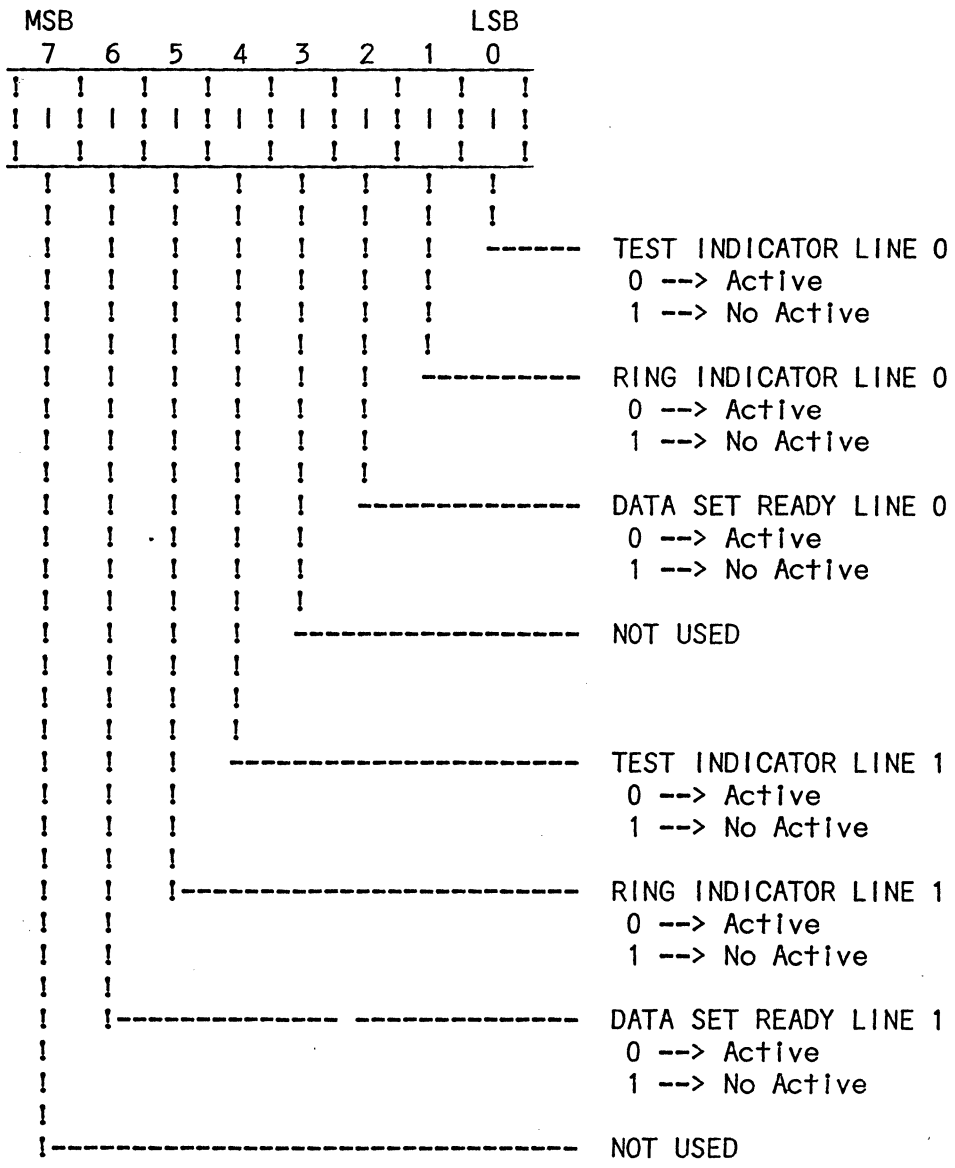
and they bring this signal:



MODEM REGISTER LINE 0/1 INPUT is located at:

Hex 74.00.01

and it brings this signal:



1.1.4 TIMER and PARALLEL PRINTER INTERFACE

This hardware block consists essentially of Motorola MC68230 PI/T chip, which provides a programmable timer plus a versatile double buffered parallel interfaces.

a) TIMER - The PI/T timer contains a 24-bit synchronous down counter that is loaded from three 8-bit Counter Preload Registers. The 24-bit counter must be clocked by the output of a 5-bit (divided-by-32) prescaler to generate periodic interrupts, a square wave, a single interrupt after a programmed time period, or it can be used for elapsed time misurement.
Also, the end of count can be checked by software without interrupt use.

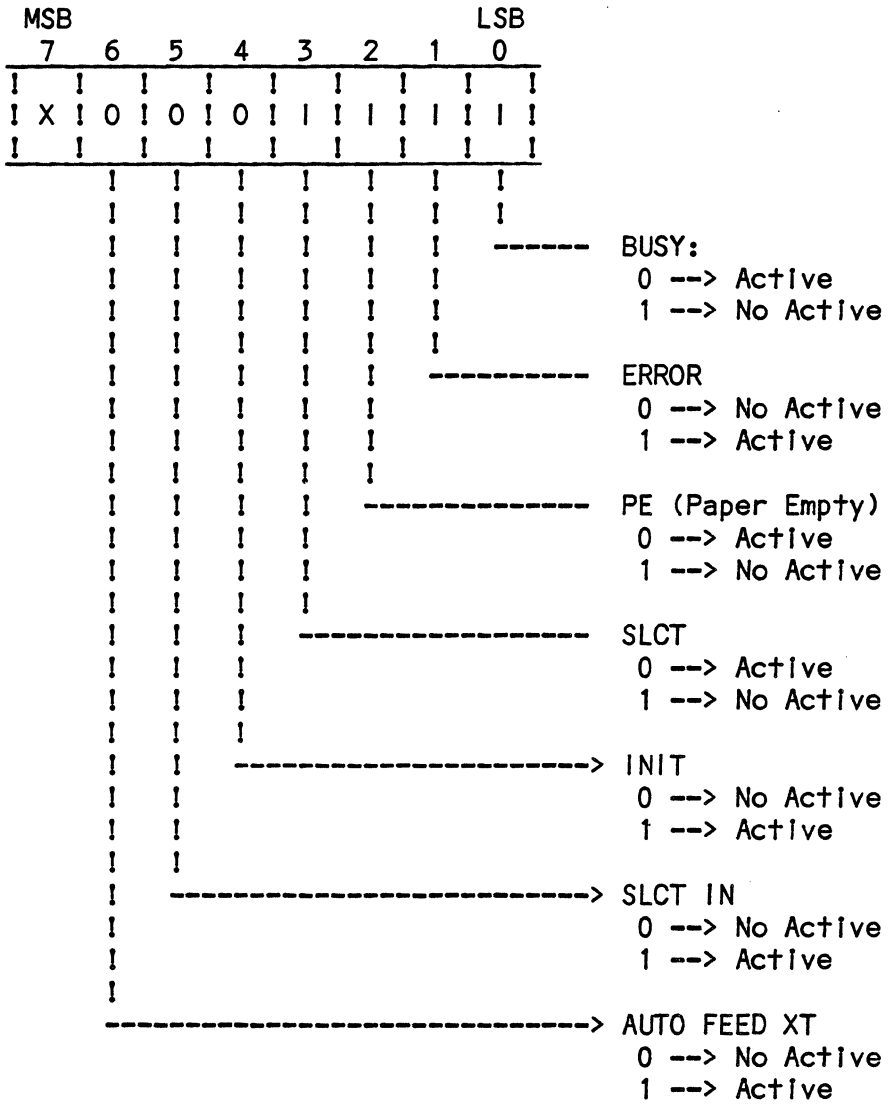
A register model that includes the corresponding Register Selects is shown in Fig. 1.1.4.3. For further information refer to "16-BIT MICROPROCESSOR DATA MANUAL - 1983 MOTOROLA" pag. 4-509 - 4-537.

b) PARALLEL PRINTER INTERFACE - The Parallel Printer Interface allows the connection to printers with electric parallel interface both of the IBM and CENTRONICS types. Fig. 1.1.4.1 shows the major block diagram.

This interface is implemented by mean a programmable parallel interface (MC68230) plus some DRIVERs and RECEIVERs of the Low Power Schottky type. The dialogue with the printer must be performed programming the MC68230 in the following mode:

- Port A must be set with Mode 0 and submode 01;
- Port B must be set with Mode 0 and submode 1X;
- all pins of the Port A must be programmed in output mode to drive the printer data;
- some pins of the Port B must be programmed in output mode to drive printer command and some ones in input mode to receive printer status, as shown below:

PI/T PORT B (bit Input/Output)



- handshake pin H4 must be used as an edge-sensitive status input pin to produce an interrupt to MC68000 when there is the trailing edge of the ACKNLG signal with the meaning of character request. The character should be sent only if the printer is ready (not busy) and no error condition is present.

N.B. - The write of the data register (Port A) involve a hardware generation of the data STROBE signal timing without software management necessity (see Fig. 1.1.4.2 which shows the timings relevant to CENTRONICS and IBM interfaces).

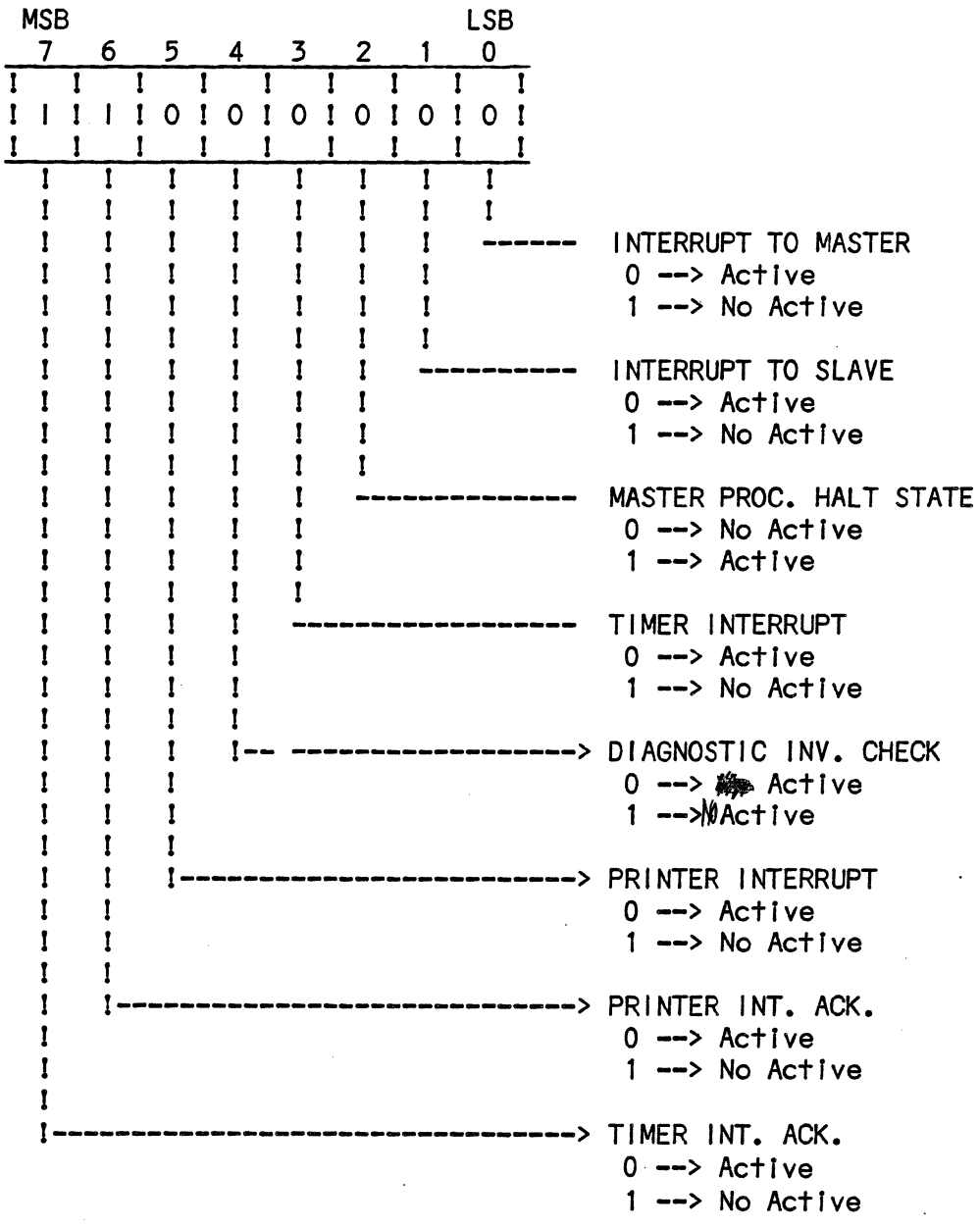
A register model that includes the corresponding Register Selects is shown in Fig. 1.1.4.3. For further information refer to "16-BIT MICROPROCESSOR DATA MANUAL - 1983 MOTOROLA" pag. 4-509 - 4-537.

The I/O signals are made available via a CANNON connector having 37 paths with female pins: the connector pin out and a detailed description of the interface signals are shown in table 1.1.4.1.

The LP0 board connects the printer unit via a signal cable (with twisted-pair and shield) 5 meters max in lenght with a 37 pin connector at the board unit end, and a 36 pin connector on the printer end.

The bits of the Port C are used in this way:

PI/T PORT C (bit Input/Output)



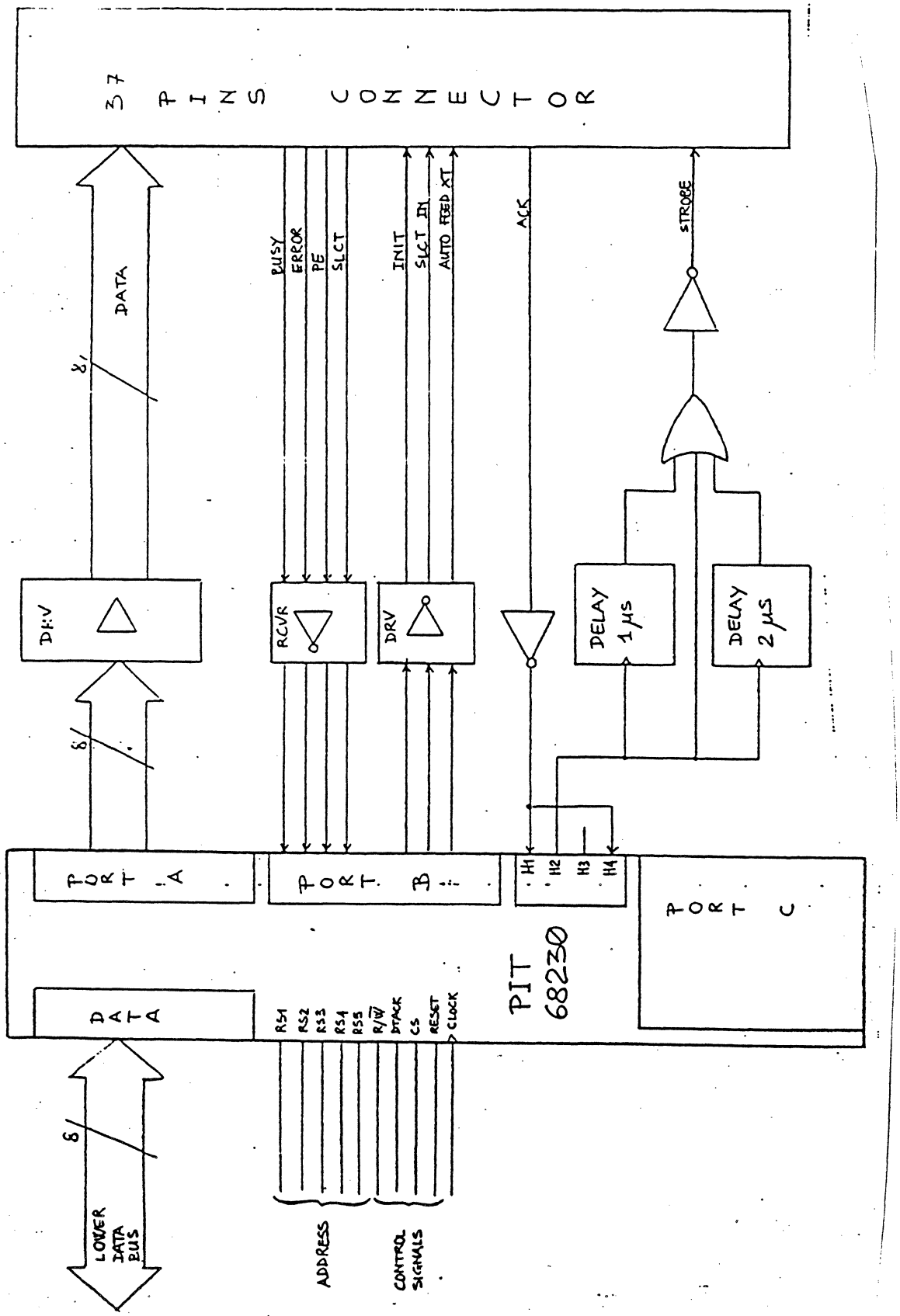
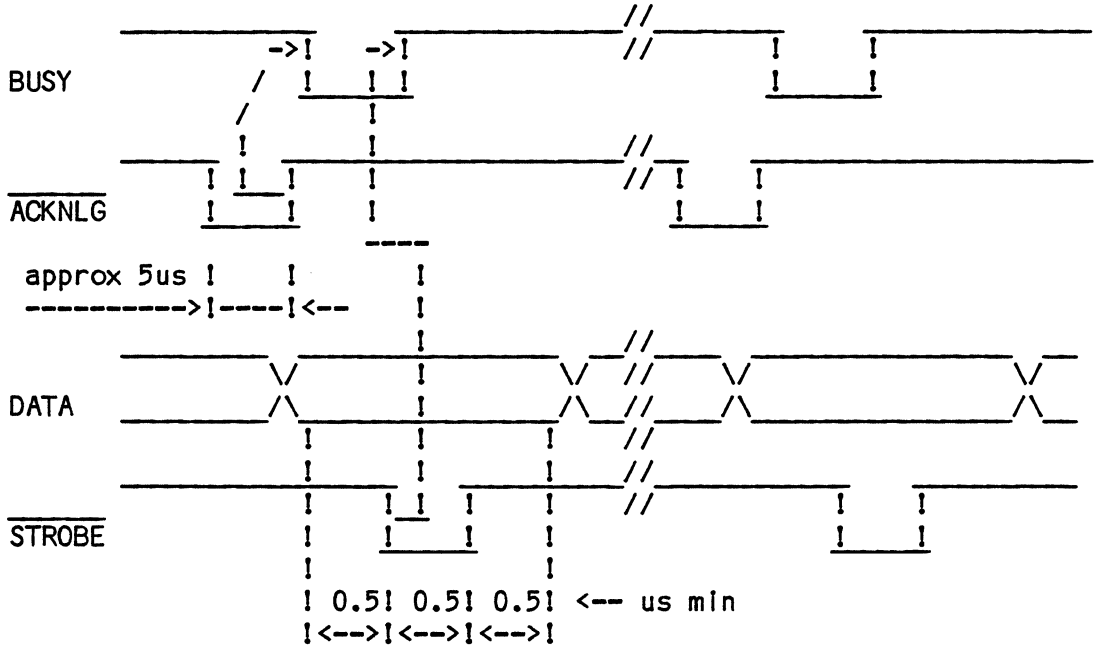
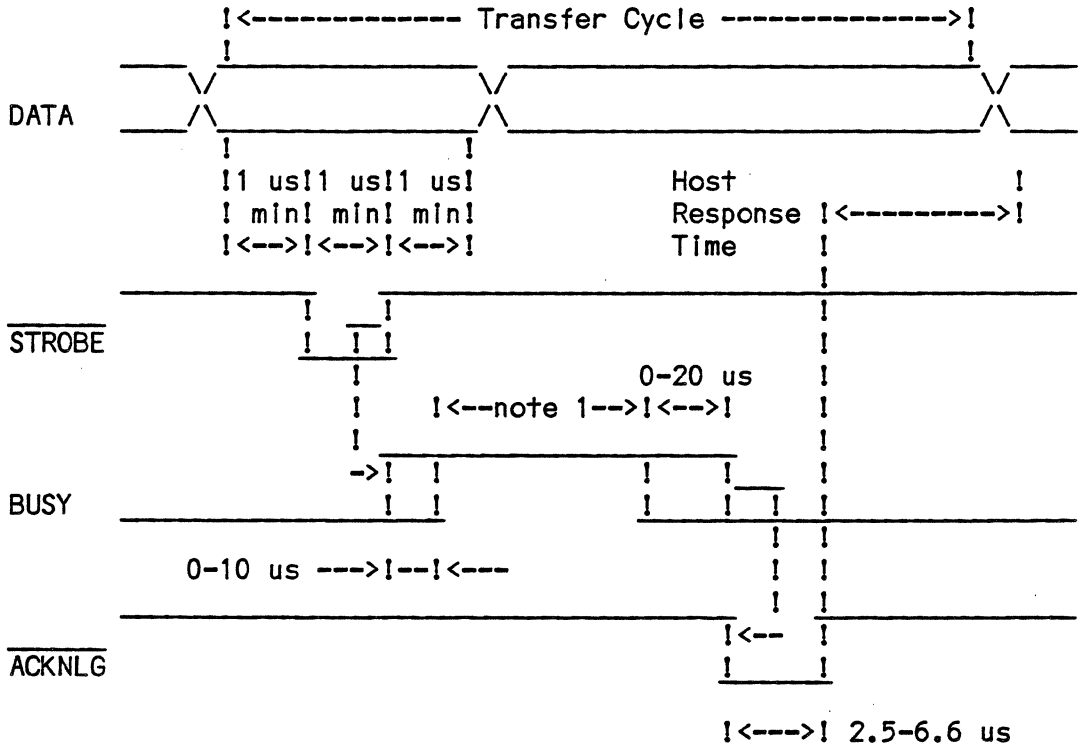


Fig. 1.1.4.1 Parallel Printer Interface Block Diagram



a) IBM Interface Timing



Note 1: Max duration is a function of the required operation.

b) CENTRONICS Interface Timing

Fig. 1.1.4.2 Parallel Printer Interface Timing Diagram

ADDRESS (Hex)	7	6	5	4	3	2	1	0	
B8.00.01	Port Mode Control		H34 Enable	H12 Enable	H4 Sense	H3 Sense	H2 Sense	H1 Sense	Port General Control Register
B8.00.03	*	SVCRQ Select		Interrupt FFS		Port Interrupt Priority Control			Port Service Request Register
B8.00.05	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Port A Data Direction Register
B8.00.07	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Port B Data Direction Register
B8.00.09	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Port C Data Direction Register
B8.00.0B	Interrupt Vector Number						*	*	Port Interrupt Vector Register
B8.00.0D	Port A Submode		H2 Control			H2 Int Enable	H1 SVCRQ Enable	H1 Stat Ctrl.	Port A Control Register
B8.00.0F	Port B Submode		H4 Control			H4 Int Enable	H3 SVCRQ Enable	H3 Stat Ctrl.	Port B Control Register
B8.00.11	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Port A Data Register
B8.00.13	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Port B Data Register
B8.00.15	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Port A Alternate Register
B8.00.17	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Port B Alternate Register
B8.00.19	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Port C Data Register
B8.00.1B	H4 Level	H3 Level	H2 Level	H1 Level	H4S	H3S	H2S	H1S	Port Status Register
B8.00.1D	*	*	*	*	*	*	*	*	(null)
B8.00.1F	*	*	*	*	*	*	*	*	(null)
B8.00.21	TOUT/TIACK Control			Z D Ctrl.	*	Clock Control		Timer Enable	Timer Control Register
B8.00.23	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Timer Interrupt Vector Register
B8.00.25	*	*	*	*	*	*	*	*	(null)
B8.00.27	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Counter Preload Register (High)
B8.00.29	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	(Mid)
B8.00.2B	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	(Low)
B8.00.2D	*	*	*	*	*	*	*	*	(null)
B8.00.2F	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Count Register (High)
B8.00.31	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	(Mid)
B8.00.33	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	(Low)
B8.00.35	*	*	*	*	*	*	*	ZDS	Timer Status Register
B8.00.37	*	*	*	*	*	*	*	*	(null)
B8.00.39	*	*	*	*	*	*	*	*	(null)
B8.00.3B	*	*	*	*	*	*	*	*	(null)
B8.00.3D	*	*	*	*	*	*	*	*	(null)
B8.00.3F	*	*	*	*	*	*	*	*	(null)

(*) - Unused, read as zero.

Fig. 1.1.4.3 MC68230 Register Model

Table 1.1.4.1 Parallel-Printer Connector Pin Assignment and Interface Signal Description

Signal Pin N.	Signal Name	Source	Description
1	$\overline{\text{STROBE}}$	HOST	Data sampling strobe: it clocks data lines into the printer interface logic. The signal level is normally HIGH; write-out of data is performed at the LOW level of this signal. STROBE pulse width requirements are shown in Fig. 1.9.5.2 for IBM and CENTRONICS interfaces.
2	DATA0	HOST	* Least Significant Bit (2^0)
3	DATA1	HOST	These signals represent the character to be printed or the control code to be executed by the printer: normally, these informations are given in ASCII code. Each signal is at HIGH level when data is logical "1" and LOW when logical "0". Data Set-up and Data Hold Times requirements are shown in Fig. 1.9.5.2 for IBM and CENTRONICS interfaces.
4	DATA2	HOST	
5	DATA3	HOST	
6	DATA4	HOST	
7	DATA5	HOST	
8	DATA6	HOST	
9	DATA7	HOST	* Most Significant Bit (2^7)
10	$\overline{\text{ACKNLG}}$	PRINTER	Active LOW pulse. It indicates that the data has been loaded into the buffer or the command has been executed and that the printer is ready to accept other data. In Fig. 1.9.5.2 are shown timing consideration for IBM and CENTRONICS interfaces.

Table 1.1.4.1 Parallel-Printer Connector Pin Assignment and Interface Signal Description (Cont'd.)

Signal Pin N.	Signal Name	Source	Description
11	BUSY	PRINTER	Active HIGH level. It indicates the printer is not ready to accept any data or control code. It is high in the following cases: - During data entry; - During printing operation; - When the printer is in OFF-LINE (or LOCAL) state; and, only for CENTRONICS interface, in these other cases: - As long as the $\overline{\text{INIT}}$ signal is LOW; - When the printer is in STAND-BY status; in this case it will anyway accept XON and DEL codes; and, only for IBM interface, in this other case: - During printer error status. Timing considerations about BUSY signal are shown in Fig. 1.9.5.2 for both interfaces.
12	PE	PRINTER	PE (Paper Empty) is active at HIGH level; it indicates that the printer is out of paper.
13	SLCT	PRINTER	SLCT (Select) signal is active at HIGH level; it indicates that the printer is in the selected state, i.e. is in READY state.

Table 1.1.4.1 Parallel-Printer Connector Pin Assignment and Interface Signal Description (Cont'd.)

Signal Pin N.	Signal Name	Source	Description
14	$\overline{\text{AUTO FEED XT}}$	HOST	This signal is applicable only for IBM interface. When this signal is driven at LOW level, the paper is automatically fed one line after printing.
32	$\overline{\text{INIT}}$	HOST	For the CENTRONICS interfaces this signal is named $\overline{\text{PRIME}}$ and a LOW level on this line causes the output signal BUSY to go high for as long as the $\overline{\text{INIT}}$ signal is low. For the IBM interface, when the level of this signal becomes LOW the printer is reset to its initial state and the printer buffer is cleared. This signal is normally at HIGH level, and its pulse width must be more than 50 us at the receiving terminal.
33	$\overline{\text{ERROR}}$	PRINTER	This line is named $\overline{\text{FAULT}}$ for the CENTRONICS interface. The level of this signal becomes LOW when the printer is in: - PAPER END state; - OFF LINE state; - Error state.
37	$\overline{\text{SLCT IN}}$	HOST	This signal is applicable only for IBM interface. Data entry to the printer is possible only when the level of this signal is LOW.

Table 1.1.4.1 Parallel-Printer Connector Pin Assignment and Interface Signal Description (Cont'd.)

Signal Pin N.	Signal Name	Source	Description
16-19 20-21 22-23 24-25 26-27 28-29 30-31 34	GND	---	Logic GND level.
15-18 35-36	R.F.U.	---	Pins not used.
17	ZVP00	---	Safety Ground.

1.1.5 VME BUS INTERRUPTER LOGIC (BIL)

The VME Bus Interrupter Logic allows to generate interrupts to the VME bus on any of the seven levels and supplies an 8-bit vector during interrupt acknowledge cycle. Besides it handles the daisy-chain configuration.

The heart of this logic is the SIGNETICS 68154 Interrupt Generator (IGOR), that provides this interface between an interrupting device and the VME Bus.

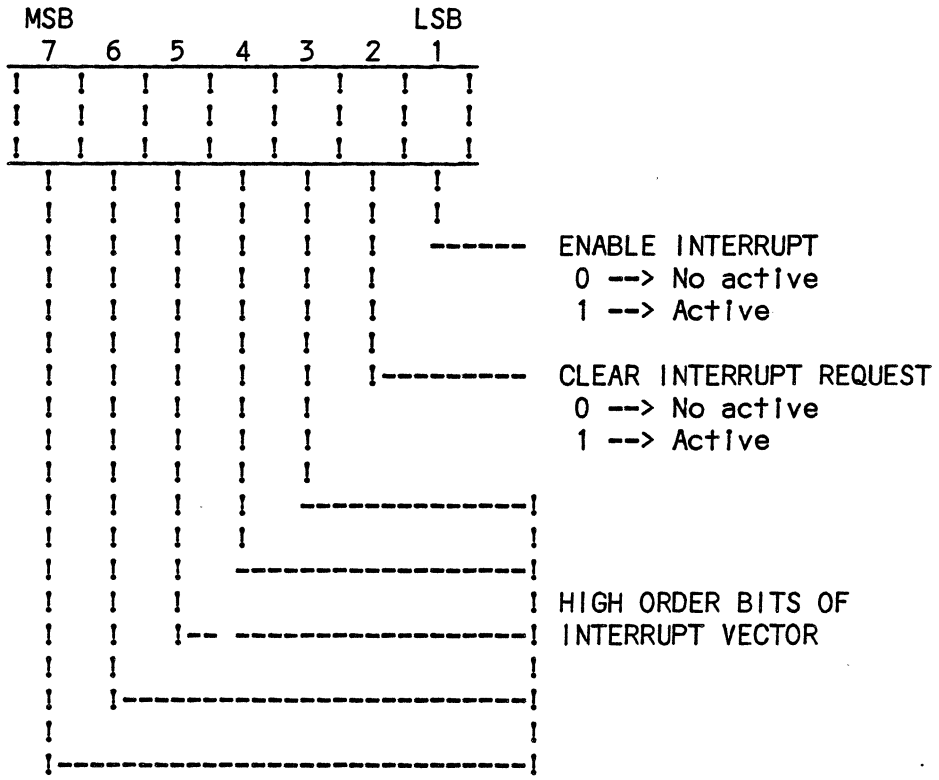
Inside it has two registers:

- the INTERRUPT VECTOR REGISTER R0 (located at Hex E8.00.01)

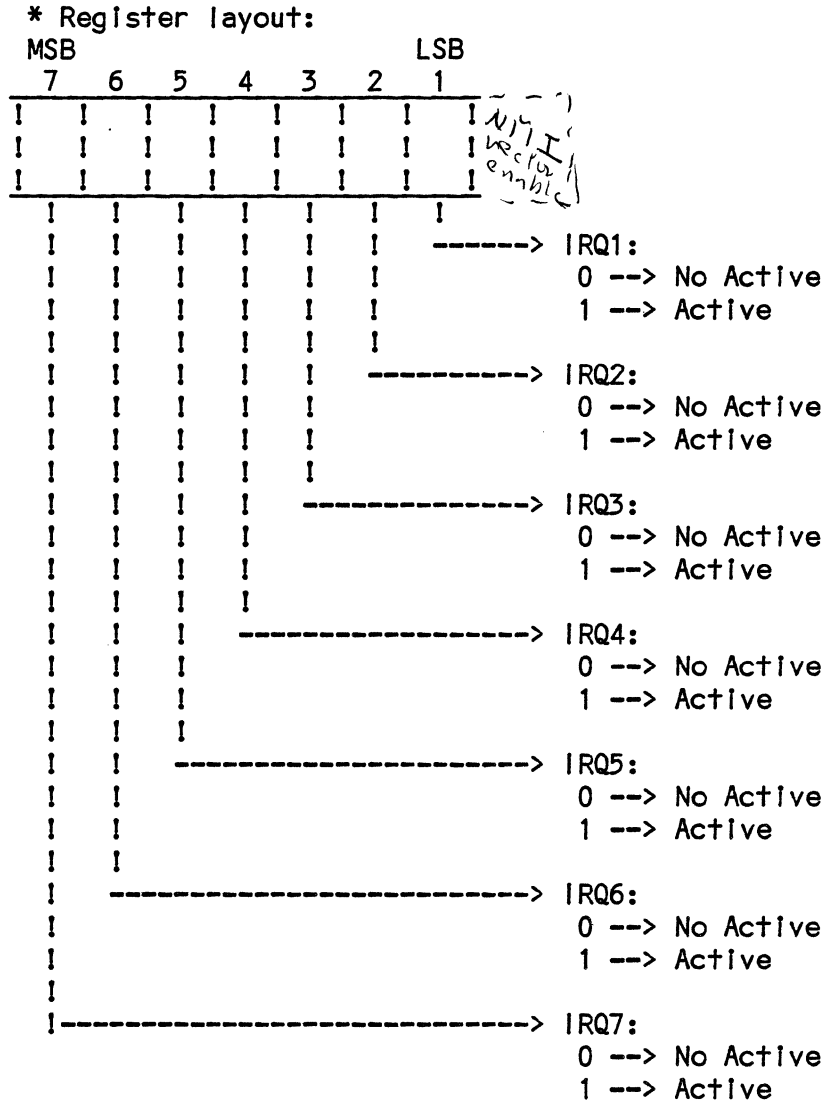
- the INTERRUPT REQUEST REGISTER R1 (located at Hex E8.00.05)

The local processor writes the interrupt vector register to enable interrupts and to program the high order bits of the interrupt vector on any interrupt request level of the VME Bus.

The R0 register has this layout:



The R1 register has this layout:



The 7/6/5/4/3 bits of the interrupt vector register plus the A3 A2 A1 address bits of the VME Bus form the interrupt vector register passed during an interrupt acknowledge cycle.

Writing a 1 to bit 2 of the R0 register are reset all interrupt levels in the interrupt request register. Bit 2 will always be read as 0.

Setting bit 1 of R0 all interrupt levels are enabled.

Any Number (up to seven) of interrupt request can be generated in single access of R1 but they are not stackable on the same level. To generate another interrupt request on a level currently asserted, the user must wait until that level has been acknowledge. During an interrupt acknowledge cycle the corresponding bit of the interrupt level will automatically cleared by the device.

Refers to the Signetics SCB68154 data sheet on Signetics book (January 1986 pages 2-358 2-368).

1.1.6 ATTENTION LOGIC AND SYSTEM FAIL DETECTION

The Attention Logic allows to generate interrupts to the MC68000 from System's CPU and supplies an 8-bit vector, loaded by the interrupting device, during interrupt acknowledge cycle.
Below are shown the registers layouts and access mode of this logic:

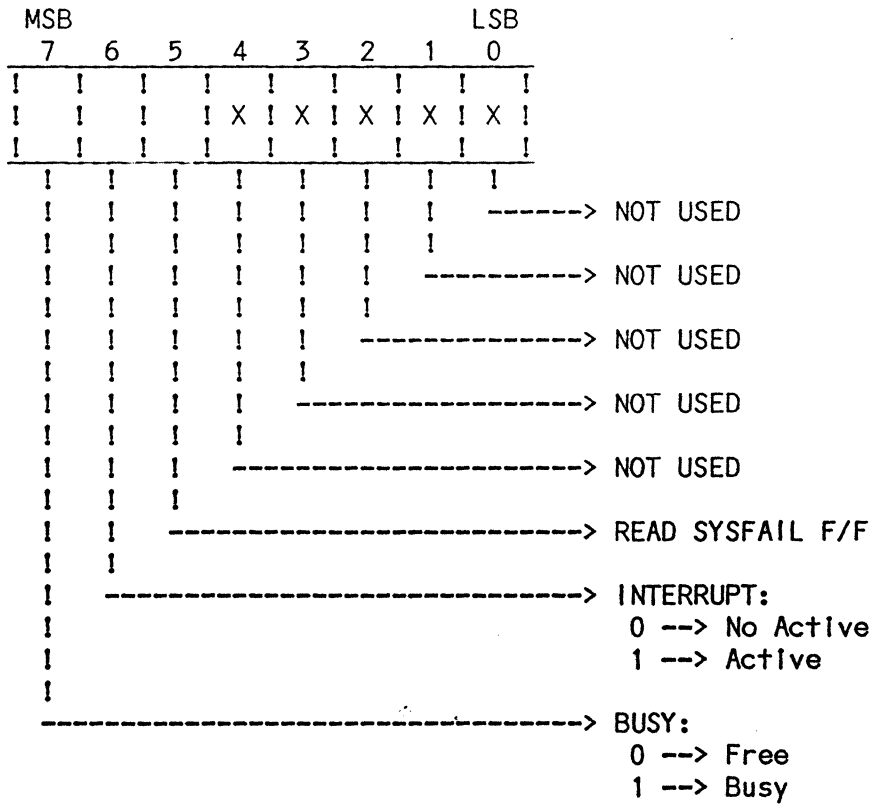
- ATTENTION INTERRUPT REGISTER:

  * Write/Read 8 bit register.

  * Access mode:

    ADDRESS: F8.00.00 Hex.
    Read/Write UPPER DATA BYTE.
    TAS.

  * Register layout:



- ATTENTION VECTOR REGISTER

  * Write/Read 8 bit register.

  * Access mode:

    ADDRESS: F8.00.01 Hex;
    Write/Read LOWER DATA BYTE.

  * Register layout:

MSB 7	6	5	4	3	2	1	LSB 0
!	!	!	!	!	!	!	!
!	!	!	!	!	!	!	!
!	!	!	!	!	!	!	!

The informations contained in these registers can be also supplied at word lenght.

It is possible to give only one interrupt at a time to the 68000 LP0 CPU, setting the bit SIX after having checked with the TAS instruction, the 7 bit (BUSY) of the Interrupt Register that no one else interrupt is active.

The reset of the Interrupt Register is made by software control accessing to the interrupt register.

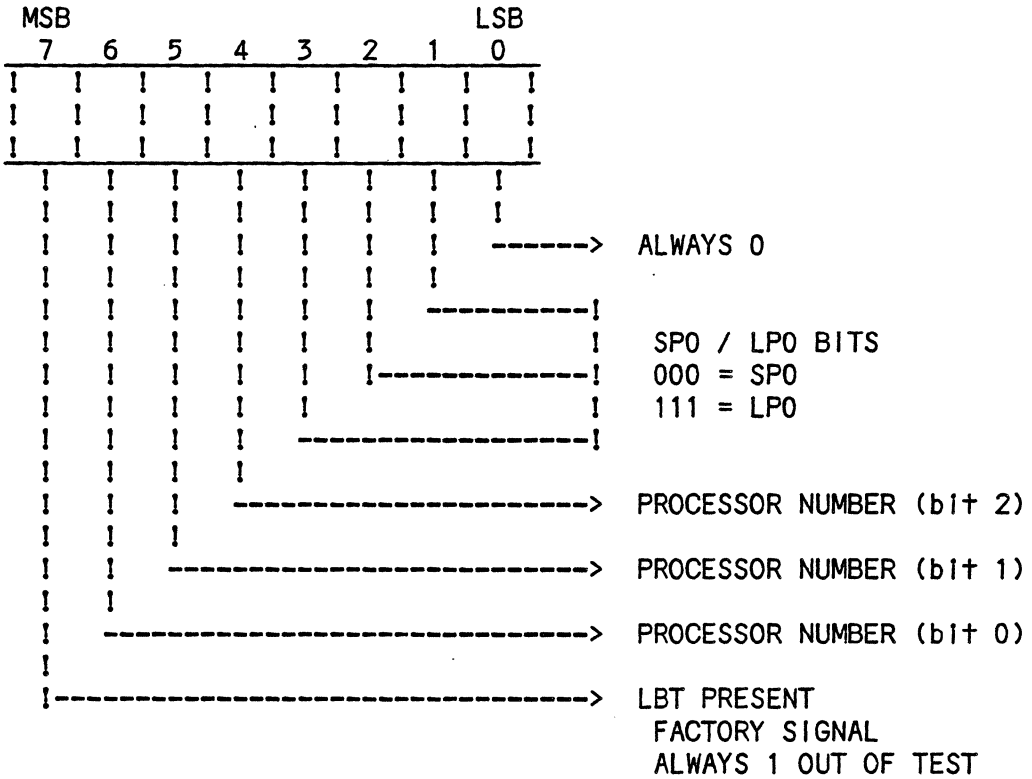
The system fail condition is detectable in the bit 5 of the ATTENTION INTERRUPT REGISTER. The software running in the SGM2 CPU can know which LP0 board in the VME Bus is in the system fail condition.

1.1.7 PROCESSOR NUMBER AND BOARD TYPE DETECTION

There is a register loacated at Hex A8.00.00 where is possible to know some information about the board:

- in which slot of the SGM2 VME Bus the LP0 board was inserted;
- the type of the board inserted (SP0 or LP0);
- testing enviroment;

The layout of this register is:



55.E8.00.00

The table below describes the VME bus addressing space of each board with the processor number bits equal to:

6	5	4	
0	0	0	PROCESSOR NUMBER 0 - LP0 located at Hex 56.00.00.00 (VME BUS ADDRESSING SPACE)
1	0	0	PROCESSOR NUMBER 1 - LP0 located at Hex 56.40.00.00 (VME BUS ADDRESSING SPACE)
0	1	0	PROCESSOR NUMBER 2 - LP0 located at Hex 56.80.00.00 (VME BUS ADDRESSING SPACE)
1	1	0	PROCESSOR NUMBER 3 - LP0 located at Hex 56.C0.00.00 (VME BUS ADDRESSING SPACE)
0	0	1	PROCESSOR NUMBER 4 - LP0 located at Hex 57.00.00.00 (VME BUS ADDRESSING SPACE)
1	0	1	PROCESSOR NUMBER 5 - LP0 located at Hex 57.40.00.00 (VME BUS ADDRESSING SPACE)
0	1	1	PROCESSOR NUMBER 6 - LP0 located at Hex 57.80.00.00 (VME BUS ADDRESSING SPACE)
1	1	1	PROCESSOR NUMBER 7 - LP0 located at Hex 57.C0.00.00 (VME BUS ADDRESSING SPACE)

2 DMA EMULATOR INTERFACE

The interface model of the DMA is located in the I/O memory and consists of the following four entries:

The first is the COMMAND LIST:

- buffer address (4 bytes): BFADR - set by MASTER
- actual data lenght (2 bytes):ADLL - RX: set by DMA
  TX: set by MASTER
- buffer size (2 bytes) : BFSZ - set by MASTER
- FU (1 byte)
- command list flag (1 byte): CLFLG

  bit(0) = 1 intermediate command list
  = 0 last command list
  set by MASTER

  bit(1) = 1 command list already executed
  set by DMA after INT..
  reset by MASTER

The second part is the RX/TX COMMON AREA and consists:

- driver type (1 byte): DRTYP

  value:
  VIP Hex 04
  BSC Hex 08
  SNA Hex 09
  X25 Hex 0A

- device status (1 byte): DVSTUS

  value:
  opened Hex 01
  closed Hex 00
  set by MASTER

- mode control register (1 byte): MDCTL
  bit(0) = 1 COMPLEX MODE

  bit(1) = 1 ONE SHOT MODE

These flag are set by MASTER at the open time; these status is kept until the close is requested.

The third section is the RX service area:

- Rx control table address (4 bytes): RXCTLA
  set by MASTER

- FU (1 byte)

- Rx command (1 byte): RXCMND

  bit(0) = 1 start RX DMA
  set by MASTER

  bit(7) = 1 command request
  set by MASTER
  set by DMA after read command

- FU (1 byte)

- RX error status (1 byte): RXERR

- RX current command list pointer(4 bytes): RXCMPT
  used by DMA

- FU (4 BYTES)

The fourth group is the TX SERVICE AREA:

- TX control table address(4 bytes): TXCTLA

- FU (1 byte)

- TX command (1 byte): TXCMND

- FU (1 byte)

- TX error status (1 byte):TXERR

- TX current command list address (4 bytes): TXCMPT

- TX current data counter (2 bytes): TXDTCNT

- FU (2 bytes)

3. VME BUS PIN ASSIGNMENT

(*) - Signal low level active

VMEbus J1/P1 PIN ASSIGNMEMENT (ROW A)

PIN NAME	SIGNAL DESIGN NAME	STANDARD VME MNEMONIC
A 1	VDAT00+00	D00
A 2	VDAT01+00	D01
A 3	VDAT02+00	D02
A 4	VDAT03+00	D03
A 5	VDAT04+00	D04
A 6	VDAT05+00	D05
A 7	VDAT06+00	D06
A 8	VDAT07+00	D07
A 9	ZGND	GND
A10	VSYCLK+00	SYSCLK
A11	ZGND	GND
A12	VDSTB1-00	DS1*
A13	VDSTB0-00	DS0*
A14	VWRITE-00	WRITE*
A15	ZGND	GND
A16	VDTACK-00	DTACK*
A17	ZGND	GND
A18	VADSTB-00	AS*
A19	ZGND	GND
A20	VINACK-00	IACK*
A21	VIACKI-00	IACKIN*
A22	VIACKO-00	IACKOUT*
A23	VADMD4+00	AM4
A24	VADD07+00	A07
A25	VADD06+00	A06
A26	VADD05+00	A05
A27	VADD04+00	A04
A28	VADD03+00	A03
A29	VADD02+00	A02
A30	VADD01+00	A01
A31	ZVN12	-12 V
A32	ZVP05	+5 V

CROSS-REFERENCE BETWEEN VME SIGNAL DESIGN NAMES
AND STANDARD VME MNEMONIC (Cont.)

VMEbus J1/P1 PIN ASSIGNMEMENT (ROW B)

PIN NAME	SIGNAL DESIGN NAME	STANDARD VME MNEMONIC
B 1		BBSY*
B 2		BCLR*
B 3		ACFAIL*
B 4	SBG0CC-00	BG0IN*
B 5	SBG0CC-00	BG0OUT*
B 6	SBG1CC-00	BG1IN*
B 7	SBG1CC-00	BG1OUT*
B 8	SBG2CC-00	BG2IN*
B 9	SBG2CC-00	BG2OUT*
B10	SBG3CC-00	BG3IN*
B11	SBG3CC-00	BG3OUT*
B12		BR0*
B13		BR1*
B14		BR2*
B15		BR3*
B16	VADMD0+00	AM0
B17	VADMD1+00	AM1
B18		AM2
B19	VADMD3+00	AM3
B20	ZGND	GND
B21		SERCLK (1)
B22		SERDAT (1)
B23	ZGND	GND
B24	VINRQ7-00	IRQ7*
B25	VINRQ6-00	IRQ6*
B26	VINRQ5-00	IRQ5*
B27	VINRQ4-00	IRQ4*
B28	VINRQ3-00	IRQ3*
B29	VINRQ2-00	IRQ2*
B30	VINRQ1-00	IRQ1*
B31	ZVP5SB	+ 5 V STDBY
B32	ZVP05	+ 5 V

CROSS-REFERENCE BETWEEN VME SIGNAL DESIGN NAMES
AND STANDARD VME MNEMONIC (Cont.)

NOTE:

(1) SERCLK and SERDAT represent provision for a special serial communication bus protocol still being finalized

VMEbus J1/P1 PIN ASSIGNMEMENT (ROW C)

PIN NAME	SIGNAL DESIGN NAME	STANDARD VME MNEMONIC
C 1	VDAT08+00	D08
C 2	VDAT09+00	D09
C 3	VDAT10+00	D10
C 4	VDAT11+00	D11
C 5	VDAT12+00	D12
C 6	VDAT13+00	D13
C 7	VDAT14+00	D14
C 8	VDAT15+00	D15
C 9	ZGND	GND
C10	VSYFAL-00	SYSFAIL*
C11	VBUERR-00	BERR*
C12	VSYRES-00	SYSRESET*
C13	VLWORD-00	LWORD*
C14	VADMD5+00	AM5
C15	VADD23+00	A23
C16	VADD22+00	A22
C17	VADD21+00	A21
C18	VADD20+00	A20
C19	VADD19+00	A19
C20	VADD18+00	A18
C21	VADD17+00	A17
C22	VADD16+00	A16
C23	VADD15+00	A15
C24	VADD14+00	A14
C25	VADD13+00	A13
C26	VADD12+00	A12
C27	VADD11+00	A11
C28	VADD10+00	A10
C29	VADD09+00	A09
C30	VADD08+00	A08
C31	ZVP12	+ 12 V
C32	ZVP05	+ 5 V

CROSS-REFERENCE BETWEEN VME SIGNAL DESIGN NAMES
AND STANDARD VME MNEMONIC (Cont.)

VMEbus J2/P2 PIN ASSIGNMEMENT (ROW A)

PIN NAME	SIGNAL DESIGN NAME	STANDARD VME MNEMONIC
A 1		USER I/O
A 2	ZGND	USER I/O (GND)
A 3		USER I/O
A 4		USER I/O
A 5		USER I/O
A 6		USER I/O
A 7		USER I/O (GND)
A 8		USER I/O
A 9		USER I/O
A10		USER I/O
A11		USER I/O
A12		USER I/O
A13	ZVP05	USER I/O (+5 V)
A14		USER I/O
A15		USER I/O
A16		USER I/O
A17		USER I/O
A18	VPNUM0+00	USER I/O (PROCESSOR NUMBER)
A19	VPNUM1+00	USER I/O (PROCESSOR NUMBER)
A20	VPNUM2+00	USER I/O (PROCESSOR NUMBER)
A21		USER I/O
A22	ZGND	USER I/O (GND)
A23		USER I/O
A24	ZGND	USER I/O (GND)
A25		USER I/O
A26		USER I/O
A27		USER I/O
A28		USER I/O
A29	ZVP05	USER I/O (+5 V)
A30	ZVP05	USER I/O (+5 V)
A31	ZGND	USER I/O (GND)
A32	ZGND	USER I/O (GND)

CROSS-REFERENCE BETWEEN VME SIGNAL DESIGN NAMES
AND STANDARD VME MNEMONIC (Cont.)

VMEbus J2/P2 PIN ASSIGNMEMENT (ROW B)

PIN NAME	SIGNAL DESIGN NAME	STANDARD VME MNEMONIC
B 1	ZVP05	+ 5 V
B 2	ZGND	GND
B 3	VRESERVD+01	RESERVED
B 4	VADD24+00	A24
B 5	VADD25+00	A25
B 6	VADD26+00	A26
B 7	VADD27+00	A27
B 8	VADD28+00	A28
B 9	VADD29+00	A29
B10	VADD30+00	A30
B11	VADD31+00	A31
B12	ZGND	GND
B13	ZVP05	+5 V
B14		D16
B15		D17
B16		D18
B17		D19
B18		D20
B19		D21
B20		D22
B21		D23
B22	ZGND	GND
B23		D24
B24		D25
B25		D26
B26		D27
B27		D28
B28		D29
B29		D30
B30		D31
B31	ZGND	GND
B32	ZVP05	+5 V

CROSS-REFERENCE BETWEEN VME SIGNAL DESIGN NAMES
AND STANDARD VME MNEMONIC (Cont.)

VMEbus J2/P2 PIN ASSIGNMEMENT (ROW C)

PIN NAME	SIGNAL DESIGN NAME	STANDARD VME MNEMONIC
C 1		USER I/O
C 2	ZGND	USER I/O (GND)
C 3		USER I/O
C 4		USER I/O
C 5		USER I/O
C 6		USER I/O
C 7		USER I/O
C 8		USER I/O
C 9		USER I/O
C10		USER I/O
C11		USER I/O
C12		USER I/O
C13	ZVP05	USER I/O (+5 V)
C14		USER I/O
C15		USER I/O
C16		USER I/O
C17		USER I/O
C18		USER I/O
C19		USER I/O
C20		USER I/O
C21		USER I/O
C22	ZGND	USER I/O (GND)
C23		USER I/O
C24	ZGND	USER I/O (GND)
C25		USER I/O
C26		USER I/O
C27		USER I/O
C28		USER I/O
C29	ZVP05	USER I/O (+5 V)
C30	ZVP05	USER I/O (+5 V)
C31	ZGND	USER I/O (GND)
C32	ZGND	USER I/O (GND)

CROSS-REFERENCE BETWEEN VME SIGNAL DESIGN NAMES
AND STANDARD VME MNEMONIC

Line processor specification

update 07/07/83
draft -3

general architecture.

shared memory definition.

local memory definition.

mainloop(device procedure dispatcher).

DMA emulator interface

note

- initialization
- open
- close
- interrupt
- read
- write
- data flow

:nroff

1. general architecture

1.1. line procedure board definition(MX)

connectors	N	connection	type	speed/bps	
25 pins	2	RS232c	Local	38,400	asy
			Remote	19,200	asy/syn
		RS422A	Local	12,200	asy
15 pins	2	RS422A	Local	76,800	asy
				100,000	syn
9 pins	2	RS422A	Local	19,200	asy
				38,400	
		RS232C	Local	19,200	asy
				38,400	

1.2. Line processor board

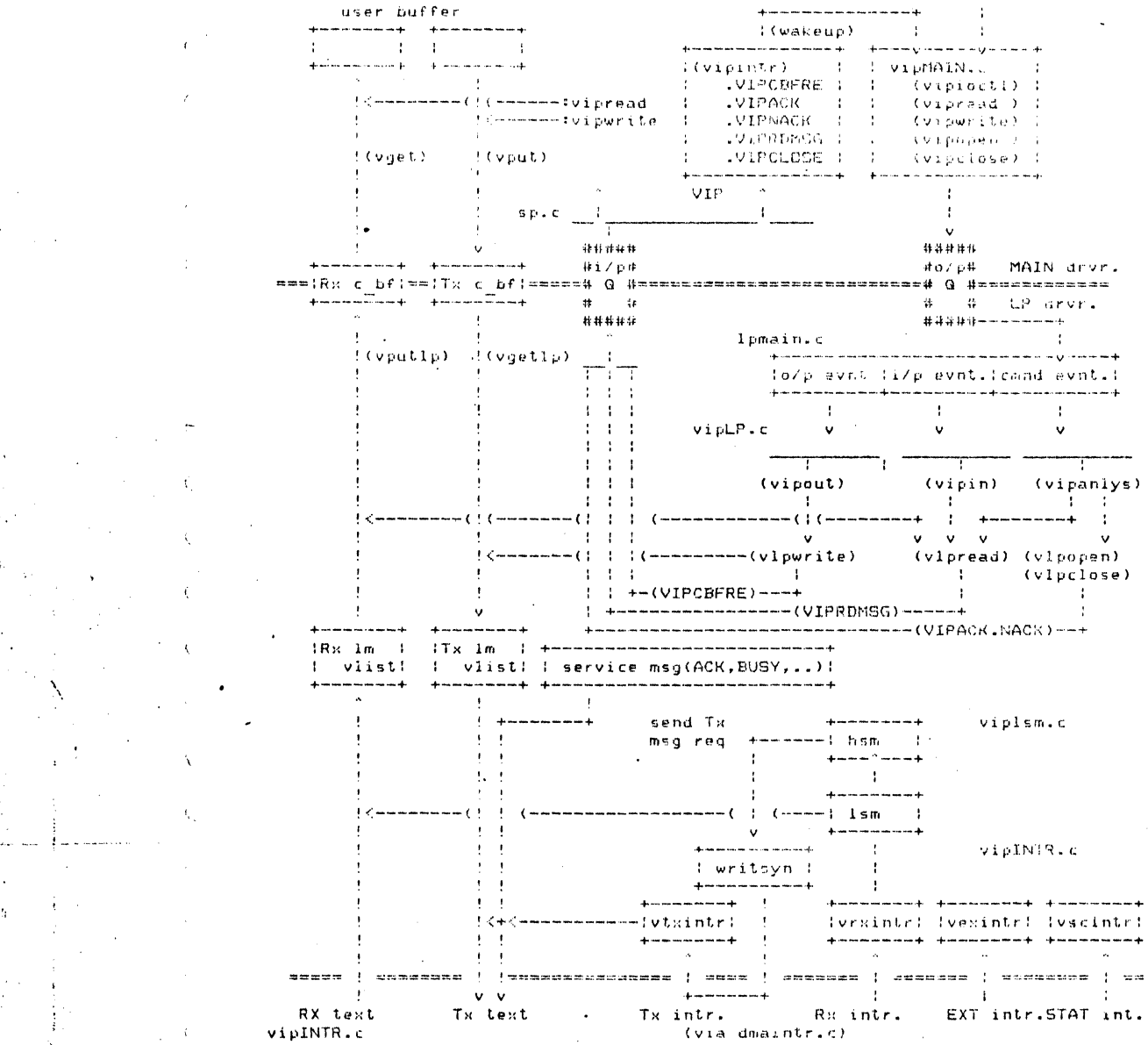
see appendix A

1.3. Interface between LP and Main CPU

The information is exchanged through the input/output command queues in the shared memory via attention interrupt/vme interrupt. Only one input and one output command queue are defined for all drivers.
An other area is allocated for each line at configuration time, it is normally used as data buffer. But a driver can use it anyway. This area is given to the driver at open time.

Attention/vme interrupt is sent when the queue is empty and a command is recorded in it.

VIP driver software structure



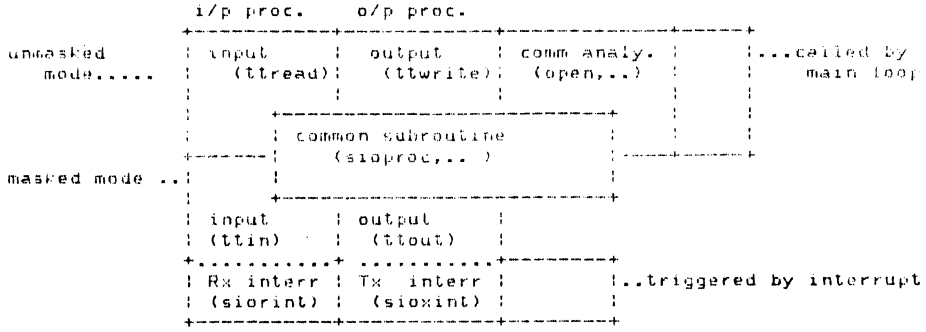
vp : node table lm : local memory
( ) : module name buffer
c_bf : circular shared memory buffer.

1.4. Driver in LP

The architecture of mini OS running into LP assumes that a driver normally may be divided into the following parts:

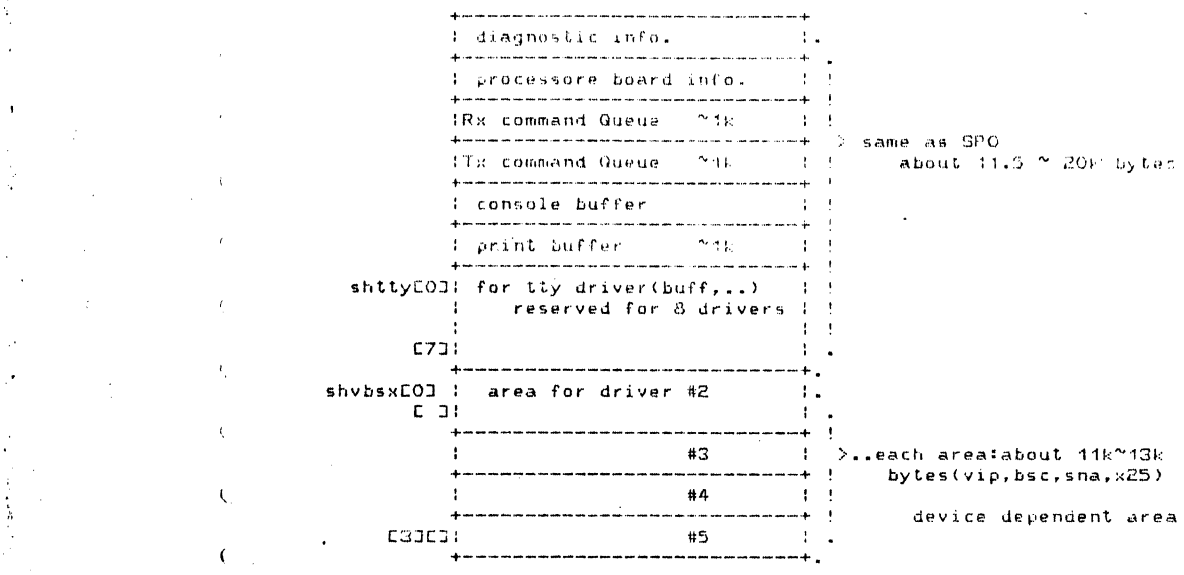
. input procedure	unmasked mode
. " "	masked mode
. output procedure	unmasked mode
. " "	masked mode
. command analyser (open/close/ioctl)	unmasked mode
. common subroutine	masked/unmasked mode

The following picture shows an example of this concept applied to a tty driver:



2. shared memory

shared memory map(64k)



struct shvbsxCSDRVNB]CLPDVSZ];

2.1. i/o subroutine in shared memory(only for vip)

2.1.1. c buffer definition

- addressing to the corresponding c-buffer

At the open time ,device number is declared in the open command. At this time the corresponding driver structure is overlapped on the struct shvbsx[][] .

```
ex vip
    device      #define LINE(x) ((x>>5)&3)
    struct vavip   *vh=(struct shvip *)&shvbsx[LINE(x)][0];

    Tx/Rx c_buffer address is gotten below;

      struct vipbuf  *vbuf=&vh->shrxvbf;    ...Rx

                     or      =&vh->shtxvbf;    ...Tx
```

2.1.2. record definition

Owing to the limitation of shared memory space, the user text record may be divided into some records.
The max record size may be defined at configuration time,but at now this size is defined with the fixed value(ie, MAXREC =1024 bytes).

Each record size is variable but not exceeded the max data size.

One record must not be divided into two parts phisically(ie. the record data is consecutive phisically).

One message associated with the node,may be, consists of some consecutive records.

A record format is:

```
struct vrec {
        uchar_t    vnode;       /*  node address
                                    bit(0-5) :node address */
        uchar_t    vflg;        /*  flag
                                    VRLAST:last text rec
                                    VRITMD:intermediate text rec
                                    LASTREC:last rec in c_buff  */
        ushort     vll;         /* text length                  */
        char       vdata[1];
};
```

C-buffer is:

```
struct  vipbuf {
        short   v_put;
        short   v_get;
        char    vdata[1];
};
```

The record address is caluculated below:

```
struct vipbuf *vbuf;
struct vrec *vr=&vbuf->vdata[vbuf->v_get];
```

see vip.h/sp0sm.h

2.1.3. i/o subroutine

At now only for vip driver:

2.1.3.1. vput

It writes one record on the c_buffer according to the specified parameter list.
It is used for Main.

format: vput(vbuf,arg)

. input:

par1 vbuf: C_buffer address to store data.

par2 arg : parameter list address.

This argument is defined in the vip_line table.

```
struct pgarg l_arg;
```

Structure pgarg is defined below:

```
struct pgarg{
        char      *a_buf;  /* i/p buffer address */
        uchar_t    a_node; /* node address*/
        uchar_t    a_flg;  /* flag for text rec  */
        ushort     a_ll;   /* record size    */
};
```

These above parameters must be compiled by caller.

. output:

After operation vput gives one of the following return codes to caller.

- -1: C_buffer was full and the record was not inserted.
- 0: The record was inserted.
- 2: the record was inserted but not completed one because of no free space.
  The remained data length and the current i/p buffer address were saved into the parameter list(a_ll/buf).

. operation

The record data is moved into c_buffer from caller's i/p buffer until one of the following conditions is occured:

- C_buffer is full, return code (-1) is given.
  (ie. free space is less than S(CBMINS) bytes)
- The data posion of c_buffer has arrived at the last point. The transfer is suspended, and the stored record is completed at this time by having flag"LASTREC" and x"FF" is put after the record.
  Put pointer is reset with binary zero if get pointer is not equal zero and the remaining data is moved again into c_buff as new record .
  If get pointer is zero, return code 2 is given.
- When a record was inserted successfully, return code 0

is given to caller.

- During the data transfer, when there isn't free space, the transfer is suspended and the already stored data is completed at this time.
  Remaining data length and current i/p buffer address are compiled into the parameter list and return code 2 is given to caller.

Anyway the put pointer is updated after record insertion below except case 2:

v_put += (4+ moved data ll)

After compiled a record , x"FF" is put in the 1st character of next record.

2.1.3.2. vget

It reads one record on the specified buffer declared in the parameter list from c_buffer . The record information is stored into the parameter list.
It is used for Main.

format vget(cbuf,arg)

.input

par1 vbuf: c_buffer address to be read the record

par2 arg : the parameter list see vput
Only a_buf must be specified by caller.

.output

After the operation, vget gives the one of the following return codes to caller.

. -1: C_buffer was empty.

. 0: One record having flg intermediate text rec. was moved into the specified buffer.

. 1: One record having flg last text rec. was moved into the specified buffer.

. 2: After reading one record having intermediate text rec flag, c_buffer became empty.

What is more the following parameters must be compiled in the parameter list:

- a_ll record length
- a_node node address
- a_flg text record flag

.operation

The data record is read into the specified buffer by caller until one of the following conditions has been occured and the record header is saved into the specified parameter list.

- C_buffer was empty. return code (-1) is given back.
- The record has the flag "last text record". Return code (1) is given back.
- The record has the flag"Intermediate text rec". Return code (0) is given back.
- After having read one record, c_buffer has becom empty. Return code(2) is given back.

The v_get pointer is updated after one record read below except case2:

```
v_get += (4+record length);
```

If a record flag is "LASTREC" or x"FF", the get pointer must be reset with zero.

2.1.3.3. vputlp

It is used only for LP. According to the indicated infomationt in node table(vp), data is move into c_buffer.

format: vputlp(vbuf,vp)

.input

par1 vbuf: Rx c_buffer address
par2 vp : node table address

used fields:

v_rqll	user request buffer size It is set when VIPREAD command is received.
v_rx	vlist of input buffer containig message.

. output

return(-1) : c_buffer is full, data is not inserted.

return(0) : data is inserted completely in c_buff.

return(2) : patial data is inserted because of no free space in c_buffer.

Except return(-1),the transfered data length is set in node table(v_rcbll).

. operation

Data buffer is accessed via vlist in node table

When a record is written in c_buffer,x"FF" must be put in the first character of next record at same time.

Data length to be moved in c_buffer is:
mll= min(vp->v_rqll,vp->v_rx.v_nc).

If c_buffer space is less than mll, return code 2 is given to caller.

Owing to no free space in c_buffer, if some characters are remained in vlist, the remained data length and next read pointer in vblock must be updated.

When a vblock becomes empty, this is released in vfreel.

2.1.3.4. vgetlp

It is used only for LP. Data in Tx c_buffer is moved into the specified buffer(vblock via vlist(v_tx) corresponding to the node within a data record.

format: vgetlp(vbuf,vp)

. input
par1 vbuf: Tx c_buffer address
par2 vp : node table address to be operated.

. output
return(-1): Tx c_buffer is empty.

(0) : a complete msg is moved into the specified buffer.

(2) : all records in c_buffer are transferd in specified buffer but msg is not complete.

(-2): no buffer in local memory.

(-3): not match node address among records in c_buffer.

. operation
The first free vblock must be prepared by caller.

This vblock is linked in vlist in v_tx.

Until a complete message is received, each record in c_buffer must have the same node address.

Every one character is transfered, LRC is caluculated in v_txlrc.

calling sub routine: putcb(lst,c)

2.2. interface command(queue)

The following commands are declared between LP and Main CPU:

for vip:

- Main to LP

```
   .. VIPOPEN      x"60"
   .. VIPCLOSE     x"61"
   .. VIPREAD      x"62"   par1= request data length
   .. VIPDTW       x"63"   Main drvr waits remaining msg
```

- LP to Main

```
   .. VIPACK       x"70"  answer for open/close
   .. VIPNACK      x"74"  for open(no buff/no ctltbl)
   .. VIPRDMSG     x"72"  Rx message present in c_buff
                          par1= actual data ll in c_buff
   .. VIPCBFRE     x"73"  Tx c-bffer free  ???
```

note: device(major+minor) definition vip

```
bit  15              8   7              0
    +----------------+----------------+
    | 0 0 0 0 0 1 z z | z y y x x x x x |
    +----------------+----------------+
```

zzz = slot number(0 - 7)
yy   = line number relative:device #= line # + 2
xxxxx= node address
bit(15 - 8) : driver type(4 - 7)

: device definition tty

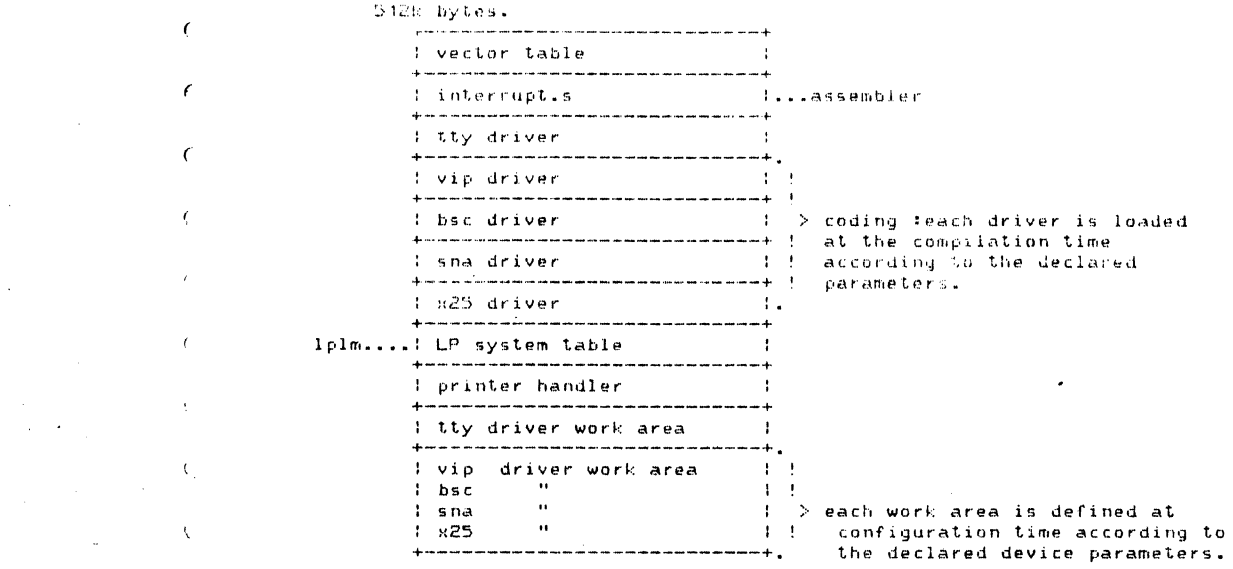
```
bit  15              8   7              0
    +----------------+----------------+
    | 0 0 0 0 0 0 1 0 | 0 0 z z z y y y |
    +----------------+----------------+
```

zzz = slot number(0 - 7)
yyy = device number(0 - 7)
bit(15 - 8) : driver type 2

2. Local memory definition

local memory map is shown below:

512K bytes.



see spsys.h

* field definition: in lm system table

. attention request flag: s_atrf
bit(0) = $\overline{1}$ attention req

.set by atten intr handler
.reset by command analyser

. input request flag: s_iprf
bit(0) = $\overline{1}$ dev#0 req
(1) = 1 dev#1 req
.
.
(5) = 1 dev#5 req

.set by i/p processore under masked mode.
.reset by i/p processore under unmasked mode.

. output request flag:

```
                    s_oprf
bit(0) = 1 dev#0 req
          .
          .
  (5) = 1 dev#5 req
  (6) = 1 printer req

      .set by open time
      .reset by close time
```

4. mainloop (dispatcher)

fig

Each event is called according to the corresponding request flag field(ie.input,output,or attention request flag)

In the input/output event, the proper driver is called using the lock_byte, which is set at open time. Each bit corresponds to a line (device).

In the attention event, the proper driver is called according to the value of device field in the command.

The corresponding request bit in the request flag field must be tested/reset by each corresponding driver.

The request flag fields are defined in the LP system table(s_iprf s_atrf,s_oprf).

The request flag scan algorithm is round robin way.

The dispatching for each driver is done via driver switch table and lock_byte,whitch is defined at open time .

lock_byte : see spsys.h

struct drvdt drvdt[NDRVCCTL]

value				
	tty	2	DTTY	
	vip	4,5,6,7	DVIP	
	bsc	8	DBSC	
	sna	9	DSNA	
	x25	A	DX25	see spOsm.h

.set by open time.
.reset by close time with x"ff"
.intialised value x"ff"

note: this value is equal to the driver value in major device.

driver switch table: see drvswt.c

drvswt[typ]

	cmnd event	i/p event	o/p event
[2]	ttyanlys	ttyinput	ttyoutput
[3]	00	00	00
[4]	vipanlys	vipin	vipout
[7]	"	"	"
[8]	bsc	bsc	bsc

1) for vip

- vipanlys

The following commands are analysed:

.. VIPOPEN — The vlpopen rtn is called. If it's ok, VIPACK command is sent back.

.. VIPCLOSE — The vlpclose rtn is called. If it's ok, VIPACK command is sent back.

.. VIPDTW — The vlpread rtn is called to give the remaining message to Main if t_buffer is locked and the record is for the same node in v_cbstat.
If return is -1, i/p request flag of spsys(s_iprf) is set on.

.. VIPREAD — A flag "VRDREQ:v_cbstat) and VDATARDY are set on , and c_buffer is not locked , vlpread routine is called.
If return of vlpread is -1, s_iprf is set on.

- vipin

The following thinges are done:

.. search the node having the VDATARDY|VRDREQ flag on in v_cbstat. If they are on ,the vipread routine is called.

VDATARDY flag is set by the lower level under masked mode. It means that Rx msg is ready in the lm.

VRDREQ flag is set by vipanlys rtn when VIPREAD command has been received from Main.

- vipout

If the line is opened and Rx c_buffer is empty,vlpwrite rtn is is called to move a message from c_buffer to local memory.

5. DMA emulator interface.

DMAe request is given by Master cpu in LP via DMA device table in I/O memory.

This common data is defined for each driver(except tty at now)below:

* Rx command lists
* Tx command lists
* Rx/Tx common service area
* Rx service area
* Tx service area

when more than two command lists are specified and when a command having the intermediate flag is finished, DMAe executes next command list automatically until a command having the last flag is detected.

After operation completed,DMAe issues one of the following interrupts according to the DMA request type to Master:

* Intermediate command list completed sussesfully.....Tx/Rx
* command list completed sussesfully .....Tx/Rx
* DMA channel errore .....Tx/Rx
* End of record character found .....RX/Tx

At now 7 interrupts are assumed.

The following interrupt vector(DMA to Master) numbers are defined below:

line#2	180	RxI	0	intermediate command list executed
	181	RXL	1	last command list executed
	182	RXER	2	DMA channel error occured
	183	RXEOR	3	End of record character found
	184	TxI	4	intermediate command list executed
	185	TXL	5	last command list executed
	186	TXER	6	DMA channel error occured
	187	TXEOR	7	End of record character found

! condition code

line#3	188 - 195
line#4	196 - 203
line#5	204 - 211

see interrupt.s

After dma interrupt, each appropriate rx/tx interrupt handler is called with the device number and the condition code via dma switch table according to the driver type in dma table.

see dmaintr.c
lplinesw.c
spconf.c

Condition code(bit(0-1)) is defined below:

0: intermediate cmnd.executed(IMCMND)
1: last commd.executed (LSCMND)
2: error occured (ERCMND)
3: end of char.found (EOTCMD)

see dma.h

5.1. DMA I/O Memory definition

1) command list

A command list consists of 10 entries and one entry is 10 bytes.

An entry format is:

- buffer address(4 bytes): BFADR .set by Master
- actual data length(2 bytes):ADLL RX: set by DMAe
  Tx: set by Master
- buffer size (2 bytes): BFSZ .set by Master
- FU(1 byte)
- command list flag(1 byte): CLFLG
  bit(0) =1 intermediate comm.list
  =0 last comm.list
  .set by Master
  (1) =1 comm.list already executed
  .set by DMAe after interr.
  reset by Master

2) Rx/Tx common service area

. driver type (1 byte): DRTYP

```
                    value
                          vip   x"04"  DVIP
                          bsc   x"08"  DBSC
                          sna   x"09"  DSNA
                          x25   x"0A"  DX25     see sp0sm.h
                    .set by Master at the open time.
```

. device status (1 byte): DVSTUS

```
                     value  opened   x"01"     L_OPENED   vip.h
                            closed   x"00"
                     .set by Master.
```

. Mode control register(1 byte): MDCTL

```
                          bit(0) = 1  character is controled via
                                      RX control table .  CTLTBRQ  dma.h
                             (1) = 1  character is controled via
                                      Tx control table.
                             (2) = 1  only one shot(one command list)
                                                          ONESHOT  dma.h
          These flags are set by Master at the open time.
          These status are kept until VIPCLOSE is requested.
```

3) Rx service area

. Rx control table address(4 bytes): RXCTLA

```
                                    .set by Master at open time.
```

. FU (1 byte)

. Rx command (1 byte) : RXCMND

```
                    bit(0) =   tbd

                       (7) = 1 command request
                               .set by Master
                               .reset by DMAe after read command.
```

. FU (1 byte)

. Rx error status (1 byte) : RXERR

```
                               .set by DMA.
                               tbd
```

. Rx current command list pointer(4 bytes): RXCMPT
used by DMAe

. FU (4 bytes)

. DMARXW1(4 bytes) used by DMAe (work area)

. FU (2 bytes)

. DMARXW2(2 bytes) used by DMAe (work area)

4) Tx service area

. Tx control table address(4 bytes): TXCTLA
.set by Master at open time.

. FU (1 byte)

. TX command(1 byte) : TXCMND see RXCMND

. FU (1 byte)

. TX error status (1 byte) : TXERR see TXERR

. TX current command slist pointer(4 bytes): TXCMPT see RXCMPT

. Tx current data counter (2 bytes): TXDTCNT
used by DMA

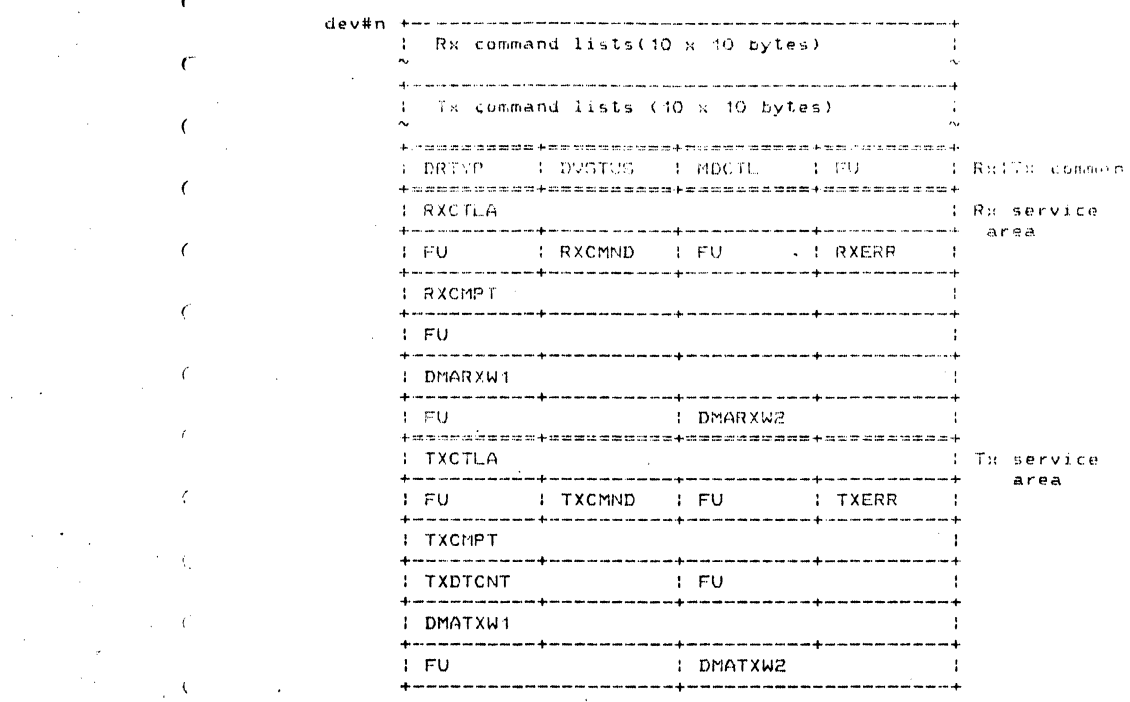
. FU (2 bytes)

. DMATXW1(4 bytes) used by DMAe (work area)

. FU (2 bytes)

. DMATXW2(2 bytes) used by DMAe (work area)

Device information area layout:



note. AT open time, the Master must issue the notify to indicate the DMAe open to Slave(DMAe) via register :

```
                read cycle  reg addr. 68.00.01 <-- vector #nn
                write cycle reg addr. A4.00.01 <-- vector #mm
                        nn: 112    go
                            113    abort   vector#=
                            114    query       nn/mm + line#*8
                            115    fu
                        mm: 116    go
                            117    abort
                            118    query
                            119    fu          ,
```

after the first Rx or Tx command and mode control register are compiled in the corresponding DMA device table of I/O memory.

.SIO registers are initialised by Master at open time.

.Each control table is allocated in the i/o memory following device information area as follows:

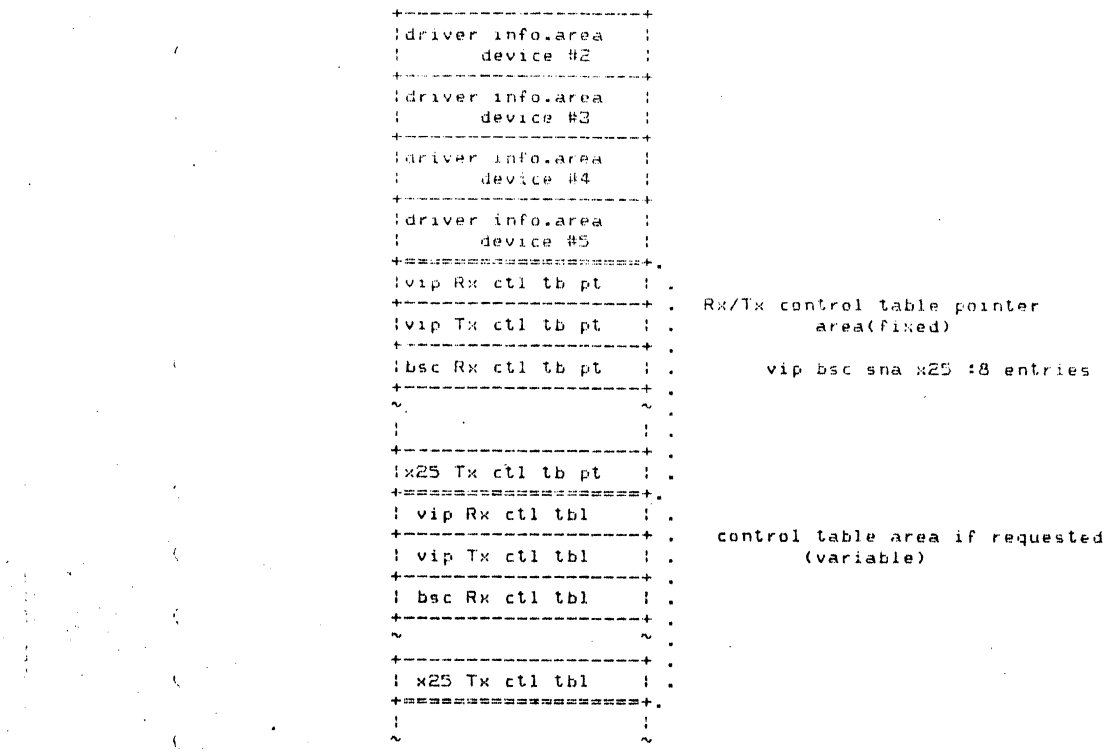
```
control table pointer: for each Rx/Tx driver
                       4 * 2(Tx/Rx)*4(driver #)=32 bytes
control table        : for each Rx/Tx driver if requested
                       one table  256 bytes
                            max 256*8= 2k bytes
```

If control table is not requested at configuration time, the associated pointer must be 0;

.dma interrupt handler — see dmaintr.c
spconf.h
lplinesw.c

.see dma.h

.i/o memory table layout is:



.....table

field name	set by	reset by	used by	
BFADR	M		D	cmnd list
ADLL R:	D		M	
T:	M		D	
BFSZ	M		D	
CLFLG b(0)=1/0	M	M	D	
(1)=1	D	M	D	
DRTYP	M(open)			common sv area
DVSTUS	M(open)			
MDCTL	M(open)		D	
RXCTLA	M(open)		D	RX serv area
RXCMND	M	D	D	
RXERR	D	M	M	
RXCMPT	D	D	D	
DMARXW1	D	D	D	
DMARXW2	D	D	D	
TXCTLA	M(open)		D	TX serv area
TXCMND	M	D	D	
TXERR	D		M	
TXCMPT	D	D	D	
TXDTCNT	D	D	D	
DMATXW1	D	D	D	
DMATXW2	D	D	D	

note: D DMA
M Master

5.2. SIO interrupt

The following interrupt vector numbers are defined for each driver:

tty	#64 - #95	sio0-3
vip	#96 - #111	sio1-2
bsc	#112 - #127	sio1-2
sna	#144 - #159	sio1-2
x25	#160 - #175	sio1-2
		see spsio.h interrupt.s

6. note

6.1. initialization

- the lock_byte with the value :

```
def:Main    sc_lock[MSPOJ[NDRVCTL] :0        see vipMAIN.c
  LP        drvdt[devJ.s_lock      :x"FF"    see spsys.h
```

- LP: initialise freevlist: buffer pool ...vblock link
  set each device area in shared memory to systam table in lm.
  (. ->*s_sm[line]) <----- &(.->shvbsx[line][0])

6.2. open

1) for vip

MAIN:
- The lock_byte, which indicates the supported driver type, is tested.
  And if it's free, the vip driver type(DVIP) is compiled in it.
  If it's not free, an error is given back to user.

- initialise the requested node/line table :

```
line table
            l_status    L_opened
            l_dev       line#(0-3)
            l_devx      driver + slot# :but no node
            l_mode      ~L_FDX
            l_pads      NPADS
  l_syns    NSYNS
  l_devs    node map.
    l_rxcb      Rx c_buff addr.
    l_txcb      Tx c_buff addr.
node table
  v_status  V_OPENED
  v_dev     node address
  v_oflg    user parameter
  v_omode   user parameter
```

LP:

The following thinges are done(in LP):

- initialise of:

  .. DMAe interface :call dmainit return(-1)--,error
  error VENOBF no buffer in lm buffer pool
  VECTLTB Rx ctl tbl not defined

  .. SIO registers see Vsiointb in vipLP.c

  .. lock byte :DVIP

- test DSR dsrpoll :if ready, l_status L_DSR L_SYN on.

- If it's all ok ,VIPACK is sent back to Main.
  If no, VIPNACK is sent and error code is compiled in line table
  (l_errlp)

6.3. close

- release the lock_byte.
  MAIN: 0
  LP : x"FF"

MAIN:
- reset line table:
  l_devs corresponding device # bit off
  l_status 0
  l_ndevs -1 node #

LP:
- reset sio registers:
  XMTCTL ~(TCRDTR!TCRRTS!TCRTXENAB)
  CMDREG CRTXINTRES

6.4. interrupt

- When an interrupt is occured, the proper interrupt handling routine address is selected using the lock_byte.

MAIN:	see sp.c	br to vipintr
LP :	see lpmain.c	br to vipin sio interrupt
		br to vipanlys attention intr.

6.5. read

- read operation

MAIN:

When Main driver received READ command from upper level, the following operations are down:

1) If Rx c_buffer is locked, delay is called until this Rx c_buffer is unlocked.

2) sends VIPREAD to LP with the request data length(u.u_count). And sleep is called.
It is waked up when VIPRDMSG has been received from LP.

3) If ready, a message is moved into the user buffer from c_buffer until the last record is detected.
Before detecting the last record ,the c_buffer became empty(ie, a message is not completed) and sleep is called until the next command(VIPRDMSG) is received.
A command (VIDTW:data wait) is sent to LP.

Command in input queue is read only one at a time...????

The following error cases are assumed:

???

LP:

When command(VIPREAD) has been received, the data requested flag (v_vrdreq=v_cbstat)is set in the appropriate node table. At this time, if the input message is already stored in local memory associated with the node,it is moved into c_buffer in shared memory until reaching the requested data length ,which is passed through via VIPREAD par. And command(VIPRDMSG) is sent to Main.
During transfer of a message,when there is no free space in c_buffer, this node processing is waited until command (VIPDTW) is received.

vlpread(vl,vp,par)
vl: line table address
vp: node table address
par: 0= caller is vipin
1= caller is vipanlys(VIPREAD)
2= caller is vipanlys(VIPDTW)

Before calling this routine ,the following checkes are done by caller:

vipin:
- V_DATARDY & V_VRDREQ on
- Rx c_buffer associated with this line is not locked .

vipanlys:
VIPREAD comm.:
- V_RDREQ on
- Rx c_buffer is not locked.

VIPDTW comm.:
- Rx c_buffer is locked associated with line and node.

- If par is 1/0, the Rx c_buffer is locked.
- call vlpget
- If return code is 2, VIPRDMSG command is sent to Main with the transfered data length. Wait VIPDTW command.
- If return code is 0, VIPRDMSG command is sent with the data length .
This Rx c_buffer is unlocked.
- If a message is sent completely to Main(ie:vp->v_rx.v_nc =0),next flags are set/reset:
v_cbstat ~VDATARDY
v_status ~V_BUSY
V_NOMSG
If some data remains in vlist(in local memory), this data is sent when next VIPREAD command is received.

- If retun code is not 0/2, errore procedure is called.
.....tbd

6.8. write

Main:

- check whether or no Tx c_buffer is locked.
  If it's locked, delay is called to check this flag.
  If no, Tx c_buff is locked and user data is moved
  into c_buffer according to the c_buffer state(l_cbstat).
  During data transfer, if there is no space, sleep is called
  This is wakeduup when VIPCBFRE command is received.
  And the remained data is moved into c_buffer again.
  This is released when the complete data is transferd into
  c_buffer.

LP:

format: vlpwrite(sp,vl)
  sp: spsys address in lm
  vl: line table address

- called by vipout in lpmain

- get node address from a record in Tx c_buffer

- when a message is the first time(V_TXCBWAIT:v_cbstat off),one vblock is fetched from vfreelist in local memory(by allocvb()) and it is linked to the vlist in v_tx.
  Before reading the c-buffer, text header(S_TEXT) is store in the vblock and character length/read pointer are updated:vb->v_nc,vb->v_buf and vp->v_tx.v_nc).
  Lrc will be calucilated until EOT is put in buffer.

- call vgetlp(read Tx c_buff-->vlist)

- If a message is not complete, V_TXCBWAIT is set on and it waits the next call from lpmain. And VIPCBFRE command is sent to Main.

- If a message is moved completely, V_TXCBWAIT flag is reset and tail message(ETB/ETX + LRC + EOT + nPAD) is put in the vblock in local memory.

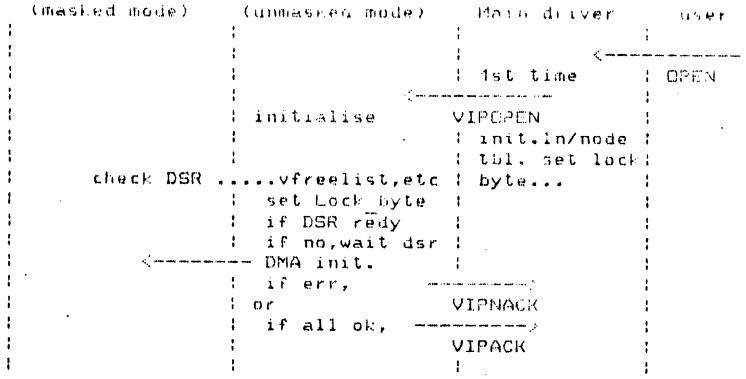
- V_TXMSG is set on(v_state) and VIPCBFRE command is send to Main.

note: If L_block is on(vl->l_status) ETB is put,viceversa ETB.

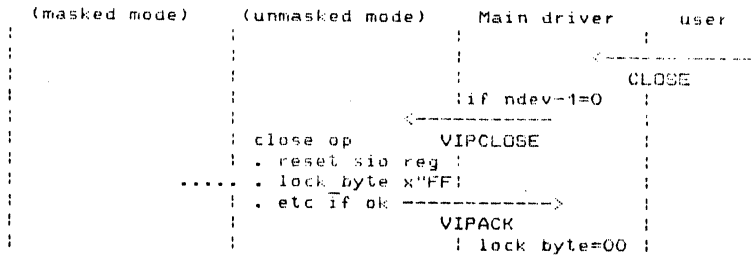
This message is sent to line when polling message is received.

6.7. data flow

6.7.1. OPEN operation data flow

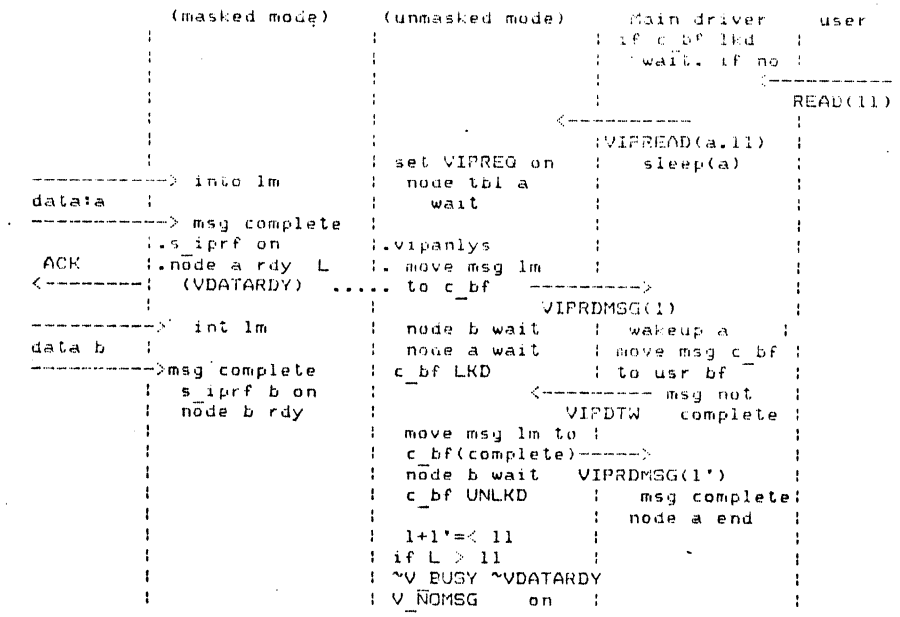


6.7.2. CLOSE operation data flow



9.2.3.

READ operation data flow:



note: 11 user request data length
a,b node address
l recorded data length

6.2.1. WRITE operation data flow: