MOTOROLA
Semiconductor Products Inc.

AN-882
Application Note

THE BISYNC PROTOCOL AND A BISYNC DATA LINK BETWEEN TWO MC68000 MPU BASED SYSTEMS

John Vaglica
Systems Applications Engineer
Motorola Inc.
Austin, Texas

INTRODUCTION

This application note describes the hardware and software required to implement a Binary Synchronous Communications (BISYNC) data link between two MC68000 based systems. Background information is included to introduce the user to the BISYNC protocol and the transmission sequences used. An MC68661/MC2661 Enhanced Programmable Communications Interface (EPCI) controls the data link at one end, and an MC68652/MC2652 Multi-Protocol Communications controller (MPCC) controls the other. The generation and checking of the cyclic redundancy character (CRC) is done by an MC68653/MC2653 Polynomial Generator/Checker (PGC) used at each end. A block diagram of the data link system is shown in Figure 1.

The hardware discussion includes schematic diagrams and a description of the interface circuitry necessary to form the data link. The hardware consists of the following circuitry: MC68000 asynchronous bus interface, interrupt prioritizing logic, interrupt vector generation logic, and ancillary support circuitry required by the data communications devices.

Software listings are provided for the following routines: initialization, interrupt service, and transmitter and receiver I/O drivers. The transmitter and receiver driver routines are set up to run as tasks, with all I/O being interrupt driven. Flowcharts and a description of the algorithm complete the software documentation.

A final topic explores the interfacing exceptions which are not readily discernible from the data sheets. Because MC686xx devices were originally designed to operate on the synchronous bus of the Signetics 2650, several minor interfacing differences exist when used with the MC68000 microprocessor. In addition to the hardware exceptions, several software irregularities are also discussed.

THE BISYNC PROTOCOL

The BISYNC protocol belongs to a group of character oriented protocols known as byte controlled protocols (BCPs). A BCP message consists of a header or control field, a text field, and an error checking field. A typical message is shown in Figure 2. Each message is transmitted as a block consisting of both control and data characters. The special control characters define the beginning of the block, the end of the block, and delineate the various fields within a block.

Each message must be preceded by a minimum of two synchronizing (SYN) characters, to allow the receiver to synchronize itself with the transmitted data. Following the SYN characters, a start of header (SOH) is sent, marking the beginning of the header field. The header contains the control information necessary for the receiver to interpret the remainder of the message. Information in the header includes: the secondary station address, the block sequence number, and control and message acknowledgement information. A start of text (STX) character terminates the header field and begins the text field.

The text field can be of any length, and may contain any character not reserved for data link control. If unrestricted data transmission is required (i.e., the data contains control characters), it is possible to transmit in the transparent mode. The transparent mode is entered by preceding the STX character by a data link escape (DLE) character. Within the remainder of the message, any control or fill characters must be preceded by a DLE. Any control character not preceded by a DLE will be interpreted as data. The text block must be properly terminated by an end of text (ETX), an end of transmission block (ETB), or an intermediate transmission block (ITB) character. In the transparent mode, the block termination character must be preceded by a DLE.

An error checking field follows the termination character. This block check character (BCC) is calculated from one of several polynomials. If the transmitted data were ASCII, the error checking can be either a vertical redundancy check (VRC/parity) on each character and a longitudinal redundancy check (LRC) over the entire message; or a cyclic redundancy check (CRC) over the entire message. Error checking on transmitted EBCDIC data is normally restricted to CRC. The BISYNC protocol requires that all SYN characters and SOH control characters be excluded from the

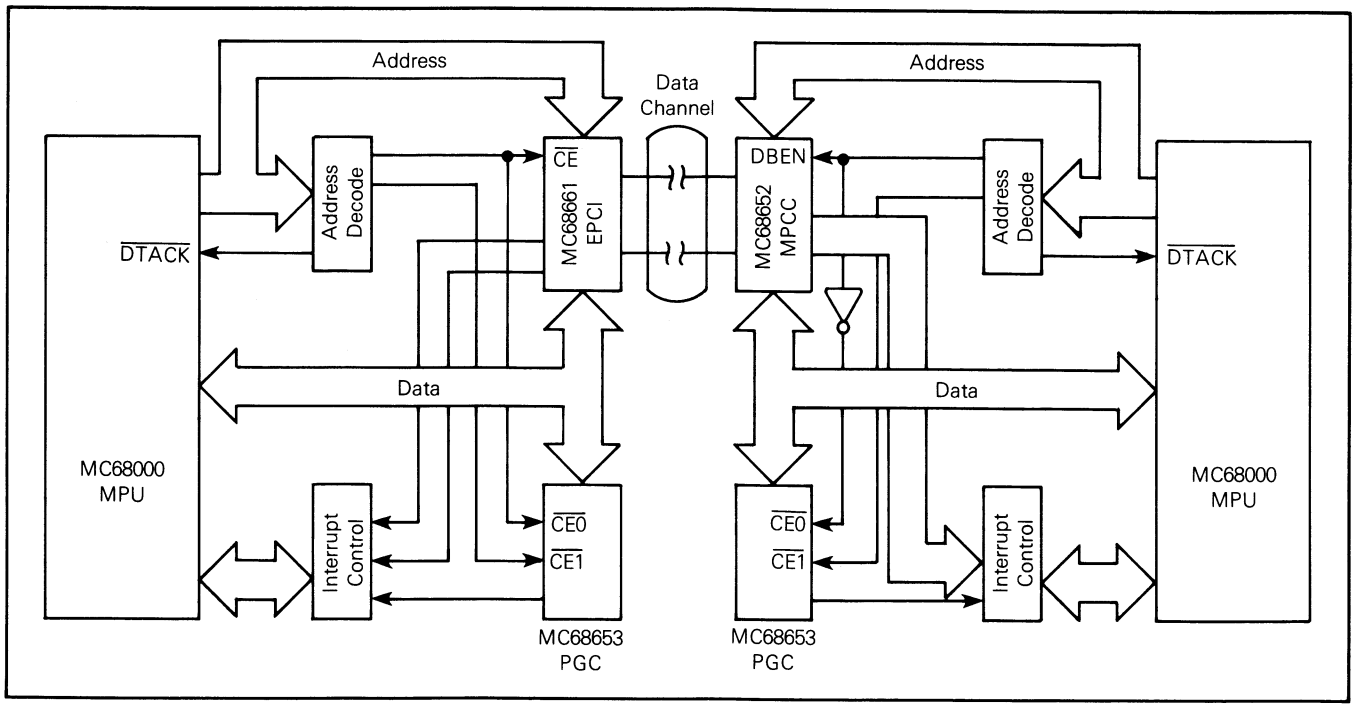


FIGURE 1 — BISYNC Data Link Block Diagram

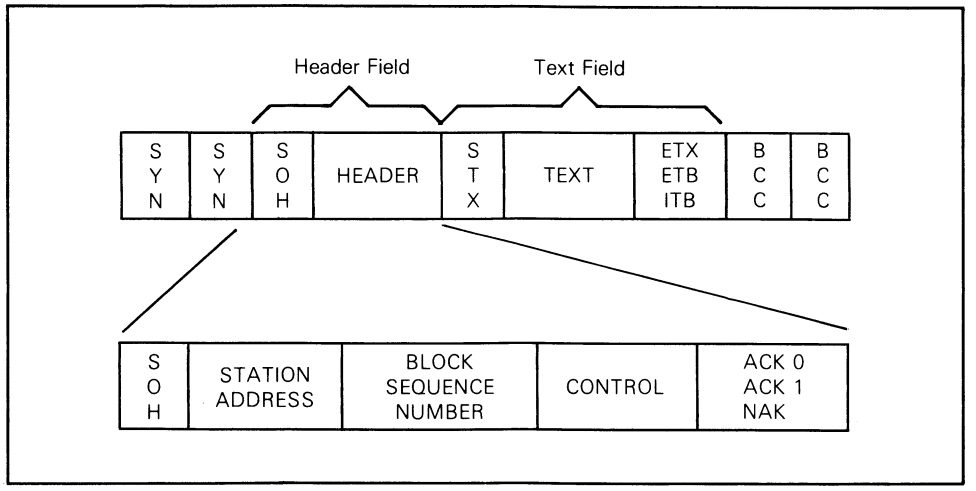


FIGURE 2 — BISYNC Message Block Format

error checking calculation. In the transparent mode, the DLE contained in DLE/control character pairs is excluded from the error checking calculation as well.

When the receiver has completed the BCC calculation, an acknowledgement must be sent to the transmitter, indicating if the message was received without error. The BISYNC protocol does not allow transmission of a new block to begin until an affirmative acknowledgement is received for all previously transmitted blocks. Because of this acknowledgement requirement, the data link is forced to operate in the half duplex mode; therefore, line utilization suffers. The receiver signals the transmitter that it has received the message by sending either an ACK or NAK control message. Providing that the receiving station has a block of data ready to transmit, the acknowledgement for the received block can be embedded in the header field of the next transmitted message. Otherwise, a control message with no data field, should be sent (see Figure 3). Only the latter option is available under the software presented in this application note. If the block sequence number for the received message was even, an ACK 0 is sent; or if the sequence number was odd, an ACK 1 is sent. In the case that the message was received in error, a NAK would be sent.

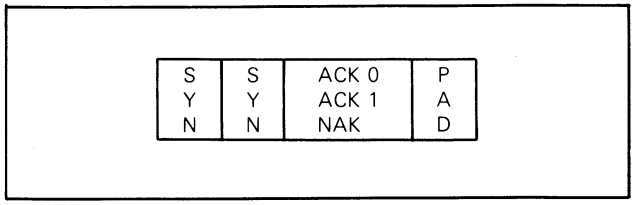


FIGURE 3 — BISYNC Control Message

The data communications peripherals (the MC68652 MPCC, MC68653 PGC, and MC68661 EPCI) each support BISYNC to some extent. The enhanced programmable communications interface (EPCI) is a bus oriented, universal synchronous/asynchronous communications controller. It accepts parallel data from the microprocessor, via the data bus, and converts it into transmit-serial data. Conversely, the EPCI receiver can convert receive-serial data back into character data to be read by the MPU.

When operating in the synchronous mode, the EPCI is designed to handle byte controlled protocols. Internal registers allow the user to program the number of SYN characters, mode of operation (transparent or non-transparent), and automatic DLE stuffing and detection when in the transparent mode. In order to accommodate various character codes, it is also possible to program the values for the SYN and DLE characters. An internal baud rate generator is available as the source of the Tx and Rx clocks, generating frequencies up to 614 kHz (data rate up to 38.4 kilobaud). Alternately, the TxC and RxC pins can be driven by an external oscillator with clock speeds up to one megahertz. The EPCI fully supports BISYNC, with the exception of CRC error checking. If CRC generation and checking is required in a particular application, an external generator/checker, such as the MC68653/MC2653, must be used.

The MC68652/MC2652 is a multi-protocol communications controller (MPCC). The MPCC supports both BCPs and bit oriented protocols (BOPs) at data rates up to one Mbps. A standard synchronous bus interface is provided, allowing the MPCC to communicate with an MPU. Data bus width is eight or 16 bits, selected via the BYTE control input.

The MPCC performs only a minimum number of the functions required to fully support the BISYNC protocol. The MPCC receiver is capable of detecting the initial SYN characters in a message, but will not strip any SYN characters used as line fill during the course of a message. Due to this limitation, the CRC error checking facilities built into the MPCC cannot be used under the BISYNC protocol. In the transmit mode, the MPCC will generate leading SYN characters and insert SYN characters as fill characters during periods of transmitter underrun. The MPCC does not support the BISYNC transparent mode.

Neither the MPCC nor EPCI will properly generate the BCC (block check character), but by using the MC68653/MC2653, both the MPCC and EPCI can fully support BISYNC. The PGC is used for generating the block check sequences and performing parity checks on the parallel data passed between a receiver/transmitter and an MPU. The maximum character accumulation rate is 500,000 characters per second. Three different polynomials can be selected: CRC-16, CRC-12, and LRC-8. Independent of the polynomial selected, four maskable conditions are available as interrupts, via an open drain output. The four conditions allow the flagging of CRC errors, VRC errors, Block Termination Character (BTC) detection, and second search character (SSC) detection.

The PGC can be dynamically programmed to recognize specific characters as belonging to one class or another. There are four classes to which a character can be assigned: normal, SYN/BISYNC not included, block termination character (BTC)/search character (SC), and second search character (SSC). Characters belonging to the normal class are any normal data characters not reserved for control of the data link. The SYN/BISYNC not included characters are those characters which are used to synchronize the receiver hardware to the incoming bit stream. The SYN and SOM characters are included in this class. The BTC characters are those control characters which are used to indicate the end of the data block. Examples of BTCs are ETX, ETB, ITB and ENQ. The secondary search character (SSC) is the final class and contains the second character of control procedures represented by a sequence of two characters. The first character of these sequences must be a DLE. An example of a member of this class is a DLE-STX pair, signaling the initiation of the transparent mode.

The character classes are used to determine whether or not a character, presented to the PGC, should be included under a specific accumulation mode. Up to 128 characters can be assigned in the character class array. Characters presented to the PGC can be accumulated in one of four modes. The modes are: BISYNC normal, BISYNC transparent, automatic accumulate, and single accumulate. In the BISYNC normal mode, all characters presented to the PGC are accumulated except those in the SYN/BISYNC not included class. In the BISYNC transparent mode, characters not included in the calculation are the first DLE of a DLE/non-SYN pair not preceded by an odd number of DLEs. All characters presented to the PGC are accumulated in the automatic accumulate mode. If the single accumulation mode is selected, the start accumulation command must be issued for every character which is to be included.

MC68000 ASYNCHRONOUS BUS INTERFACE

The asynchronous bus structure of the MC68000 maximizes throughput by matching the processor speed to that of the memory or peripheral devices. By signaling the MC68000 when to complete a bus cycle, the peripheral device can vary the length of the bus cycle to match its own access time. Asynchronous control of the bus is accomplished through the use of four control lines: address strobe ($\overline{AS}$), lower data strobe ($\overline{LDS}$), upper data strobe ($\overline{UDS}$) and data transfer acknowledge ($\overline{DTACK}$). The timing for normal read and write bus cycles is shown in Figure 4. The $\overline{AS}$ line is asserted during S2, signaling that the address placed on the address bus during the previous half cycle (S1) is valid. The data strobes ($\overline{UDS}$ and/or $\overline{LDS}$) are then asserted (during S2 for a read or S4 for a write), indicating the length of the operand for this bus cycle. The addressed peripheral device can now be selected and $\overline{DTACK}$ returned such as to ensure that the access time for the device is met. If the peripheral device does not assert $\overline{DTACK}$ at least a setup time before the falling edge of state S4, wait states are inserted for an integral

number of clock periods. This continues until $\overline{DTACK}$ is asserted. A slow read cycle is shown as the final bus cycle of Figure 4. After $\overline{DTACK}$ is recognized, the MC68000 terminates the bus cycle by negating $\overline{AS}$, $\overline{UDS}$, and $\overline{LDS}$. The peripheral completes the cycle by negating $\overline{DTACK}$.

An interrupt acknowledge cycle differs from a normal bus cycle in several ways. Refer to the timing diagram of Figure 5. If an interrupt is pending at the end of an instruction cycle, which is of a higher priority than the current value of the interrupt mask, interrupt exception processing will begin. Interrupt exception processing begins by entering the supervisory state. The machine status, including the current program counter and status register, is saved on the supervisor stack, the interrupt mask is updated to reflect the current interrupt level, and the function codes (FC0-2) are changed to indicate an interrupt acknowledge cycle. The interrupting level is placed on the lowest three bits of the address bus (A1-A3), the remainder of the address bus is driven high, and

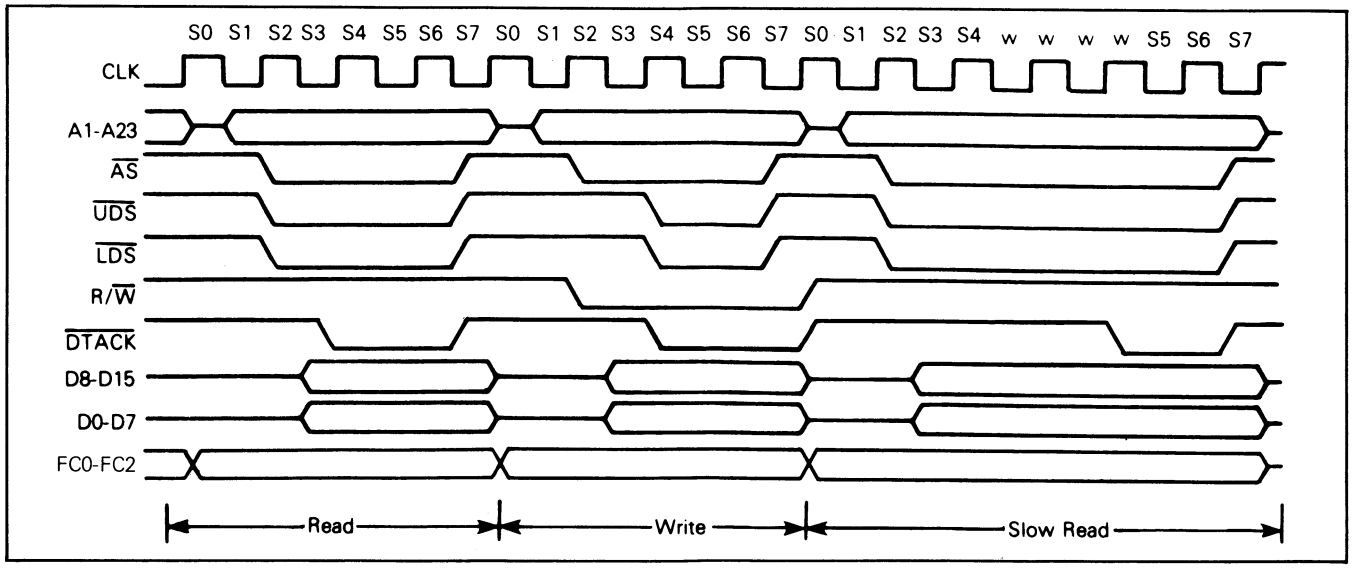
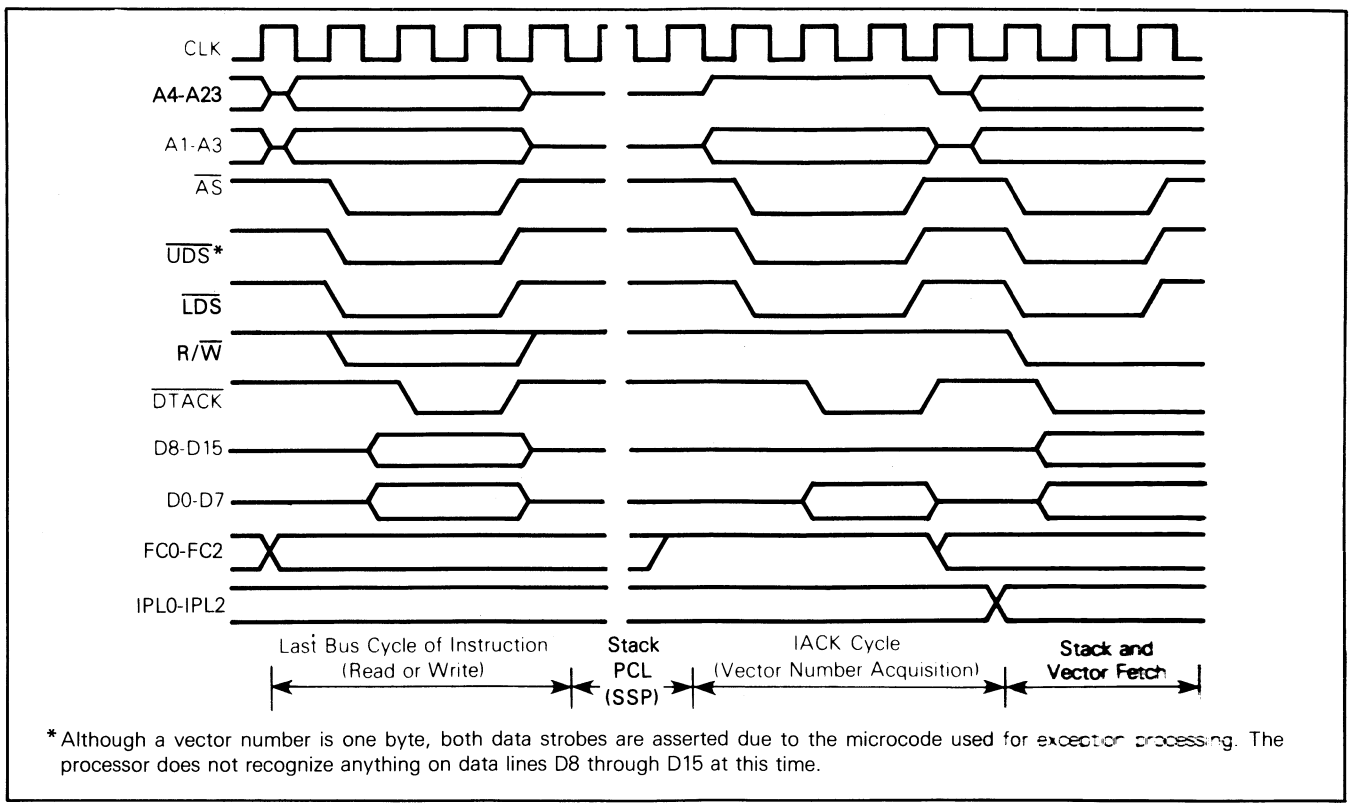


FIGURE 4 — MC68000 Read and Write Bus Cycle Timing



*Although a vector number is one byte, both data strobes are asserted due to the microcode used for exception processing. The processor does not recognize anything on data lines D8 through D15 at this time.

FIGURE 5 — Vectored Mode Interrupt Acknowledge Cycle Timing

$\overline{AS}$ and $\overline{LDS}$ are asserted. Either the vectored or auto-vectored mode can be entered at this point.

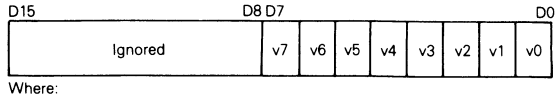
If auto-vectoring is selected, the peripheral will respond by asserting valid peripheral address ($\overline{VPA}$). This causes the MC68000 to internally generate an exception vector number between 25 and 31 (see Table 1), which corresponds to the current interrupt level. At the starting address contained in this vector, execution of the interrupt service routine begins. If the vector number is to be supplied by the peripheral (vectored mode), it must be placed on the low order data lines and $\overline{DTACK}$ asserted. The bus timing for a vectored mode (vectored number supplied by peripheral) interrupt acknowledge cycle is shown in Figure 5.

HARDWARE

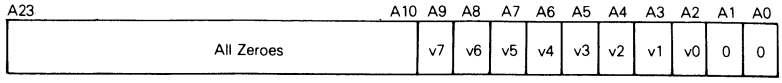
The data communications devices were designed to operate on a synchronous bus similar to the M6800 bus structure; therefore, no provisions were made for the asynchronous bus of the MC68000. In the next few paragraphs, the hardware comprising the asynchronous bus interface and interrupt vector generation circuitry is described.

The data link is comprised of two receiver/transmitters, each under the control of an MC68000 microprocessing system. One terminus contains an MC68661 and an MC68653, while the other contains an MC68652 and an MC68653. Each station also contains the necessary $\overline{DTACK}$ and interrupt service hardware. The data link prototype was tested on a single MC68000 microprocessor system operating on a VERSAbus. The VERSAbus is an asynchronous bus, defined by Motorola, to be used with the MC68000 and its peripherals. Several signals of note on the bus are the $\overline{IACK}$ and $\overline{IRQ}$ lines. The $\overline{IACK}$ signal is generated from the function code outputs, and is asserted only during an interrupt acknowledge cycle. The $\overline{IRQ1}$ through $\overline{IRQ7}$ are vectored interrupt request lines. On the CPU board, these seven lines are encoded into IPL0 thru IPL2. The final note concerning the

TABLE 1 — MC68000 Exception Vectors



Where:
v7 is the MSB of the Vector Number
v0 is the LSB of the Vector Number



Exception Vector Assignment

Vector Number(s)	Address			Assignment
	Dec	Hex	Space	
0	0	000	SP	Reset: Initial SSP
–	4	004	SP	Reset: Initial PC
2	8	008	SD	Bus Error
3	12	00C	SD	Address Error
4	16	010	SD	Illegal Instruction
5	20	014	SD	Zero Divide
6	24	018	SD	CHK Instruction
7	28	01C	SD	TRAPV Instruction
8	32	020	SD	Privilege Violation
9	36	024	SD	Trace
10	40	028	SD	Line 1010 Emulator
11	44	02C	SD	Line 1111 Emulator
12*	48	030	SD	(Unassigned, Reserved)
13*	52	034	SD	(Unassigned, Reserved)
14*	56	038	SD	(Unassigned, Reserved)
15	60	03C	SD	Uninitialized Interrupt Vector
16-23*	64	04C	SD	(Unassigned, Reserved)
	95	05F		–
24	96	060	SD	Spurious Interrupt
25	100	064	SD	Level 1 Interrupt Autovector
26	104	068	SD	Level 2 Interrupt Autovector
27	108	06C	SD	Level 3 Interrupt Autovector
28	112	070	SD	Level 4 Interrupt Autovector
29	116	074	SD	Level 5 Interrupt Autovector
30	120	078	SD	Level 6 Interrupt Autovector
31	124	07C	SD	Level 7 Interrupt Autovector
32-47	128	080	SD	TRAP Instruction Vectors
	191	0BF		–
48-63*	192	0C0	SD	(Unassigned, Reserved)
	255	0FF		–
64-255	256	100	SD	User Interrupt Vectors
	1023	3FF		–

*Vector numbers 12, 13, 14, 16 through 23, and 48 through 63 are reserved for future enhancements by Motorola. No user peripheral devices should be assigned these numbers.

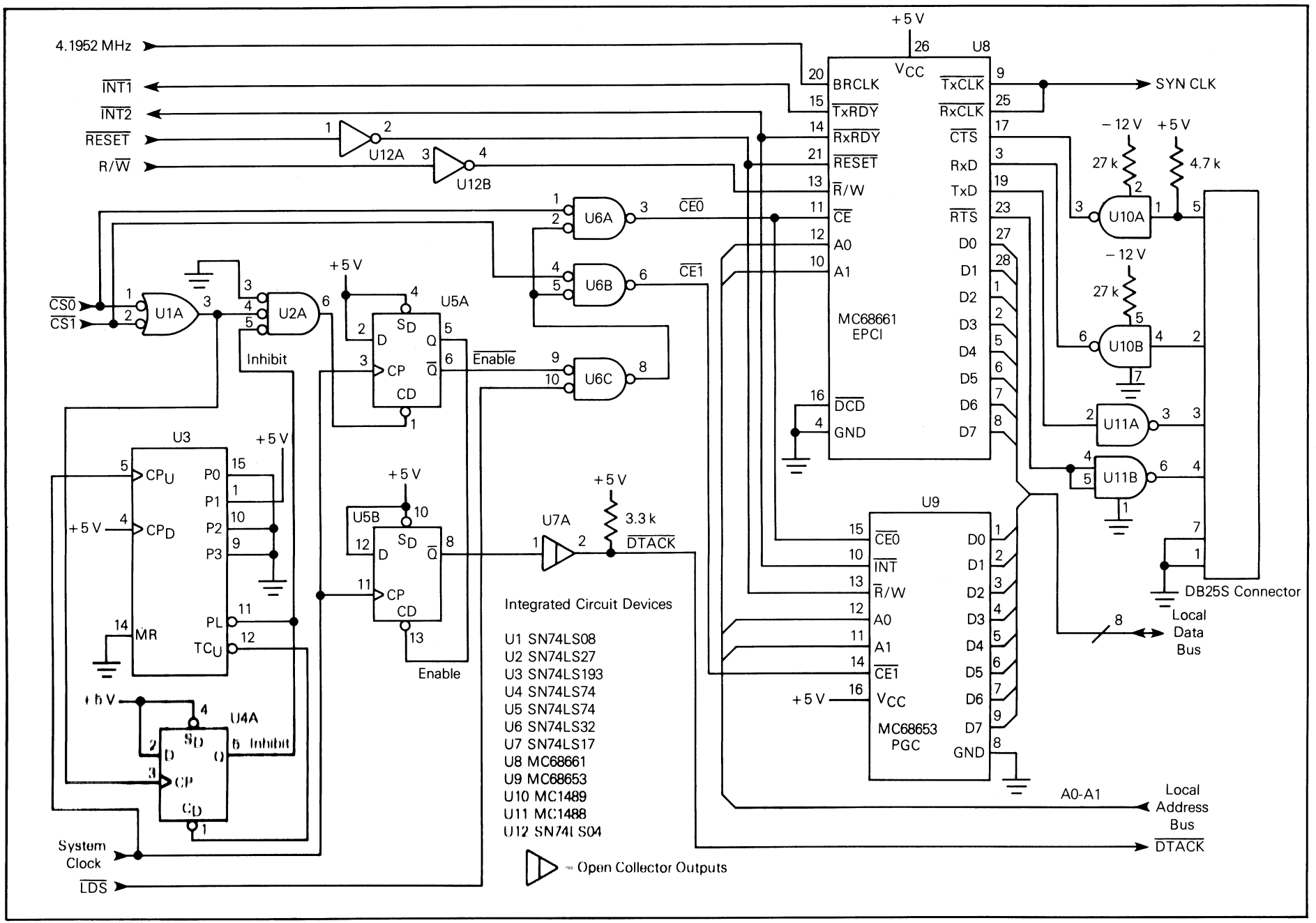


FIGURE 6 — MC68661/MC68653 to MC68000 Interface Schematic

VERSAbus is that all address and data lines are inverted. This accounts for the need to have both local and system, data and address buses as shown in the schematic diagrams.

MC68661 EPCI/MC68653 PGC INTERFACE

In addition to merely supplying $\overline{\text{DTACK}}$, the MC68661/MC68653 interface circuitry must also force the MPU to meet the chip enable delay period, t_{CED}, specified in the data sheet. Figure 6 contains the circuitry required to meet these requirements. The PGC has two chip enables, $\overline{\text{CE0}}$ and $\overline{\text{CE1}}$. The $\overline{\text{CE0}}$ enable signal controls access to the character register used in accumulating the CRC. This signal can also be used to access the data registers of the EPCI, and thus write to both the PGC and the EPCI at the same time. The $\overline{\text{CE1}}$ enable signal is used to select the command and status registers of the PGC and is decoded at a different address. A number of the MC68000 instructions can access the same or consecutive addresses on succesive bus cycles. In these cases, a double read or write could reaccess the same location within 2.5 clock cycles (187 nanoseconds at 8-megahertz clock rate). This violates t_{CED} and t_{CEC} on the MC68653 and t_{CED} on the MC68661. The minimum t_{CED} for the EPCI is 600 nanoseconds, whereas the minimum t_{CED} for the PGC is 1750 nanoseconds. In order to prevent the PGC from being reaccessed too soon, the chip enable signals must be delayed, as discussed below.

The $\overline{\text{CS0}}$ and $\overline{\text{CS1}}$ inputs are the unqualified chip select signals, generated as a function of the address lines and address strobe. The state machine (SN74LS193) shown in Figure 6, guarantees that these chip selects are held off for 14-clock cycles after the trailing edge of the previous chip enable. The 14-cycle delay corresponds to a period of 1750 nanoseconds (PGC t_{CED}), assuming an 8-MHz system clock. For other system clock frequencies, the number of delay cycles has been compiled in Table 2. Following the trailing (rising) edge of the first chip select, INHIBIT is asserted. As well as holding off subsequent chip enables, INHIBIT also loads the SN74LS193 counter with a binary value of two (parallel load 0010). The state machine counts for fourteen system clock cycles before negating INHIBIT (TC_U goes low) and halting. The falling edge of INHIBIT is synchronized with the system clock, creating the ENABLE signal. The ENABLE output is gated with $\overline{\text{LDS}}$, allowing $\overline{\text{CE0}}$ or $\overline{\text{CE1}}$ to be asserted. Since the chip enables are gated with the lower data strobe only, both byte and word instructions can be used. The high ENABLE disables the SN74LS74 (U5B) clear input and permits $\overline{\text{DTACK}}$ to be asserted on the next rising edge of the system clock. A timing diagram of a read bus cycle is shown in Figure 7. The first cycle of a double read is shown with INHIBIT low, allowing a normal read to occur. In the next cycle, $\overline{\text{CE}}$ is held off by INHIBIT until the 14-cycle delay time is met.

TABLE 2 — Delay for Various Clock Rates

MPU Clock Rate (MHz)	Delay (MPU Clock Cycles)
4	7
6	11
8	14
10	18
12	21
14	25
16	28

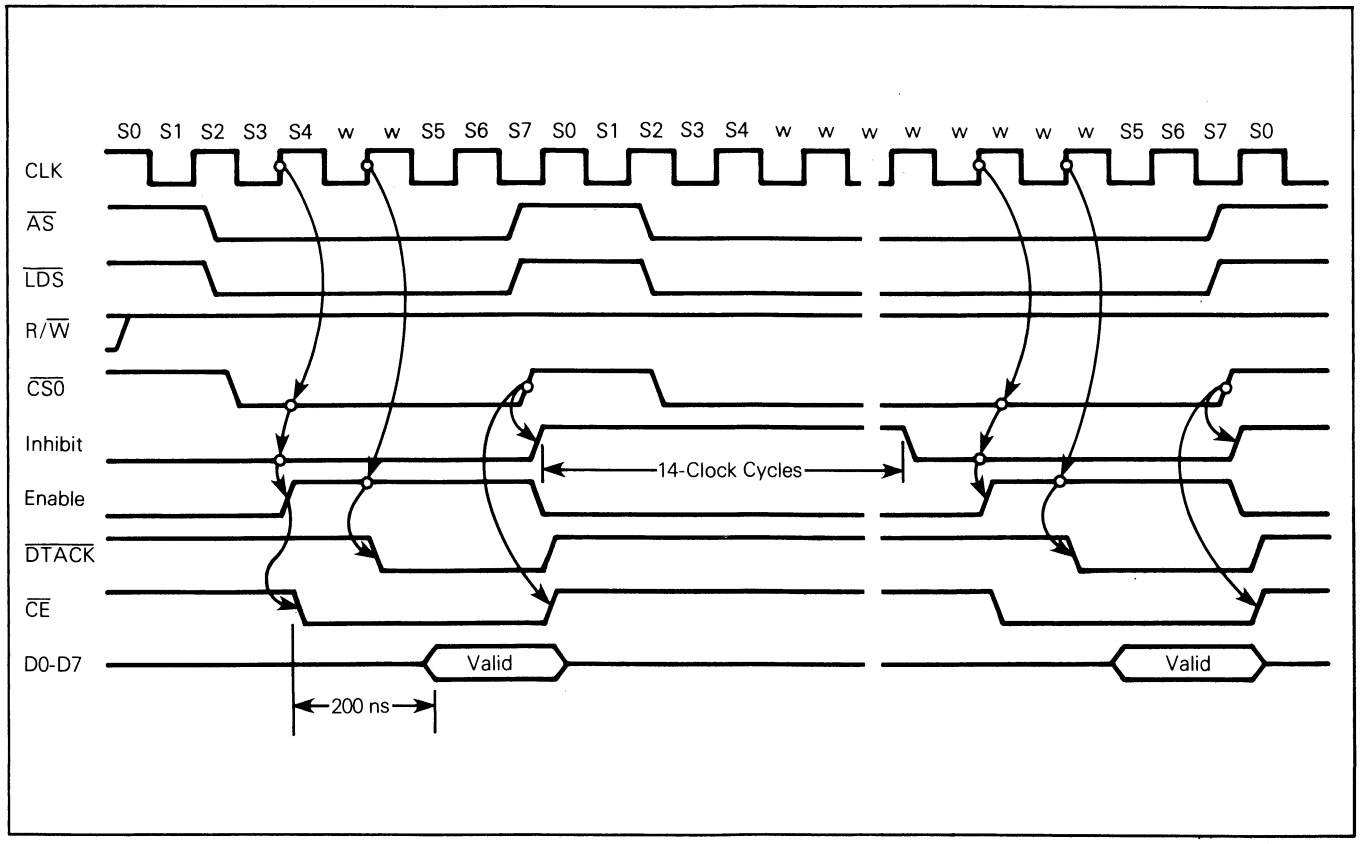


FIGURE 7 — MC68661/MC68653 Read Cycle Timing

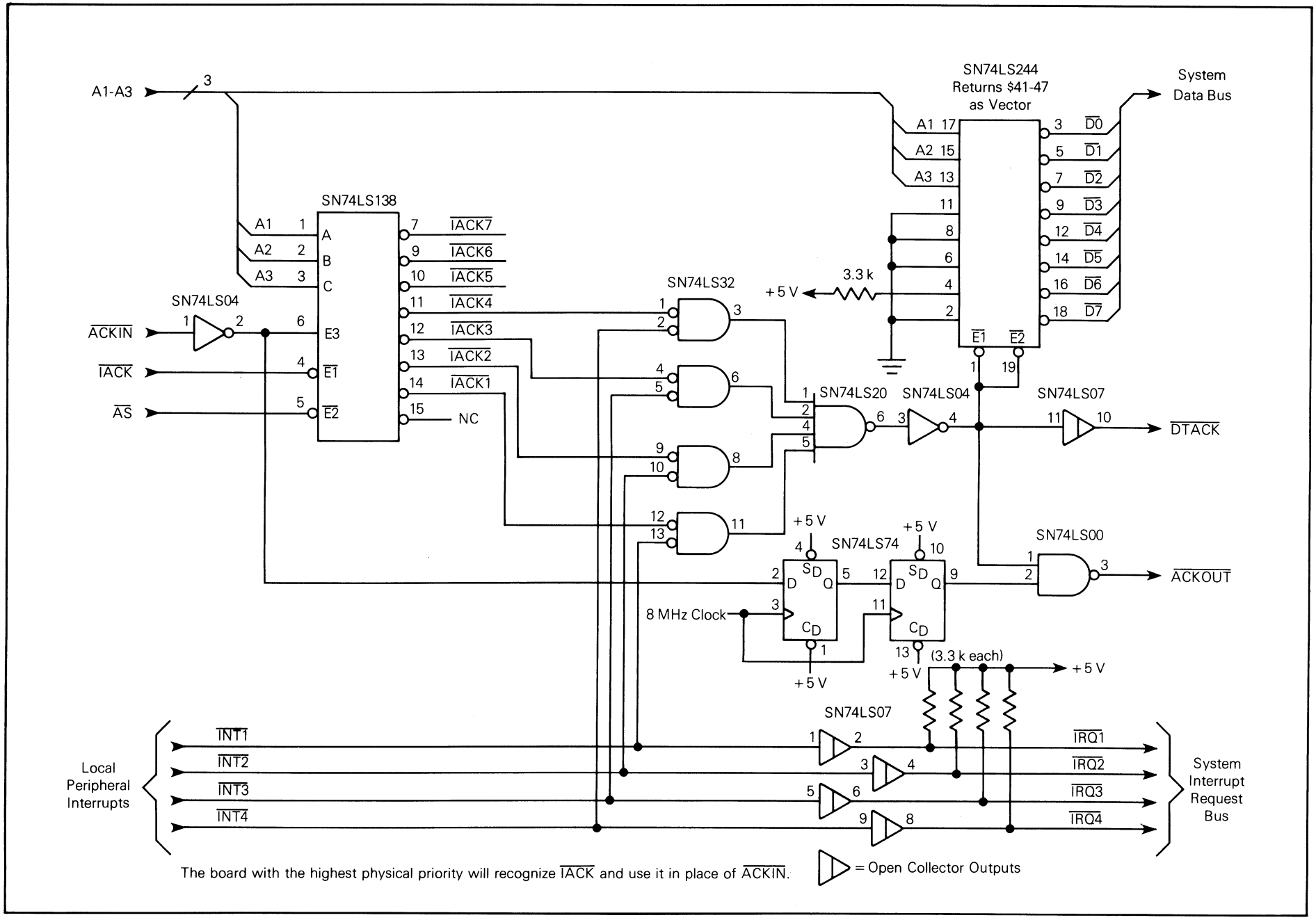


FIGURE 8 — Interrupt Prioritizing and Vector Generation Logic

The remainder of the PGC interface is straightforward. The MC68000 R/$\overline{W}$ and $\overline{RESET}$ lines must be inverted to match the corresponding lines of the peripherals. The interrupt request lines, shown in Figure 8, are connected to the appropriate level of the vectored interrupt hardware. The internal baud rate generator of the MC68661A requires that a frequency of 4.9152 MHz be applied to the BRCLK input. A simple crystal oscillator, shown in Figure 9, may be used to generate the clock. The SYNCLK signal is the internally generated transmit clock that is used as the synchronous clock in the data link. The MC1488 and MC1489 provide the necessary level shifting to conform to EIA RS-232-C standards. This level shifting is not required, but it does allow for the data link to be used as a standard asynchronous serial port. Only the software configuration of the EPCI needs to be changed to switch to the asynchronous operating mode.

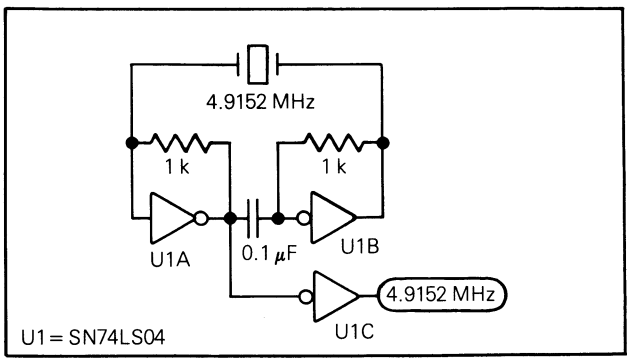


FIGURE 9 — 4.9152 MHz Baud Rate Oscillator

MC68652 MPCC/MC68653 PGC SUPPORT CIRCUITRY

The MC68652/MC68653 bus interface is very similar to the MC68661/MC68653 interface, even though the chip enable to the MPCC, during the second read of a double read, need not be delayed. A PGC is being used in conjunction with the MPCC; therefore, the 14-cycle delay is still necessary. Figure 10 depicts the chip select circuitry. The data bus enable signal (DBEN) is required by the MPCC and is used as the reference for all internal activity (as opposed to CE, which only controls power consumption). To simplify the interface circuitry, CE is tied high and DBEN used as the chip enable signal. In addition to the bus interface circuitry, the MPCC requires additional hardware to minimize software overhead.

The transmitter and receiver enables (TxE, RxE) are not bits of a status register, but are external pins. An addressable latch (SN74LS74) is required to permit enabling and disabling the R/T. By incorporating $\overline{UDS}$ in the latch decoding, the latch appears as the even byte in the same address space as the PGC (which used the $\overline{LDS}$). The R/$\overline{W}$ signal is also used in the decoding scheme, thus allowing the same location to be used for a status buffer as well. The SN74LS240 buffer allows the processor to read all of the status bits that appear as external pins to the MC68652. The latch and buffer are shown as part of Figure 11. Additional circuitry in the figure includes a portion of an SN74LS32 which is used to generate the BYTE control signal. The BYTE signal indicates to the MPCC the width of the data in a peripheral read or write. Through BYTE and the address lines (A1 and A3), the MPCC registers are selected as shown in Table 3. The A0 input is generated by the lower data strobe since the odd and even bytes of the MPCC are reversed with respect to those of the MC68000. In order that the PGC character register and the MPCC data registers could be accessed simultaneously, address lines A1 and A2 of the MPCC, and A0 and A1 of the PGC are offset, as shown in Figure 11.

TABLE 3 — MC68652 Register Addressing

A3	A1	LDS	Register
Byte = 0: 16 Bit Data Bus			
0	0	X	RDSR
0	1	X	TDSR
1	0	X	PCSAR
1	1	X	PCR*
Byte = 1: 8 Bit Data Bus			
0	0	0	RDSR(L)
0	0	1	RDSR(H)
0	1	0	TDSR(L)
0	1	1	TDSR(H)
1	0	0	PCSAR(L)
1	0	1	PCSAR(H)
1	1	0	PCR(L)*
1	1	1	PCR(H)

* — PCR lower byte does not exist. It will be all "0s" when read.

A common serial clock is used throughout the data link; however, to properly interface the MPCC with another MC68652 or an MC68661, it is necessary to invert the MPCC transmit clock (TxC). The inversion is required because the MPCC uses the same clock edge to strobe data into the receiver and out of the transmitter. Using a common edge is a problem, because transmission line effects, inherent in the data channel, could result in a loss of data integrity.

The MC68652 does not have any signals which were meant to be used directly as interrupts. Instead, several of the status lines (TxBE, RxSA, and RxDA) were used. The TxBE status line is inverted, passed through a open collector driver and prioritized on level 3 ($\overline{INT3}$). The RxDA and RxSA status lines are NORed to produce a single logic low output signal, which was passed through an open collector driver and vectored onto level 4 ($\overline{INT4}$).

The serial data was level shifted similar to the EPCI. As stated earlier, if the EPCI is not intended to be used as an asynchronous device, TTL levels could be used for the serial data.

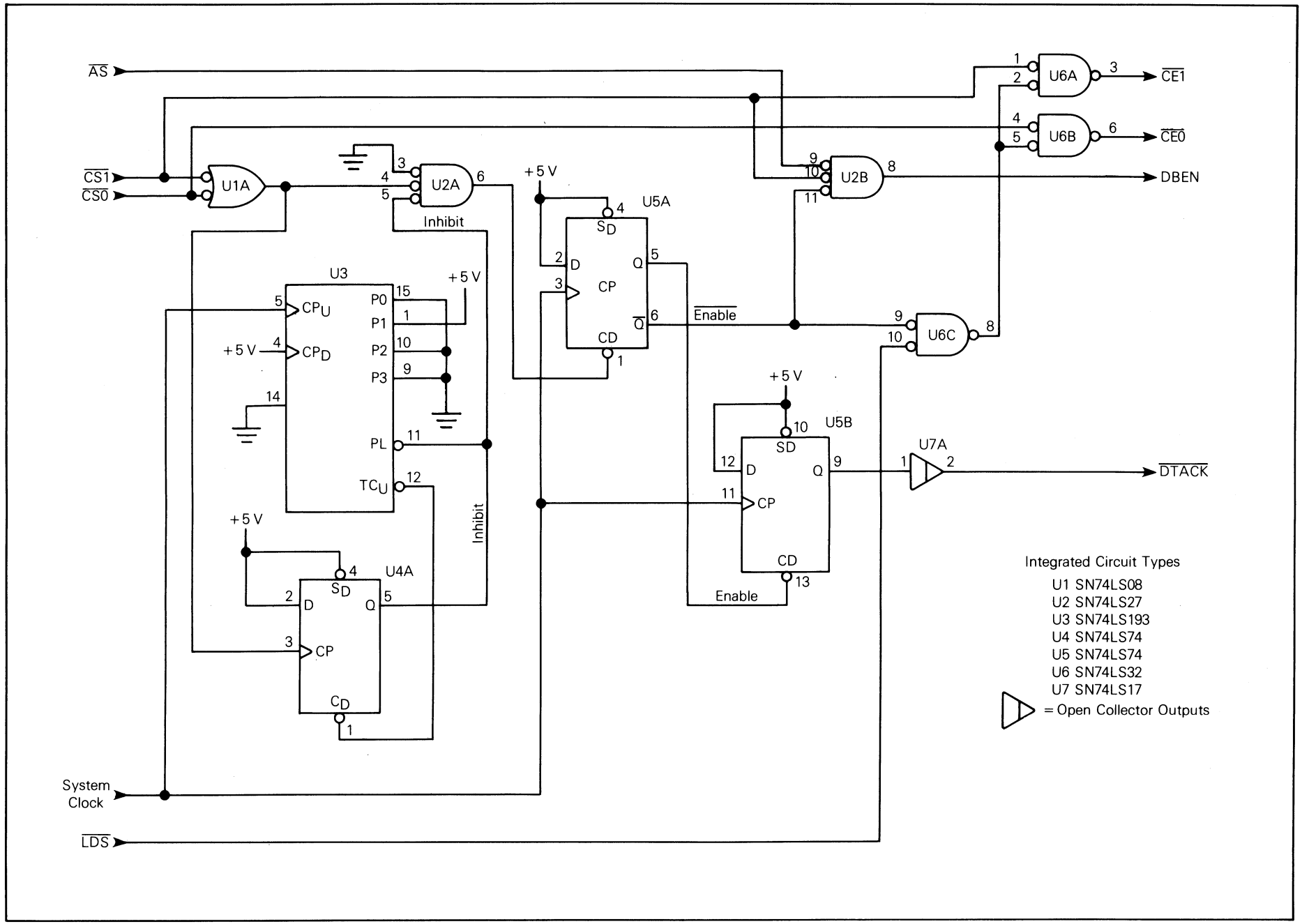


FIGURE 10 — MC68652/MC68653 Bus Interface Schematic Diagram.

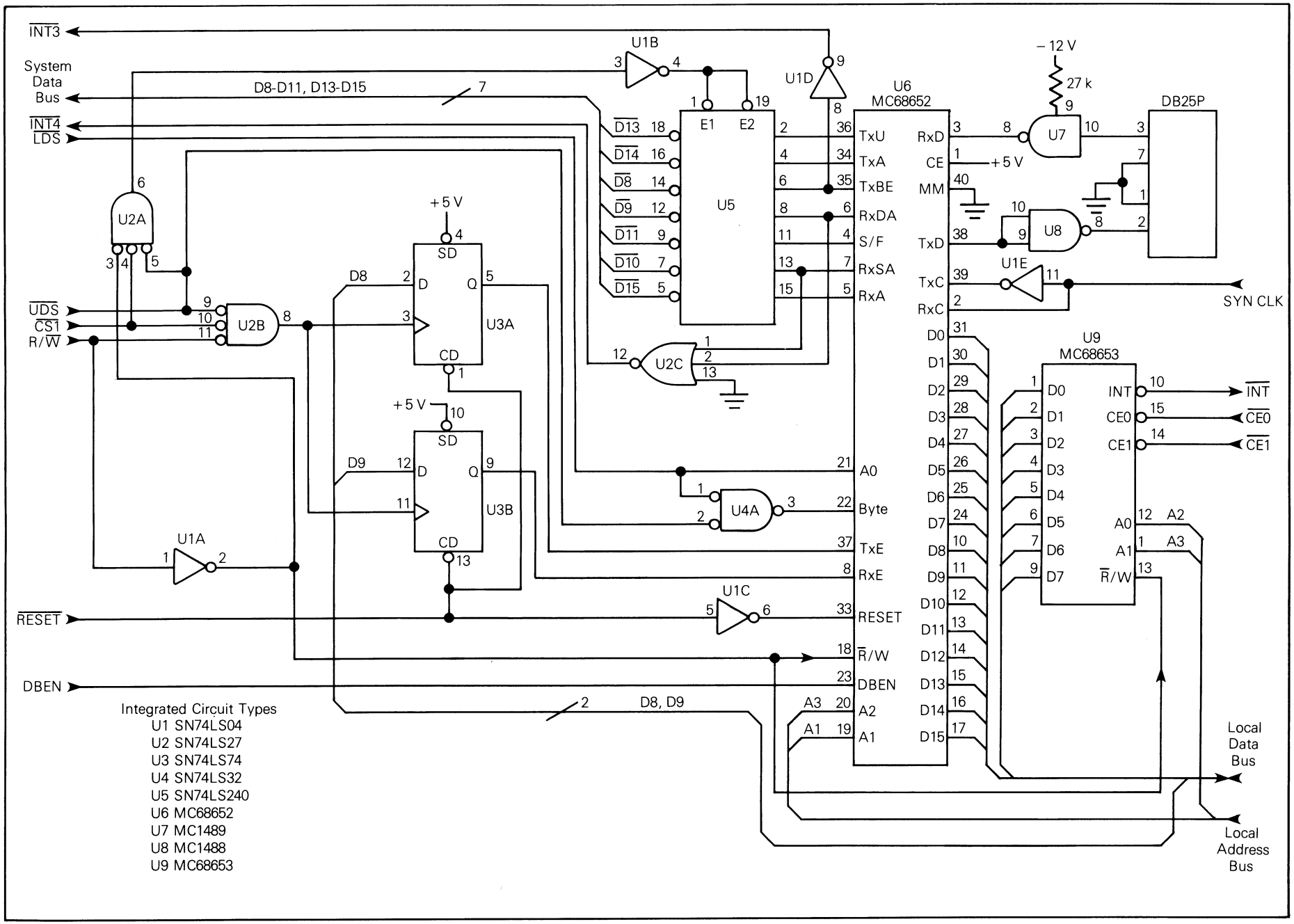


FIGURE 11 — MC68652 Support Circuitry Schematic Diagram

INTERRUPT VECTORING HARDWARE

The vectored interrupt mode of operation was chosen for the interface rather than the autovectored mode. In this mode, the vector number must be supplied via the data bus during the interrupt acknowledge cycle. The circuitry described here provides seven levels of interrupts, each with its own unique vector. The vector number is alterable, such that the circuitry can be easily duplicated on additional boards. This allows "daisy chaining" of interrupts on the same level while maintaining unique interrupt vectors for each device.

The interrupt prioritizing/vector generation circuitry is illustrated in Figure 8 and functions as follows. A one-of-eight priority decoder (SN74LS138) determines which of the four locally generated interrupts ($\overline{INT1}$ through $\overline{INT4}$) will assert $\overline{DTACK}$ (and also negate $\overline{ACKOUT}$) during an interrupt acknowledge cycle. One output line of the priority decoder is asserted low by the A1-A3 inputs (assuming $\overline{E1}$, $\overline{E2}$, and E3 are enabled). The selected line is then compared to the corresponding interrupt line inputs ($\overline{INT1}$, $\overline{INT2}$, $\overline{INT3}$, or $\overline{INT4}$) in one of the SN74LS32 OR gates. If both inputs to one of the OR gates are low, $\overline{DTACK}$ is asserted ($\overline{ACKOUT}$ is negated) and the interrupt vector number is placed on the data bus (by three-state inverter/buffer SN74LS240). The lower three bits of the vector number are selected from within a given range by the A1-A3 inputs to the SN74LS240. The upper five bits are hardwired during system design to provide the correct range of vectors (as shown in Figure 8, vector numbers $41 through $47 have been selected on this board). Seven separate $\overline{IACK}$ outputs ($\overline{IACK1}$ through $\overline{IACK7}$), each corresponding to an interrupt level, are generated by the one-of-eight priority decoder; however, only four are used. If an interrupt currently requesting service was generated on this board, the output of the SN74LS20 (4-input NAND gate) goes high to assert $\overline{DTACK}$. The interrupt requests, which are passed down the system bus, are driven by SN74LS07 open collector drivers to allow wire-ORing of multiple interrupts on the same level. The "daisy chaining" of interrupts during the interrupt acknowledge cycle is accomplished between boards by controlling the propogation of $\overline{ACKIN}$/$\overline{ACKOUT}$ signal along the bus. The $\overline{ACKIN}$ input of each board reflects the $\overline{ACKOUT}$ level of the previous board in the system (the first board in the system receives its $\overline{ACKIN}$ from the MPU). When a low $\overline{ACKIN}$ signal reaches a board, it indicates that no devices of higher priority are currently interrupting the processor on the same level. If no interrupt is pending at this priority level during the interrupt cycle, the low $\overline{ACKIN}$ will simply pass through as a low $\overline{ACKOUT}$ to the next board. Once the board that requested the interrupt receives a low $\overline{ACKIN}$, it asserts $\overline{DTACK}$ ($\overline{ACKOUT}$ cannot go low) and places its vector number on the data bus. This breaks the "daisy chain" propogation path and the interrupt acknowledge cycle is completed.

SOFTWARE

Three distinct subprograms exist within the data link software. The three divisions are the initialization routines, the I/O drivers and the interrupt service routines. The initialization routines define the operating conditions of the data communications devices, define the interrupt vectors, and move the header block from ROM into RAM. The transmitter and receiver driver subroutines handle the non-real time overhead required in supporting the data link. For the transmitters, this involves little more than controlling the transmitter and receiver enables. The receiver driver subroutines must check the BCC bytes and setup and monitor acknowledgements. The interrupt service routines handle the real time processing required by the EPCI and the MPCC. In addition to handling the data transfers to an from the MC68000s, the service routines must also maintain the data pointers. In the following paragraphs, each section of the software is described individually. The complete, annotated listings are included at the end of this applications note.

The data link software was written to demonstrate the ability of the data communications peripherals to support the BISYNC protocol. As such, only a subset of the protocol is fully supported by the routines presented in this applications note. The limitations imposed are that the acknowledgements must be a separate control message, rather than part of the data message header and the BISYNC transparent mode is not supported. A few minor software changes, to monitor the DLE characters, is the only requirement to support the transparent mode.

As has already been shown, the BISYNC protocol requires the use of a number of control characters. The following table matches the control character class as programmed in the PGC, with the ASCII abbreviation and the hexadecimal value of the code. Note that all characters not listed in the table fall under the BISYNC Normal class.

TABLE 4 — ASCII Control Codes

Class	ASCII Mnemonic	Hex Value
SYN/BISYNC Not Included	SYN	16
	SOH	01
	ETX	03
Block Termination Character/	EOT	04
Search Character	ENQ	05
	ETB	17
Secondary Search Character	STX	02

SYSTEM INITIALIZATION

During the power up reset sequence, a portion of the data link initialization takes place. After entering the supervisory state and masking the interrupts, the power up reset initialization sequence depicted in Figure 12 is executed. The header block, which is normally stored in ROM, is moved to RAM to facilitate updating the station address and block sequence count during each message. At this point, the interrupt and trap vectors are also initialized to point to the proper service routines. Each data communications device is then initialized to the proper mode with the transmitters and receivers disabled. The ASCII control codes are programmed into the PGC character class array and the error checking polynomial is set to CRC-16, completing the device initialization.

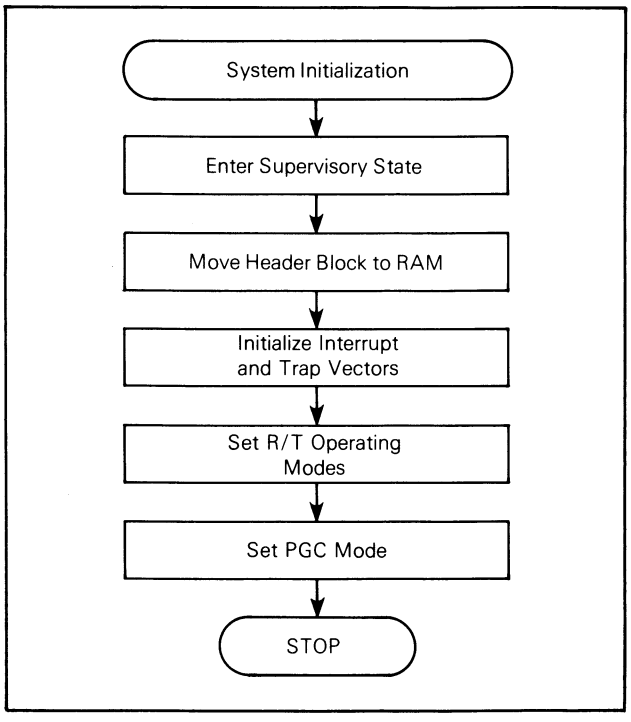


FIGURE 12 — Power Up Reset Initialization Sequence Flowchart

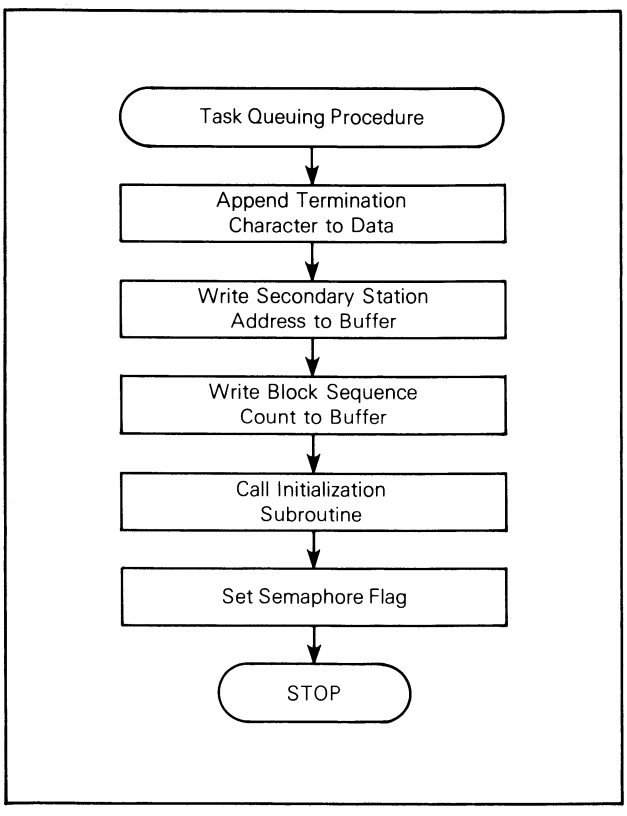


FIGURE 13 — Task Queuing Procedure Flowchart

The applications dependent initialization (which includes the clearing of flags, setting the initial value of counters and pointers, enabling interrupts, and enabling the transmitters and receivers) is not done at this time. Those functions are provided by the task initialization subroutines, which are called as each task is placed in the queue. The task queuing procedure flowchart is shown in Figure 13. Flowcharts for the transmitter task and receiver task initialization subroutines, are found in Figures 14 and 15, respectively.

In general, an Operating System will keep track of tasks within the system through the use of a task queue. The queue contains information on all current tasks, including the priority of each. Quasi-real-time operations, such as I/O drivers, are usually assigned the highest priority. Active tasks in the system are denoted by a non-zero value in the semaphore register for that task. Semaphore flags are used in the data link software to indicate if a receiver or transmitter is active; and if it is active, whether the message type is data or control. Single byte locations are used for all semaphore flags. The semaphore registers are also used as byte counters in several instances, since only a non-zero value is required for a device to be active.

To properly queue a transmitter task, a data block must be prepared for transmission as follows: an ETX character must be appended to the end of the data block, the starting address of the block must be moved to the appropriate transmitter data pointer (TC61DATP or TC52DATP), the secondary station address must be written into the header block, and the block sequence count updated (see Figure 13). Setting the transmitter semaphore flags (TX61SMPH or TX52SMPH) indicates that the task has been queued and transmission can commence. Receiver tasks should be continually enabled, anytime the corresponding transmitter is disabled. The actual I/O transfers are interrupt driven and are handled by the interrupt service routines.

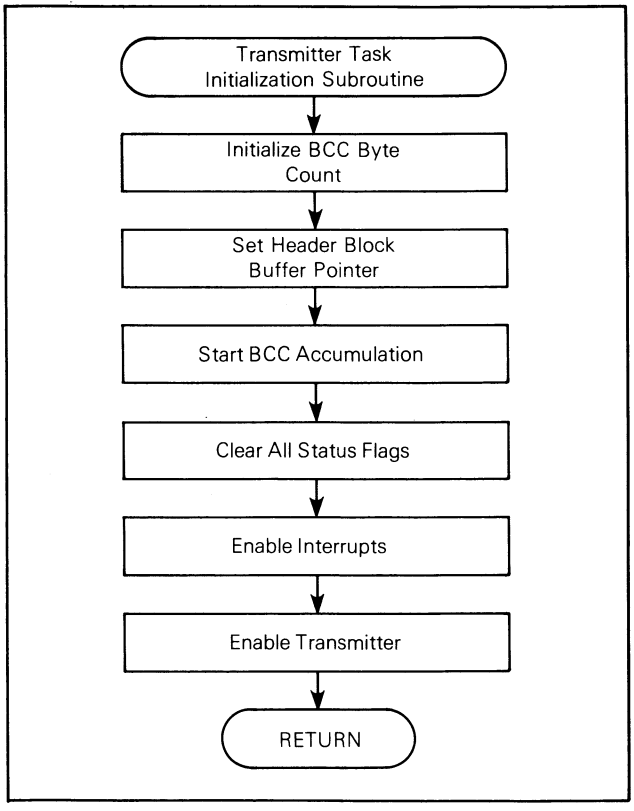


FIGURE 14 — Transmitter Task Initialization Flowchart

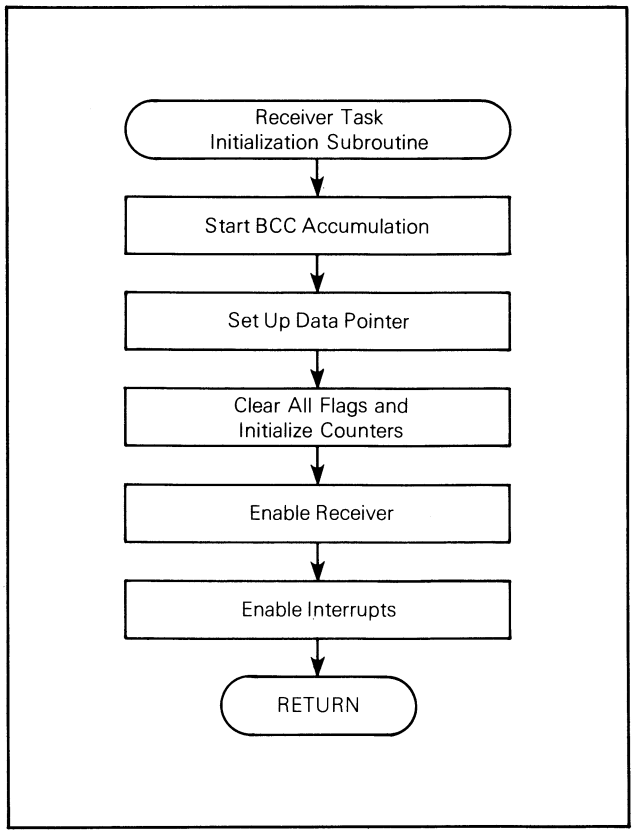


FIGURE 15 — Receiver Task Initialization Flowchart

INTERRUPT SERVICE ROUTINES

As the Receiver/Transmitter (R/T) completes the processing of a character, the CPU is interrupted to signal the completion. The CPU acknowledges the interrupt and is vectored off to a particular service routine. Each receiver and transmitter requires a separate service subroutine. Flowcharts for these subroutines are presented in Figures 16 and 17.

To service a transmitter interrupt, the work registers must be saved and the data pointer restored. Using the restored pointer, a byte of data is read from the buffer and written to the transmitter. The pointer is then incremented and saved. Depending on the type of message being sent, two paths are available. For an acknowledgement control message, the byte counter is incremented, the work registers restored, and the routine exited. In the case of a standard data message, checks are done to determine if the end of either the header or data block has been reached. At the completion of the service routine, the data pointer is updated as necessary, the interrupt serviced flag set, and the work registers restored. The receiver interrupt service routine also saves the work registers and restores the data pointers upon entry. The data is then read from the receiver. In the case of the MPCC, an additional segment of code must be executed. The MPCC does not strip SYN characters received after the initial two; therefore, a short section of code is required to sense and strip them. This is accentuated by the dotted lines in the flowchart. Valid characters cause the interrupt serviced flag to be set and the remaining functions performed. If this is part of an acknowledgement control message, the byte counter is incremented and a normal exit performed. For a data message, the PGC status is read and saved before exiting the routine. If an ETX was sensed by the PGC, a flag is set before returning.

This completes the real time processing of information by the data link software. The execution time of the remaining software is not critical, as long as it is completely executed before another interrupt (corresponding to the unexecuted code) is serviced.

I/O DRIVERS SUBROUTINES

Each I/O driver contains two paths through the code, corresponding to acknowledgement and data messages. The transmitter drivers (see Figure 18), clear the interrupt serviced flag before branching to the appropriate path. Until the fifth byte of an acknowledgement has been transmitted, no special processing is required and a return is simply executed. Following the fifth byte, the transmitter is turned off and the reply semaphore flag is cleared. Data messages also return, unless the BTC character has been transmitted. On subsequent passes after the BTC (ETX) transmission, a BCC byte is moved to the buffer and the byte count incremented. After the entire BCC has been transmitted, the transmitter is disabled, the receiver enabled, and the acknowledgement semaphore flag set.

A flowchart for the receiver I/O drivers is presented in Figure 19. Immediately after the interrupt serviced flag is cleared, a check is done to see whether the current message is data or an acknowledgement. If an acknowledgement, nothing happens unless the entire block has been received. At that point, a decision is made whether to retransmit the current block, or queue the next block for transmission. The decision is based upon the reception of a valid ACK. Data messages also cause the routine to terminate, unless the entire block check sequence has been received. When it has been received, the receiver is disabled, and a check done to see what the response should be. The proper response is loaded into the transmit buffer, the transmitter enabled and the reply semaphore flag is set.

ASYNCHRONOUS SUPPORT

As an asynchronous peripheral, the EPCI can be programmed for a variety of data formats. The number of stop bits and the baud rate clock divisor are selectable through software. Other software alterable control bits allow the user to force a break level on the transmitter output, enable an auto echo mode, and toggle modem control lines. The receiver monitors the receiver-serial data for parity, overrun, and framing errors, as well as detecting false start bits.

Either an external clock, or the built in baud rate generator can be selected as the source of the receive and transmit clocks. The internal baud rate generator allows the user to select any of 16 baud rates under program control. Three versions of the EPCI are available. Each version contains a different set of 16 baud rates ranging from 50 to 38.4 K baud.

Features common to both the synchronous and asynchronous modes of operation include a double buffered receiver and transmitter, full or half-duplex operation, and two maintenance loopback modes. Character length is software programmable to be from five to eight bits plus parity, and may be changed dynamically. All inputs and outputs are TTL compatible, except for three open-drain MOS outputs which are available for use in interrupt driven environments. Also, the RxC and TxC pins are both protected against short circuits.

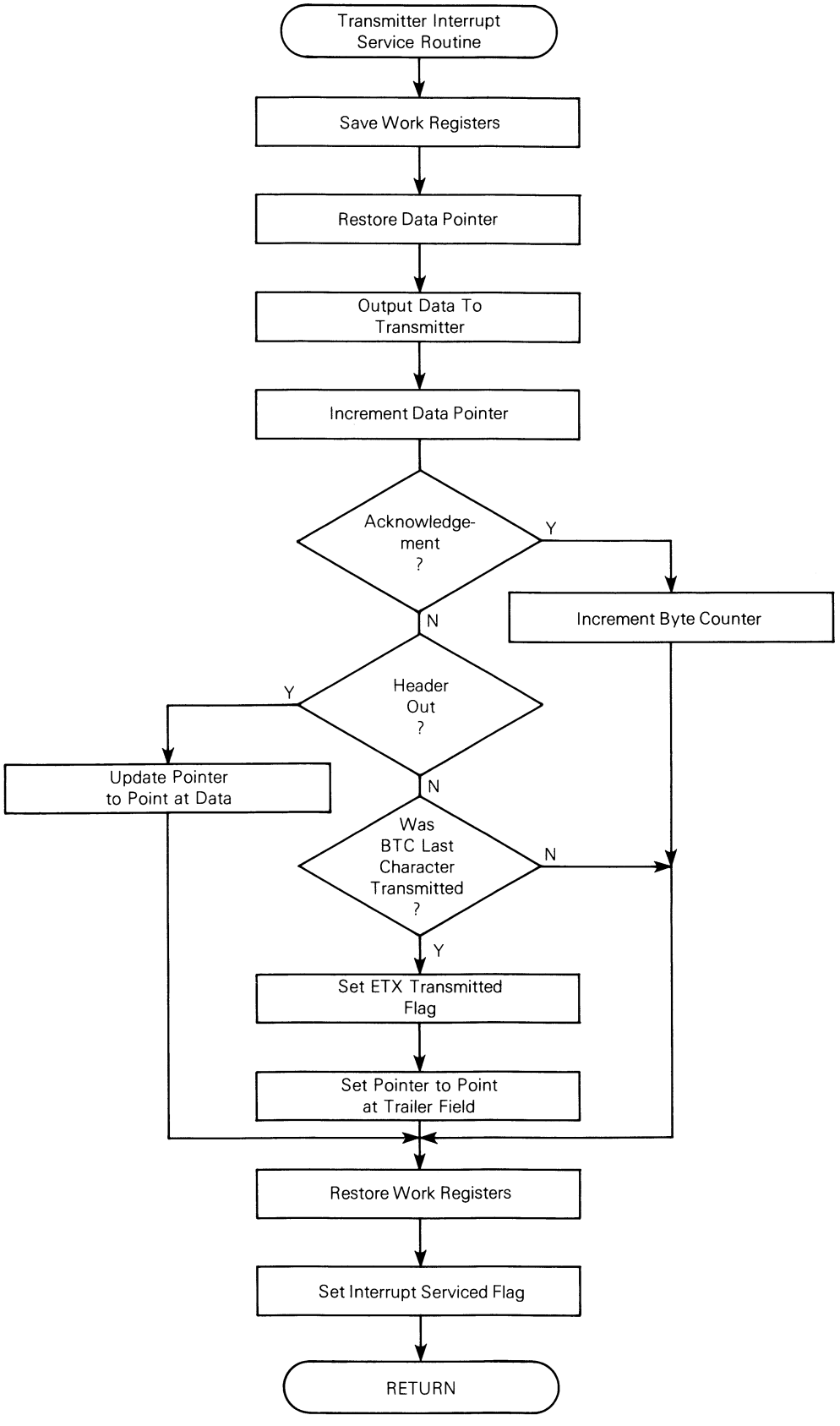


FIGURE 16 — Transmitter Interrupt Service Routine Flowchart

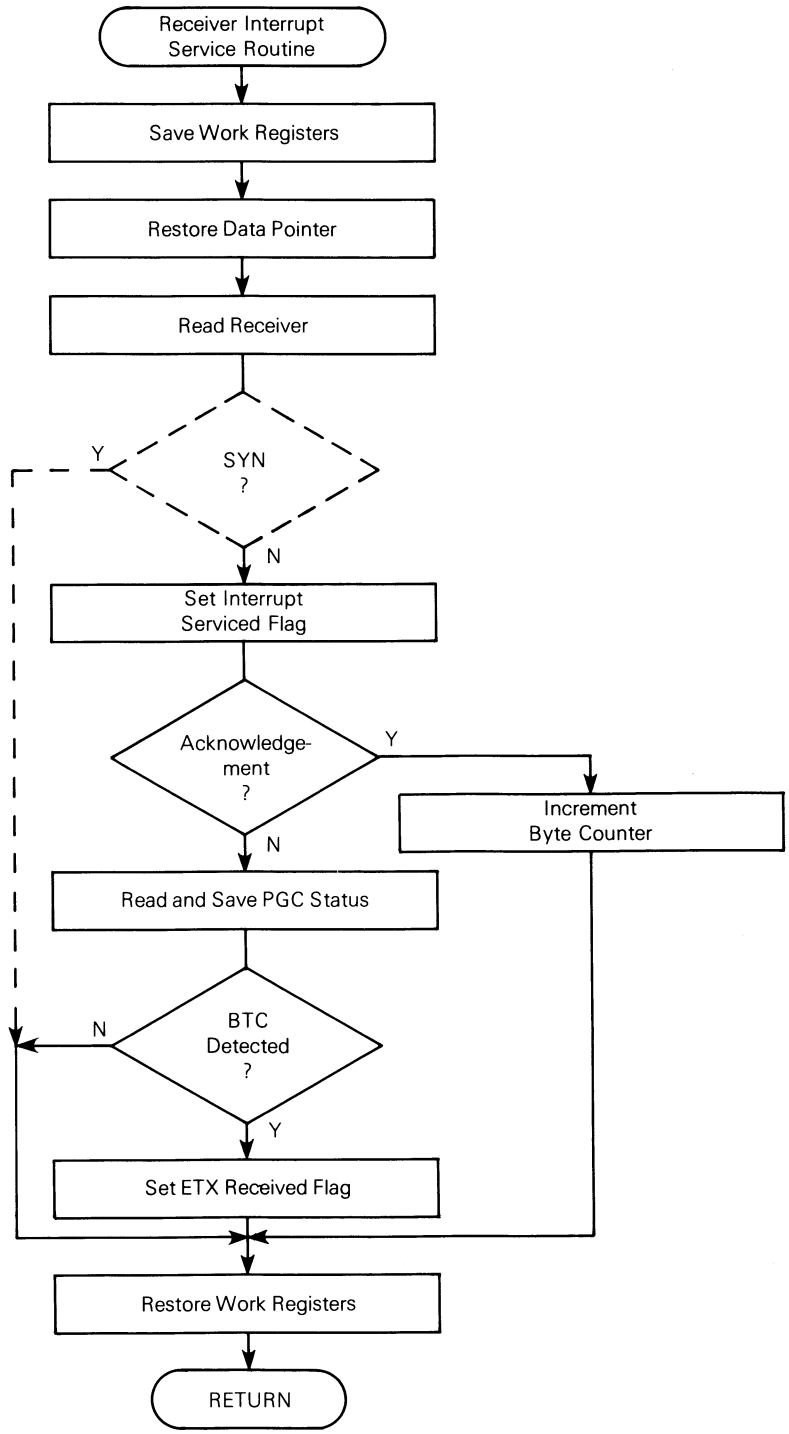


FIGURE 17 — Receiver Interrupt Service Routine Flowchart

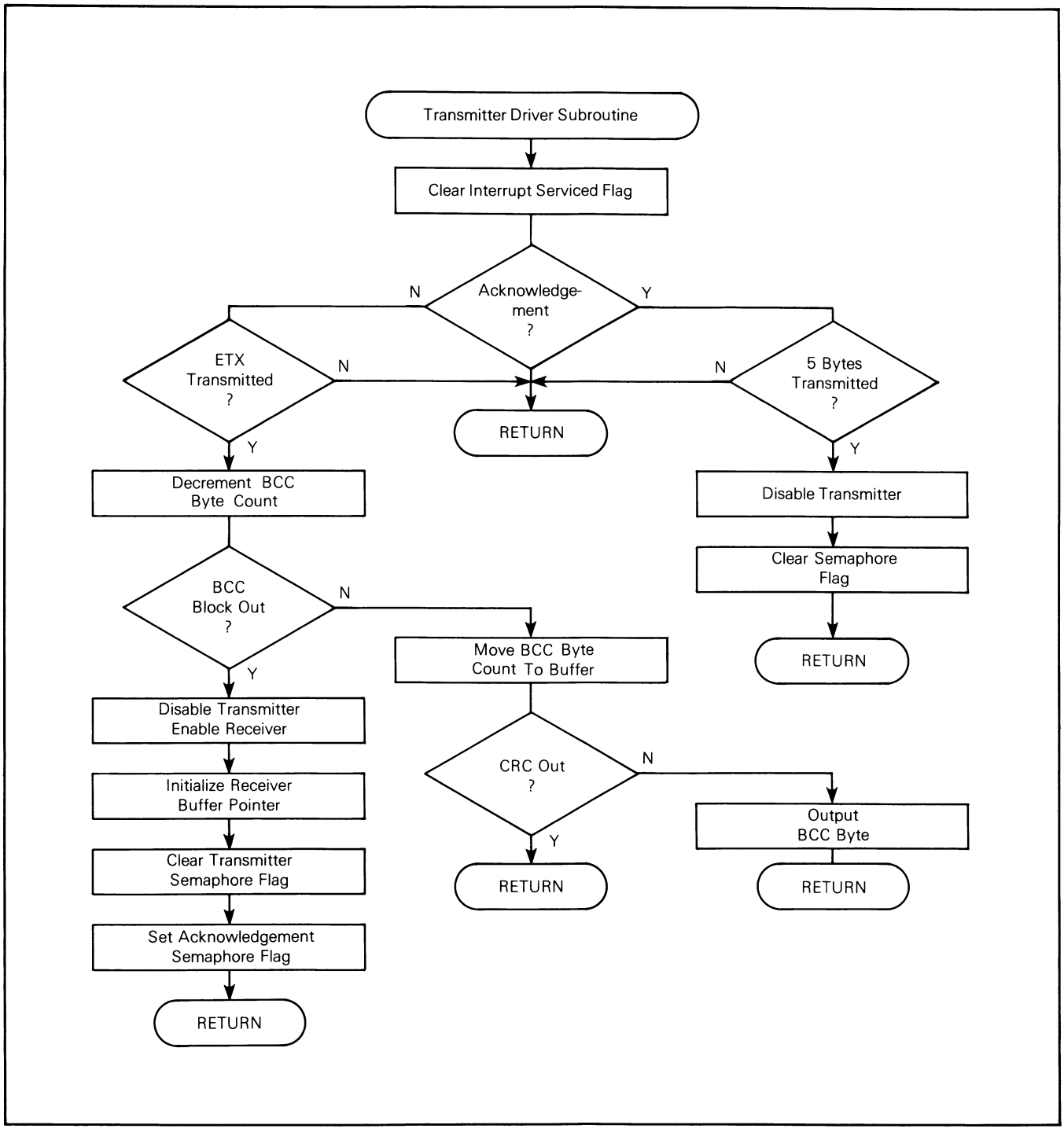


FIGURE 18 — Transmitter I/O Driver Flowchart

The MC68661/MC2661 can be programmed to operate asynchronously simply by re-initializing the device. This is done by executing the code shown in Figure 20 in place of the initialization routines presented earlier. Examples of typical interrupt service routines are also included in Figure 20. After executing the code, the EPCI will be programmed to operate asynchronously on 8-bit data characters without parity. The TxC and RxC are provided by the internal baud rate generator, with 9600 baud being the selected rate. If a different baud rate is desired, bits 0 thru 3 of Mode Register 2 should be changed according to Table 1 in the EPCI data sheet. Interrupts operate the same as in the synchronous mode.

BIT ORIENTED PROTOCOL (BOP) SUPPORT

The MC68652 is capable of supporting not only BCP, but several BOP protocols as well: SDLC, HDLC, and ADCCP. The features of the part, unique to the BOP mode of operation, include: secondary station address detection, zero insertion and deletion, and short character detection at the end of a message. Automatic generation and detection of special BOP control sequences such as FLAG, ABORT, and GA are

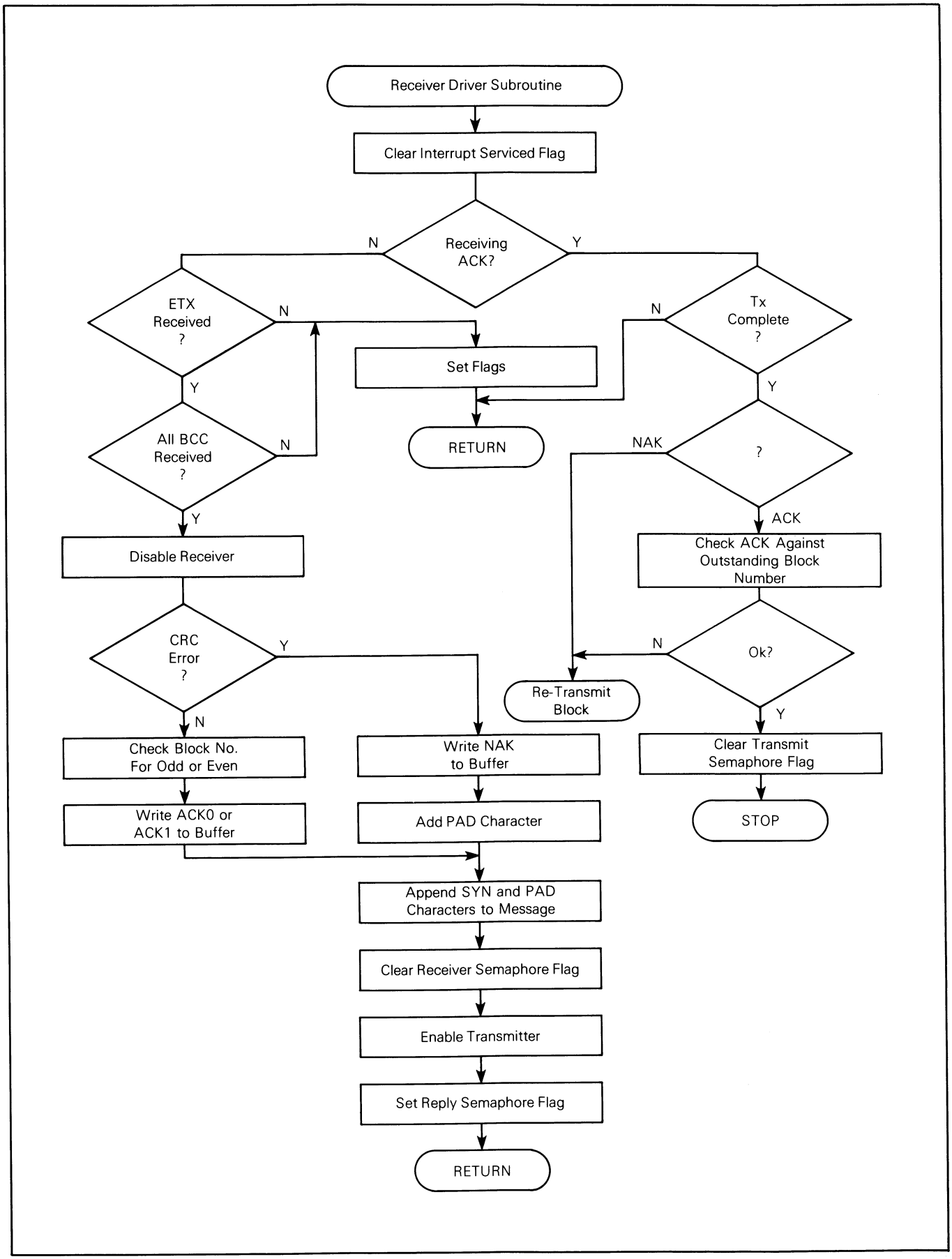


FIGURE 19 — Receiver I/O Driver Flowchart

```
MOTOROLA M68000 ASM VERSION  1.20 SYS :   8.        .FIGURE20.SA 01/06/82 11:59:00


  1              00000000              ORG       0
  2              00FFFE05      MR61    EQU       $FFFE05              MODE REGISTER 1 AND 2
  3              00FFFE07      CR61    EQU       $FFFE07              COMMAND REGISTER
  4              00FFFE03      SR61    EQU       $FFFE03              STATUS REGISTER
  5              00FFFE01      RHR61   EQU       $FFFE01              RECEIVE HOLDING REGISTER
  6              00FFFE01      THR61   EQU       $FFFE01              TRANSMIT HOLDING REGISTER
  7              00000A00      RX61PT  EQU       $A00                 RECEIVER DATA BUFFER POINTER
  8              00000A04      TX61PT  EQU       $A04                 TRANSMIT DATA POINTER
  9                            *
 10                            *       MC68661 ASYNCHRONOUS INITIALIZATION SUBROUTINE
 11                            *       DATA FORMAT IS ASYNCHRONOUS, 9600 BAUD, 8 DATA
 12                            *       BITS, 2 STOP BITS AND NO PARITY.  CLOCK SOURCE
 13                            *       IS THE INTERNAL BAUD RATE GENERATOR.
 14                            *
 15  00000000 13FC00CE00FF INIT      MOVE.B    #$CE,MR61            INITIALIZE OPERATING MODE
                FE05
 16  00000008 13FC007E00FF            MOVE.B    #$7E,MR61            SET BAUD RATE & CLOCK SOURCE
                FE05
 17  00000010 13FC002700FF            MOVE.B    #$27,CR61
                FE07
 18  00000018 4E75                    RTS
 19                            *
 20                            *       RECEIVER INTERUPT SERVICE ROUTINE
 21                            *
 22  0000001A 48E78080        RINT61  MOVEM.L   D0/A0,-(SP)          SAVE WORK REGISTERS
 23  0000001E 20780A00                MOVEA.L   RX61PT,A0            RESTORE DATA POINTER
 24  00000022 103900FFFE03            MOVE.B    SR61,D0              CHECK EPCI STATUS
 25  00000028 E458                    ROR.W     #2,D0
 26  0000002A 6410                    BCC.S     RERR                 ERROR IF NO CHARACTER AVAILABLE
 27  0000002C 10F900FFFE01            MOVE.B    RHR61,(A0)+          MOVE CHARACTER TO BUFFER
 28  00000032 21C80A00                MOVE.L    A0,RX61PT            SAVE NEW DATA POINTER
 29  00000036 4CDF0101                MOVEM.L   (SP)+,D0/A0          RESTORE WORK REGISTERS
 30  0000003A 4E73                    RTE                            BEFORE RETURNING
 31  0000003C 4E4F            RERR    TRAP      #15                  RECOVER FROM ERROR
 32                            *
 33                            *       TRANSMITTER INTERRUPT SERVICE ROUTINE
 34                            *
 35  0000003E 48E78080        TINT61  MOVEM.L   D0/A0,-(SP)          SAVE WORK REGISTERS
 36  00000042 20780A04                MOVEA.L   TX61PT,A0            RESTORE DATA POINTER
 37  00000046 103900FFFE03            MOVE.B    SR61,D0              CHECK EPCI STATUS
 38  0000004C E258                    ROR.W     #1,D0
 39  0000004E 6410                    BCC.S     TERR                 ERROR IF TX NOT READY
 40  00000050 13D800FFFE01            MOVE.B    (A0)+,THR61          WRITE CHAR TO EPCI
 41  00000056 21C80A04                MOVE.L    A0,TX61PT            AND INCREMENT ADATA POINTER
 42  0000005A 4CDF0101                MOVEM.L   (SP)+,D0/A0          RESTORE THE REGISTERS
 43  0000005E 4E73                    RTE
 44  00000060 4E4F            TERR    TRAP      #15
 45                                    END
```

FIGURE 20 — EPCI Asynchronous Initialization Software

also included. As would be expected for a bit oriented protocol, character length can range from one to eight bits. Error checking facilities, which are built into the MPCC, enable the device to perform parity and CRC checks. The available error checking polynomials include odd and even parity (VRC), CRC-CCITT, and CRC-16.

As with the EPCI, only simple software changes are required to change the operating parameters of the peripheral. No hardware changes are required. Depending on the protocol which is to be supported, the value written to the MPCC during initialization will vary. Consult the MC2652/MC68652 data sheet for the proper values. Additional information on BOP applications is available from Motorola Applications Note AN-839, "A Data Communications System Using an MC6809 MPU, MC68652 MPCC, and/or the MC68661 EPCI."

POTENTIAL PROBLEM AREAS

Several potential problems exist if the data sheets are not read closely before using the data communications peripherals. The next few paragraphs cover several situations in which problems could develop.

Problems can start to develop even as the EPCI is being initialized. The SYN1, SYN2 and DLE character registers are programmed by writing to the same memory location. Internally, the EPCI rotates from the SYN1 register, to the SYN2

register, to the DLE register, on subsequent writes. If more than the required number of accesses are made, the internal pointer sequences back to the first register. To ensure that the pointer starts in the correct place, it is good practice to read the command register before programming these characters. A read of the command register resets the pointer to the SYN1 register. Mode registers 1 and 2 are accessed in a similar manner.

The EPCI is double buffered on both the receiver input and transmitter output. In the transmitting mode, one character is shifted out of the Transmit Shift Register (TSR) as the next character is being held in the Transmit Holding Register (THR). If the transmitter is disabled while both the TSR and the THR are full, the contents of the TSR will continue to be shifted out until the register is empty; however, the contents of the THR will not be transferred to the TSR and will be lost. When disabling the transmitter after writing the last character of a message to the EPCI, the user must wait until TxRDY is asserted again, before disabling TxEN (in Command Register).

During periods of transmitter underrun, while in the BISYNC transparent mode, line fill characters will consist of a DLE/SYN pair. The MC68661 receiver must be switched to the transparent mode, for these character pairs to be properly detected and stripped. Entering the transparent mode too early can cause problems if line fill is required during the header block. To alleviate these problems, the receiver should not be switched to the transparent mode until after the DLE/STX pair has been received.

The MPCC is a powerful peripheral; however, several precautions should be observed when using it. Besides the hardware anamolies which have been described previously, the system programmer must be made aware of several software exceptions. The MC68652 pins which have been designated as possible interrupt sources are not truly interrupts, but status signals. As a result, it is not sufficient to disable the transmitter to disable the transmitter interrupts. Instead, a dummy byte of data must be written to the transmitter, clearing the TxRE status flag.

A final precaution when using the MPCC is that address line A0 on the MC68652 is the logical inversion of the MC68000 internal A0. This places MPCC odd addresses in the high byte and even addresses in the low byte. In the data link hardware, this problem was avoided by using the lower data strobe instead of the upper data strobe, in generating A0 for the MPCC.

CONCLUSION

A Binary Synchronous Communications (BISYNC) data link, requiring a minimal amount of hardware and software overhead, was described in this application note. The hardware consists of an MC68000 asynchronous bus interface, interrupt prioritizing and vector generation logic, and the auxillary support circuitry required by the peripherals. The I/O drivers and interrupt service routines are provided as an example of the software intervention that is actually required to support the data link.

Using the hardware already described with different software drivers, virtually any common serial protocol can be supported. Using the MC68661 EPCI, asynchronous and byte controlled synchronous protocols can be supported. In the same light, the MC68652 MPCC can support either byte controlled or bit oriented synchronous protocols. If the internal error checking facilities are insufficient for a given protocol, an MC68653 PGC will, in most cases, provide the necessary functions.

A scheme for vectoring interrupt requests was also described. This method of prioritizing the interrupts is in no way limited to the peripherals employed in this application. Any Motorola family peripheral can be provided with vectored interrupts using this scheme. Multiple interrupts on the same level are allowed, provided that each device is assigned a unique vector number and interrupts are "daisy chained" between boards.

REFERENCES

1. "Seminar on Basic Data Communications Techniques and Introduction to LSI communications Circuits," Student Notes, Signetics Corporation, Sunnyvale, California, 1981, pp. 1.2-2.14.

2. "Data Communications Handbook," Signetics Corporation, Sunnyvale, California, 1981, pp. 43-54, 141.

3. "MC68000 16-Bit Microprocessing Unit" Data Sheet, Motorola Inc., Austin, Texas, 1981.

4. "MC2661/MC68661 Enhanced Programmable Communications Interface" Data Sheet, Motorola Inc., Austin, Texas, 1981.

5. "MC2653/MC68653 Polynomial Generator/Checker" Data Sheet, Motorola Inc., Austin, Texas, 1981.

6. "MC2652/MC68652 Multi-Protocol Communications Controller" Data Sheet, Motorola Inc., Austin, Texas, 1981.

7. "Using the 2653 Polynomial Generator and Checker," Applications Note 400, Signetics Corporation, Sunnyvale, California, 1981.

8. "Communicating Data with Protocols," Sandra E. Traylor, Digital Design, July 1980, 40-44.

9. "MC68000 16-Bit Microprocessor User's Manual," Third Edition, Motorola Inc., Austin, Texas, 1982.

10. "A Data Communications System Using an MC6809 MPU, MC68652 MPCC, and/or MC68661 EPCI," AN-839, Motorola Inc., Austin, Texas, 1981.

11. "General Information — Binary Synchronous Communications," IBM Systems Reference Library, Publication number GA27-3004, International Business Machines, White Plains, N.Y.

```
MOTOROLA M68000 ASM VERSION  1.20 SYS :   8.        .SOFTART .SA 01/06/82 10:59:47


 1              00000A00                ORG     $A00
 2                              **********************************************************************
 3                              *                                                                    *
 4                              *       MC68661/MC2661 INTERNAL REGISTERS                            *
 5                              *                                                                    *
 6                              **********************************************************************
 7              00FFFE05        MR61    EQU     $FFFE05                 MODE REGISTERS 1 AND 2
 8              00FFFE07        CR61    EQU     $FFFE07                 COMMAND REGISTER
 9              00FFFE03        SR61    EQU     $FFFE03                 STATUS/SYNC REGISTER
10              00FFFE01        RHR61   EQU     $FFFE01                 RECEIVE DATA HOLDING REGISTER
11              00FFFE01        THR61   EQU     $FFFE01                 TRANSMIT DATA HOLDING REGISTER
12                              **********************************************************************
13                              *                                                                    *
14                              *       MC68652/MC2652 INTERNAL AND AUXILLARY REGISTERS              *
15                              *                                                                    *
16                              **********************************************************************
17              00FFFE4A        PCR     EQU     $FFFE4A                 PARAMETER CONTROL REGISTER
18              00FFFE48        PCSAR   EQU     $FFFE48                 PARAMETER CONTROL SYNC/ADDRESS REGISTER
19              00FFFE48        PCSARH  EQU     $FFFE48                 HIGH BYTE OF PCSAR
20              00FFFE49        PCSARL  EQU     $FFFE49                 LOW BYTE OF PCSAR
21              00FFFE43        TDR     EQU     $FFFE43                 TRANSMIT DATA REGISTER
22              00FFFE42        TSR     EQU     $FFFE42                 TRANSMIT STATUS REGISTER
23              00FFFE41        RDR     EQU     $FFFE41                 RECEIVE DATA REGISTER
24              00FFFE40        RSR     EQU     $FFFE40                 RECEIVE STATUS REGISTER
25              00FFFE60        ASR     EQU     $FFFE60                 AUXILLARY STATUS REGISTER
26              00FFFE60        TREN    EQU     $FFFE60                 TRANSMITTER/RECEIVER ENABLE LATCH
27                              **********************************************************************
28                              *                                                                    *
29                              *       MC68653/MC2653 INTERNAL REGISTERS                            *
30                              *                                                                    *
31                              **********************************************************************
32              00FFFE25        MR531   EQU     $FFFE25                 MODE REGISTER MC68653-1
33              00FFFE69        MR532   EQU     $FFFE69                 MODE REGISTER MC68653-2
34              00FFFE23        CR531   EQU     $FFFE23                 COMMAND REGISTER MC68653-1
35              00FFFE65        CR532   EQU     $FFFE65                 COMMAND REGISTER MC68653-2
36              00FFFE23        SR531   EQU     $FFFE23                 STATUS REGISTER MC68653-1
37              00FFFE65        SR532   EQU     $FFFE65                 STATUS REGISTER MC68653-2
38              00FFFE21        CCA531  EQU     $FFFE21                 CHARACTER CLASS ARRAY MC68653-1
39              00FFFE61        CCA532  EQU     $FFFE61                 CHARACTER CLASS ARRAY MC68653-2
40              00FFFE27        BCC531  EQU     $FFFE27                 BLOCK CHECK CHARACTER MC68653-1
41              00FFFE6D        BCC532  EQU     $FFFE6D                 BLOCK CHECK CHARACTER MC68653-2
42                              **********************************************************************
43                              *                                                                    *
44                              *       TASK CONTROL BLOCK FLAGS AND TEMPORARY STORAGE               *
45                              *                                                                    *
46                              **********************************************************************
47  00000A00 00                 TX61SMPH DC.B   0                       TRANSMIT BUFFER READY SEMAPHORE
48  00000A01 00                 TCB61DRI DC.B   0                       TRANSMIT INTERRUPT SERVICED FLAG
49  00000A02 00                 TCB61RVI DC.B   0                       RECEIVED INTERRUPT SERVICED FLAG
50  00000A03 00                 TCSR531F DC.B   0                       PGC STATUS READ FLAG
51  00000A04 00                 TCBSR531 DC.B   0                       PGC STATUS TEMPORARY STORAGE
52  00000A05 00                 ETXF61   DC.B   0                       END OF TEXT RECEIVED FLAG
53  00000A06 00000000           TC61RXPT DC.L   0                       RECEIVE BUFFER POINTER
54  00000A0A 00000000           TC61TXPT DC.L   0                       TRANSMIT BUFFER POINTER
55                              *
56                              *       THE OUTPUT DATA POINTER MUST BE INITIALIZED FOR THE
57                              *       STARTING ADDRESS FOR EACH MESSAGE BLOCK.
58                              *
```

```
MOTOROLA M68000 ASM VERSION  1.20 SYS :   8.        .SOFTART .SA 01/06/82 10:59:47


  59   00000A0E 00000000      TC61DATP DC.L     0                    MC68661 OUTPUT DATA POINTER
  60   00000A12 0000          D61CNT   DC.W     0                    OUTPUT CRC COUNT
  61            00000940      TXTBUF61 EQU      $940                 OVERHEAD BUFFER START
  62            00000820      RXBUF61  EQU      $820                 MC68661 INPUT BUFFER START
  63            00000062      CR61MSK  EQU      $62                  MC68661 CONTROL REGISTER MASK
  64   00000A14 00            TX52SMPH DC.B     0                    TRANSMIT BUFFER READY SEMAPHORE
  65   00000A15 00            TCB52DRI DC.B     0                    TRANSMIT INTERRUPT SERVICED FLAG
  66   00000A16 00            TCB52RVI DC.B     0                    RECEIVED INTERRUPT SERVICED FLAG
  67   00000A17 00            TCSR532F DC.B     0                    PGC STATUS READ FLAG
  68   00000A18 00            TCBSR532 DC.B     0                    PGC STATUS TEMPORARY STORAGE
  69   00000A1A 00000000      TC52RXPT DC.L     0                    RECEIVE BUFFER POINTER
  70   00000A1E 00000000      TC52TXPT DC.L     0                    TRANSMIT BUFFER POINTER
  71   00000A22 00000000      TC52DATP DC.L     0                    MC68652 OUTPUT DATA POINTER
  72   00000A26 0000          D52CNT   DC.W     0                    OUTPUT CRC COUNT
  73   00000A28 00            ETXF52   DC.B     0                    ETX RECEIVED FLAG
  74            00000980      TXTBUF52 EQU      $980                 OVERHEAD BUFFER START
  75            00000700      RXBUF52  EQU      $700                 MC68652 INPUT BUFFER START
  76   00000A29 00            REP61SPH DC.B     0                    TRANSMIT ACKNOWLEDGE SEMAPHORE
  77   00000A2A 00            REP52SPH DC.B     0                    TRANSMIT ACKNOWLEDGE SEMAPHORE
  78   00000A2B 00            TCB61SPH DC.B     0                    GENERAL PURPOSE SEMAPHORES
  79   00000A2C 00            TCB52SPH DC.B     0
  80   00000A2D 00            ACK61SPH DC.B     0
  81   00000A2E 00            ACK52SPH DC.B     0
  82                          *
  83                          *         THE REMAINDER OF THE TABLE IS ROMABLE
  84                          *
  85   00000A2F 03            INITBUF  DC.B     03                   53 START UP MODE
  86   00000A30 04                     DC.B     04                   53 SYN/NI CLASS
  87   00000A31 16                     DC.B     $16                  SYN
  88   00000A32 01                     DC.B     01                   SOH
  89   00000A33 08                     DC.B     08                   BTC/SC CLASS
  90   00000A34 03                     DC.B     03                   ETX
  91   00000A35 17                     DC.B     $17                  ETB
  92   00000A36 05                     DC.B     05                   ENQ
  93   00000A37 04                     DC.B     04                   EOT
  94   00000A38 0C                     DC.B     $0C                  SSC CLASS
  95   00000A39 02                     DC.B     02                   STX
  96   00000A3A 16            DUMDAT   DC.B     $16                  SYN
  97   00000A3B 16                     DC.B     $16                  SYN
  98   00000A3C 16                     DC.B     $16                  SYN
  99   00000A3D 01                     DC.B     01                   SOH
 100   00000A3E 00            STRTADR  DC.B     0                    STATION ADDRESS
 101   00000A3F 00                     DC.B     0
 102   00000A40 00            BLKSEQ   DC.B     0                    BLOCK SEQUENCE
 103   00000A41 02                     DC.B     02                   STX
 104   00000A42 00                     DC.B     0                    BCC
 105   00000A43 00                     DC.B     0
 106   00000A44 16                     DC.B     $16                  SYN AS PAD
 107                          ********************************************************************************
 108                          *                                                                              *
 109                          *         MC68661/MC2661 TRANSMITTER DRIVER INITIALIZATION                     *
 110                          *                                                                              *
 111                          ********************************************************************************
 112   00000A46 31FC00030A12  INIT61TX MOVE.W   #3,D61CNT            INITIALIZE BYTE COUNT
 113   00000A4C 21FC00000940          MOVE.L   #TXTBUF61,TC61TXPT    OVERHEAD BUFFER POINTER
                0A0A
 114   00000A54 13FC008500FF          MOVE.B   #$85,MR531            SET '61 PGC MODE
                FE25
```

```
MOTOROLA M68000 ASM VERSION  1.20 SYS :   8.        .SOFTART .SA 01/06/82 10:59:47


 115   00000A5C 13FC000200FF          MOVE.B   #02,CR531             START BCC ACCUMULATION
                FE23
 116   00000A64 4200                  CLR.B    D0
 117   00000A66 11C00A05              MOVE.B   D0,ETXF61             ZERO OUT FLAGS
 118   00000A6A 11C00A01              MOVE.B   D0,TCB61DRI
 119   00000A6E 11C00A2D              MOVE.B   D0,ACK61SPH
 120   00000A72 46FC2000              MOVE     #$2000,SR             ENABLE INTERRUPTS
 121   00000A76 103C0062              MOVE.B   #CR61MSK,D0
 122   00000A7A 00000001              OR.B     #01,D0
 123   00000A7E 13C000FFFE07          MOVE.B   D0,CR61               ENABLE '61 TRANSMITTER
 124   00000A84 4E75                  RTS
 125                          ********************************************************************************
 126                          *                                                                              *
 127                          *        MC68661/MC2661 RECEIVER DRIVER INITIALIZATION                         *
 128                          *                                                                              *
 129                          ********************************************************************************
 130   00000A86 13FC008100FF INIT61RX MOVE.B   #$81,MR531            SET '61 PGC MODE
                FE25
 131   00000A8E 13FC000200FF          MOVE.B   #$02,CR531            START BCC ACCUMULATION
                FE23
 132   00000A96 21FC00000820          MOVE.L   #RXBUF61,TC61RXPT     POINTER TO DATA BUFFER
                0A06
 133   00000A9E 4200                  CLR.B    D0
 134   00000AA0 11C00A05              MOVE.B   D0,ETXF61             ZERO OUT FLAGS
 135   00000AA4 11C00A02              MOVE.B   D0,TCB61RVI
 136   00000AA8 11C00A03              MOVE.B   D0,TCSR531F
 137   00000AAC 11C00A29              MOVE.B   D0,REP61SPH
 138   00000AB0 103C0062              MOVE.B   #CR61MSK,D0
 139   00000AB4 00000004              OR.B     #04,D0                ENABLE '61 RECEIVER
 140   00000AB8 13C000FFFE07          MOVE.B   D0,CR61
 141   00000ABE 46FC2000              MOVE     #$2000,SR             ENABLE INTERRUPTS
 142   00000AC2 4E75                  RTS
 143                          ********************************************************************************
 144                          *                                                                              *
 145                          *        MC68652/MC2652 TRANSMITTER DRIVER INITIALIZATION                      *
 146                          *                                                                              *
 147                          ********************************************************************************
 148   00000AC4 31FC00030A26 INIT52TX MOVE.W   #3,D52CNT              INITIALIZE BYTE COUNT
 149   00000ACA 21FC00000980          MOVE.L   #TXTBUF52,TC52TXPT    OVERHEAD BUFFER POINTER
                0A1E
 150   00000AD2 13FC008500FF          MOVE.B   #$85,MR532            SET '52 PGC MODE
                FE69
 151   00000ADA 13FC000200FF          MOVE.B   #02,CR532             START BCC ACCUMULATION
                FE65
 152   00000AE2 4200                  CLR.B    D0
 153   00000AE4 11C00A28              MOVE.B   D0,ETXF52             CLEAR STATUS FLAGS
 154   00000AE8 11C00A15              MOVE.B   D0,TCB52DRI
 155   00000AEC 11C00A2E              MOVE.B   D0,ACK52SPH
 156   00000AF0 46FC2000              MOVE     #$2000,SR             ENABLE INTERRUPTS
 157   00000AF4 13FC000100FF          MOVE.B   #01,TREN              ENABLE '52 TRANSMITTER
                FE60
 158   00000AFC 4E75                  RTS
 159                          ********************************************************************************
 160                          *                                                                              *
 161                          *        MC68652/MC2652 RECEIVER DRIVER INTIALIZATION                          *
 162                          *                                                                              *
 163                          ********************************************************************************
 164   00000AFE 13FC008100FF INIT52RX MOVE.B   #$81,MR532            SET '52 PGC MODE
```

```
MOTOROLA M68000 ASM VERSION  1.20 SYS :   8.        .SOFTART .SA 01/06/82 10:59:47


                FE69
 165   00000B06 13FC000200FF          MOVE.B   #$02,CR532             START BCC ACCUMULATION
                FE65
 166   00000B0E 21FC00000700          MOVE.L   #RXBUF52,TC52RXPT      POINTER TO DATA BUFFER
                0A1A
 167   00000B16 4200                  CLR.B    D0
 168   00000B18 11C00A28              MOVE.B   D0,ETXF52              ZERO OUT FLAGS
 169   00000B1C 11C00A16              MOVE.B   D0,TCB52RVI
 170   00000B20 11C00A17              MOVE.B   D0,TCSR532F
 171   00000B24 11C00A2A              MOVE.B   D0,REP52SPH
 172   00000B28 13FC000200FF          MOVE.B   #02,TREN               ENABLE '52 RECEIVER
                FE60
 173   00000B30 46FC2000              MOVE     #$2000,SR
 174   00000B34 4E75                  RTS
 175                          ********************************************************************************
 176                          *                                                                              *
 177                          *       HEADER BLOCK INITIALIZATION                                            *
 178                          *                                                                              *
 179                          ********************************************************************************
 180   00000B36 21FC00000B40 START    MOVE.L   #START1,$080           INITIALIZE TRAP 0 VECTOR
                0080
 181   00000B3E 4E40                  TRAP     #0                     GO TO SUPERVISORY MODE
 182   00000B40 46FC2700      START1  MOVE     #$2700,SR              MASK OFF INTERRUPTS
 183   00000B44 247C00000980          MOVEA.L  #TXTBUF52,A2           SETUP DATA POINTERS
 184   00000B4A 207C00000940          MOVEA.L  #TXTBUF61,A0
 185   00000B50 227C00000A3A          MOVEA.L  #DUMDAT,A1
 186   00000B56 303C000A              MOVE     #10,D0
 187   00000B5A 10D1          DBINIT  MOVE.B   (A1),(A0)+             MOVE OVERHEAD DATA
 188   00000B5C 14D9                  MOVE.B   (A1)+,(A2)+            TO RAM BUFFER
 189   00000B5E 51C8FFFA              DBRA     D0,DBINIT
 190                          ********************************************************************************
 191                          *                                                                              *
 192                          *       INTERRUPT VECTOR INITIALIZATION                                        *
 193                          *                                                                              *
 194                          ********************************************************************************
 195   00000B62 307C0104              MOVEA    #$104,A0
 196   00000B66 20FC00000F4C          MOVE.L   #IS61TX,(A0)+          IRQ 1 VECTOR
 197   00000B6C 20FC00000F10          MOVE.L   #IS61RX,(A0)+          IRQ 2 VECTOR
 198   00000B72 20FC00000F9A          MOVE.L   #IS52TX,(A0)+          IRQ 3 VECTOR
 199   00000B78 20FC00000ECE          MOVE.L   #IS52RX,(A0)+          IRQ 4 VECTOR
 200                          ********************************************************************************
 201                          *                                                                              *
 202                          *       MC68661/MC2661 INITIALIZATION                                          *
 203                          *                                                                              *
 204                          ********************************************************************************
 205   00000B7E 103900FFFE07 INIT61   MOVE.B   CR61,D0                 RESET REGISTER POINTER
 206   00000B84 207C00FFFE03          MOVEA.L  #SR61,A0
 207   00000B8A 13FC008C00FF          MOVE.B   #$8C,MR61              DEFINE FORMAT
                FE05
 208   00000B92 13FC002C00FF          MOVE.B   #$2C,MR61              CHANGE TO $0C FOR EXT. CLOCK
                FE05
 209   00000B9A 10BC0016              MOVE.B   #$16,(A0)              SYN1
 210   00000B9E 10BC0016              MOVE.B   #$16,(A0)              SYN2
 211   00000BA2 10BC0010              MOVE.B   #$10,(A0)              DLE
 212   00000BA6 13FC006200FF          MOVE.B   #CR61MSK,CR61          DISABLE TX AND RX TILL INIT COMPLETE
                FE07
 213                          ********************************************************************************
 214                          *                                                                              *
```

```
MOTOROLA M68000 ASM VERSION  1.20 SYS :   8.        .SOFTART .SA 01/06/82 10:59:47


                                *       MC68653/MC2653 INITIALIZATION                                   *
                                *                                                                       *
                                ************************************************************************
00000BAE 207C00FFFE23 INIT531   MOVEA.L  #CR531,A0
00000BB4 227C00FFFE21           MOVEA.L  #CCA531,A1
00000BBA 247C00000A2F           MOVEA.L  #INITBUF,A2
00000BC0 109A                   MOVE.B   (A2)+,(A0)             RESET TO START UP MODE
00000BC2 109A                   MOVE.B   (A2)+,(A0)             SYN/NI CLASS
00000BC4 129A                   MOVE.B   (A2)+,(A1)             WRITE SYN TO CCA
00000BC6 129A                   MOVE.B   (A2)+,(A1)             SOH
00000BC8 109A                   MOVE.B   (A2)+,(A0)             BTC/SC CLASS
00000BCA 129A                   MOVE.B   (A2)+,(A1)             ETX
00000BCC 129A                   MOVE.B   (A2)+,(A1)             ETB
00000BCE 129A                   MOVE.B   (A2)+,(A1)             ENQ
00000BD0 129A                   MOVE.B   (A2)+,(A1)             EOT
00000BD2 109A                   MOVE.B   (A2)+,(A0)             SSC CLASS
00000BD4 129A                   MOVE.B   (A2)+,(A1)             STX
                                ************************************************************************
                                *                                                                       *
                                *       MC68653/MC2653 INITIALIZATION                                   *
                                *                                                                       *
                                ************************************************************************
00000BD6 207C00FFFE65 INIT532   MOVEA.L  #CR532,A0
00000BDC 227C00FFFE61           MOVEA.L  #CCA532,A1
00000BE2 247C00000A2F           MOVEA.L  #INITBUF,A2
00000BE8 109A                   MOVE.B   (A2)+,(A0)             RESET TO START UP MODE
00000BEA 109A                   MOVE.B   (A2)+,(A0)             SYN/NI CLASS
00000BEC 129A                   MOVE.B   (A2)+,(A1)             WRITE SYN TO CCA
00000BEE 129A                   MOVE.B   (A2)+,(A1)             SOH
00000BF0 109A                   MOVE.B   (A2)+,(A0)             BTC/SC CLASS
00000BF2 129A                   MOVE.B   (A2)+,(A1)             ETX
00000BF4 129A                   MOVE.B   (A2)+,(A1)             ETB
00000BF6 129A                   MOVE.B   (A2)+,(A1)             ENQ
00000BF8 129A                   MOVE.B   (A2)+,(A1)             EOT
00000BFA 109A                   MOVE.B   (A2)+,(A0)             SSC CLASS
00000BFC 129A                   MOVE.B   (A2)+,(A1)             STX
                                ************************************************************************
                                *                                                                       *
                                *       MC68652/MC2652 INITIALIZATION                                   *
                                *                                                                       *
                                ************************************************************************
00000BFE 13FC000000FF INIT52    MOVE.B   #00,TREN               DISABLE TRANSMITTER AND RECEIVER
         FE60
00000C06 33FCE71600FF           MOVE     #$E716,PCSAR           DEFINE PROTOCOL
         FE48
00000C0E 13FC000000FF           MOVE.B   #$00,PCR               DEFINE CHARACTER LENGTH
         FE4A
00000C16 13FC001800FF           MOVE.B   #$18,PCR
         FE4A
00000C1E 33FC011600FF           MOVE.W   #$0116,TSR             TRANSMIT DUMMY SYNC
         FE42
                                ************************************************************************
                                *                                                                       *
                                *       TASK DISPATCHING LOOP                                           *
                                *                                                                       *
                                ************************************************************************
00000C26 4A380A00       TASKLP  TST.B    TX61SMPH               61 READY TO TRANSMIT ?
00000C2A 6610                   BNE.S    TX61LP
```

```
MOTOROLA M68000 ASM VERSION  1.20 SYS :   8.        .SOFTART .SA 01/06/82 10:59:47


 268   00000C2C 4A380A14              TST.B    TX52SMPH              52 READY TO TRANSMIT ?
 269   00000C30 67F4                  BEQ.S    TASKLP
 270   00000C32 6100FE52      TX52LP  BSR      INIT61RX
 271   00000C36 6100FE8C              BSR      INIT52TX
 272   00000C3A 6008                  BRA.S    LOOP
 273   00000C3C 6100FEC0      TX61LP  BSR      INIT52RX
 274   00000C40 6100FE04              BSR      INIT61TX
 275   00000C44 4A380A01      LOOP    TST.B    TCB61DRI              61 TRANSMITTER READY ?
 276   00000C48 6702                  BEQ.S    LOOP1
 277   00000C4A 612C                  BSR.S    DRV61
 278   00000C4C 4A380A16      LOOP1   TST.B    TCB52RVI              52 RECEIVER READY ?
 279   00000C50 6704                  BEQ.S    LOOP2
 280   00000C52 61000082              BSR      RCV52
 281   00000C56 4A380A15      LOOP2   TST.B    TCB52DRI              52 TRANSMITTER READY ?
 282   00000C5A 6704                  BEQ.S    LOOP3
 283   00000C5C 61000144              BSR      DRV52
 284   00000C60 4A380A02      LOOP3   TST.B    TCB61RVI              61 RECEIVER READY ?
 285   00000C64 6704                  BEQ.S    LOOP4
 286   00000C66 61000192              BSR      RCV61
 287   00000C6A 4A380A14      LOOP4   TST.B    TX52SMPH              TRANSMISSION COMPLETE ?
 288   00000C6E 66D4                  BNE      LOOP
 289   00000C70 4A380A00              TST.B    TX61SMPH
 290   00000C74 66CE                  BNE      LOOP
 291   00000C76 4E4A                  TRAP     #10
 292                        ********************************************************************************
 293                        *                                                                              *
 294                        *       MC68661/MC2661 TRANSMITTER DRIVER SUBROUTINE                           *
 295                        *                                                                              *
 296                        ********************************************************************************
 297   00000C78 42380A01      DRV61   CLR.B    TCB61DRI              CLEAR INTERRUPT FLAG FIRST
 298   00000C7C 4A380A05              TST.B    ETXF61
 299   00000C80 6752                  BEQ.S    DRV61RTN
 300   00000C82 4A380A29              TST.B    REP61SPH              IS THIS A REPLY ?
 301   00000C86 6624                  BNE.S    DRV612
 302   00000C88 30380A12              MOVE     D61CNT,D0             GET BYTE COUNT
 303   00000C8C 51C80034              DBRA     D0,DRV611
 304   00000C90 103C0062              MOVE.B   #CR61MSK,D0           DISABLE TX AFTER PAD CHAR
 305   00000C94 00000004              OR.B     #04,D0
 306   00000C98 13C000FFFE07          MOVE.B   D0,CR61               AND ENABLE RECEIVER
 307   00000C9E 21FC00000820          MOVE.L   #RXBUF61,TC61RXPT     SETUP FOR RESPONSE
               0A06
 308   00000CA6 50F80A2D              ST       ACK61SPH
 309   00000CAA 4E75                  RTS
 310   00000CAC 0C3800050A2B  DRV612  CMP.B    #5,TCB61SPH           ALL OUT ?
 311   00000CB2 6620                  BNE.S    DRV61RTN
 312   00000CB4 13FC006200FF          MOVE.B   #CR61MSK,CR61         DISABLE TX AFTER ACK/NAK
               FE07
 313   00000CBC 42380A29              CLR.B    REP61SPH
 314   00000CC0 4E75                  RTS
 315   00000CC2 31C00A12      DRV611  MOVE     D0,D61CNT             SAVE NEW BYTE COUNT
 316   00000CC6 4A00                  TST.B    D0
 317   00000CC8 670A                  BEQ.S    DRV61RTN              IGNORE PAD CHARACTER
 318   00000CCA 20780A0A              MOVEA.L  TC61TXPT,A0           GET BUFFER POINTER
 319   00000CCE 10B900FFFE27          MOVE.B   BCC531,(A0)           BUT MOVE BCC TO BUFFER
 320   00000CD4 4E75          DRV61RTN RTS
 321                        ********************************************************************************
 322                        *                                                                              *
 323                        *       MC68652/MC2652 RECEIVER DRIVER SUBROUTINE                              *
```

```
MOTOROLA M68000 ASM VERSION  1.20 SYS :   8.        .SOFTART .SA 01/06/82 10:59:47


 324                            *                                                              *
 325                            ********************************************************************************
 326   00000CD6 42380A16        RCV52    CLR.B    TCB52RVI
 327   00000CDA 4A380A2E                 TST.B    ACK52SPH            RECEIVING ACKNOWLEDGEMENT ?
 328   00000CDE 661A                     BNE.S    RCV521
 329   00000CE0 4A380A17                 TST.B    TCSR532F            ETX RECEIVED ?
 330                            *
 331                            *         ADDITIONAL CODE MAY BE INSERTED AS REQUIRED, TO CHECK THE
 332                            *         SECONDARY STATION ADDRESS, MESSAGE ACKNOWLEDGEMENT, OR BLOCK
 333                            *         SEQUENCE NUMBER INFORMATION.
 334                            *
 335   00000CE4 670000BA                 BEQ      RCV52RTN
 336   00000CE8 4AF80A28                 TAS      ETXF52              TEST FLAG
 337   00000CEC 6648                     BNE.S    RCV522              BCC CHAR. IF ALREADY SET
 338   00000CEE 10380A28                 MOVE.B   ETXF52,D0
 339   00000CF2 E208                     LSR.B    #1,D0               ETX SETS BIT 6 ONLY
 340   00000CF4 11C00A28                 MOVE.B   D0,ETXF52
 341   00000CF8 4E75                     RTS
 342   00000CFA 0C3800020A2C RCV521      CMP.B    #02,TCB52SPH        ALL OF ACKNOWLEDGEMENT RECEIVED ?
 343   00000D00 6600009E                 BNE      RCV52RTN
 344   00000D04 13FC000000FF             MOVE.B   #0,TREN             DISABLE RECEIVER
                FE60
 345   00000D0C 207C00000700             MOVEA.L  #RXBUF52,A0
 346   00000D12 0C180006                 CMP.B    #06,(A0)+           COMPARE AGAINST ACK
 347   00000D16 661C                     BNE.S    RCV523              RETRANSMIT IF NOT ALLRIGHT
 348   00000D18 10380986                 MOVE.B   TXTBUF52+6,D0       CHECK BLOCK SEQUENCE
 349   00000D1C 1210                     MOVE.B   (A0),D1             AGAINST ACK
 350   00000D1E 02000001                 AND.B    #01,D0
 351   00000D22 02010001                 AND.B    #01,D1
 352   00000D26 42380A14                 CLR.B    TX52SMPH            CLEAR SEMAPHORES WHEN DONE
 353   00000D2A 42380A2E                 CLR.B    ACK52SPH
 354   00000D2E B200                     CMP.B    D0,D1
 355   00000D30 6602                     BNE.S    RCV523              RETRANSMIT IF NOT CORRECT
 356   00000D32 4E47                     TRAP     #7
 357   00000D34 4E46            RCV523   TRAP     #6
 358   00000D36 6A68            RCV522   BPL.S    RCV52RTN            SET BIT 6 AND 7 ON 1ST BCC
 359   00000D38 13FC000000FF             MOVE.B   #0,TREN             DISABLE RECEIVER
                FE60
 360   00000D40 10380A18                 MOVE.B   TCBSR532,D0         AND
 361   00000D44 E258                     ROR      #1,D0               CHECK STATUS AFTER SECOND BCC
 362   00000D46 6524                     BCS.S    REPLY521
 363                            ********************************************************************************
 364                            *                                                                              *
 365                            *         MC68652/MC2652 ACKNOWLEDGEMENT SETUP                                 *
 366                            *                                                                              *
 367                            ********************************************************************************
 368   00000D48 207C00000980 REPLY52    MOVEA.L  #TXTBUF52,A0
 369   00000D4E 117C00060002             MOVE.B   #$06,2(A0)          FILL BUFFER WITH ACK
 370   00000D54 227C00000700             MOVEA.L  #RXBUF52,A1
 371   00000D5A 10290003                 MOVE.B   3(A1),D0            CHECK BLOCK # FOR ODD OR EVEN
 372   00000D5E 02000001                 AND.B    #01,D0
 373   00000D62 00000030                 OR.B     #$30,D0
 374   00000D66 11400003                 MOVE.B   D0,3(A0)            ASCII 0 OR 1
 375   00000D6A 6012                     BRA.S    REPLY522
 376   00000D6C 207C00000980 REPLY521    MOVEA.L  #TXTBUF52,A0
 377   00000D72 117C00150002             MOVE.B   #$15,2(A0)          OR NAK
 378   00000D78 117C00000003             MOVE.B   #0,3(A0)            AND NULL
 379   00000D7E 117C00160004 REPLY522    MOVE.B   #$16,4(A0)          ADD PAD CHARACTER
```

```
MOTOROLA M68000 ASM VERSION  1.20 SYS :   8.        .SOFTART .SA 01/06/82 10:59:47


380   00000D84 21C80A1E              MOVE.L   A0,TC52TXPT
381   00000D88 10FC0016              MOVE.B   #$16,(A0)+
382   00000D8C 10FC0016              MOVE.B   #$16,(A0)+
383   00000D90 42380A2C              CLR.B    TCB52SPH
384   00000D94 13FC000100FF          MOVE.B   #01,TREN              ENABLE 52 TRANSMITTER
               FE60
385   00000D9C 50F80A2A              ST       REP52SPH              SET REPLY SEMAPHORE
386   00000DA0 4E75          RCV52RTN RTS
387                                  ********************************************************************************
388                                  *                                                                              *
389                                  *       MC68652/MC2652 TRANSMITTER DRIVER SUBROUTINE                           *
390                                  *                                                                              *
391                                  ********************************************************************************
392   00000DA2 42380A15      DRV52     CLR.B    TCB52DRI
393   00000DA6 4A380A2A              TST.B    REP52SPH              IS THIS A REPLY ?
394   00000DAA 6624                  BNE.S    DRV522
395   00000DAC 4A380A28              TST.B    ETXF52                CHECK FOR ETX SENT
396   00000DB0 6746                  BEQ.S    DRV52RTN
397   00000DB2 30380A26              MOVE     D52CNT,D0             GET BYTE COUNT
398   00000DB6 51C8002E              DBRA     D0,DRV521
399   00000DBA 13FC000200FF          MOVE.B   #02,TREN              DISABLE TX AFTER PAD CHAR
               FE60
400   00000DC2 21FC00000700          MOVE.L   #RXBUF52,TC52RXPT     SETUP FOR RESPONSE
               0A1A
401   00000DCA 50F80A2E              ST       ACK52SPH
402   00000DCE 4E75                  RTS
403   00000DD0 0C3800050A2C  DRV522    CMP.B    #5,TCB52SPH
404   00000DD6 6620                  BNE.S    DRV52RTN
405   00000DD8 13FC000000FF          MOVE.B   #0,TREN               DISABLE TX AFTER ACK/NAK
               FE60
406   00000DE0 42380A2A              CLR.B    REP52SPH
407   00000DE4 4E75                  RTS
408   00000DE6 31C00A26      DRV521    MOVE     D0,D52CNT             SAVE NEW BYTE COUNT.
409   00000DEA 4A00                  TST.B    D0
410   00000DEC 670A                  BEQ.S    DRV52RTN              IGNORE PAD CHAR
411   00000DEE 20780A1E              MOVEA.L  TC52TXPT,A0           GET BUFFER POINTER
412   00000DF2 10B900FFFE6D          MOVE.B   BCC532,(A0)           GET BCC BYTE
413   00000DF8 4E75          DRV52RTN RTS
414                                  ********************************************************************************
415                                  *                                                                              *
416                                  *       MC68661/MC2661 RECEIVER DRIVER SUBROUTINE                              *
417                                  *                                                                              *
418                                  ********************************************************************************
419   00000DFA 42380A02      RCV61     CLR.B    TCB61RVI
420   00000DFE 4A380A2D              TST.B    ACK61SPH              RECEIVING ACKNOWLEDGEMENT ?
421   00000E02 661A                  BNE.S    RCV611
422   00000E04 4A380A03              TST.B    TCSR531F              ETX RECEIVED ?
423                                  *
424                                  *       ADDITIONAL CODE MAY BE INSERTED AS REQUIRED, TO CHECK THE
425                                  *       SECONDARY STATION ADDRESS, MESSAGE ACKNOWLEDGEMENT, OR BLOCK
426                                  *       SEQUENCE COUNT INFORMATION.
427                                  *
428   00000E08 670000C2              BEQ      RCV61RTN
429   00000E0C 4AF80A05              TAS      ETXF61                TEST FLAG
430   00000E10 664A                  BNE.S    RCV612                BCC CHAR IF ALREADY SET
431   00000E12 10380A05              MOVE.B   ETXF61,D0
432   00000E16 E208                  LSR.B    #1,D0                 ETX SETS BIT 6 ONLY
433   00000E18 11C00A05              MOVE.B   D0,ETXF61
```

```
MOTOROLA M68000 ASM VERSION  1.20 SYS :   8.        .SOFTART .SA 01/06/82 10:59:47


 434   00000E1C 4E75                    RTS
 435   00000E1E 0C3800020A2B RCV611     CMP.B    #02,TCB61SPH          ALL OF ACKNOWLEDGEMENT RECV'D ?
 436   00000E24 660000A6                BNE      RCV61RTN
 437   00000E28 13FC006200FF            MOVE.B   #CR61MSK,CR61         DISABLE RECEIVER
                FE07
 438   00000E30 207C00000820            MOVEA.L  #RXBUF61,A0
 439   00000E36 0C180006                CMP.B    #06,(A0)+             COMPARE AGAINST ACK
 440   00000E3A 661E                    BNE.S    RCV613                RETRANSMIT IF NOT OK
 441   00000E3C 10380946                MOVE.B   TXTBUF61+6,D0         CHECK BLOCK SEQUENCE
 442   00000E40 1210                    MOVE.B   (A0),D1               AGAINST ACK
 443   00000E42 02000001                AND.B    #01,D0
 444   00000E46 02010001                AND.B    #01,D1
 445   00000E4A 42380A00                CLR.B    TX61SMPH              CLEAR SEMAPHORES WHEN DONE
 446   00000E4E 42380A2D                CLR.B    ACK61SPH
 447   00000E52 6606                    BNE.S    RCV613                RETRANSMIT IF NOT CORRECT
 448   00000E54 42380A00                CLR.B    TX61SMPH              CLEAR SEMAPHORE WHEN DONE
 449   00000E58 4E48                    TRAP     #8
 450   00000E5A 4E49            RCV613  TRAP     #9
 451   00000E5C 6A6E            RCV612  BPL.S    RCV61RTN              SET BIT 6 AND 7 ON 1ST BCC
 452   00000E5E 13FC006200FF            MOVE.B   #CR61MSK,CR61         DISABLE '61 RECEIVER
                FE07
 453   00000E66 10380A04                MOVE.B   TCBSR531,D0
 454   00000E6A E258                    ROR      #1,D0                 CHECK STATUS AFTER 2ND BCC
 455   00000E6C 6524                    BCS.S    REPLY611
 456                          **********************************************************************
 457                          *                                                                    *
 458                          *        MC68661/MC2661 ACKNOWLEDGEMENT SETUP                        *
 459                          *                                                                    *
 460                          **********************************************************************
 461   00000E6E 207C00000940  REPLY61  MOVEA.L  #TXTBUF61,A0
 462   00000E74 117C00060002            MOVE.B   #$06,2(A0)            FILL BUFFER WITH ACK
 463   00000E7A 227C00000820            MOVEA.L  #RXBUF61,A1
 464   00000E80 10290003                MOVE.B   3(A1),D0
 465   00000E84 02000001                AND.B    #01,D0                IS BLOCK ODD OR EVEN ?
 466   00000E88 00000030                OR.B     #$30,D0
 467   00000E8C 11400003                MOVE.B   D0,3(A0)              ASCII 0 OR 1
 468   00000E90 6012                    BRA.S    REPLY612
 469   00000E92 207C00000940  REPLY611  MOVEA.L  #TXTBUF61,A0
 470   00000E98 117C00150002            MOVE.B   #$15,2(A0)            OR NAK
 471   00000E9E 117C00000003            MOVE.B   #0,3(A0)              AND NULL
 472   00000EA4 117C00160004  REPLY612  MOVE.B   #$16,4(A0)            ADD PAD TO END OF MESSAGE
 473   00000EAA 21C80A0A                MOVE.L   A0,TC61TXPT
 474   00000EAE 10FC0016                MOVE.B   #$16,(A0)+            INSERT INITIAL SYNCS
 475   00000EB2 10BC0016                MOVE.B   #$16,(A0)
 476   00000EB6 42380A2B                CLR.B    TCB61SPH
 477   00000EBA 103C0062                MOVE.B   #CR61MSK,D0
 478   00000EBE 00000001                OR.B     #01,D0
 479   00000EC2 13C000FFFE07            MOVE.B   D0,CR61               ENABLE TRANSMITTER
 480   00000EC8 50F80A29                ST       REP61SPH              SET REPLY SEMAPHORE
 481   00000ECC 4E75            RCV61RTN RTS
 482                          **********************************************************************
 483                          *                                                                    *
 484                          *        MC68652/MC2652 RECEIVER - INTERRUPT SERVICE ROUTINE         *
 485                          *                                                                    *
 486                          **********************************************************************
 487   00000ECE 48E78080        IS52RX  MOVEM.L  D0/A0,-(SP)           SAVE ALTERED REGISTERS
 488   00000ED2 20780A1A                MOVEA.L  TC52RXPT,A0           RESTORE DATA POINTER
 489   00000ED6 103900FFFE41            MOVE.B   RDR,D0                GET THE CHARACTER
```

```
MOTOROLA M68000 ASM VERSION  1.20 SYS :   8.        .SOFTART .SA 01/06/82 10:59:47


490   00000EDC 0C000016              CMP.B    #$16,D0
491   00000EE0 6722                  BEQ.S    IS52RTN              STRIP OUT SYN CHARACTERS
492   00000EE2 10C0                  MOVE.B   D0,(A0)+             SAVE THE CHARACTER
493   00000EE4 21C80A1A              MOVE.L   A0,TC52RXPT          AND INCREMENT THE POINTER
494   00000EE8 50F80A16              ST       TCB52RVI
495   00000EEC 4A380A2E              TST.B    ACK52SPH
496   00000EF0 6618                  BNE.S    RTN1
497   00000EF2 103900FFFE65          MOVE.B   SR532,D0             CHECK STATUS OF PGC
498   00000EF8 11C00A18              MOVE.B   D0,TCBSR532          SAVE THE STATUS
499   00000EFC E658                  ROR      #3,D0
500   00000EFE 6404                  BCC.S    IS52RTN
501   00000F00 50F80A17              ST       TCSR532F             SET INTERRUPT SERVICED FLAG
502   00000F04 4CDF0101      IS52RTN MOVEM.L  (SP)+,D0/A0          RESTORE REGISTERS
503   00000F08 4E73                  RTE
504   00000F0A 52380A2C      RTN1    ADDQ.B   #01,TCB52SPH
505   00000F0E 60F4                  BRA      IS52RTN
506                        ********************************************************************************
507                        *                                                                              *
508                        *       MC68661/MC2661 RECEIVER - INTERRUPT SERVICE ROUTINE                    *
509                        *                                                                              *
510                        ********************************************************************************
511   00000F10 48E78080      IS61RX  MOVEM.L  D0/A0,-(SP)          SAVE ALTERED REGISTERS
512   00000F14 20780A06              MOVEA.L  TC61RXPT,A0          RESTORE DATA POINTER
513   00000F18 103900FFFE01          MOVE.B   RHR61,D0             GRAB AND HOLD DATA
514   00000F1E 10C0                  MOVE.B   D0,(A0)+             MOVE DATA TO BUFFER
515   00000F20 21C80A06              MOVE.L   A0,TC61RXPT          AND SAVE NEW POINTER
516   00000F24 50F80A02              ST       TCB61RVI
517   00000F28 4A380A2D              TST.B    ACK61SPH             PART OF ACKNOWLEDGEMENT ?
518   00000F2C 6618                  BNE.S    RRTN1
519   00000F2E 103900FFFE23          MOVE.B   SR531,D0             CHECK PGC STATUS,
520   00000F34 11C00A04              MOVE.B   D0,TCBSR531          BUT SAVE IT FIRST
521   00000F38 E658                  ROR      #3,D0
522   00000F3A 6404                  BCC.S    IS61RRTN
523   00000F3C 50F80A03              ST       TCSR531F             ETX RECEIVED IF CARRY SET
524   00000F40 4CDF0101      IS61RRTN MOVEM.L  (SP)+,D0/A0          RESTORE REGISTERS
525   00000F44 4E73                  RTE
526   00000F46 52380A2B      RRTN1   ADDQ.B   #01,TCB61SPH
527   00000F4A 60F4                  BRA      IS61RRTN
528                        ********************************************************************************
529                        *                                                                              *
530                        *       MC68661/MC2661 TRANSMITTER - INTERRUPT SERVICE ROUTINE                 *
531                        *                                                                              *
532                        ********************************************************************************
533   00000F4C 48E78080      IS61TX  MOVEM.L  D0/A0,-(SP)          SAVE WORK REGISTERS
534   00000F50 20780A0A              MOVEA.L  TC61TXPT,A0          RESTORE DATA POINTER
535   00000F54 13D800FFFE01          MOVE.B   (A0)+,THR61          OUTPUT DATA TO TRANSMITTER
536   00000F5A 21C80A0A              MOVE.L   A0,TC61TXPT
537   00000F5E 4A380A29              TST.B    REP61SPH
538   00000F62 6630                  BNE.S    IS61TX2
539   00000F64 B1FC00000948          CMPA.L   #TXTBUF61+8,A0       HEADER OUT ?
540   00000F6A 6608                  BNE.S    IS61TX1
541   00000F6C 21F80A0E0A0A          MOVE.L   TC61DATP,TC61TXPT    POINT TO DATA IF IT IS
542   00000F72 6016                  BRA.S    IS61RTN
543   00000F74 103900FFFE23 IS61TX1  MOVE.B   SR531,D0             CHECK IF ETX WAS TRANSMITTED
544   00000F7A E658                  ROR      #3,D0
545   00000F7C 640C                  BCC.S    IS61RTN
546   00000F7E 50F80A05              ST       ETXF61
547   00000F82 21FC00000948          MOVE.L   #TXTBUF61+8,TC61TXPT  POINT TO TRAILER FIELD
```

```
MOTOROLA M68000 ASM VERSION  1.20 SYS :   8.        .SOFTART .SA 01/06/82 10:59:47


                0A0A
548   00000F8A 50F80A01      IS61RTN  ST        TCB61DRI
549   00000F8E 4CDF0101               MOVEM.L   (SP)+,D0/A0          RECOVER FROM INTERRUPT
550   00000F92 4E73                   RTE
551   00000F94 52380A2B      IS61TX2  ADDQ.B    #01,TCB61SPH
552   00000F98 60F0                   BRA       IS61RTN
553                          ********************************************************************************
554                          *                                                                              *
555                          *         MC68652/MC2652 TRANSMITTER - INTERRUPT SERVICE ROUTINE               *
556                          *                                                                              *
557                          ********************************************************************************
558   00000F9A 48E78080      IS52TX   MOVEM.L   D0/A0,-(SP)           SAVE ALTERED REGISTERS
559   00000F9E 20780A1E               MOVEA.L   TC52TXPT,A0          RESTORE DATA POINTER
560   00000FA2 4240                   CLR.W     D0
561   00000FA4 1018                   MOVE.B    (A0)+,D0
562   00000FA6 33C000FFFE42           MOVE.W    D0,TSR               CLEAR TSOM WHILE WRITING DATA
563   00000FAC 21C80A1E               MOVE.L    A0,TC52TXPT
564   00000FB0 4A380A2A               TST.B     REP52SPH
565   00000FB4 6630                   BNE.S     IS52TX2
566   00000FB6 B1FC00000988           CMPA.L    #TXTBUF52+8,A0       HEADER OUT ?
567   00000FBC 6608                   BNE.S     IS52TX1
568   00000FBE 21F80A220A1E           MOVE.L    TC52DATP,TC52TXPT    POINT TO DATA IF IT IS
569   00000FC4 6016                   BRA.S     IS52TRTN
570   00000FC6 103900FFFE65  IS52TX1  MOVE.B    SR532,D0             CHECK IF ETX WAS TRANSMITTED
571   00000FCC E658                   ROR       #3,D0
572   00000FCE 640C                   BCC.S     IS52TRTN
573   00000FD0 50F80A28               ST        ETXF52               FLAG IT
574   00000FD4 21FC00000988           MOVE.L    #TXTBUF52+8,TC52TXPT AND POINT TO THE TRAILER
                0A1E
575   00000FDC 50F80A15      IS52TRTN ST        TCB52DRI             SHOW INTERRUPT WAS SERVICED
576   00000FE0 4CDF0101               MOVEM.L   (SP)+,D0/A0          BEFORE RETURNING
577   00000FE4 4E73                   RTE
578   00000FE6 52380A2C      IS52TX2  ADDQ.B    #01,TCB52SPH
579   00000FEA 60F0                   BRA       IS52TRTN
580                                   END
```

```
MOTOROLA M68000 ASM VERSION  1.20 SYS :   8.        .SOFTART .SA 01/06/82 10:59:47


SYMBOL TABLE LISTING

SYMBOL NAME      SECT   VALUE       SYMBOL NAME      SECT   VALUE

ACK52SPH               00000A2E    RCV52                  00000CD6
ACK61SPH               00000A2D    RCV521                 00000CFA
ASR                    00FFFE60    RCV522                 00000D36
BCC531                 00FFFE27    RCV523                 00000D34
BCC532                 00FFFE6D    RCV52RTN               00000DA0
BLKSEQ                 00000A40    RCV61                  00000DFA
CCA531                 00FFFE21    RCV611                 00000E1E
CCA532                 00FFFE61    RCV612                 00000E5C
CR531                  00FFFE23    RCV613                 00000E5A
CR532                  00FFFE65    RCV61RTN               00000ECC
CR61                   00FFFE07    RDR                    00FFFE41
CR61MSK                00000062    REP52SPH               00000A2A
D52CNT                 00000A26    REP61SPH               00000A29
D61CNT                 00000A12    REPLY52                00000D48
DBINIT                 00000B5A    REPLY521               00000D6C
DRV52                  00000DA2    REPLY522               00000D7E
DRV521                 00000DE6    REPLY61                00000E6E
DRV522                 00000DD0    REPLY611               00000E92
DRV52RTN               00000DF8    REPLY612               00000EA4
DRV61                  00000C78    RHR61                  00FFFE01
DRV611                 00000CC2    RRTN1                  00000F46
DRV612                 00000CAC    RSR                    00FFFE40
DRV61RTN               00000CD4    RTN1                   00000F0A
DUMDAT                 00000A3A    RXBUF52                00000700
ETXF52                 00000A28    RXBUF61                00000820
ETXF61                 00000A05    SR531                  00FFFE23
INIT52                 00000BFE    SR532                  00FFFE65
INIT52RX               00000AFE    SR61                   00FFFE03
INIT52TX               00000AC4    START                  00000B36
INIT531                00000BAE    START1                 00000B40
INIT532                00000BD6    STRTADR                00000A3E
INIT61                 00000B7E    TASKLP                 00000C26
INIT61RX               00000A86    TC52DATP               00000A22
INIT61TX               00000A46    TC52RXPT               00000A1A
INITBUF                00000A2F    TC52TXPT               00000A1E
IS52RTN                00000F04    TC61DATP               00000A0E
IS52RX                 00000ECE    TC61RXPT               00000A06
IS52TRTN               00000FDC    TC61TXPT               00000A0A
IS52TX                 00000F9A    TCB52DRI               00000A15
IS52TX1                00000FC6    TCB52RVI               00000A16
IS52TX2                00000FE6    TCR52SPH               00000A2C
IS61RRTN               00000F40    TCB61DRI               00000A01
IS61RTN                00000F8A    TCB61RVI               00000A02
IS61RX                 00000F10    TCB61SPH               00000A2B
IS61TX                 00000F4C    TCBSR531               00000A04
IS61TX1                00000F74    TCBSR532               00000A18
IS61TX2                00000F94    TCSR531F               00000A03
LOOP                   00000C44    TCSR532F               00000A17
LOOP1                  00000C4C    TDR                    00FFFE43
LOOP2                  00000C56    THR61                  00FFFE01
LOOP3                  00000C60    TREN                   00FFFE60
LOOP4                  00000C6A    TSR                    00FFFE42
```

```
MOTOROLA M68000 ASM VERSION  1.20 SYS :   8.        .SOFTART .SA 01/06/82 10:59:47


MR531                00FFFE25    TX52LP               00000C32
MR532                00FFFE69    TX52SMPH             00000A14
MR61                 00FFFE05    TX61LP               00000C3C
PCR                  00FFFE4A    TX61SMPH             00000A00
PCSAR                00FFFE48    TXTBUF52             00000980
PCSARH               00FFFE48    TXTBUF61             00000940
PCSARL               00FFFE49
```

MOTOROLA ***Semiconductor Products Inc.***

3501 ED BLUESTEIN BLVD., AUSTIN, TEXAS 78721 • A SUBSIDIARY OF MOTOROLA INC.

A16016-1 PRINTED IN USA 8-83 IMPERIAL LITHO C15332 10,000