March 1993

M68HC705X32 PROGRAMMER BOARD

(REVision A PWBs only)

APPLICATION NOTE

1. INTRODUCTION

This application note describes the technique used to program and verify a X32 or B series microcontroller (MCU), and how to construct the programmer board (PGMR) used in conjunction with this application note. All that is required to program an MCU is the PGMR and a +5 volt and a V PP dc power supply.

The X32 MCU is available in a number of packages. To accommodate these different packages on the PGMR board a package adapter board approach has been adopted. For each different package type there exists a package adapter board. The PGMR is normally supplied with the 64pin QFP adapter board.

This board allows for either parallel or serial bootstrap operation. In the parallel mode the data required to be written to the internal PROM (OTPROM/EPROM) is held in an external EPROM (SKT3), while in the serial mode the data to be written is supplied from a remote computer via the serial connector (P2).

2. PARALLEL MODE

This PGMR programming mode allows the user program, contained in an external EPROM, to be copied into the internal PROM (OTPROM/EPROM) of the MC68HC705B, MC68HC705X16 and MC68HC705X32 MCU device (or EEPROM in the case of the 805B6).

2.1 68HC805B6 PARALLEL PROGRAMMING MODE

Jumper settings required:

JP1	805B6	
JP2	805B6	
JP3	MODE B	
JP4	EPROM	
JP5	64/128	**Use only a 27C64 chip in SKT3**
JP6	64	
JP7	NOT X32	
JP8	NORMAL	
JP9	PARALLEL	

Apply power to the PGMR board. Insert the B series MCU device into the PGMR. Apply power to the MCU via switch S1. The MCU is taken out of reset and placed in the run mode via switch S2, and MCU control is transferred to the bootstrap ROM. The selected programming routine is then executed.

The EEPROM6 will first be erased then checked for complete erasure. If a non $FF byte is detected, the RED LED D3 will be turned on, and erase will be performed a second time, and so on until total erasure. Before proceeding to erase the EEPROM1 the RED LED D3 is turned off, then erasure of EEPROM1 takes place in a similar fashion to the EEPROM6, thus clearing the security bit if needed.

At that point, both EEPROMs being completely erased, the programming operation follows. Both EEPROMs will be loaded in turn, in increasing address order from the external EPROM. Segments containing no EEPROM will be skipped by the loader.

A ‘fast’ alogorithm is used that skips over $FF bytes, thus speeding up the loading process for short programs. During programming, the GREEN LED D2 will flash at about 3Hz (much faster if DATA = $FF).

Upon completion of the programming operation, the EEPROM’s content will be checked against the external EPROM. If the device verifies correctly, the GREEN LED D2 will stay ON, else if an error is detected the RED LED D3 will turn ON (GREEN LED off).

2.2 68HC705B5 PARALLEL PROGRAMMING MODE

Jumper settings required:

JP1	705B5/X16/X32	
JP2	705B5/X16/X32	
JP3	MODE A	
JP4	EPROM	
JP5	64/128	**Use only a 27C64 chip in SKT3**
JP6	64	
JP7	NOT X32	
JP8	NORMAL	
JP9	PARALLEL	

Apply power to the PGMR board. Insert the B series MCU device into the PGMR. Apply power to the MCU via switch S1. The MCU is taken out of reset and placed in the run mode via switch S2, and MCU control is transferred to the bootstrap ROM. The selected programming routine is then executed.

During programming, the GREEN LED D2 will flash at about 3 Hz.

Upon completion of the programming operation, the internal EPROM content will be checked against the external EPROM. If the device verifies correctly, the GREEN LED D2 will stay ON, else if an error is detected the RED LED D3 will turn ON (GREEN LED off).

2.3 68HC705X16 PARALLEL PROGRAMMING MODE

Jumper settings required:

JP1	705B5/X16/X32	
JP2	705B5/X16/X32	
JP3	MODE A	
JP4	EPROM	
JP5	64/128	**Use only a 27C128 chip in SKT3**
JP6	128/256	
JP7	NOT X32	

JP8	NORMAL
JP9	PARALLEL

Apply power to the PGMR board. Insert the 705X16 MCU device into the PGMR. Apply power to the MCU via switch S1. The MCU is taken out of reset and placed in the run mode via switch S2, and MCU control is transferred to the bootstrap ROM. The selected programming routine is then executed.

The EPROM will first be verified as being blank. At the first byte not erased to $00, the RED LED D3 will turned ON and the routine will stay in a loop. When the whole EPROM content is checked as erased, the GREEN LED D2 will be turned ON.

The contents of EEPROM1 will then be erased. If a non $FF byte is detected, the RED LED will be turned on, and erase will be performed a second time, and so on until total erasure.

After both the EPROM and the EEPROM1 are completely erased and the security bit is cleared. The programming operation can be performed.

All EPROM and EEPROM1 locations will be loaded, in increasing address order. Segments containing no EPROM or EEPROM will be skipped by the loader.

During programming, the GREEN LED D3 will flash at about 3 Hz.

Upon completion of the program operation, the EPROM and EPROM1 content will be checked against the external EPROM. If the device verifies correctly, the GREEN LED D2 will stay ON, else if an error is detected the RED LED D3 will turn ON (GREEN LED off).

2.4 68HC705X32 PARALLEL PROGRAMMING MODE

Jumper settings required:

JP1	705B5/X16/X32	
JP2	705B5/X16/X32	
JP3	MODE A	
JP4	EPROM	
JP5	256	**Use only a 27C256 chip in SKT3**
JP6	128/256	

JP7	X32
JP8	NORMAL
JP9	PARALLEL

Apply power to the PGMR board. Insert the 705X32 MCU device into the PGMR. Apply power to the MCU via switch S1. The MCU is taken out of reset and placed in the run mode via switch S2, and MCU control is transferred to the bootstrap ROM. The selected programming routine is then executed.

The EPROM will first be verified as being blank. At the first byte not erased to $00, the RED LED D3 will turned ON and the routine will stay in a loop. When the whole EPROM content is checked as erased, the GREEN LED D2 will be turned ON.

The contents of EEPROM1 will then be erased. If a non $FF byte is detected, the RED LED will be turned on, and erase will be performed a second time, and so on until total erasure.

After both the EPROM and the EEPROM1 are completely erased and the security bit is cleared. The programming operation can be performed.

All EPROM and EEPROM1 locations will be loaded, in increasing address order. Segments containing no EPROM or EEPROM will be skipped by the loader.

During programming, the GREEN LED D3 will flash at about 3 Hz.

Upon completion of the program operation, the EPROM and EPROM1 content will be checked against the external EPROM. If the device verifies correctly, the GREEN LED D2 will stay ON, else if an error is detected the RED LED D3 will turn ON (GREEN LED off).

2.5 MCU BLANK CHECK (705B5/705X16 ONLY)

To check that the EPROM in a device is erased follow the steps outlined below:

2.4.1 Set the power OFF/ON switch (S1) to the POWER OFF position and the reset switch (S2) to the RESET IN position.

2.4.1 Set up the jumpers as shown , with the MCU device in the programming socket and no EPROM fitted to the SKT3.

JP1	705B5/705B16
JP2	705B5/705B16
JP3	MODEB
JP4	EPROM
JP5	27C64/27C128
JP6	27C64 (for B5) or 27C128 (for X16)
JP7	PARALLEL

2.4.2 Apply power by moving the power OFF/ON switch (S1) to the POWER ON position, then remove the reset by moving the reset switch (S2) to the RESET OUT position.

3. If the device is in the erased state the green LED will light, if the part is non blank the red LED will light.

4. Return the reset switch(S2) to the RESET IN position. Then remove the power by setting the power OFF/ON switch(S1) to the OFF position.

3. SERIAL MODE

3.1 805B6 SERIAL PROGRAMMING MODE

Jumper settings required:

JP1	805B6
JP2	805B6
JP3	MODE A
JP4	EPROM
JP5	64/128
JP6	64
JP7	NOT X32
JP8	NORMAL
JP9	SERIAL

Apply power to the PGMR board. Insert the B series MCU device into the PGMR. Apply power to the MCU via switch S1. The MCU is taken out of reset and placed in the run mode via switch S2, and MCU control is transferred to the serial bootstrap.

The EEPROM6 will first be erased. Then, it is checked for complete erasure; if a non $FF byte is detected, the RED LED D3 will be turned on, and erase will be performed a second time, and so on until total erasure. Before proceeding to erase the EEPROM1, the RED LED D3 is turned off, then erasure of EEPROM1 takes place in a similar fashion, thus clearing the security bit if needed.

At that point, both EEPROMs being completely erased, control is given to the serial routine. The serial routine communicates through the SCI to an external host, typically a PC, by means of an RS232 link at 9600 baud, 8 bit, no parity, full duplex.

Attention: DATA format is not ASCII, but 8-bit binary, so a complementary program must be run by the host to supply the required format. Such a program is available for the IBM PC from MOTOROLA. The format accepted by the serial loader is as follows:

<address high> <address low> <data>

These three bytes must be sent by the host for each address to be programmed.

The protocol is as follows:

3.1.1 The MC68HC805B6 sends the last byte programmed to the host as a prompt (the first
time,this data is undermined), this allows verification by the host of proper programming.

3.1.2 In response to this prompt, the host sends the 3 bytes as outlined above.

3.1.3 If the data if different from $FF, it is programmed to the address provided, be it PORT, RAM ($50-$7C used by the loader - not available) or EEPROM.

3.1.4 Loop to 1.

If the data sent is $FF, programming does not take place (EEPROMs only), and the content of the accessed location is actually returned as a prompt.

See 805B specification for further information.

3.2 705B5 SERIAL PROGRAMMING MODE

Jumper settings required:

JP1	705B5/X16/X32
JP2	705B5/X16/X32
JP3	MODE A
JP4	RAM
JP5	64/128
JP6	64
JP7	NOT X32
JP8	NORMAL

JP9 SERIAL

Apply power to the PGMR board. Insert the B series MCU device into the PGMR. Apply power to the MCU via switch S1. The MCU is taken out of reset and placed in the run mode via switch S2, and MCU control is transferred to the serial bootstrap.

The serial bootstrap routine communicates through the SCI to an external host, typically a PC, by means of an RS232 link at 9600 baud, 8 bit, no parity, full duplex.

Attention: DATA format is not ASCII, but 8-bit binary, so a complementary program must be run by the host to supply the required format. Such a program is available for the IBM PC from MOTOROLA. The format accepted by the serial loader is as follows: <address high> <address low> <data0> <data1> <data2> <data3>

These six bytes must be sent by the host for each group of four address to be programmed.

The protocol is as follows:

3.2.1 The MC68HC705B5 sends the last two bytes programmed to the host as a prompt, this allows for verification by the host of proper programming. NOTE: After reset the MC68HC705B5 serial bootstrap routine will first echo two blocks of 4 bytes at $00, as no data is programmed yet.

3.2.2 In response to the first byte prompt, the host sends the first address byte.

3.3.3 After receiving the first address byte, the MC68HC705B5 sends the next byte programmed.

3.3.4 The exchange of data continues until the MC68HC705B5 has sent the four data bytes, and the host has sent 2 address bytes and 4 data bytes.

3.3.5 If the data is different from $00, it is programmed, at the address provided, while the next address and bytes are received and the previous data is echoed.

3.3.6 Loop to 1.

If the data sent in is $00, no programming in the EPROM takes place, and the content of the accessed location is returned as a prompt. The entire EPROM memory can be read in this fashion. The RED LED D3 will be on is the data read is not $00.

During programming, the GREEN LED D2 will flash.

3.3 705X16 SERIAL PROGRAMMING MODE

Jumper settings required:

JP1	705B5/X16/X32
JP2	705B5/X16/X32
JP3	MODE A
JP4	RAM
JP5	64/128
JP6	128
JP7	NOT X32
JP8	NORMAL
JP9	SERIAL

Apply power to the PGMR board. Insert the X series MCU device into the PGMR. Apply power to the MCU via switch S1. The MCU is taken out of reset and placed in the run mode via switch S2, and MCU control is transferred to the bootstrap ROM. The selected programming routine is then executed.

The serial bootstrap routine communicates through the SCI to an external host, typically a PC, by means of an RS232 link at 9600 baud, 8 bit, no parity, full duplex.

Attention: DATA format is not ASCII, but 8-bit binary, so a complementary program must be run by the host to supply the required format. Such a

program is available for the IBM PC from MOTOROLA. The format accepted by the serial loader is as follows: <address high> <address low> <data0> <data1> <data2> <data3>

These six bytes must be sent by the host for each group of four address to be programmed.

For programming information on the X16, please refer to the X16 reference document.

NOTE: The PC program differs from the B5 program in that it must program both the EPROM and EEPROM locations of the X16.

3.4 705X32 SERIAL PROGRAMMING MODE

Jumper settings required:

JP1	705B5/X16/X32
JP2	705B5/X16/X32
JP3	MODE A
JP4	RAM
JP5	64/128
JP6	128
JP7	X32
JP8	NORMAL
JP9	SERIAL

Apply power to the PGMR board. Insert the X series MCU device into the PGMR. Apply power to the MCU via switch S1. The MCU is taken out of reset and placed in the run mode via switch S2, and MCU control is transferred to the bootstrap ROM. The selected programming routine is then executed.

The serial bootstrap routine communicates through the SCI to an external host, typically a PC, by means of an RS232 link at 9600 baud, 8 bit, no parity, full duplex.

Attention: DATA format is not ASCII, but 8-bit binary, so a complementary

program must be run by the host to supply the required format. Such a program is available for the IBM PC from MOTOROLA. The format accepted by the serial loader is as follows: <address high> <address low> <data0> <data1> <data2> <data3>

These six bytes must be sent by the host for each group of four address to be programmed.

For programming information on the X32, please refer to the X32 reference document.

NOTE: The PC program differs from the B5 program in that it must program both the EPROM and EEPROM locations of the X32.

4. PROGRAMMING MODULE PREPARATION

The PGMR must be prepared/configured prior to any operations. Board preparation consists of the external power source (+5V and V_{PP}), EPROM installation (if parallel mode), QFP PGMR configuration, or PLCC PGMR configuration.

4.1 External Power Source

Power connector P1 is used to connect an external power supply to the PGMR. A +5 Vdc @ 100 mA power source is connected to connector P1 pins labeled +5V and GND. The programming voltage power source is connected to pins labeled V_{PP} and GND. Refer to the specific device data sheet for programming voltage (V_{PP}) specifications.

NOTE

The programming voltage (V_{PP}) must be measured at JP2 (centre pin) during the programming cycle (GREEN LED illuminated).

4.2 EPROM Installation

The basic EPROM device used on the PGMR (at location SKT3) is a EPROM, 28-pin device. This EPROM device contains the user code to be programmed into the MCU device.

4.3 64 pin QFP PGMR Configuration

For the 64 pin quad flat pack (QFP) device programming the 64 pin QFP package adapter board must be connected to the PGMR. Made sure that the package adapter reference mark is in the same corner as that on the PGMR board.

5. PROGRAMMING OPERATION

To program a X32/B series MCU, perform the following steps:

CHECK JUMPERS ARE CORRECTLY SET

5.1 Place switch S1 to POWER-OFF (right) position.

5.2 Place switch S2 to RESET-IN (left) position.

5.3 Apply power to programming board.

5.4 Insert MCU and EPROM.

5.5 Place switch S1 to POWER-ON (left) position.

5.6 Place switch S2 to RESET-OUT (right) position.
GREEN LED flashes signifying that programming is taking place.

5.7 Place switch S2 to RESET-IN (left) position.

5.8 Remove power (via S1).

5.9 Remove MCU.

5.9 Repeat from 5.4 if further MCU's require to be programmed.

APPENDIX A

A.1 Information on 705B16/705X16 parts

Due to firmware errors in the initial silicon mask set C97K , these parts when programmed in parallel mode fail to verify or to program page 0 ROM. To get around these problems the EPB16 PC program is enclosed. This program can be used to verify and/or program these parts. Mask set D28J has corrected these errors and eliminates the need for the EPB16 program.

Note: The EPB16 programs only allows for the programming and verification of 705B16/705X16 parts - it cannot be used on 705B5 or 805B6 parts.

A.2 SERIAL VERIFICATION/PROGRAMMING

A.2.1 Software Requirements

EPB16.EXE Revision C.0 This version is enclosed with the programmer board.

A2.2 Hardware Requirements

The hardware requirements to run the EPB16 program are as follows:

1 x IBM® PC Compatible Personal Computer
1 x Serial Port (must be communication port 1 - COM1)
1 x Connecting Cable
1 x M68HC705X16PGMR programming board

A2.3 Running the program

To use the EPB16 program follow the following instructions.

A.2.3.1 Setup board for serial operation, as per section 3.

A.2.3.2 Connect the programming board to the PC using the connecting cable.

A.2.3.3 Install the appropriate 705X16 part into the programming board.

A.2.3.4 Apply power to the programming board and set the power off/on switch (S1) to the ON position.

A.2.3.5 Move the reset switch to the RESET OUT position.

A.2.3.6 On the PC execute the EPB16 program as follows:

EPB16 [xxx] [file] <return>

where "-xxx" is an optional loop count multiplier to be specified on fast PC's in the case of serial errors and "file" specifies the name of an S-record formatted file to be loaded.

Notes:

a. Specify -2 for the loop count multiplier on an PC386 at 25MHz when running EPB16 as detailed above.

b. The S-record file must contain data only at EPROM or EEPROM locations. Downloading data into registers or into RAM will disturb the correct serial operation. (The S-record file must be in Motorola EXORCISOR format no. 82).

A.2.3.7 If a file name was not specified when invoking EPB16 program, the "F" command must be executed. The program prompts for the name of a EXORciser formatted file.

Note: The S-record file must only contain data only at EPROM or EEPROM locations.
Downloading data into registers or into RAM will disturb the correct serial operation.

A.2.3.8 Enter "D" to download the file data to the MCU. As the data is downloaded the EPROM and/or EEPROM of the MCU is programmed.

Note:
a. The download process is automatically followed by a verification process which compares the bytes sent to the MCU with those echoed by the MCU. Differences, if any, are reported in the following format:

Addr Sent Recv

where Addr is the location which was sent the value Sent, and Recv is the value which was echoed. The three values are in hexadecimal.

b. The "D" command is only feasible after an S-record file has been successfully loaded with the "F" command (or as a program argument).

A.2.3.9 Enter "V" to verify the MCU.

Note:
The "V" command is only feasible after a file has been downloaded into the MCU.

A.2.3.10 The "S" command is used to show the MCU memory contents.

A.2.3.11 Set the reset switch to RESET-IN. Then remove the power using the power
off switch.

A.2.3.12 Quit EPB16 using either the "X" or "Q" command.

APPENDIX B

B.1 serial programming cable
The following pages give details of a serial programming cable . If a nine way connector is used the signals must be mapped correctly.

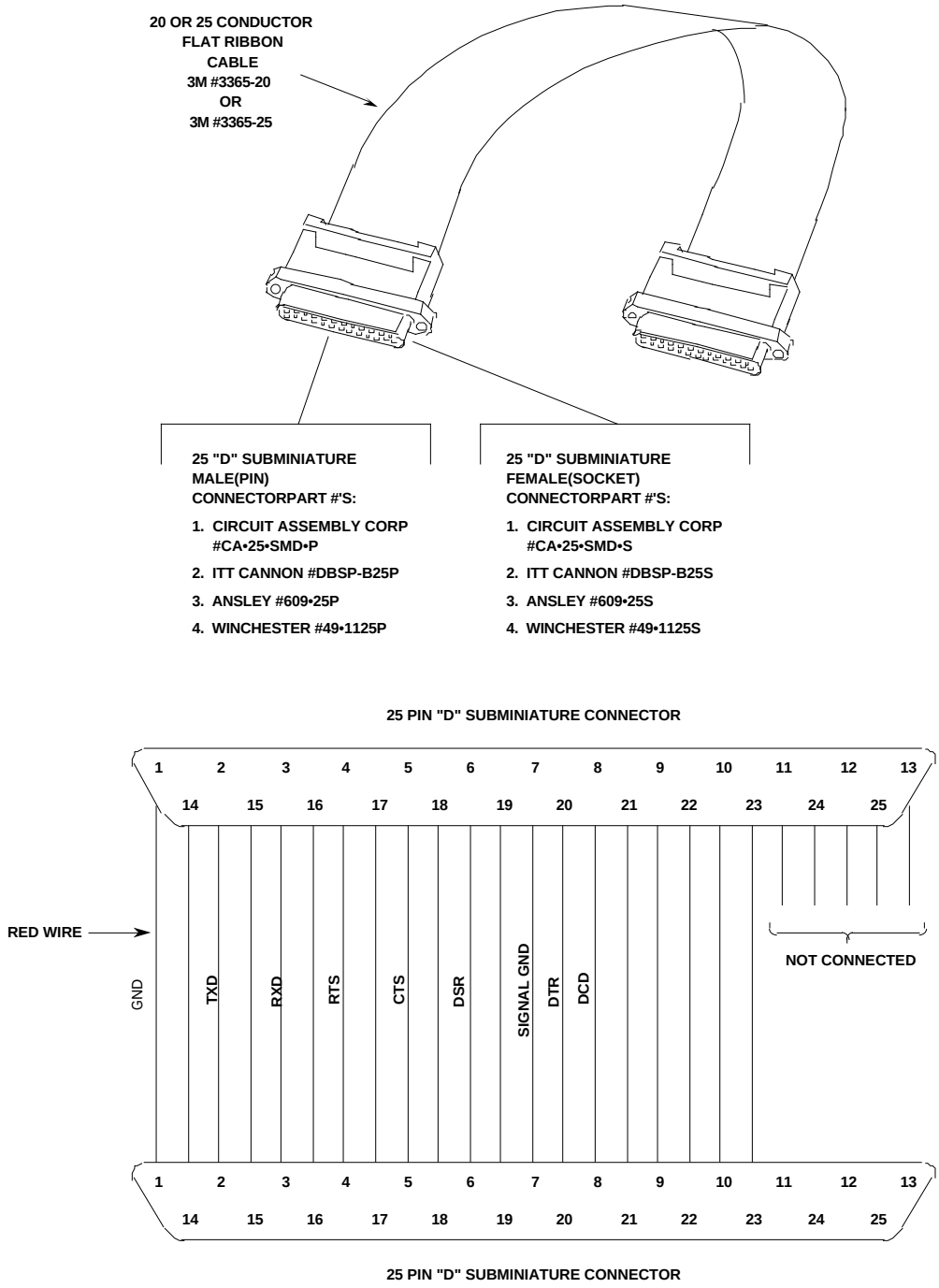

FIGURE 1-1. PGMR/Host Computer Cable Assembly Diagram

A Hayes compatible modem cable, purchased from a local computer store, can be used to connect the PGMR to the host computer.

The PGMR is wired as data communication equipment (DCE) whereas a terminal and most serial modem ports on host computers are wired as data

terminal equipment (DTE). This lets a straight-through cable be used for most setups.

If a different type of cable is used to connect the PGMR to the host computer, a null modem adapter (shown below) may be required to match the cable to the EVS terminal port connector.

A null modem adapter reverses the roles of various data and control signals to make a DTE device appear as a DCE device, or vice versa.

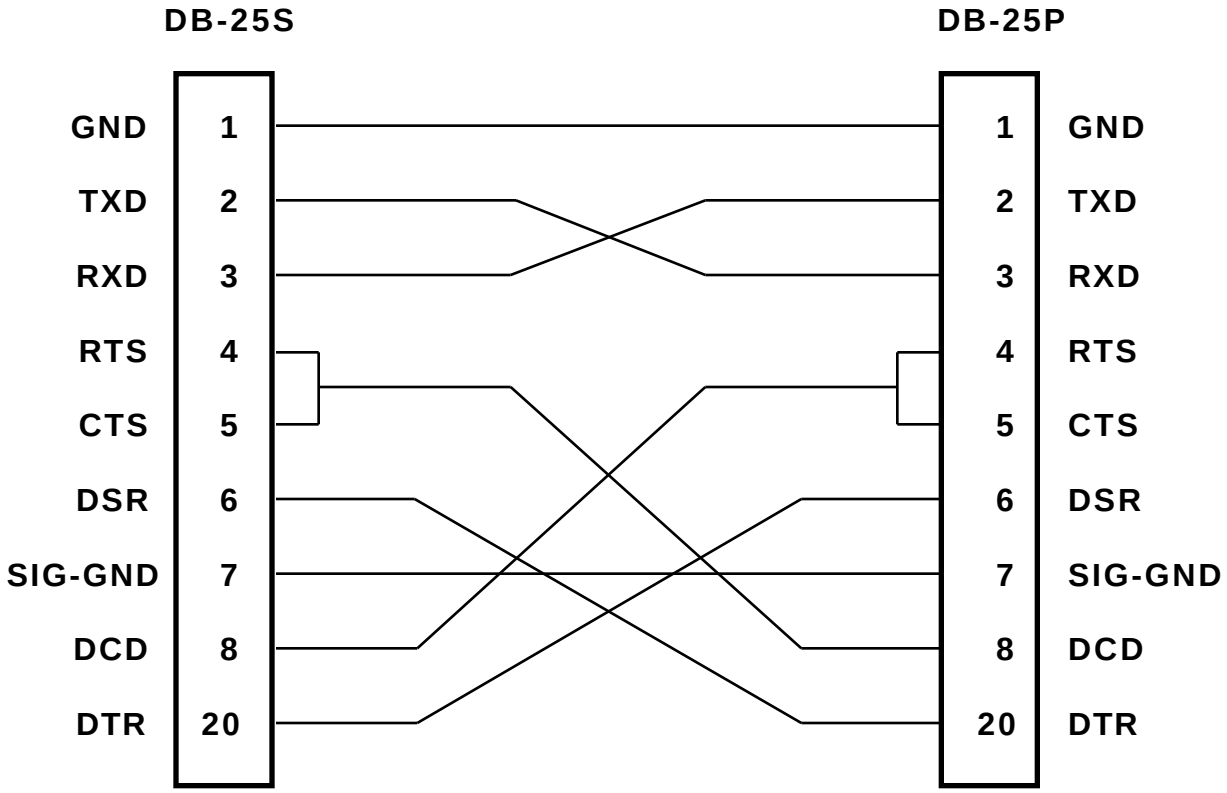

FIGURE 1-2 Null Modem Adapter

Motorola reserves the right to make changes without further notice to any products herein to improve reliability, function or design. Motorola does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death

may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part.