April 4, 2002

How Nohau supports the Philips 8051MX Microcontroller

There are two options for the emulation of the 8xC51MB2 and 8xC51MC2 micros. One is the emulator with 768k of emulation memory that has been configured to support up to 512k of code and 256k of external data memory. The other is the 256k emulator which is configured for 128k for code memory and 128k for external data memory.

In most cases customers will typically map all of the external data to the target and leave the code memory mapped to emulation RAM to emulate their on chip / off-chip ROM memory. The current software is **preliminary** for supporting the 51MX family we have changed the memory mapping screen for the MX family to have the following options. This will be the normal screen displayed, if you use the advanced setup and then re-enter the mapping configuration then the advanced setup screen from Diagram 2 will appear.

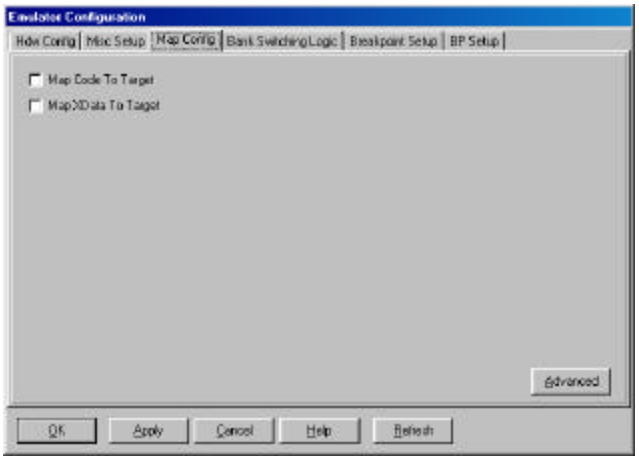


Diagram 1

With the XDTATA all mapped to target the on-chip XDATA memory will be visible to both the user and the emulators trace memory. The trace will read the correct data regardless of the mapping of the XDATA for the on-chip XDATA memory.

In the advanced setup the “advanced” memory mapping screen will appear (diagram 2), this is also our current memory mapping screen for the EMUL51-PC emulator family, and we will be documenting how the operation of this works in both the manuals and the on-line help. Refer to diagram 3 for information on memory mapping with the advanced setup option and how it reflects on the available memory.

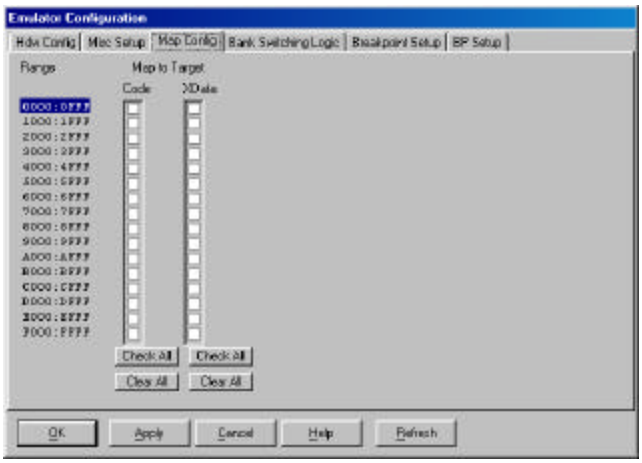


Diagram 2

Code Memory block mapping layout			256k emulator board boundary ends here				768K emulator board boundary ends here
80:0000 - 80:0FFF	81:0000 - 81:0FFF	82:0000 - 82:0FFF	83:0000 - 83:0FFF	84:0000 - 84:0FFF	85:0000 - 85:0FFF	86:0000 - 86:0FFF	87:0000 - 87:0FFF
80:1000 - 80:1FFF	81:1000 - 81:1FFF	82:1000 - 82:1FFF	83:1000 - 83:1FFF	84:1000 - 84:1FFF	85:1000 - 85:1FFF	86:1000 - 86:1FFF	87:1000 - 87:1FFF
80:2000 - 80:2FFF	81:2000 - 81:2FFF	82:2000 - 82:2FFF	83:2000 - 83:2FFF	84:2000 - 84:2FFF	85:2000 - 85:2FFF	86:2000 - 86:2FFF	87:2000 - 87:2FFF
80:3000 - 80:3FFF	81:3000 - 81:3FFF	82:3000 - 82:3FFF	83:3000 - 83:3FFF	84:3000 - 84:3FFF	85:3000 - 85:3FFF	86:3000 - 86:3FFF	87:3000 - 87:3FFF
80:4000 - 80:4FFF	81:4000 - 81:4FFF	82:4000 - 82:4FFF	83:4000 - 83:4FFF	84:4000 - 84:4FFF	85:4000 - 85:4FFF	86:4000 - 86:4FFF	87:4000 - 87:4FFF
80:5000 - 80:5FFF	81:5000 - 81:5FFF	82:5000 - 82:5FFF	83:5000 - 83:5FFF	84:5000 - 84:5FFF	85:5000 - 85:5FFF	86:5000 - 86:5FFF	87:5000 - 87:5FFF
80:6000 - 80:6FFF	81:6000 - 81:6FFF	82:6000 - 82:6FFF	83:6000 - 83:6FFF	84:6000 - 84:6FFF	85:6000 - 85:6FFF	86:6000 - 86:6FFF	87:6000 - 87:6FFF
80:7000 - 80:7FFF	81:7000 - 81:7FFF	82:7000 - 82:7FFF	83:7000 - 83:7FFF	84:7000 - 84:7FFF	85:7000 - 85:7FFF	86:7000 - 86:7FFF	87:7000 - 87:7FFF
80:8000 - 80:8FFF	81:8000 - 81:8FFF	82:8000 - 82:8FFF	83:8000 - 83:8FFF	84:8000 - 84:8FFF	85:8000 - 85:8FFF	86:8000 - 86:8FFF	87:8000 - 87:8FFF
80:9000 - 80:9FFF	81:9000 - 81:9FFF	82:9000 - 82:9FFF	83:9000 - 83:9FFF	84:9000 - 84:9FFF	85:9000 - 85:9FFF	86:9000 - 86:9FFF	87:9000 - 87:9FFF
80:A000 - 80:AFFF	81:A000 - 81:AFFF	82:A000 - 82:AFFF	83:A000 - 83:AFFF	84:A000 - 84:AFFF	85:A000 - 85:AFFF	86:A000 - 86:AFFF	87:A000 - 87:AFFF
80:B000 - 80:BFFF	81:B000 - 81:BFFF	82:B000 - 82:BFFF	83:B000 - 83:BFFF	84:B000 - 84:BFFF	85:B000 - 85:BFFF	86:B000 - 86:BFFF	87:B000 - 87:BFFF
80:C000 - 80:CFFF	81:C000 - 81:CFFF	82:C000 - 82:CFFF	83:C000 - 83:CFFF	84:C000 - 84:CFFF	85:C000 - 85:CFFF	86:C000 - 86:CFFF	87:C000 - 87:CFFF
80:D000 - 80:DFFF	81:D000 - 81:DFFF	82:D000 - 82:DFFF	83:D000 - 83:DFFF	84:D000 - 84:DFFF	85:D000 - 85:DFFF	86:D000 - 86:DFFF	87:D000 - 87:DFFF
80:E000 - 80:EFFF	81:E000 - 81:EFFF	82:E000 - 82:EFFF	83:E000 - 83:EFFF	84:E000 - 84:EFFF	85:E000 - 85:EFFF	86:E000 - 86:EFFF	87:E000 - 87:EFFF
80:F000 - 80:FFFF	81:F000 - 81:FFFF	82:F000 - 82:FFFF	83:F000 - 83:FFFF	84:F000 - 84:FFFF	85:F000 - 85:FFFF	86:F000 - 86:FFFF	87:F000 - 87:FFFF

XDATA Memory block mapping layout	256k emulator boundary ends here.		768k emulator boundary ends here
00:0000 - 00:0FFF	01:0000 - 01:0FFF	02:0000 - 02:0FFF	03:0000 - 03:0FFF
00:1000 - 00:1FFF	01:1000 - 01:1FFF	02:1000 - 02:1FFF	03:1000 - 03:1FFF
00:2000 - 00:2FFF	01:2000 - 01:2FFF	02:2000 - 02:2FFF	03:2000 - 03:2FFF
00:3000 - 00:3FFF	01:3000 - 01:3FFF	02:3000 - 02:3FFF	03:3000 - 03:3FFF
00:4000 - 00:4FFF	01:4000 - 01:4FFF	02:4000 - 02:4FFF	03:4000 - 03:4FFF
00:5000 - 00:5FFF	01:5000 - 01:5FFF	02:5000 - 02:5FFF	03:5000 - 03:5FFF
00:6000 - 00:6FFF	01:6000 - 01:6FFF	02:6000 - 02:6FFF	03:6000 - 03:6FFF
00:7000 - 00:7FFF	01:7000 - 01:7FFF	02:7000 - 02:7FFF	03:7000 - 03:7FFF
00:8000 - 00:8FFF	01:8000 - 01:8FFF	02:8000 - 02:8FFF	03:8000 - 03:8FFF
00:9000 - 00:9FFF	01:9000 - 01:9FFF	02:9000 - 02:9FFF	03:9000 - 03:9FFF
00:A000 - 00:AFFF	01:A000 - 01:AFFF	02:A000 - 02:AFFF	03:A000 - 03:AFFF
00:B000 - 00:BFFF	01:B000 - 01:BFFF	02:B000 - 02:BFFF	03:B000 - 03:BFFF
00:C000 - 00:CFFF	01:C000 - 01:CFFF	02:C000 - 02:CFFF	03:C000 - 03:CFFF
00:D000 - 00:DFFF	01:D000 - 01:DFFF	02:D000 - 02:DFFF	03:D000 - 03:DFFF
00:E000 - 00:EFFF	01:E000 - 01:EFFF	02:E000 - 02:EFFF	03:E000 - 03:EFFF
00:F000 - 00:FFFF	01:F000 - 01:FFFF	02:F000 - 02:FFFF	03:F000 - 03:FFFF

Diagram 3

The memory mapping controls operate on a basic 64k address range. Each of the 4k mapping regions will mirror in the same address range for all of the pages. Mapping can be set for either the emulator's memory or the target system's memory.

DATA RAM

(This memory is completely handled by the Emulation Microcontroller without interference from the emulator.)

The P87C51MB2 and P87C51MC2 have 2 Kbytes and 3 K bytes of on-chip RAM respectively. Refer to Philips Documentation for usages of the different data segments under the **51MX Architecture Reference** section.

Data Memory		Size (in Bytes)	
Type	**Description**	**P87C51MB2**	**P87C51MC2**
DATA	memory that can be addressed directly and indirectly	128	128
IDATA	memory that can be addressed indirectly (where direct address is for SFRs only)	128	128
EDATA	memory that can only be addressed indirectly	1024	1024
XDATA	memory (**on-chip** "External Data") that is accessed using the MOVXinstructions	768	1792
	Total Memory	2048	3072

Table 1: On-Chip Data Memory Usage.

Extended SFRs in Seehau

Nohau has incorporated a flag in our symbol table for all special function registers that accessed via the A5H extended operation. This is by telling the software that the SFRs address is 0xFF + the normal address. This means that the SFRs in table 2 will appear with these in Seehau and the BIT symbols for each of these are tagged as BIT and not EBIT:

P4	S1CON	S1BUF	S1ADDR	S1ADEN	S1STAT
BRGCON	BRGR0	BRGR1	S0STAT	WDCON	SPE
EPL	EPM	EPH	MXCON.		

Table 2: Extended-SFRs

Shadow RAM

Nohau Currently supports one 64k page range of "Shadow RAM" that will see writes to the external data area. There is an unimplemented feature that would give the ability to have a second 64k page based on a specified extended address line giving the total shadow memory to be 128k bytes.

Currently if a write to XDATA at address 00:0000h and then a write to XDATA to address 04:0000h would result in the same cell in the data window to be displayed.

Memory / Code Coverage

Nohau's current setup is to have the ability to cover one megabyte of memory range. The hardware and software work together to set the mode, cycle type, for the coverage analysis. The available modes are the shown in table 3.

Mode	**Description**
Code Access	Any access to code memory be it a fetch, pre-fetch or read
Opcode Fetch	First true fetch of the Opcode byte for an instruction.
XDATA Write	Any write cycle to the XDATA memory space.
XDATA Read	Any read cycle to the XDATA memory space.

Table 3: Code Coverage Operational modes

April 4, 2002

Hardware Section

This document covers both the POD-C51MX and the EA256-51MX-BSW emulator board. This system is designed to emulate the Philips 8xC51MB/C enhanced microcontroller from Philips Semiconductors.

Note:

The emulator board has been specially modified and will work **only** with the POD-C51MX. The EPROM on the emulator board must be version COM 1.6. This board is also labeled with "51MX USE ONLY"

The solder side of the board has a 44-pin PGA plug. The plug goes into the target system via a 44 pin PGA to 44 pin PLCC or 44-pin QFP adapter. (To *be ordered separately*.)

The pod has five LEDs, named MON, EMUL, PD, IDLE, RES:

- MON – is red, and means that the system is in monitor mode. In monitor mode, the processor is executing code that is internal to the emulator. This code is not user code, and is used to communicate with the host PC, to set up breakpoints, etc.

- EMUL – is green, and means that the system is in emulation mode. In emulation mode, the processor is executing the user's code from the emulation RAM or the targets PROM depending on the mapping in the emulator software.

- PD – is yellow, and means that the microcontroller is in the *Power-down* state.

- IDLE – is yellow, and means that the microcontroller is in the *IDLE* state.

If you break emulation during either the Power-down or Idle states the emulator will loose control of the system and the emulation software will yield an error condition. RESET is the only way to regain control of the microcontroller and emulator again.

- RES – is red, and means that the microcontroller RESET pin is LOW (active state).

JP5 determines the mode(s) of operation for this pod board: (*refer to figure 1*)

Jumper ID	IN/ OUT	Default	Description of function
M1	**IN** **OUT**	**IN**	Trace the special function register (SFR) writes
M2	**IN** **OUT**	**OUT**	Trace the normal P1 and P3 header signals on the pod board. Trace the A16 through A23 address lines
M3	**IN** **OUT**	**IN**	PSEN is not gated. PSEN is gated
M4	**IN** **OUT**	**IN**	Ports 0 and 2 are used for General I/O. Ports 0 and 2 are used for Address / Data bus.

The board has three jumpers for power and crystal connection to the emulation chip:

- PWR – is used to select power for the processor. Power should be supplied from the emulator and the jumper should be in either the **5v** or **3v** position depending on the type of microcontroller installed in the pod. With the power jumper in the left two pins, **EXT** position, then the power will come from the target system.

- XTAL – determines if the crystal or clock is taken from the target system or the on pod crystal**.** The jumpers should be in the top two positions for crystal to be supplied from the pod or the **INT** position. If the jumpers are in lower positions, **EXT** position, then the crystal/clock is supplied from the target system.

If you use an external clock, note that XTAL1 is an input and XTAL2 is left open.

All jumpers for PWR and XTAL should be in the **INT, "internal"** position when no target is connected to the pod board.

RST connects the target RESET pin to the emulator. If the target system has a watchdog, it will probably interfere with the emulation and **RST** should then be removed.

JP2 is used to connect external signals to the emulator. They are used for trace function. The pin to the right (away from the pods cable connector) is **SY1**. The name of the pin in the center is **SY0**, can also be used in the breakpoint logic. The pin on the left (closest to the pod cable connector) is the **EM/** signal. This signal goes low during the RUN mode and goes high when you enter monitor mode of the emulator.

The header labeled FLF/, ANB/ carries signals from the emulator. The ANB/ pin is used in conjunction with the Enhanced trace boards as a signal output signal from the state machine logic.

Jumper P1/P0 upper pins carry the processor port 1 pins in order with P1.0 on the left, and P1.7 on the right. Port 1 is connected directly to the processor. The lower pins carry port 0 with P0.0 on the left and P0.7 on the right. Port 0 is emulated in the LCA. This means that the P0 has better sink capability than the processor. Output signals also appear a few clock cycles later than the real part.

The Middle row of the jumper P1/P0 is normally connected to P1 using the jumpers. The signal level on the middle pins will be reflected on the trace display. The board is delivered with the jumpers, so that P1 will be traced. External signals can be connected to the middle pins if the jumpers are removed. The input load is approximately 200k ohms.

Jumper P2/P3 upper pins carry the processor port 3 pins in order with P3.0 on the left, and P3.7 on the right. Port 3 is connected directly to the processor, except for P3.6 & P3.7 that go high speed switch logic and carries approx. 100 ohms of series resistance. The lower pins carry port 2 with P2.0 on the left and P2.7 on the right. Port 2 is emulated in the LCA. This means that the P2 has better sink capability than the processor. Output signals also appear a few clock cycles later than the real part.

The Middle row of the jumper P2/P3 is normally connected to P3 using the jumpers. The signal level on the middle pins will be reflected on the trace display. The board is delivered with the jumpers, so that P3 will be traced. External signals can be connected to the middle pins if the jumpers are removed. The input load is approximately 200k ohms.

S1 push-button is used to reset the processor, and can be used instead of a target system reset.

JP1 is used to connect the upper address lines, A16-A23, to the emulator boards bank switching memory logic. There are only 4 input banking lines that can be used, they are labeled on the J1 connectors top row as 0, 1, 2, and 3. The bottom row is the address lines A16 through A23. The **B0/** and **B1/** jumpers are used for the connection of the A16 and A17 lines to the emulators bank logic

Note:

Ports 0 and 2 are emulated, and have a slightly different AC and DC characteristics:

> Emulated I/O pins sink a minimum of 4 mA, and source (P2) minimum 4 mA for two clock cycles after an output low to high transition. As high outputs or inputs, P2 ports have 22 kohm pull-up resistors.
>
> Emulated output will change state three clock cycles later than a normal output.

P3.6 and P3.7 go through the high speed switch logic which adds approx. 100 ohms to their output impedance.

When accessing target memory, P0, and P2 will be delayed approximately 10 ns.

The clock circuitry is emulated, but is very close to the processor clock.

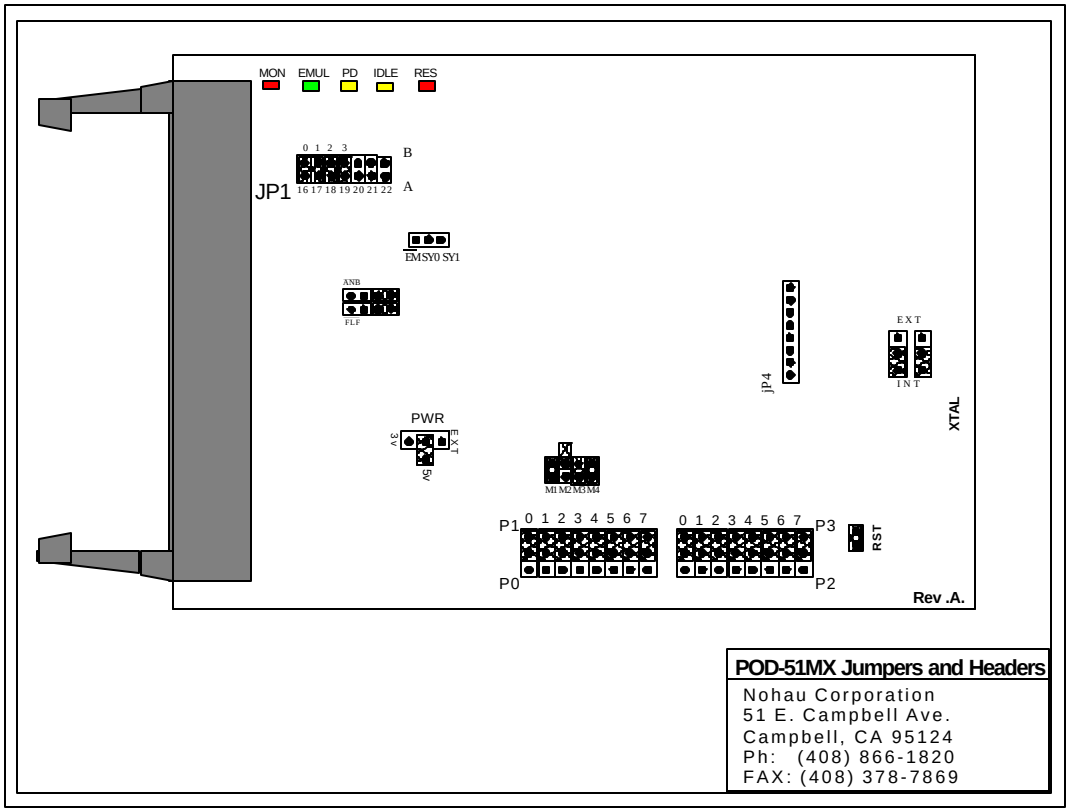
The timer/counters are stopped at breakpoints. This usually means that the serial port also stops at breakpoints. If a character is received or sent at this moment is with be distorted.

If you break emulation, during the time that the PD (power-down) or IDLE leds are lit, a number of communication errors with cause the software to loose control of the emulation system. The system will have to be restarted at this point.

Connecting the pod board to the target system:

Make sure that the power is turned off to both the emulator and the target system. Connect the black ground wire to the target first, to assist in discharging and static electricity. Now you may connect the pod to the processor connection and power up both the emulator and target systems.

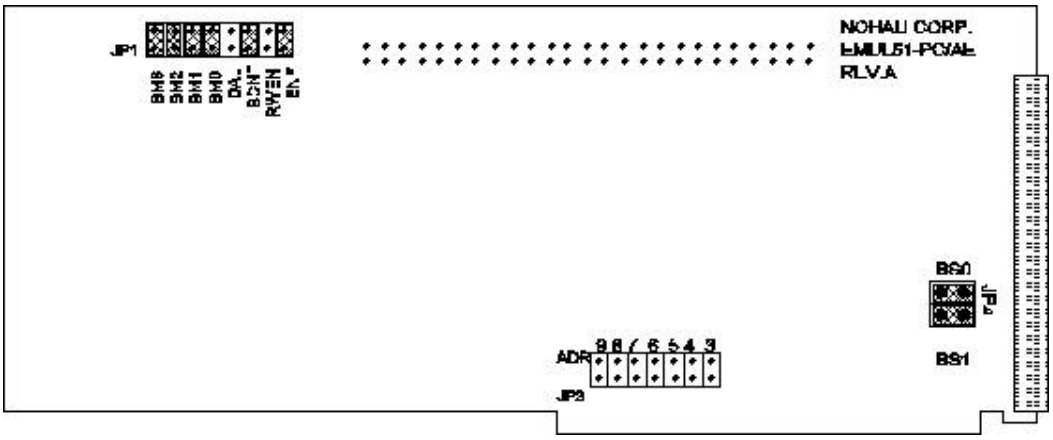
If you have the jumper for power selected for target you should power up the target first then the emulator, or both together. If selected for internal power, then power up the emulator first then the target. Power-down in the reverse sequence.



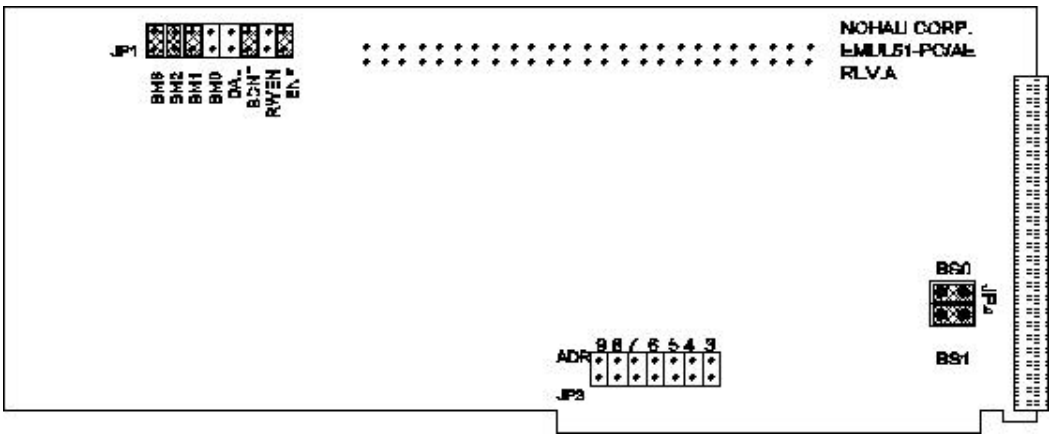
Jumpers and headers – *Figure 1* – POD-C51MX

Emulator board(s):

The EA series emulator board has two configurations and are shown below:



Configuration show for the 256k Memory model.



Configuration show for the 768k Memory model.

The JP1 jumper header has many jumpers. The BM3 and BM2 are not used with this emulation setup at this time. BM1 will decide if the memory spaces of code and data are **overlaid**or separate. BM2 is used to differentiate the difference between a board with 256k of emulation RAM or 786k. The positions for ENF, SCNT, RWEN, and DAL are factory set and should be left in their default positions.

BM0	**BM1**	**FUNCTION**
IN	IN	256K emulation memory with separate CODE (128K) and DATA (128K).
IN	OUT	256K emulation memory with **overlaid** CODE and DATA all 256K.
OUT	IN	768K emulation memory with separate CODE (512K) and DATA (256K).
OUT	OUT	768K emulation memory with **overlaid** CODE and DATA all 768K.

The jumpers labeled BS0 and BS1 should always be in so that the A16 and A17 lines are properly terminated.

April 4, 2002

Special Hardware Considerations:

Due to current design information we have found a difference in the operation of the P2 in reference to using the 8-bit MOVX operations (MOVX @Rx,A & MOVX @A,Rx). Do this issue if you have XDATA memory region mapped to the emulator and you attempt to use these operations they will fail and the data will be incorrectly written and read. We have been able with the current hardware to make this operation work if the XDATA is mapped to the target.

Special Software Considerations:

Currently, if you choose to use the option to trace the SFRs the information is no decoded in the trace buffer. A user will have to manually decode this information, as it will appear under the P1 and P3 columns.

The extended address information for operations like the EMOV instruction for the 51MX will appear under the P3 column in the trace buffer display window.