Jurnal Kejuruteraan 4 (1992) 13-24

Designing 68008 Emulator Using the VME68K8/CPU-1 Board

Zainol Abidin Abdul Rashid

ABSTRACT

Emulators is an essential tool in designing microprocessor based systems. Its value lies in its functional and electrical similarity with the actual processor in the target system. This paper describes the design of a 68008 emulator for emulating any Motorola 68008 microprocessor based system. The emulator is designed as add-on version by modifying the existing 68008 microprocessor based systems, which is the VME68K8/CPU-1 board. This add-on emulator is simple to design, able to meet the electrical and functional requirements of a good emulator and is relatively cheap.

ABSTRAK

Emulator merupakan alatan yang sangat penting dalam rekabentuk sistem-sistem yang berasaskan mikroprosesor. Nilai sesebuah emulator itu bergantung kepada kesamaan fungsian dan elektrikal dengan prosesor yang sebenar di dalam sistem sasaran. Kertas ini menerangkan rekabentuk emulator 68008 untuk meniru sebarang sistem berasaskan mikroprosesor Motorola 68008. Emulator ini direkabentuk sebagai versi tambahan dengan mengubahsuai sistem berasaskan mikroprosesor 68008 yang telah sedia ada, iaitu papan VME68K8/CPU-1. Emulator jenis ini, mudah direkabentuk, mampu untuk mencapai keperluan elektrikal dan fungsian sebuah emulator yang baik, dan lagi ia adalah murah.

INTRODUCTION

Designing and debugging microprocessor-based systems can be a very difficult task, especially when the size of the projects grow geometrically. What designers really want is a tool that makes their job easier, maximize their efficiency and effectiveness in designing their products. This tool is an emulator.

Emulators are commercially available for most of the microprocessors in use today. These, however, tend to be costly and usually require fairly complex hosts for their operation. Designers tend to develop a stand-alone in-circuit emulators because they are easy to implement and are relatively cheaper. However, the design of the 68008 emulator described in this paper is rather different. It is an in-circuit emulator that is designed as an add-on version using the existing 68008 microprocessor based system, which is the VME68K8/CPU-1 board.

IN-CIRCUIT EMULATION

In-circuit emulation is a technique that is widely used for debugging the hardware and evaluating software in a microprocessor based system. It is an important support tool provided by the microprocessor development system to help debugging the hardware and software problems encountered during the development phase of a project.

In principle, emulation is the replacement of the microprocessor of the prototype microcomputer by a piece of test equipment which is called an 'emulator'. This is intended to provide functionality of the microprocessor along with the additional capabilities to assist in the integration of the hardware and software components of the prototype. Webster's definition of emulate is 'to be as good as or better than the thing you are replacing'. To emulate a microprocessor means that the microprocessor can be replaced by something else (an emulator) and the system will run exactly the same. However this is a necessary, but not sufficient condition for debugging microprocessor systems. It is the debugging capabilities that give the emulators their great value [1].

The emulator ought to operate in the same way as the original microprocessor of the system under test. However, this may not be possible in all aspects of the operation. The degree to which the target microprocessor in the emulator environment resembles an unattached operation is known as transparency of emulation.

Transparency of emulation is measured in both functional and electrical terms. Functional transparency guarantees that the user is not deprived of or restricted in the use of any address space, instructions, interrupt system or other features normally available in the microprocessor being emulated. Electrical transparency means that the design of the emulator will operate in the user's system as much like the emulation processor as possible [2].

The emulator's value lies in its functional and electrical similarity with the actual procesor in the target system. A good emulator will maintain a high degree of transparency to the user, even at full operating speed [3].

DESIGN CONCEPTS

An emulator is actually a microprocessor-based system tool. This means that, it has a CPU, RAM, ROM and I/O on-board. What makes it different from the other microprocessor-based systems is that, it has an extra feature incorporated in its hardware intended for emulating the microprocessor of the target system.

From this idea, it can be seen that, emulators can be designed from any existing microprocesor-based system, and the designer only needs to add an extra feature on-board to make it works. Figure 1 shows the architecture of a typical emulator, the extra features on-board is shown in the dashed box. These features are the kind of interface circuitry that provides the emulation functions electrically and functionally similar to the target processor, and is known as the emulator interface.

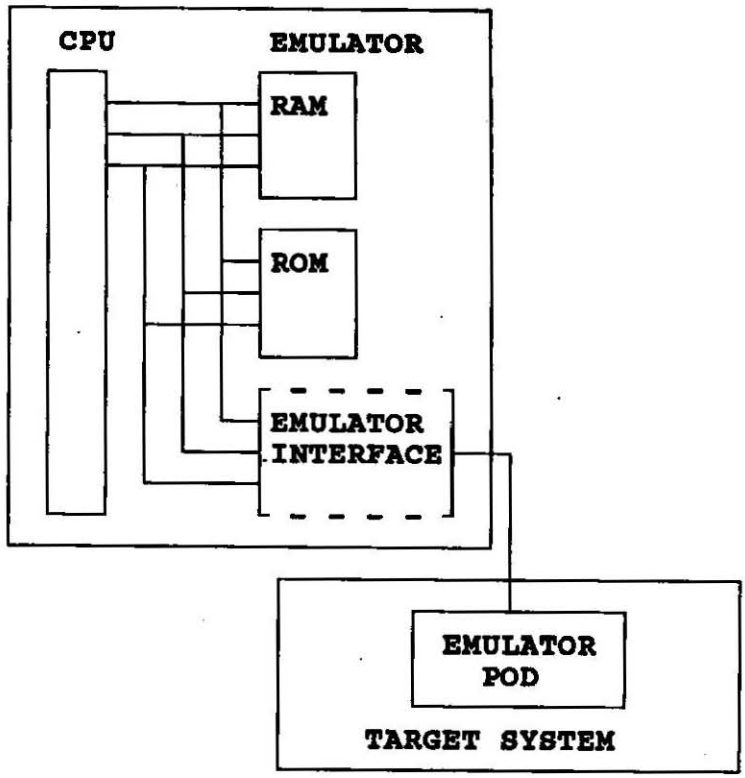

FIGURE 1. Architecture of a typical emulator

Designing emulators using an existing microprocessor-based system is not a straightforward task, in that the designer has to compromise in all aspects of electrical and functional characteristics of the system used in order to provide the functions and for the requirements of a good emulator. There are several factors which has to be taken when designing emulator from any microprocessor-based system.

The first factor is the type of processor to be emulated. Since the emulator is designed using an existing microprocessor-based system, the type of processor which can be emulated is the same as the processor used by the system board.

The second factor is how to bring all the signals of that processor on the emulator pod for emulation. If the microprocessor-based system used brought all the processor signals on the bus, the designer can buffer all the signals on the bus and bring it straight to the emulator pod. Some microprocessor-based system did not bring all processor signals on the bus and some modified the signals for some other reasons on the bus. If this is the case, the designer needs to modify the system bus in order to bring all the processor signal to the emulator pod. Clock signal can be provided from either the system board used or from the target system.

The third factor is that, all the signals brought to the emulator pod must be present at the correct timing. Care must be taken to make sure that all delayed signals are present within the clock state required by the processor timing. If the delayed signals are present at the wrong time, the

rest of the signals need to be delayed to make sure that they present at the correct timing. This is the most difficult factor and it must be fullfilled in order to make the emulator works correctly.

The fourth factor is how to provide linear address space of the microprocessor being emulated. Most microprocessor-based systems uses most of the address space for its on-board hardware. The only address space left for emulation is the off-board address. To fullfill this task, some address mapping mechanism is required. The address mapper map the off-board address in order to provide linear address space of the processor.

All of these factors must be considered and taken into the design in order for the emulator to meet the electrical and functional transparency of the target processor. This design concept will be illustrated with the design of the 68008 emulator in the following section.

68008 EMULATOR CIRCUIT DESIGN

The 68008 emulator is designed as an add-on version using the 68008 microprocessor based system, which is the VME68K8/CPU-1 board. Since the board has a CPU, RAM, and ROM on-board, the design does not need to provide all these devices. Instead it will make use of the facilities that are already available including the system monitor provided by the VME68K8/CPU-1 system. The block diagram of the 68008 emulator designed, is shown in Figure 2 [4]. It consists of three main components, the VME68K8/CPU-1 board, the emulator interface card and the cable assembly (emulator pod).

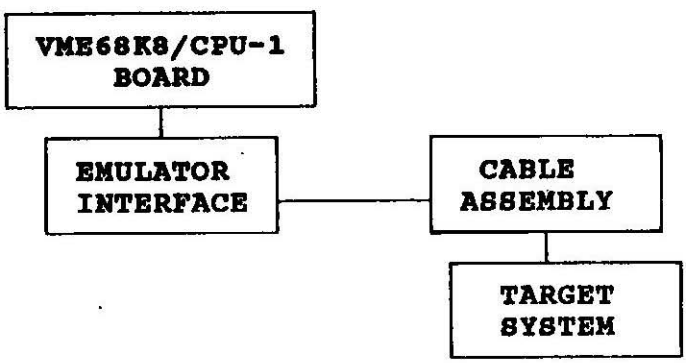

FIGURE 2. Block diagram of the 68008 emulator

THE VME68K8/CPU-1 BOARD

The VME 68K8/CPU-1 is a single board computer which uses the Motorola 68008 microprocessor as its processor and it runs on a 4MHz clock. A system block diagram of the VME68K8/CPU-1 board is shown in Figure 3 [5], the VME68K8/CPU-1 board uses the VME bus as its bus interface as shown in Table 1 [5]. Although the MC68008 processor has an 8-bit data bus (D_0 - D_7), the on-board hardware permits 16-bit data bus (D_0 - D_{15}). The data lines D_0 - D_7 will be activated when the

processor is addressing the odd address (A_0 = 1) while D_8 - D_{15} will be activated when the processor is addressing the even address (A_0 = 0).

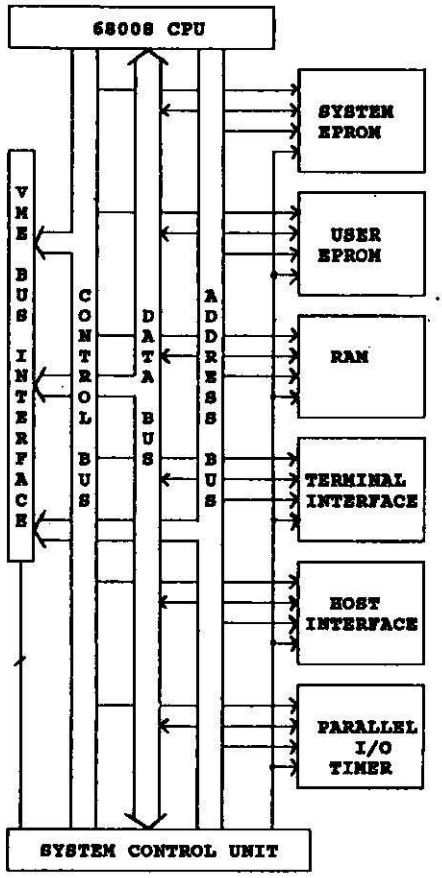

FIGURE 3. VME68K8/CPU-1 system block diagram

THE EMULATOR INTERFACE

The emulator interface section is designed as a plug-in card which can be plugged directly onto the system mother board over the VME68K8/CPU-1 bus. It is a buffer card which buffers all the 68008 signals from the VME68K8/CPU-1 bus and brings it to the emulator pod for emulation.

Since not all 68008 signals are defined on the VME68K8/CPU-1 bus, the design must provide all these undefined 68008 signals on the VME68K8/CPU-1 bus, and some modification to the board and the bus has to be made. This modification can be done since the VME68K8/CPU-1 does not use all the defined VME bus signals as shown in Table 1. These signals which are not defined on the VME68K8/CPU-1 bus are A_0, VPA, E, BR*, BG*, HALT*, and FC_0 - FC_3. There are some signals which has to be defined again on the VME68K8/CPU-1 bus, these are the data bus (D_0 - D_7), IPL0/2*, IPL1* and DS*. All these signals are taken directly from the 68008 processor on board through direct connection, buffered and brought to the VME68K8/CPU-1 bus as a piggy-back card mounted at the

TABLE 1. The VME 68K8/CPU-1 system bus

PIN	ROWA	ROWB	ROWC
1	D_0		D_8
2	D_1		D_9
3	D_2		D_{10}
4	D_3		D_{11}
5	D_4		D_{12}
6	D_5		D_{13}
7	D_6		D_{14}
8	D_7		D_{15}
9	GND		GND
10	SYCCLK		
11	GND		BERR
12	DS_1		SYSRESET*
13	DS_0*		
14	WRITE*		
15	GND		
16	DTACK*		
17	GND		
18	AS*		
19	GND		A_{19}
20	IACK*		A_{18}
21	IACKIN*		A_{17}
22	IACKOUT*		A_{16}
23			A_{15}
24	A_7	IRQ_7*	A_{14}
25	A_6		A_{13}
26	A_5	IRQ_5*	A_{12}
27	A_4		A_{11}
28	A_3		A_{10}
29	A_2	IRQ_2*	A_9
30	A_1		A_8
31	−12V	+5V	+12V
32	+5V	+5V	+ 5V

bottom of the VME68K8/CPU-1 board. All these signals are defined on row B on connector P1 of the VME 68K8/CPU-1 bus. This modified bus is listed in Table 2 and the hardware diagram for the piggy-back card is shown in Figure 4.

To provide linear address space of the 68008 processor for emulation, a hardware circuitry known as the address translator is developed. This hardware circuitry translates the off-board address of the VME68K8/CPU-1 to the 68008 linear address space. The address translator uses a look-up table which contains the address pattern for translating the address.

Looking at the address map of the VME68K8/CPU-1 board as shown in Figure 5[5], the off-board address that are available for emulation are address $01000 to $07FFFF, address $090000 to $0BFFFF, and address $0C4000 to $0FFFFF. Since most of the off-board address are located in the high-order area of the VME68K8/CPU-1 address space, the designed utilizes the high-order bits of the address lines, A_{16} - A_{19}, for translating

TABLE 2. The modified bus

PIN	ROWA	ROWB	ROWC
1	D_0	A_0	D_8
2	D_1	FC0	D_9
3	D_2	FC1	D_{10}
4	D_3	FC2	D_{11}
5	D_4	IPL2/0*	D_{12}
6	D_5	IPL1*	D_{13}
7	D_6	VPA	D_{14}
8	D_7	E	D_{15}
9	GND	BR*	GND
10	SYCCLK	BG*	
11	GND	DS*	BERR
12	DS_1	D_0	SYSRESET*
13	DS_0*	D_1	
14	WRITE*	D_2	
15	GND	D_3	
16	DTACK*	D_4	
17	GND	D_5	
18	AS*	D_6	
19	GND	D_7	A_{19}
20	IACK*	GND	A_{18}
21	IACKIN*	HALT*	A_{17}
22	IACKOUT*		A_{16}
23			A_{15}
24	A_7	IRQ_7*	A_{14}
25	A_6		A_{13}
26	A_5	IRQ_5*	A_{12}
27	A_4		A_{11}
28	A_3		A_{10}
29	A_2	IRQ_2*	A_9
30	A_1		A_8
31	−12V	+5V	+12V
32	+5V	+5V	+ 5V

the address. These four bits high-order address lines are brought as A'_{16} – A'_{19} on the interface. The translation of address A_{16} – A_{19} as address A'_{16} – A'_{19} is as follows; If the processor is addressing on address $1XXXX ($A_{19}A_{18}A_{17}A_{16}$ = 0001), the mapping hardware will translate it to address $0XXXX ($A_{19}A_{18}A_{17}A_{16}$ = 0000), and if the processor is addressing on address $2XXXX ($A_{19}A_{18}A_{17}A_{16}$ = 0010), the hardware will translate it to $1XXXX ($A_{19}A_{18}A_{17}A_{16}$ = 0001), and so on. Althrough this method capable of translating address to address 0 to $EFFFF, the address space that is available for emulation are address $0 to $6FFFF, address $80000 to $BFFFF, and address $B4000 to $FFFFF.

This address translation is done by writting the bit pattern as shown in Table 3 into the look-up table of the address translator. To hold the bit pattern, the address translator uses a hardware look-up table. The design uses the 74LS189 bipolar RAM (16 word × 4-bit) as a look-up table device. The bipolar RAM must first be written to with the required bit pattern in the particular location, and then enabled for reading to get the translated

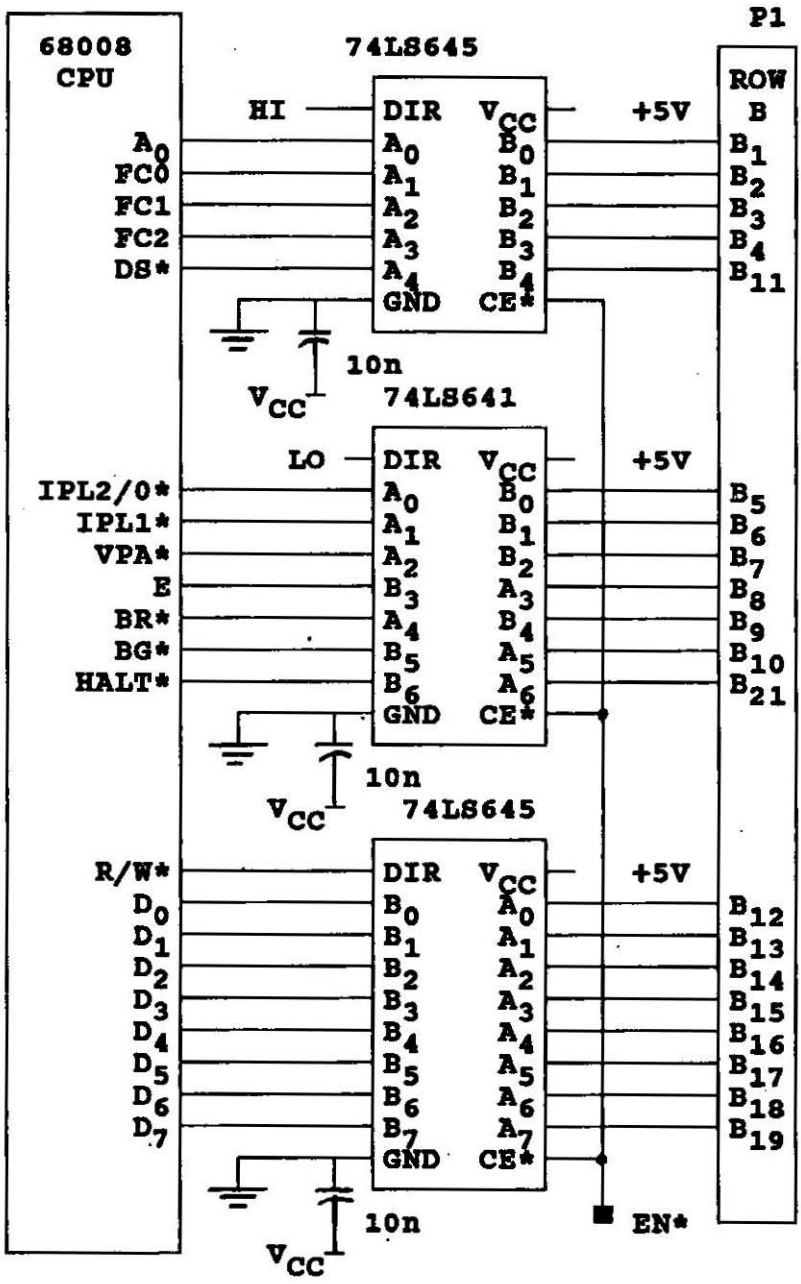

FIGURE 4. Hardware diagram of the piggy-back card.

address. For writing the bit pattern into the bipolar RAM, any parallel I/O devices can be used. The design chooses The Parallel Interface/Timer (PI/T) chip, which is available on board. This chip is selected because it works on a high frequency clock, and is actually a 68000 family processor. Figure 6 shows the hardware diagram of the address translator.

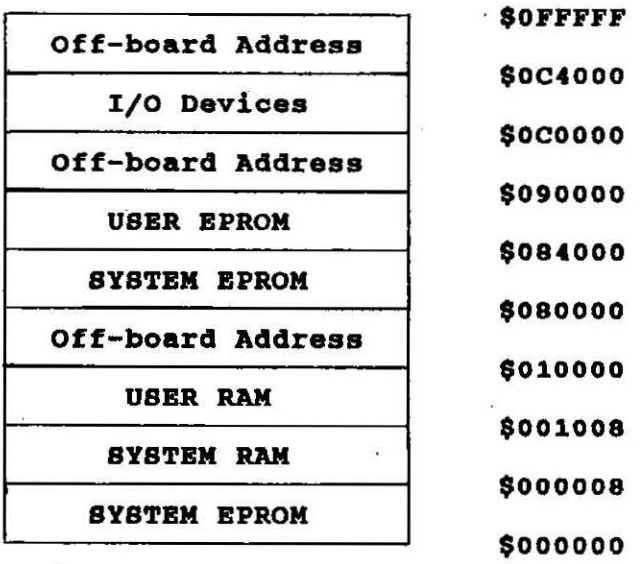

FIGURE 5. The VME68K8/CPU-1 Address Map

TABLE 3. Bit pattern for the look-up table

Actual		Address		Mapping		Address	
0	0	0	0	X	X	X	X
0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	1
0	0	1	1	0	0	1	0
0	1	0	0	0	0	1	1
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	1
0	1	1	1	0	1	1	0
1	0	0	0	0	1	1	1
1	0	0	1	1	0	0	0
1	0	1	0	1	0	0	1
1	0	1	1	1	0	1	0
1	1	0	0	1	0	1	1
1	1	0	1	1	1	0	0
1	1	1	0	1	1	0	1
1	1	1	1	1	1	1	0

Once all the 68008 signals are defined on the interface, they are buffered accordingly depending on their nature on the interface card before bringing them to the emulator pod. These signals must only activated when the valid address is not an on-board address in order to avoid bus conflict. The types of buffer used in the design is the octal

tristate and open-collector types transceiver. Figure 7 shows the hardware diagram of the emulator interface.

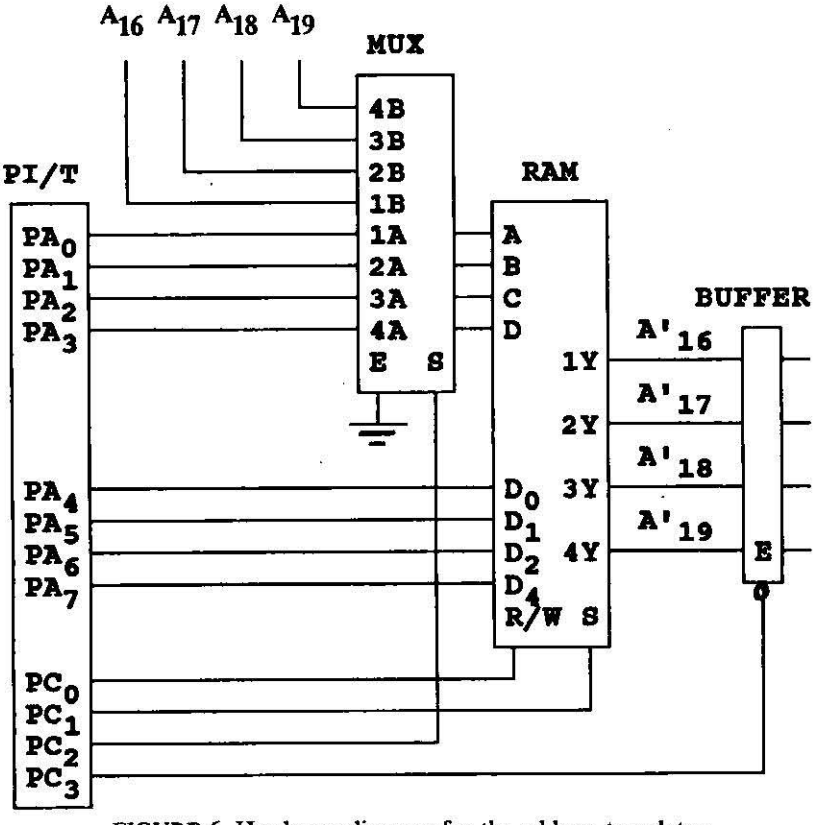

FIGURE 6. Hardware diagram for the address translator

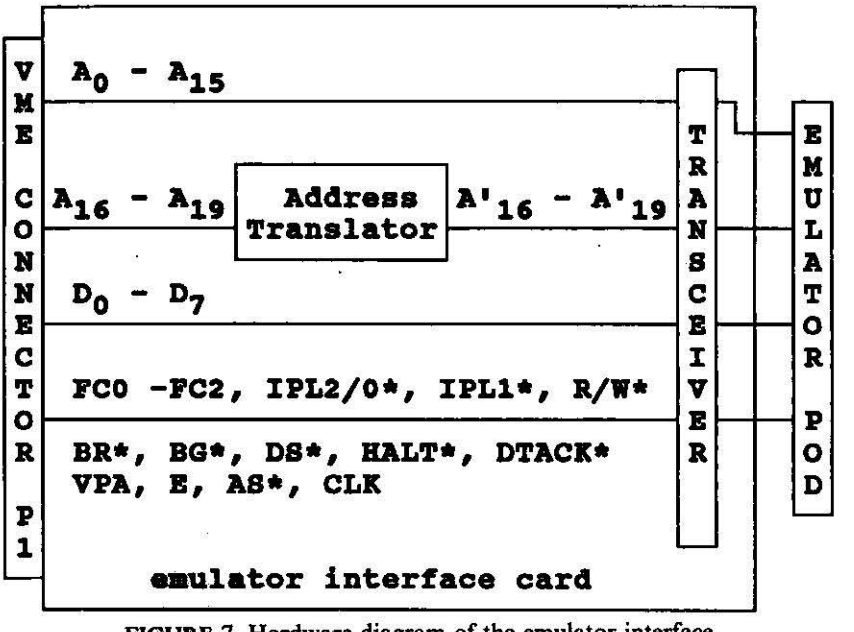

FIGURE 7. Hardware diagram of the emulator interface

As mentioned earlier, all the processor signals must be present at the emulator pod at the correct timing. In the design, all the signals brought by the piggy card are delayed about 10ns, and the translated addresses (A'_{19} - A'_{16}) are delayed about 45ns compared with the rest of the signals. Since one clock cycle is 250ns and one state is 125ns (4MHz clock), these delayed signals are present within the clock state at the emulator pod.

EMULATOR POD

To carry out all the 68008 signals for emulation, all these signals are carried out to the emulator pod through a ribbon cable. The emulator pod is a 48 way pin male connector, suited to the 48 pin 68008 processor. All the 68008 signals are positioned on the pod according to the 68008 processor pin configuration.

RESULT

The 68008 emulator designed has been tested for emulation on different 68008 microprocessor target systems. The system monitor functions provided by the VME68K8/CPU-1 system used by the emulator were also tested for debugging the target system. The testing was successful, the emulator capable of carrying out the emulation function. Although the delay occurs because of using emulator interface and emulator cable, but the performance of the emulator is measured from the emulator pod (the things that replaces the target processor) and not from the existing board. The designed emulator met the electrical and functional transparency requirements of a good emulator. because all the processor signals are present within the clock timing required by the processor. Although it can emulate only on certain 68008 address space (i.e. address $0 to $6FFFF, $80000 to $BFFFF, and $B4000 - $FFFFF), but this is sufficient enough to emulate the target systems area. The emulator designed using this technique can be categorised as a single-processor architecture emulator.

CONCLUSION

Designing an emulator from an existing microprocessor based system is not a straight forward task. Designers are restricted with the existing board design architecture and layout, and this reduces the flexibility of the design of an emulator. However, the emulator can be readily implemented before the commercial one's to be available on the market following the release of the processor and its microprocessor based system. The other advantage is that the emulator is relatively cheap to design.

REFERENCES

1. Steven E. R. 1984. Option In Microprocessor Emulation, Electronic Show and Convention, Massachusetts, USA, May 15 - 17.

2. Fahir Ergincan & Ali Saatci. 1986. *A Stand-alone In-circuit Emulator, Microprocessing and Microprogramming*. Vol. 17. North Holland.
3. Graham B. 1986. Emulators - The Essential Link, Microprocessor Development Systems. Special Report, 30 January.
4. Zainal Abidin Abdul Rashid. 1987. The Design of a 68008 Emulator Using a VME68008 Card. MSc. Dissertation, University of Bradford, U.K.
5. T.J.P. Electronics Ltd. 1985. VME68K8/CPU-1 USER MANUAL. United Kingdom.

Jabatan Kejuruteraan Elektrik, Elektronik dan Sistem
Fakulti Kejuruteraan
Universiti Kebangsaan Malaysia
43600 UKM Bangi
Selangor D.E., Malaysia.