design note

6800 coprocessor for a 6502 bus

The 6800 is a popular model for teaching microprocessors. **B T G Tan and L C Sia** have designed a board that interfaces this processor to the popular Apple II computer for practical teaching exercises

The 6800 is favoured as a teaching model in many educational establishments. Often, these establishments will have Apple II microcomputers installed, and these systems could be useful for teaching the 6800 architecture in practice if a suitable interface board was available. This paper describes the construction of a 6800 board designed for this purpose which interfaces the Apple's 6502 processor. A test program has been run on the board to assure its operation.

microprocessors   interfacing   6800   Apple II

The 8-bit 6502 microprocessor has been one of the most widely used microprocessors, particularly in small personal computers. One of the most popular personal computers using the 6502 is the Apple II computer and its derivatives. The Apple II motherboard incorporates an eight-slot I/O bus which greatly facilitates I/O interfacing. The I/O bus is allocated a 4k space in the 6502's 64k memory map, and each slot has a portion of this 4k space allocated to it and decoded on the motherboard. Though the slots are primarily meant for I/O, the I/O bus actually includes the complete 6502 bus including the entire 16-bit address bus.

Several interface boards have been designed for the 6502 which make use of the accessibility of the 6502 bus on the I/O slots to provide coprocessors for the 6502. The most popular coprocessor used in this way is the Z80, which enables the widely used CP/M operating system to be run on the Apple II[1]. Other processors which have been interfaced to the bus include the 6809, and 16-bit processors with 8-bit data buses such as the 8088 and 68008.

The 6502 is itself a derivative of the Motorola 6800, both processors having similar bus structures. The 6800 is still widely used in dedicated applications, especially in its single-chip microcomputer version, the 6801. Its elegant architecture and well structured instruction set have favoured it as a microprocessor teaching model, and it is still widely used as such in microprocessor courses and textbooks[2]. The availability of a 6800 board for the Apple II would be useful for teaching 6800 architecture and machine code, in situations where an Apple II was already at hand; however, such a 6800 board is at present not available commercially. This paper describes the design and construction of such a 6800 board for the Apple II which was intended to fulfil these aims.

Department of Physics, National University of Singapore, Kent Ridge, Singapore 0511

The 6800 board was designed initially to take over the Apple II bus automatically from the Apple's 6502 when power was switched on. The design of the board was subsequently improved to include a 'softswitch' which enables the 6502 to transfer control of the bus to the 6800 under 6502 software control. This is accomplished by incorporating an onboard control register which resides within the device address space allocated to the slot into which the board is plugged. In addition, the board includes a 2k 2716 EPROM which may contain a monitor program for the 6800.

6800 BOARD DESIGN

The circuit design of the 6800 board is shown in Figure 1. There are a total of 12 ICs on the board, including the 6800 and the 2716 EPROM. The external connections are all made to the Apple II bus, the lines of which are identified by a designation and number[3]. The data bus lines from the 6800 are connected to the Apple II data bus via a 74LS245 8-bit bidirectional tristate buffer. Similarly, the 6800's address bus and R/W line are connected to the equivalent Apple II bus lines via three 74LS367 hex unidirectional tristate buffers. These four buffers can be enabled by the 6800's VMA line via an inverter. The direction of data flow through the 74LS245 buffer when it is enabled is determined by the R/W line and hence by whether a read or write operation is taking place with respect to the 6800.

The 2716 EPROM was included on the board to provide a means of carrying a dedicated 6800 program, such as a simple monitor program. The address range for the 2716 was allocated to the range $F800 to $FFFF so that it would include the 6800's interrupt vectors which are from $FFF8 to $FFFF. The chip enable ($\overline{CE}$) and output enable ($\overline{OE}$) lines of the 2716 are pulled low, thus enabling the chip, when the address lines A11 to A15 from the 6800 and the VMA line go low. This ensures that the 2716 responds to the address range $F800 to $FFFF only when the 6800 is generating a valid memory address.

The 6800 is activated by a 74LS259 memory register chip. This chip has one input line and eight output lines each of which is enabled by a 3-bit binary address. The three address lines are connected to A0, A1 and A2 of the Apple II address bus. The 74LS259's enable ($\overline{E}$) line is connected to the device select ($\overline{DS}$) line of the Apple II bus. This effectively places the 74LS259 in the address range $C080 + $n0 to $C087 + $n0 where n is the slot number (n = 0–7), ie within the space in the Apple II memory map reserved for the slot I/O. The input line is
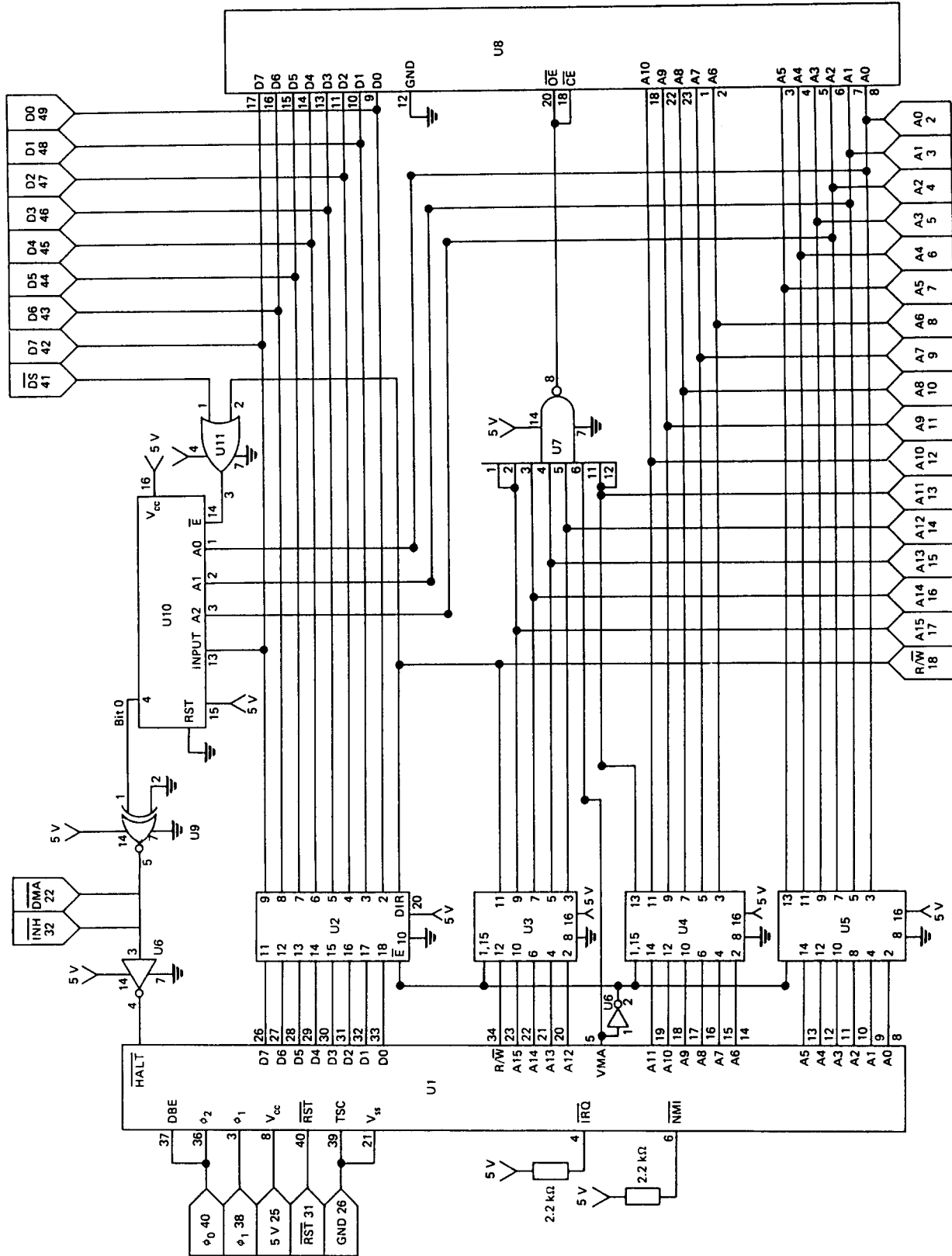
Figure 1. Circuit diagram for the 6800 board. U1 = 6800 microprocessor; U2 = 74LS245; U3, U4, U5 = 74LS367; U6 = 74LS04; U7 = 74LS30; U8 = 2716; U9 = 74LS266; U10 = 74LS259; U12 = 74LS232. Decoupling capacitors (1 μF) at U1, U2, U3, U4, U5 and U7.

connected to the D7 line of the data bus, and the reset line to the reset (RST) of the Apple II bus. Thus when the Apple II is powered up, all the eight output lines of the 74LS259 are reset to '0'. If data is now placed on the data bus to make D7 carry a '1', then any one of the above eight addresses would enable the corresponding output line to go to a '1'. The device select line is ORed with the R/W line to ensure that data from D7 will be input to the 74LS259 only during a write cycle, otherwise spurious data might be input.

Only one of the output lines of the 74LS259, ie that corresponding to I/O address $C080 + $n0 is actually used. This output line is connected to the DMA and INH lines of the Apple II bus, and to the HALT line of the 6800 via an inverter. An open-collector NOR gate was used to buffer the 74LS259 output line from the DMA and INH lines, as the latter are pulled high by resistors to the 5 V power line.

At the moment of powering up the Apple II, all the output lines of the 74LS259 are reset to 0. Thus the DMA and INH lines are held high and the HALT line is pulled low. In this state the 6800 is halted and isolated from the Apple II bus. By addressing the 74LS259 appropriately and putting a 1 on the D7 line of the data bus, the output line of the 74LS259 can be set to 1. This will then pull the DMA and INH lines low; the INH will then isolate the Apple II's ROMs on the motherboard from the bus and the DMA line will halt the Apple II's 6502 and isolate it from the bus as well. The HALT line is then pulled low, thus activating the 6800 and connecting it to the bus. The converse operation, that of deactivating the 6800 and activating the 6502, can be performed by the 6800 by addressing the 74LS259 and setting D7 to 0.

TESTING

A test program was written and programmed into the 2716 onboard EPROM (Appendix 1). The EPROM occupies the addresses $F800 to $FFFF. The program first sets the Apple II screen to the text mode and clears the screen. It then puts a blinking cursor at the top of the blank screen and waits for the input of four hex characters from the Apple II keyboard. When these four characters are received, they are used as pointers to a particular address in the memory map. The contents of this address will then be retrieved and displayed in hex characters on the screen. In this way, the program enables the contents of any memory location to be displayed.

This simple program worked successfully as intended, showing that the 6800 was truly in control of the Apple II bus and memory map. The EPROM could easily be made to carry a complete monitor program which would enable the contents of memory locations to be changed as well as displayed, and machine language programs to be run. This would enable the board to be used to teach 6800 machine language programming using the Apple II. It would also be possible to write a disc booting routine which would boot up one of the standard 6800 operating systems such as FLEX, which is already available for commercial 6809 coprocessor boards.

It is hoped that this simple 6800 coprocessor board design will be of use to those who may wish to run 6800 machine language programs on the Apple II bus or any other microcomputer using a 6502-based bus.

REFERENCES

1 *Microsoft Softcard (Z80 card) reference manual*
2 **Levanthal, L A** *Introduction to microprocessors: software, hardware, programming* Prentice-Hall Englewood Cliffs, NJ, USA (1978)
3 *Apple II reference manual*

APPENDIX 1: LISTING OF TEST PROGRAM

F800					ORG		$F800	
F800	B6	C0	51		LDA	A	$C051	set screen mode to text
F803	B6	C0	51		LDA	A	$C051	clear keyboard strobe
F806	CE	04	00		LDX		#$0400	first location of screen
F809	86	A0			LDA	A	#$A0	ASCII code for blank
F80B	A7	00		LOOP	STA	A	$0, X	clear the screen location
F80D	08				INX			
F80E	8C	07	F8		CPX		#$07F8	last location?
F811	26	F8			BNE		LOOP	no, do again
F813	CE	03	01		LDX		#$0301	
F816	86	60		START	LDA	A	#$60	ASCII code for blinking cursor
F818	A7	FF			STA	A	$FF, X	at top line of the screen
F81A	F6	C0	10	KEY	LDA	B	$C010	read keyboard register
F81D	2A	FB			BPL		KEY	no key depressed, do again
F81F	E7	FF			STA	B	$FF, X	got key, display it at cursor position
F821	B6	C0	10		LDA	A	$C010	clear keyboard strobe
F824	C1	C0			CMP	B	#$C0	greater than 9?
F826	2B	02			BMI		PLUS	no

F828	CB	09			ADD	B	#$09	yes, add offset
F82A	C4	0F		PLUS	AND	B	#$0F	masking to get actual key value
F82C	E7	00			STA	B	$0, X	and store it
F82E	08				INX			next address byte
F82F	8C	03	05		CPX		#$0305	four bytes?
F832	26	E2			BNE		START	no, do again
F834	86	BA			LDA	A	#$BA	ASCII code for colon
F836	B7	04	04		STA	A	#$0404	display it
F839	86	04			LDA	A	#$04	
F83B	78	03	01	AGN	ASL		$0301	
F83E	78	03	03		ASL		$0303	
F841	4A				DEC	A		
F842	26	F7			BNE		AGN	shift values
F844	B6	03	01		LDA	A	$0301	
F847	BA	03	02		ORA	A	$0302	
F84A	B7	03	01		STA	A	$0301	
F84D	B6	03	03		LDA	A	$0303	
F850	BA	03	04		ORA	A	$0304	
F853	B7	03	02		STA	A	$0302	address bytes stored in 0301 and 0302
F856	FE	03	01		LDX		$0301	
F859	A6	00			LDA	A	$0, X	data byte stored in accumulator
F85B	36				PSH	A		
F85C	84	F0			AND	A	#$F0	get most significant bit first
F85E	44				LSR	A		
F85F	44				LSR	A		
F860	44				LSR	A		
F861	44				LSR	A		
F862	81	0A			CMP	A	#$0A	greater than 9?
F864	2B	02			BMI		ADD2	no
F866	8B	07			ADD	A	#$07	yes and offset
F868	8B	B0		ADD2	ADD	A	#$B0	add offset
F86A	B7	04	05		STA	A	$0405	display it
F86D	32				PUL	A		
F86E	84	0F			AND	A	#$0F	get least significant bit
F870	81	0A			CMP	A	#$0A	greater than 9?
F872	2B	02			BMI		ADD3	no
F874	8B	07			ADD	A	#$07	yes, add offset
F876	8B	B0		ADD3	ADD	A	#$B0	add offset
F878	B7	04	06		STA	A	$0406	display it
F87B	01			STOP	NOP			
F87C	20	FD			BRA		STOP	
					END			
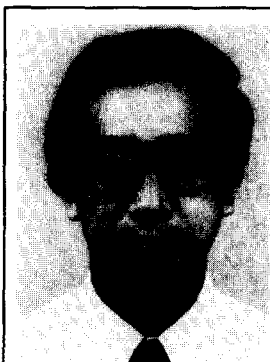
Bernard Tan graduated in 1965 with an honours degree in physics from the University of Singapore, and in 1968 with a DPhil from Oxford University, UK. He is a chartered engineer and a member of the IEE and the IERE. Since 1968 he has taught at the National University of Singapore where he is now an associate professor in physics. His research interests include microwave properties of semiconductors, computer-assisted acoustical analysis and synthesis, and microprocessor applications.
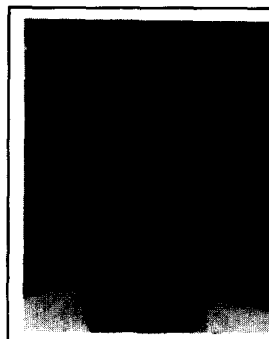
L C Sia obtained an honours degree in physics from the National University of Singapore in 1984. He is now teaching physics at St. Andrew's Junior College, Singapore. He is also doing part-time research in high energy physics for an MSc at the National University of Singapore.