Designing with the 68008 microprocessor

Ideal for use in low-cost, high-volume applications like personal computers and small business machines, the 68008 is an eight-bit microprocessor with a 32-bit architecture. This two-part article describes its main features and how it's used with other microcomputer components like rom, ram and peripheral devices.

by A. J. Barth

All of the microprocessors in the M68000 family of high-performance processors and peripherals, including the 68008, are based on the same 32-bit architecture. The 68008 has an eight-bit external data bus; others have 16 or 32-bit buses. The once-clear divisions between eight, 16 and 32-bit microprocessors are becoming blurred; with the 68008 the designer is now able to have a high-performance microprocessor with a 32-bit architecture in small cost-effective systems using eight-bit data buses.

The architecture of the 68008 is identical to the original member of the family, the 68000, a processor with a 16-bit external data bus. From the programmer's point of view the two processors look identical, so that the 68008 is completely code-compatible with the 68000. This means that any program developed for one processor will run on the other. This is true for object code as well as source code. Other 68000-family microprocessors such as the 68010, virtual memory version of the 68000, and the 68020, very high performance 32-bit mpu, have achitectures which are upward-compatible with that of the 68008, making it easier to upgrade 68008-based systems. For example, any user program written for the 68008 will execute correctly on the 68000, 68010 and 68020 without need for modification.

Using standard rams and roms a smaller minimum-sized system can be constructed with the 68008 than with the 68000. Cost savings are made by producing the 68008 in a 48-pin dual-in-line package as opposed to the 64-pin version for the 68000. Eight pins are saved by halving the reduced data bus. Other minor hardware differences allow more pins to be shed (Fig. 1), for instance a few of the high-order address lines of the 68000 are not brought off-chip on the 68008. Even so, this still allows direct addressing of over one megabyte of memory – huge compared to that of conventional eight-bit microprocessors and more than enough for the low-end applications for which the 68008 is intended.

The 68008 is as fast as the 68000 when accessing byte-sized operands. However, because of its byte-sized data bus the 68008 needs to access 16-bit words as two successive bytes. As a result, the overall throughput of the 68008 is less than that of the 68000. For the same processor clock and for a typical mix of instructions, the 68008's performance is about 60% of that of the 68000. This is still a lot of raw processing power and will meet a need for low-end applications.

Because the architectures of the 68000-family microprocessors are so similar, knowing the 68008 means knowing much about the other processors. The 68008 is characterized by its 'clean', regular and consistent structure and in particular, emphasis was given to the architecture to make it regular with respect to the registers, instructions, addressing modes and data types.

Register set

The 68008 programming model has a large number of general-purpose 32-bit data and address registers, Fig. 2. There are eight general-purpose 32-bit data registers, D_0-D_7, for byte (8-bit), word (16-bit), and long word (32-bit) operations. Seven address registers A_0-A_6 and two system stack pointers A_7, may be used as software stack pointers and base address registers. In addition these registers may be used for word and long-word address operations. All 17 registers may be used as index registers.

High-performance microprocessors are expected to rapidly handle complex functions having a large number of parameters. The 68008 can maintain most or all of these parameters in processor registers, which is both fast and makes the programs efficient and elegant. Microprocessors with only a few registers in such situations need to continuously swap parameters between registers and external memory.

As the 68008 has general-purpose registers, the programmer and not the microprocessor-chip designer decides which registers are used. It does not dictate that certain instructions use certain registers. This not only eases the task of the assembler language programmer but also makes high-level language compiler-generated code more efficient. Many of the instructions and addressing modes which

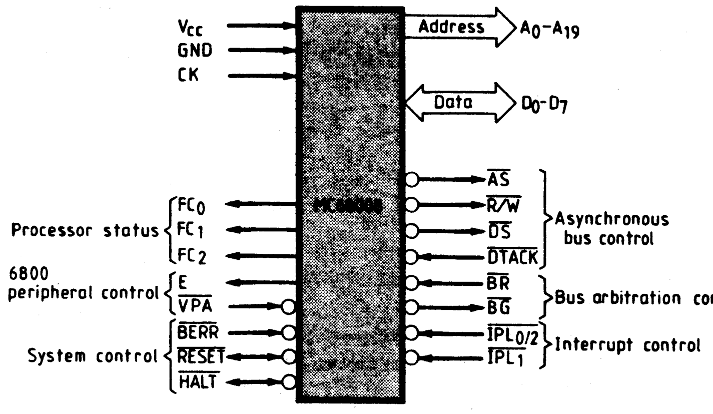


Fig. 1 68008 signal lines. The microprocessor has an internal 32-bit architecture and an eight-bit external data bus. Packaged in a 48-pin package the 68008 has non-multiplexed buses and a non-segmented 1 megabyte address space.

Reprinted with permission 5-22-85

operate on the address registers may also be used with the 32-bit program counter. This makes it easy to write position-independant software that will execute correctly no matter where the code is loaded in memory.

The 68008 has a 16-bit status register which consists of two parts: a user byte and a system byte. The user byte is accessible by any program and contains the usual condition code flags associated with the execution of instructions, condition like Negative, Zero, Carry, Overflow, etc. The system byte of the status register is accessible only by a supervisory program (usually the operating system) and is used to control the operating mode.

Addressing range

The 68008 has a large linear addressing range. It can directly access one megabyte of external memory without paging or segmentation. Many microprocessors are able to access fairly large memory space but need to do so via a narrow window called a segment or page. This may be useful in a few applications, but in most situations it is an irritating handicap because the programmer is obliged to keep repositioning the window to access the desired location.

Like other Motorola microprocessors the 68008 has memory-mapped i/o. This enables the programmer to use the m.p.u's sophisticated instructions and addressing modes to operate on i/o as well as memory.

Instruction set

The 68008 has a powerful, flexible and easy-to-use instruction set. There are 56 basic instructions (Table 1). This is actually less than the ten-year-old 6800 microprocessor. However because of the regularity of the 68008 architecture, those instructions which use registers may use any register with almost any addressing mode and data type. These permutations yield many thousands of useful operations, compared to less than 100 for the 6800; the 68000 family philosophy being to provide a small number of easy-to-remember and flexible general-purpose instructions.

The instruction set covers the following classes of operations:

- data movement
- integer arithmetic
- logic
- shift and rotate
- bit manipulation
- binary-coded decimal
- high-level language support
- program control and
- system control.

Operations on data in registers and memory are independent of the data size and usually involve a source and a destination operand. The programmer need only remember one mnemonic for each type of operation and then specify data size and addressing modes for both the source and destination operands. Consistency is maintained as all data registers and memory locations may be a source or destination for most operations on integer data.

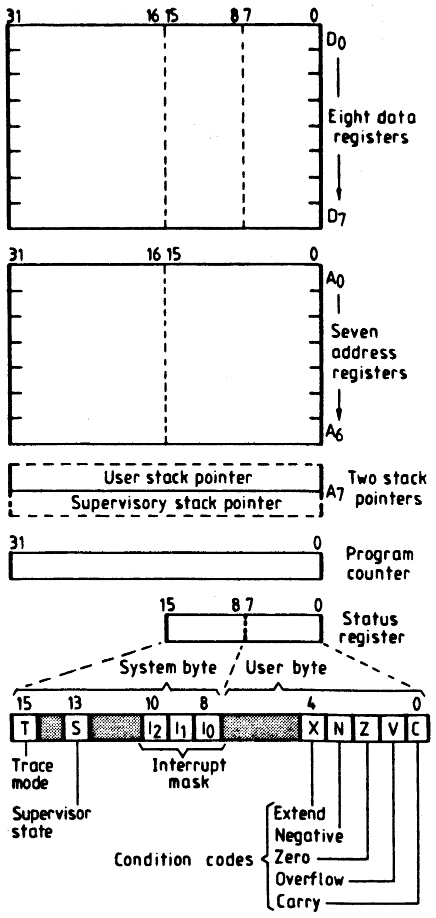


Fig. 2 68008 programming model: eight general-purpose data registers for 8, 16 and 32-bit operations, seven address registers, and two system stack pointers for software stack pointers and base address register. All 17 maybe used as index registers.

Like all M68000 microprocessors the 68008 instructions are implemented by microcode rather than random logic, so that, for example, the execution of undefined instructions no longer leads to unpredictable results.

Data types

The 68008 operates on five main data types. Operations may be on bits, b.c.d. digits, bytes, words and long-words. For integer arithmetic the programmer need not remember different instructions for different data sizes. The required data size is simply appended to the instruction as the program is written.

Addressing modes

An addressing mode is the method by which the data or other operand is accessed by the processor. The availability of powerful and flexible modes usually means performing an operation with just one instruction which would otherwise take many. This results in programs executing faster, being smaller, easier to read and to maintain. The 68008 has 14 powerful addressing modes (See Table 2). They operate consistently and are independent of the instruction itself.

Program privilege scheme

A two-level program privilege scheme provides security and high reliability. Programs should access only their own code and data areas, and ought to be restricted from accessing the information which they do not need and must not modify. Such a scheme not only prevents the deliberate tampering with data but also guards against a faulty program running wild and altering other programs.

The 68008 operates at one of two privilege levels, the supervisor level or at the user level. At the supervisor level programs have access to all processor resources and can execute any instruction or access any register. Normally, only the controlling operating system or its kernel runs at this level. This code is normally relatively small, well-tested and therefore reliable. All the rest of the software, which includes both the utility and application programs, executes at the user level and has access only to a subset of the total processor unit resources, the resources governing control of the system being protected from these programs. If a user-level program attempts to execute a 'privileged' instruction or to access a supervisor register, control is immediately taken away and given to the controlling supervisor program which can take some corrective action.

Interrupt structure

In most applications, programs are seldom executed instruction-after-instruction without a break. The need frequently arises to respond to an event or exception. Such exceptions may be the hardware interrupts caused by external logic, or the software interrupts caused by the recognition of some condition internal to the processor unit. High-performance microprocessors must be able to respond rapidly to a large variety of exceptions with varying degrees of priority.

Three levels of priority are provided for external hardware interrupts. By use of the three-bit interrupt mask in the 68008 status register (Fig. 2) the supervisor program may postpone handling external interrupts with priorities less than that contained in the mask. When an interrupt

Andrew Barth is a senior staff engineer with Motorola's systems engineering group in East Kilbride. He graduated in physics from Leeds University and since 1976 has worked in microprocessor systems design in Germany and the USA, as well as in the UK.

Table 1. 68008 instruction set

Mnemonic	Description
ADBC	add decimal with extend
ADD	add
AND	logical and
ASL	arithmetic shift left
ASR	arithmetic shift right
B$_{cc}$	branch conditionally
BCHQ	bit test and change
BCLR	bit test and clear
BRA	branch always
BSET	bit test and set
BSR	branch to subroutine
BTST	bit test
CHK	check register against bounds
CLR	clear operand
CMP	compare
DB$_{cc}$	test condition, decrement & branch
DIVS	signed divide
DIVU	unsigned divide
EOR	exclusive or
EXG	exchange registers
EXT	sign extend
JMP	jump
JSR	jump to subroutine
LEA	load to effective address
LINK	link stack
LSL	logical shift left
LSR	logical shift right
MOVE	move
MOVEM	move multiple registers
MOVEP	move peripheral data
MULS	signed multiply
MULU	unsigned multiply
NBCD	negate decimal with extend
NEG	negate
NOP	no operation
NOT	one's complement
OR	logical or
PEA	push effective address
RESET	reset external deuces
ROL	rotate left without extend
ROR	rotate right without extend
ROXL	rotate left with extend
ROXR	rotate right with extend
RTE	return from exception
RTR	return and restore
RTS	return from subroutine
SBCD	subtract decimal with extend
S$_{cc}$	set conditional
STOP	stop
SUB	subtract
SWAP	swap data register halves
TAS	test and set operand
TRAP	trap
TRAPV	trap on overflow
TST	test
UNLK	unlink

is recognised the processor performs an interrupt acknowledgement sequence (IACK). During IACK the peripheral being acknowledged may indicate that program control should be given to any one of 256 interrupt service routines (vectored interrupt method), or to one of three service routines corresponding with the hardware interrupt priority level (autovector interrupt method). Most of the M68000-family peripherals use the vectored interrupt method in which the peripheral provides an eight-bit vector number on the data bus. The 68008 uses this vector number to determine which of the 256 interrupt routine addresses in its interrupt vector table to use. The less sophisticated peripherals use the autovector interrupt method which have seven vectors reserved for them. In either case the external hardware needed to interface both kinds of perpherals to the 68008 is minimal.

Some 68008 instructions are designed to cause internal interrupts, some always and others only upon detection of certain conditions. An example of the last is the execution of a 'privileged' instruction. If a supervisor-level program executes such an instruction, the instruction will execute normally and no exception will occur. However, if a user-level program attempts to execute it, a prividege violation exception occurs and program control is given immediately to the appropriate interrupt service routine.

A number of exceptions correspond to error conditions, either those detected by external hardware or by the processor itself. For example, the Bus Error input (BERR) may be used by external hardware to cause the 68008 to abandon the current bus cycle and give program control to the Bus Error interrupt routine, or, by the simultaneous use of BERR and HALT, to retry the current bus cycle.

A very useful feature is the trace exception, which enables a supervisor-level program to step through a target program on an instruction-by-instruction basis. Each time the target program executes an instruction, no matter which instruction it may be, control is returned to the supervisory program. No external hardware is required to implement the program tracing as it is part of the processor architecture.

Asynchronous data bus

Like most of the other Motorola microprocessors, the 68008 data bus is not multiplexed – the m.p.u. pins used for the data bus are not shared with other signals. Some microprocessors multiplex the data and address buses onto the same pins to reduce the total number of pins. Non-multiplexed microprocessors such as the 68008 require more pins and sometimes a more expensive package. However, non-multiplexed buses have many advantages: they can operate much faster, dissipate less chip power, and do not require external demultiplexing hardware. Analysis shows that multiplexed microprocessor systems are more costly than non-multiplexed systems because the microprocessor and demultiplexing i.cs together cost more, occupy more board space and have more pins overall.

The 68008 has an asynchronous data bus. The time taken to transfer data to or from a memory or peripheral device via the data bus is variable. The memory or peripheral device signals the processor when it is ready to make the data transfer by use of a special handshake line called data transfer acknowledge (DTACK). The advantage of this asynchronous scheme is that each bus cycle can be fine-tuned to the speed of the particular device being accessed. If the device is rather slow, the processor simply marks time until the device is ready. In this way the 68008 runs at the fastest rate that memory and peripherals can go, which maximizes system throughput.

Most M68000-family peripherals have a pin for the DTACK handshake signal, and interfacing such parts to the 68008 microprocessor is simple. Even those peripherals originally designed to work with synchronous processors, like the 6800 or 6809 microprocessor units, may be interfaced to the 68008 with minimal hardware. This is because the 68008 has several signal pins specially for this purpose. By use of these signals, the M6800-type peripheral device signals the processor to perform the current bus cycle synchronously, making the 68008 behave like a synchronous microprocessor for this one bus cycle.

The 68008 m.p.u. uses a two-line bus arbitration scheme which enables the data bus to be shared efficiently with other microprocessors unitss in a multiprocessor system and with other bus masters such as d.m.a. controllers. WW

To be continued with interfacing details.

Table 2. 68008 addressing modes

Mode	Generation
Register direct addressing	
Data register direct	$EA=D_n$
Address register direct	$EA=A_n$
Absolute data addressing	
Absolute short	EA=(next word)
Absolute long	EA=(next two words)
Program counter relative addressing	
Relative with offset	$EA=(PC)+d_{16}$
Relative with index and offset	$EA=(PC)+(X_n)+d_8$
Register indirect addressing	
Register indirect	$EA=(A_n)$
Postincrement register indirect	$EA=(A_n), A_n \leftarrow A_n+N$
Predecrement register indirect	$A_n \leftarrow A_n-N, EA=(A_n)$
Register indirect with offset	$EA=(A_n)+d_{16}$
Indexed register indirect with offset	$EA=(A_n)+(X_n)+d_8$
Immediate data addressing	
Immediate	Data=next word(s)
Quick immediate	Inherent data
Implied addressing	
Implied register	EA=SR, USP, SP, PC

EA effective address
A_n address register
D_n data register
X_n address or data register used as index register
SR status register PC program counter
d_8 eight-bit offset (displacement)
d_{16} sixteen-bit offset (displacement)
N=1 for byte 2 for words and 4 for long words
( ) contents of
$\leftarrow$ replaces

Designing with the 68008 microprocessor

A member of the 68000 8/16/32 bit processor family, the 68008 has an internal 32-bit architecture and an eight-bit external data bus. This second article shows how it is used with other microcomputer components – rom, ram and peripheral devices.

by A. J. Barth

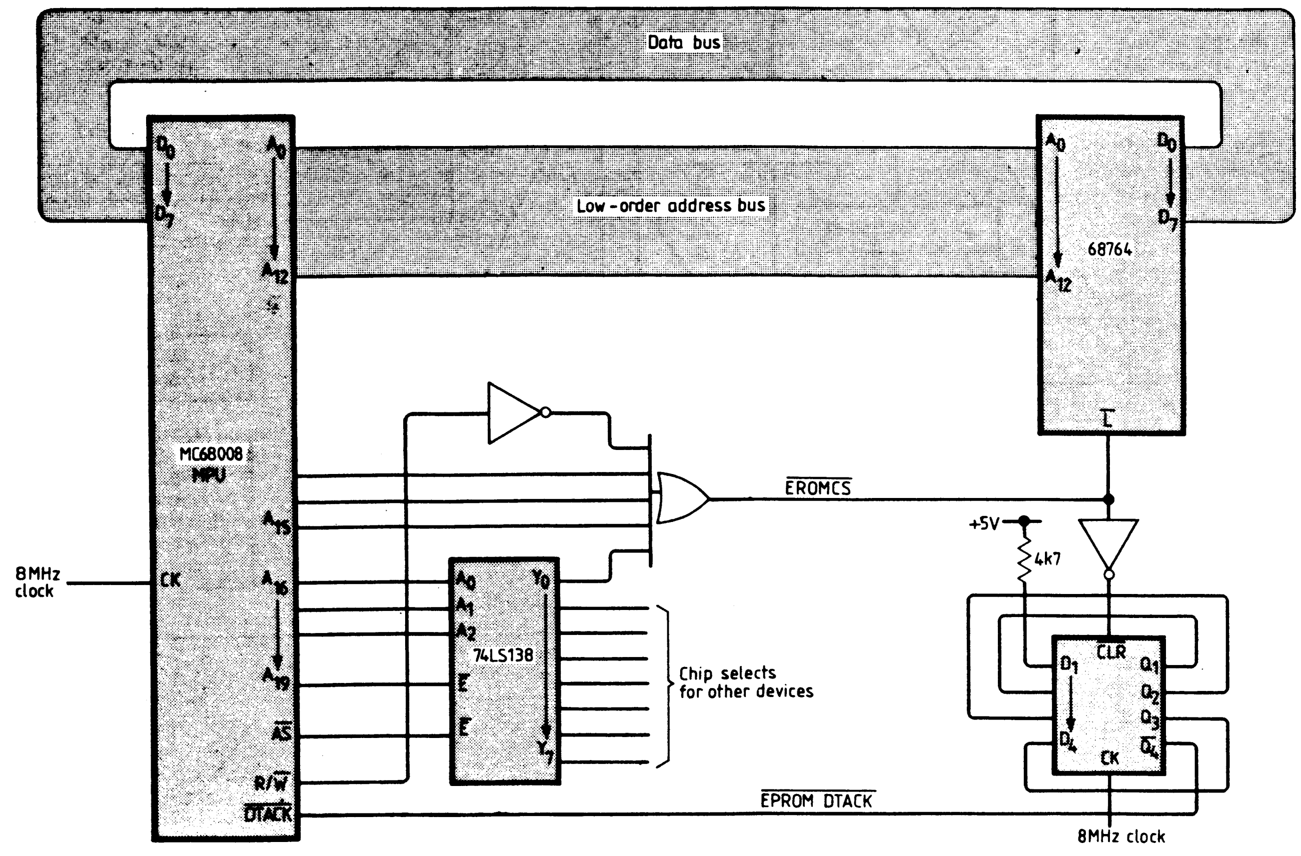
Interfacing the 68008 to memory, peripherals and other microcomputer devices is straightforward. The examples shown here are practical circuits using the popular 74LS z.t.l. family. When the separate examples are brought together to form a complete system numerous circuit rationalizations can be made. In the case of very high volume designs f.t.l. tends to be replaced by custom circuits.

68000 with rom

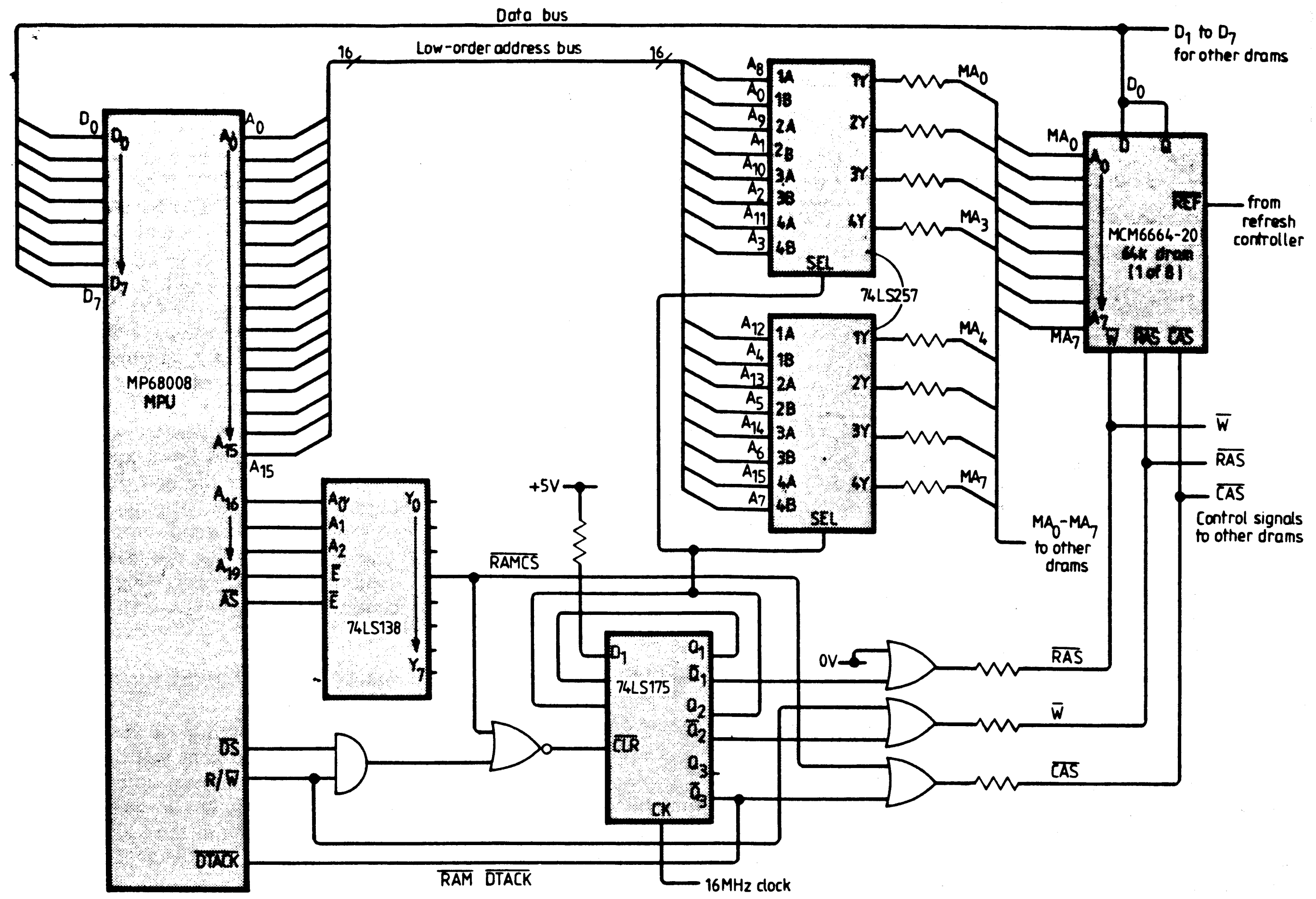
A practical minimum 68008-based system could employ one 8kbyte rom to contain program code and reset vectors. Fig. 3 shows the 68008 interfaced to an 8kbyte rom, MCM68764. The eight-bit data bus and the low-order address lines of the MPU are connected to the rom on a one-to-one basis. The high-order address lines are used to allocate different portions of the one megabyte address space to various memory and peripheral devices in the system. A 74LS138 1-of-8 demultiplexer is driven by these lines to generate a chip-select signal for each device. The demultiplexer is enabled by the 68008 address strobe signal (AS) which indicates when a valid address is on the address bus.

During a read operation on the rom the m.f.u. sets up the read/write signal (R/W) and places the appropriate address on the address bus. Signal AS is activated by the processor enabling the demultiplexer to chip-select the rom. As part of the asynchronous bus cycle, hardware must activate data-transfer-acknowledge (DTACK), to indicate to the processor that data from the rom will shortly be valid on the data bus. A 74LS175 quad D-type flip-flop acts as the DTACK generator for the rom. The rom chip-select signal EROM/CS releases the flip-flops from their cleared state which allows a logic zero to propagate from one $\overline{Q}$

Fig. 3. 68008 interfaced to rom. Asynchronous data bus enables the bus cycle time to be ne-tuned to the speed of the currently accessed memory or peripheral.



Reprinted with permission 5-22-85



output to the next on successive rising edges of the 8MHz clock. The fourth $\overline{Q}$ output generates the active low DTACK which signals the 68008 to read the data on the data bus and to terminate the bus cycle. The DTACK delay time is governed by the number of flip-flops and the frequency of the clock, and is chosen to suit the rom access time (approximately 450ns in this case).

In this example the R/W signal and all of the high-order address lines have been included in the rom chip-select logic. This enables the detection of illegal operations such as writes-to-rom and access to some unused memory space. In a price-sensitive product this may be considered a luxury and the invertor and the five-input or-gate could be omitted.

68008 with ram

Interfacing the 68008 to static ram is similar to the rom case except that the read/write signal connects directly to the ram or rams and is not included in the chip-select logic. Dynamic ram interfacing is more complex because they require refreshing and address multiplexing logic. Fig. 6 shows the 68008 connected to eight MCM6664 64kbit dynamic rams providing 64kbytes of read/write memory. The D input and Q output of each dram are connected to a different 68008 data line – only one dram is shown in Fig. 4 for clarity. The 16 low-order address lines are multiplexed together using two 74LS257 quad two-input multiplexers to form the row address and column address needed by the drams.

Like the rom circuit a 74LS175 quad D-type flip-flop is used as DTACK generator. Where the ram and rom, or any other device used, have the same data access time a single generator may be used for these devices. If a single generator is shared among several devices having different access times much of the benefits of an asynchronous bus are lost since the processor executes an unnecessarily long bus cycle for the faster devices. However, in a very price-conscious design where minimum device count is paramount, this may be an acceptable engineering compromise. As well as generating the ram DTACK the 74LS175 also provides the dram control signals, row address strobe (RAS), column address strobe (CAS), write (W), and the switching signal SEL for the multiplexers. AS, R/W and data

Fig. 4. 68008 interfaced to ram. High-density rams require the m.p.u. address to be multiplexed into a column and row address

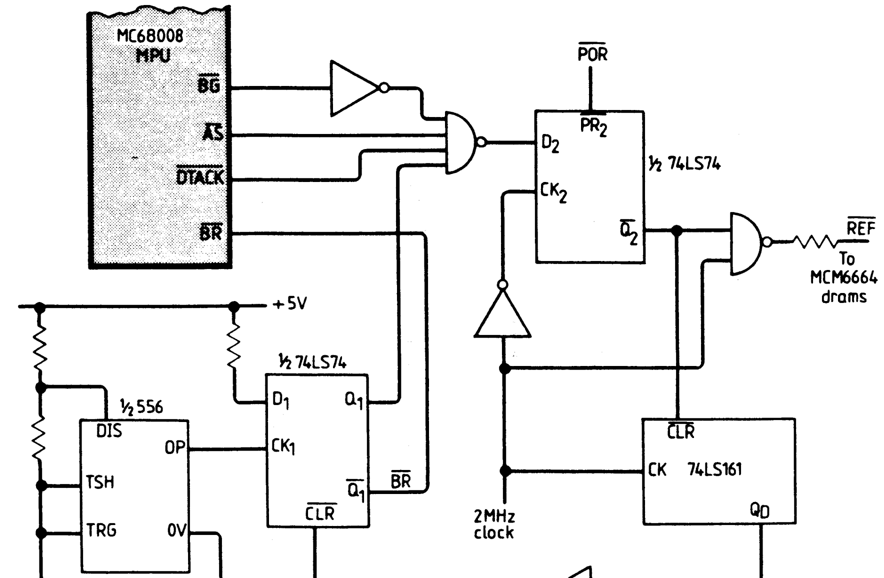


Fig. 5. 68008 dynamic ram refresh controller. The MCM6664 64K drams use a simple pin-1 refresh technique. Here the m.p.u. operation is temporarily suspended while a block of eight internal memory rows are refreshed.

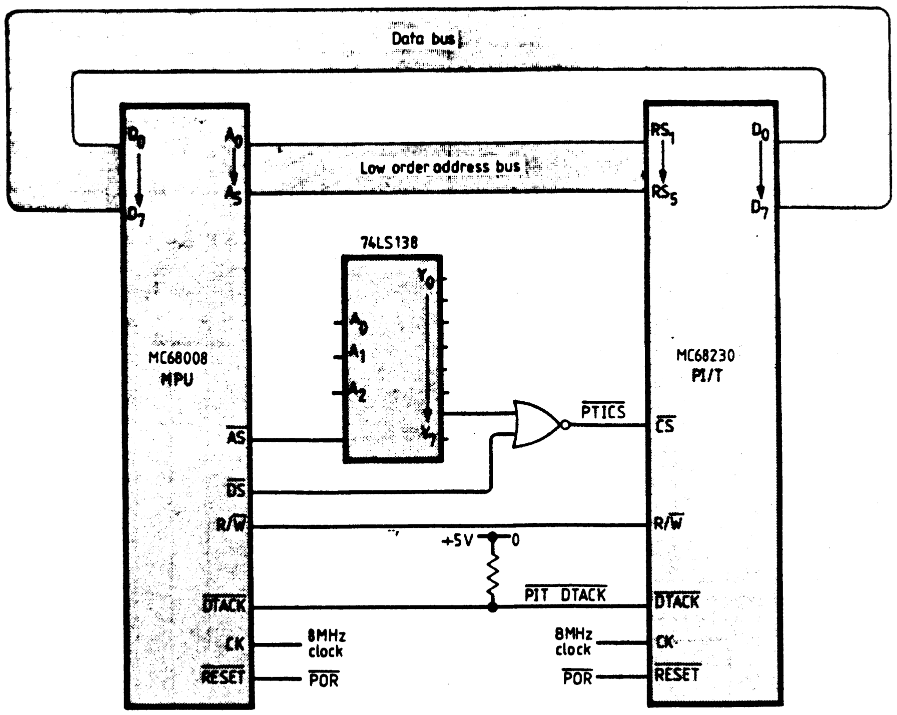


Fig. 6. 68008 with M68000 type peripherals have on-chip DTACK generators which provide the return handshake signal for the asynchronous data bus.

strobe (DS) are used to ensure the DTACK waveform is generated correctly for the three types of 68008 bus cycles – read, write and read-modify-write.

Refresh requirements for the MCM6664 drams are modest, which makes them ideal for a low-cost 68008-based system. The refresh row address counter is on-chip and refresh may be performed by providing a pulse on pin 1 (REF) to refresh the next row of memory elements. A dynamic ram refresh controller is shown in Fig. 5. The technique used is to temporarily suspend processing by requesting the 68008 bus. Eight successive rows within the drams are refreshed before allowing the 68008 to continue. Block refreshing in this way affects the longest possible interrupt latency time, the time between an interrupt occurring and the mpu entering its interrupt handler. A block refresh of eight rows is acceptable for most applications.

The 68008 bus arbitration, controlled by bus request (BR) and grant (BG), is used to suspend processing. When BR is activated by external logic the 68008 makes BG active as soon as possible, usually directly after internal synchronization. Once BR is removed BG is deactivated and the 68008 resumes processing. In Fig. 5 the multivibrator (1/2 MC3456) produces a clock REFCLK at the required refresh rate (about 9kHz). The rising edge of signal REFCLK triggers the first D-type flip-flop (1/2 74LS74) which generates the bus request to the 68008. When bus grant is received AS and DTACK must be monitored to determine the end of the current bus cycle.

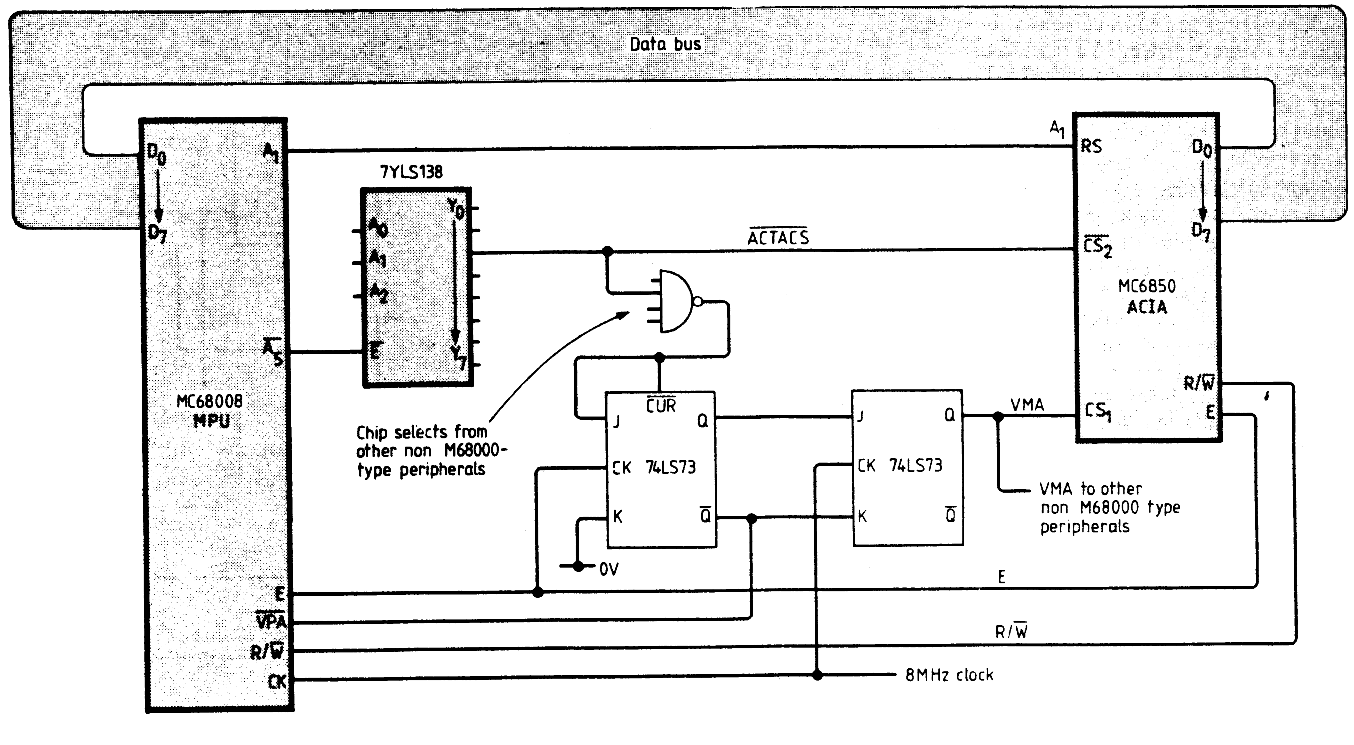
This condition, detected by the four-input nand-gate, is clocked through the second D-type flip-flop. The $\overline{Q_2}$ output goes high enabling a stream of negative-going pulses to appear on the REF inputs of the drams. And simultaneously, the 74LS161 counter is allowed to count. After eight refresh pulses have been generated the counter clears the first flip-flop. This in turn removes the bus request, and disables the REF pulses and stops the counter. The 68008 removes the bus grant and continues normal processing. This state continues until the next rising edge of REFCLK.

68008 with M68000-type peripheral

Many M68000 peripheral devices have 8-bit data buses and some may be configured for eight or 16-bit buses. In these cases the data bus lines D_0-D_7 are connected on a one-to-one basis with the 68008 data bus. For the few peripherals with only 16-bit buses, a 16-bit interface can be constructed using two octal transceivers and a couple of gates.

Fig. 6 shows a typical interconnection between the 68008 and the 68230 parallel interface/timer (PI/T). The PI/T register select inputs (RS_n) are controlled by the low-order address lines A_1-A_5. Note that address line A_0 is not used in this example. While it is quite acceptable to use A_0, making it a "don't care" places the byte-sized internal registers of the PI/T on 16- rather than eight-bit boundaries in the memory map. This preserves complete software compatibility with programs

Fig. 7. 68008 with non-M68000 type peripherals. Special m.p.u. control lines allow straight forward interfacing to these peripherals with minimal hardware. The m.p.u. performs a synchronous bus cycle when accessing such devices.



Data bus
7YLS138
ACTACS
MC6850 ACIA
MC68008 MPU
Chip selects from other non M68000-type peripherals
CUR
74LS73
74LS73
VMA
VMA to other non M68000 type peripherals
0V
E
R/W
8MHz clock

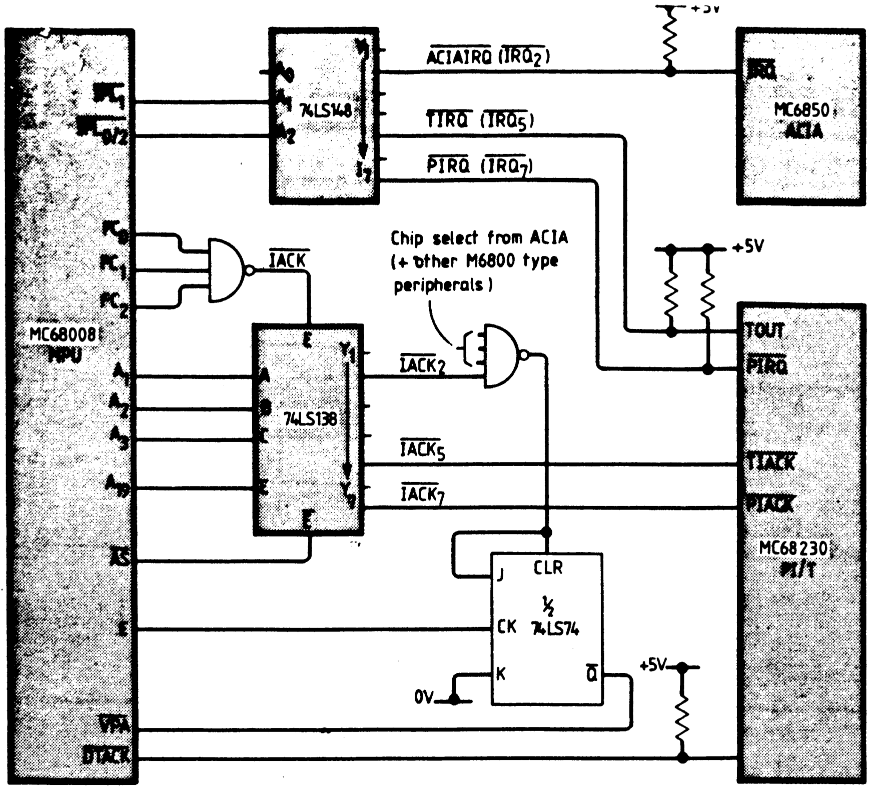


Fig. 8. 68008 interrupt interface logic. M68000-type peripheral (68230) uses the vectored interrupt scheme where it provides the vector number (0-255) on the data bus. Non-68000 type peripheral 6850 uses the auto-vector scheme.

written for the 68000 m.p.u. (which does not have an external A_0 signal).

M68000-type peripherals have an on-chip DTACK generator so the DTACK pin is connected directly to the 68008 DTACK, and may be or-wired with the DTACK signal from other peripherals.

The interfacing of interrupt signals is described later.

68008 with non-M68000-type peripheral

Non-M68000-type peripherals do not have on-chip DTACK generators and are usually accessed synchronously using the M6800-peripheral signals enable (E) and valid peripheral address (VPA). Fig. 7 shows the interconnection of the 68008 with a 6850 a.c.i.a. Each time such a device is accessed during a m.p.u. bus cycle, the 68008 VPA input is used for the returning handshake instead of DTACK. VPA signals the 68008 to perform the current bus cycle synchronously. Data transfer is made on the falling edge of the E clock which the 68008 provides for the peripheral. A J-K flip-flop (1/2 74LS73) generates VPA whenever any M6800-type peripheral in the system, such as the a.c.i.a. is chip-selected. A second flip-flop provides a valid memory address (VMA) used by many of these devices.

68008 interrupt logic

Fig. 8 shows the interrupt connections for a typical small system. A M68000-type peripheral (68230 PI/T) uses interrupt priority level 7 for its parallel interface and level 5 for its timer. A M6800-type peripheral (6850 ACIA) uses level 2. The three interrupt request signals (PIRQ, TIRQ and ACIAIRQ) are priority encoded by the 74LS148 priority encoder which generates a three-bit binary number corresponding to the interrupt level. Two bits of the three-bit number are presented to the 68008 via its interrupt priority level inputs IPL1 and IPL0/2. These input pins indicate the encoded priority level of the peripheral requesting an interrupt. The 68000 uses three pins to encode a range of 0-7 but due to pin limitations only two pins are available on the 68008. By connecting the IPL0/2pin to both the IPL0 and IPL2 inputs internally the 68008 encodes values of 0, 2, 5 and 7. Level zero is used to indicate that there is no interrupt depending and level seven is a non-maskable edge-triggered interrupt. Except for level seven, the requesting level must be greater than the level contained in the processor status register before the 68008 will acknowledge the request.

Interrupt acknowledgement (IACK) is indicated by the processor function code outputs FC_0, $FC_1$and FC_2 all logic high. The address lines A_1, A_2 and A_3 contain a three-bit binary number corresponding to the interrupt priority level being acknowledged. In Fig. 8 a 74LS138 1-of-8 demultiplexer decodes the individual IACK for each peripheral device ($IACK_n$). For M68000-type peripherals $IACK_n$ is connected to an input pin on the peripheral for this purpose (e.g. PIACK or TIACK on the 68230 PI/T). When this input is activated the peripheral places the vector number (binary 0-255) on the data bus and then it activates the DTACK signal. The 68008 reads the vector number and uses it to nd the start address of the required interrupt handler from its interrupt vector table. Non-M68000 peripherals, not capable of this vectored interrupt method, need hardware to provide the v.p.a. signal. Asserting VPA during IACK signals the m.p.u. to use one of the autovectors to find the appropriate interrupt handler. The nand-gate J-K flip-flop used as VPA generator here is the same as that in Fig. 7

Other circuit elements

No special multi-phase clocks are required by the 68008. An 8MHz t.t.l. clock generated from a crystal and several inverters is sufficient. A double-frequency master clock is useful for producing certain timings such as those for controlling dynamic memories. A watch-dog timer should be used in asynchronous microprocessor systems to detect attempts by software to access non-existant memory. It may be constructed in similar fashion to the DTACK generators shown but connected to the 68008 bus error input (BERR). Its time-out period should be longer than the slowest memory or peripheral device in the system. Power-on reset logic could comprise a 555 timer and several invertors.

Microcomputers as 68008 peripherals

Low-cost single-chip microcomputers, like the MC6805 family, may readily be cus-

continued on page 41

Motorola peripheral devices

6821	PIA	Peripheral interface adapter
6822	IIA	Industrial interface adapter
6829	MMU	Memory management unit
6835	CRTC	Mask programmed c.r.t. controller
6840	PTM	Programmable timer module
6844	DMAC	Direct memory access controller
6845	CRTC	CRT controller
6847	VDG	Video display generator
6850	ACIA	Asynchronous communications interface adapter
6852	SSDA	Synchronous serial data adapter
6854	ADLC	Advanced data link controller
68120	IPC	Intelligent peripheral controller
68153	BIM	Buss interrupter module
68230	PIT	Parallel interface and timer
68440	DDMA	Dual direct memory access controller
68450	DMAC	Direct memory access controller
68451	MMU	Memory management unit
68452	BAM	Bus arbitration module
68454	IMDC	Intelligent multiple disc controller
68459	DPLL	Disc phase-locked loop
68488	GPIA	General-purpose interface adapter
68561	M P C CII	Multi-protocol communications controller.II
68562	DUSC C	Dual universal serial communications controller
68564	SIO	Serial input output
68590	LANCE	Local area network controller for Ethernet
68652	MPCC	Multi-protocol communications controller
68653	PGC	Polynomial generator and checker
68661	EPCI	Enhanced programmable communications interface
68681	DUAR T	Dual universal asynchronous receiver transmitter
68802	LNET	Local area network controller
68901	MFP	Multi-function peripheral
14442	A to D	Microprocessor compatible a-to-d convertor
146818	RTC	Real time clock + ram

Designing with 68008

continued from page 33

tomized as intelligent peripherals having a 68008-type interface; the interface being emulated by m.c.u. software. Some of the m.c.u. i/o ports are used to provide the peripheral data bus D_0-D_7, chip-select (CS), read/write (R/W) DTACK, IACK, and register select (RS_n) lines. Port lines programmed as inputs (CS, R/W and RS_n) are monitored by the on-chip software to determine 68008 accesses. And with the port lines used for D_0-D_7, DIACK and IACK, the m.c.u. may be programmed by the 68008, and provide vectors during interrupt acknowledgement cycles. To the 68008 the m.c.u. behaves just like any other M68000 family peripheral. WW