Electronic components & applications

Vol. 6, No
198

ISSN 1041 – 6219

Electronic components & applications

Volume 6, No. 3 1984

The modern industrial world is seeing a revolution every bit as significant as the industrial revolution that started the whole thing off. A revolution in methods and philosophies. A revolution that sees factories functioning increasingly without need of human intervention. That sees electronic intelligence taking over most of the repetitive functions, and much of the decision making so essential in a modern industrial plant. A prime example of this revolution can be found in the assembly plant for the new BMW model 300. To handle the growing demand for customized cars, BMW have installed Philips' award-winning Programmable Remote Identification system. With this system, specially programmed 'recognition' modules carried by the vehicles, direct the production line to include special transmissions, interiors etc, allowing the production of customized cars on the normal assembly line.

Contents

An architectural contrast - 68000 versus iAPX 286

B. VERNOOIJ

In a previous article (see Ref.), we discussed the design philosophy of the 68000 microprocessor. That article showed that the surest route to true 16-bit capability was the development of a totally new architecture, one unhindered by the preconceptions of earlier 8-bit system designers. By freeing themselves of the restrictions imposed by 8-bit architectures, the designers of the 68000 were able to produce a microprocessor more powerful than any of its contemporaries, and one with tremendous scope for future expansion.

This article provides further evidence of the power and versatility of the 68000 by comparing it with another of today's advanced 16-bit architectures: the iAPX 286. When comparing microprocessor architectures, attributes that must be considered include:

- register set type and size
- virtual memory support
- data types and operations
- instruction set
- family compatibility
- memory management.

THE REGISTER SET

Registers were first introduced into microprocessors to help speed up operation. It follows that since registers are used frequently, they must be accessed quickly and easily to enable straightforward programming and fast program execution. If instructions are tied to specific registers they may be easier to encode (no difficulties of on-chip implementation) but this approach confines the programmer to rigid planning of register use.

For optimal use, the registers must be available to any instruction almost without exception. A truly general-purpose register set allows just this, any instruction can use any register as an operand or as an address to an operand, in combination with any data.

The 68000 is the foundation for one of the most sophisticated personal computers — Apple's Macintosh.

Each microprocessor member of the 68000 family has sixteen 32-bit general-purpose registers, divided equally between address and data register types. The 68000 instruction set includes no instruction that requires the use of a specific register, so the programmer is free to choose which of the eight data or address registers to use, thus providing maximum flexibility.

The restrictions imposed by a dedicated register set can be shown by considering the iAPX 286 and its predecessors. Multiply and divide operations, for example, are tied to the AX register in the architecture of these microprocessors. If an instruction needs to use the AX register, the register contents may have to be saved before the new value is inserted. Even if the new value is in an adjacent register like CX, it still needs to be moved to the AX register. Following instruction execution, the result will probably have to be saved to allow the AX register to be used for other operations, only to be re-entered a few instructions later for further processing.

This overhead of constantly swapping register contents, necessary with dedicated registers, restricts the programmer and slows down processing. In addition, unsigned multiply and both signed and unsigned divide don't offer an immediate addressing mode for the source operand, so even more time is lost loading another register just to hold the source.

The register set of the iAPX 286, shown in Fig.1, has four 16-bit data registers (doubling as eight 8-bit registers), each assigned special functions, as indicated. There are four 16-bit registers used as index and stack pointers and each of these have specific properties for use by certain instructions. For example the stack pointer is the only register that can be automatically incremented and decremented when data is accessed from it. Moreover, frequent operations require a specific data register to be tied to one of the pointer registers.

To increase the addressing range of the iAPX 286 within its segmented architecture, four segment registers are provided: one for the 64 Kbyte code, data and stack segments and one for an extra segment. Once again, on many occasions these registers are tied to specific pointer registers by the architecture. With the iAPX 286, the programmer must

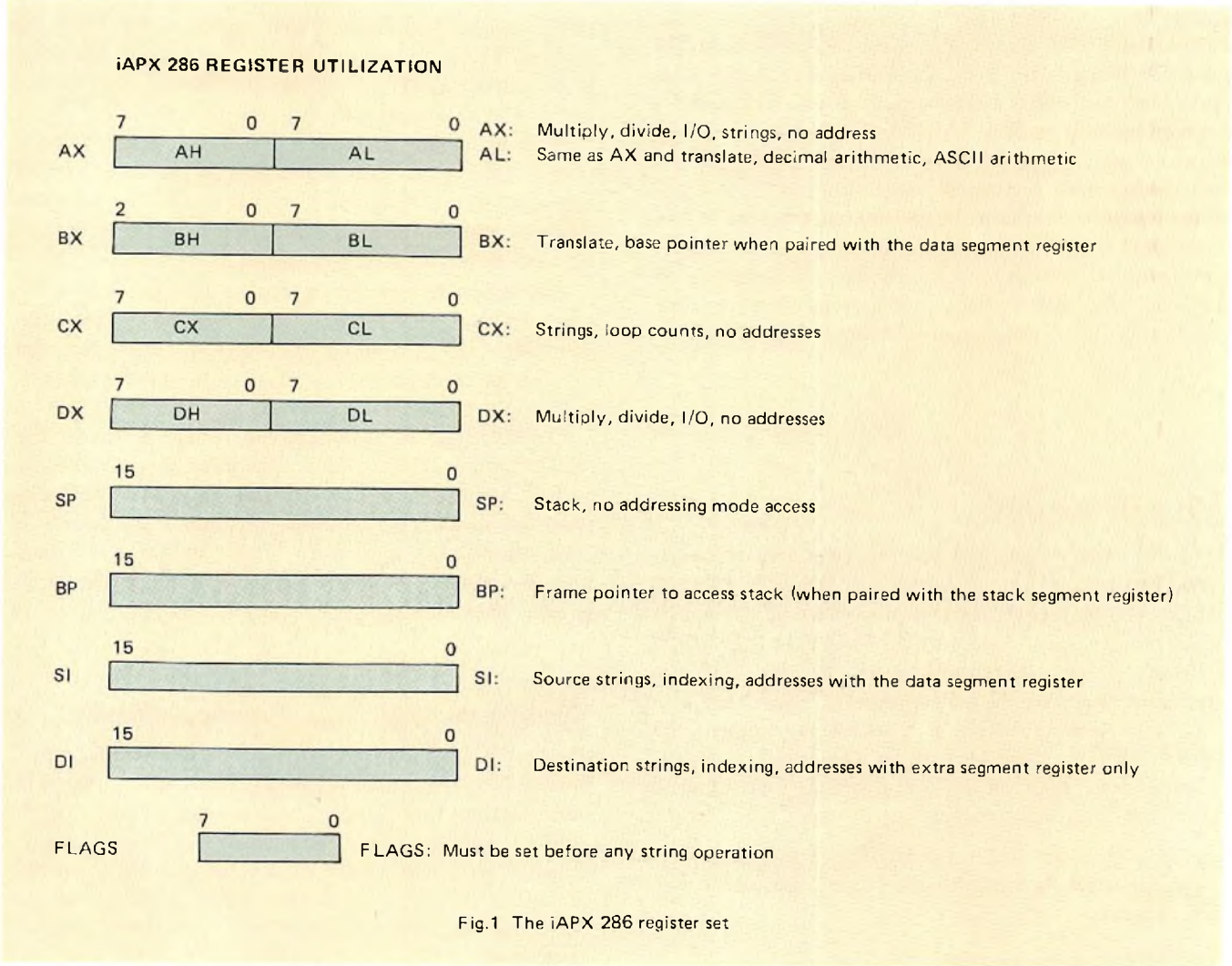


Fig.1 The iAPX 286 register set

ensure that data segment registers are matched with the correct offset, and this must be achieved with the four registers (BX, BP, SI and DI) that are permitted to hold the 16-bit offset values. However, the architecture places additional special demands upon these registers.

Register BX is used repeatedly in arithmetic and logic operations, and its content is destroyed each time the translate instruction is used. Except for a few specific operations the BP register functions permanently as a frame pointer and therefore is usually unavailable to most programs. The SI and DI registers are available to the programmer, but unfortunately they are overwritten every time a string operation is performed. Hence frequent loading and unloading of offset values is required coupled with set-up and overriding of the proper segment registers, limiting the performance of the iAPX 286/8086 severely.

Contrast all the restrictions of the dedicated registers of the iAPX 286/8086 architecture with the freedom provided by the general-purpose register set of the 68000. With the iAPX 286/8086 architecture there isn't even a single register available for the programmer to use at will, whereas the 68000 allows unrestricted use of the complete 32-bit register set.

Examination of almost any iAPX 286/8086 code will reveal the limitations that result from an architecture that dates back to the first 8-bit microprocessors. For example, consider the independent EDN Benchmark E* of a primitive string search. For this the 68000 benchmark requires a mere quarter of its register set whereas the iAPX 286 architecture has to use its entire register set and even has to save and restore two of the working variables of this short benchmark. If mundane programs like this cause such problems, what performance problems and lengthy code would result from a program of even modest complexity?

VIRTUAL MEMORY

For today's ever expanding microprocessor applications, it is vital that a processor can support virtual memory. Virtual memory allows the execution of processes that may not be completely located in memory. The main advantage of this technique is that user programs can be larger than the physical memory of the microprocessor. Selecting a processor that cannot operate in a virtual environment will restrict possibilities for future system expansion.

A processor that can operate in a virtual environment provides the following advantages:

- a program is not tied to the physical processor memory but can act as though it has as much contiguous memory as required
- the programmer need not be aware of the virtual memory capability and is thus not concerned with the microprocessor's memory protection mechanism
- the operating system can recover completely from memory access faults and continue with the program (fault tolerance) and
- demand paging is possible.

For a program to operate on the iAPX 286 as if it had as much contiguous memory as it required, some method of circumventing the crippling 64 Kbyte segmented architecture is necessary. However, the 16-bit offset limit prevents any practical solution and programmers who attempt to overcome the problem with software are punished by the massive increase in the amount of code necessary. Furthermore, execution time is increased by an order of magnitude due to the large overhead from the required descriptor loading.

What's more, the iAPX 286 doesn't have the second advantages because it requires that the programmer describe each segment to memory management unit (MMU) before any reference to that segment can be made. And, since there are only two registers that can be used to store the segment descriptors (and these are required for other specific operations), considerable time is spent just loading and unloading descriptors.

In programming the 68010/68020, no prior knowledge of the target system is required. The program will execute correctly if the target is a virtual machine, has virtual dynamic paging or real segments, or even if the system has no memory management whatsoever.

EDN's Sort Benchmark emphasises the contrast between the restrictive iAPX 286 and 68000 with no segmentation and MMU descriptor problems. The iAPX 286 takes longer to load its segment descriptors than the 68000 takes to perform the complete benchmark.

Unfortunately it is almost impossible to make the iAPX 286 fault tolerant since its architecture provides only virtual segments and not a virtual memory. Descriptor faults can be detected but memory access faults cannot. If a bus error occurs, there is no alternative but to terminate the affected program even if it is the operating system, thus bringing the system down.

Virtual memory facilities of the 68000 family

The 68010 and 68020 microprocessors allow an interrupted bus cycle to be rerun or simulated. The simulation provides many features that permit virtual accesses to be granted to tasks or programs. With National's NS32032 providing instruction retry only, and the iAPX 268 offering no support for virtual memory, the 68000 family is unique in providing virtual machine, virtual I/O and virtual window capabilities.

* EDN April and September 1983.

Virtual I/O means that the operating system may provide a set of virtual peripherals for its application programs or even for another operating system. For a virtual machine, virtual I/O is imperative, since the entire physical hardware may have to be simulated. With virtual machine capabilities, any 68000 system can emulate any other.

Virtual windows are similar to virtual I/O, creating a virtual memory block instead of peripherals. This memory block belongs to another task or another system entirely. For example, a simple parameter area can be arranged between system tasks so that a write into the area will 'wake up' all other tasks. Creating these event or semaphore windows allows operating-system primitives to provide greater flexibility.

Demand paging

Demand paging is the most common virtual memory system. Paging divides the memory into equal-sized blocks called page frames, divisions invisible to the operator and to the application programs running on the system. It is desirable to have these divisions because an entire program need not exist in physical memory. Only the pages in current use have to be given page frames. Demand paging means that the program does not have to specify beforehand which pages of its local address space will be required since the address is interpreted by the operating system to provide the correct page as it is required.

The segmented memory approach of the iAPX 286 cannot support demand paging. Programs must inform the MMU in advance which segments will be required. Furthermore, the MMU does not provide dynamic paged memory management – burdening the programmer even more.

Demand paging lets the programmer select the most appropriate page size. In the iAPX 286 a complete segment of 64 Kbytes must be made resident even if only a few bytes are required. Therefore, time and memory space are lost because a complete segment must be loaded although most of it may never be accessed. Moreover the iAPX 286 can suffer from excessive overhead in loading segment registers which results from having too many very small segments. This is caused by the module concept of the MMU when there are a large number of small routines.

For example, high-level language run-time libraries are usually made up of many small interdependent modules and when one module is called it requires a base register to be allocated one or more segments by the iAPX 286 MMU. Since segment allocation takes from 2 to 4 μs depending on memory speed, this quickly mounts up to a significant part of execution time. Even if all the routines reside in the memory, the segment load and checking overhead is still the same. With demand paging, however, the 68000 has no overhead in accessing resident pages since no MMU set-up instructions are required.

DATA TYPES AND OPERATIONS

Flexibility of microprocessors can be gauged by the variety of data types they can operate on. Comparing the data types supported by the 68000 and the iAPX 286, given in Fig.2, it is evident that the 68000 is more flexible. Many native instructions of the 68000 can only be matched by the iAPX 286 with multiple instruction sequences and loops.

Data Type	68000	iAPX 286
Bits	X	none
Bytes	X	X
Boolean Bytes	X	none
Strings	X	X*
ASCII Arithmetic	none	X*
Word Integer	X	X
Longword Integer	X	DIV/MUL only*
8-Bit Binary Multiple Precision	X	register only
16-Bit Binary Multiple Precision	X	register only
32-Bit Binary Multiple Precision	X	none
BCD Byte	X	only with register and adjusts*
BCD String	X	none
Translate	X	Register Only*
16-Bit Blocks	X	X
32-Bit Blocks	X	none
16-Bit Pointer	X	none
32-Bit Pointer	X	X
Word Logical	X	none
Longword Logical	X	none

* Each of these data types requires exclusive use of the AX register.

Fig.2 Data types supported directly by instructions

As microprocessors have developed, so data sizes have grown steadily from 4 to 16 bits. And now the 16-bit is giving way to the 32-bit processor. Today's advanced microprocessors should therefore be able to operate on data up to 32 bits, and for systems looking to extend in the future, this is a must. Hence, the 68000 was designed to handle arithmetic and logic operations on 8, 16 and 32-bit data. Compare this with the iAPX 286 that can handle only multiplication and division on 32 bits.

Bit manipulation

Single bit manipulation is very important in I/O operations, flag control, resource allocation and many other applications. In early 8-bit microprocessors, logic operations (AND, OR and exclusive-OR) were used to modify accessable bits. The iAPX 286, like its predecessors, can only test bits using the AND instruction, after first loading the specific

bit into a dedicated register. Conditional branches are usually defined by testing bit flags in memory. This is especially true for operating systems, and in the iAPX 286, operating-system routines are slowed considerably because there are no bit test instructions.

The 68000, however, provides straightforward bit manipulation with four instructions that allow testing, or testing and setting, clearing or changing of bits in any data register or memory or I/O location. The iAPX 286 would need to run a rather complex instruction sequence to duplicate the 68000's BCLR (bit test and clear) instruction.

Registers and addressing modes

Figure 3 gives the various combinations of addressing data, illustrating the difference in programming each processor. The 68000, with its additional addressing modes and registers, has a distinct advantage over the iAPX 286. 68000 family microprocessors are more flexible – about 6 times as many variations in accessing data. The 68020 is even more flexible with 100 times the accessing options of the iAPX 286.

It is important for programmers to have available as many addressing modes as possible. With more means of expressing a location the greater is the likelyhood of finding the best option for each set of circumstances. Compare the addressing modes for the 68000 and iAPX 286 given in Fig.4. Both architectures allow direct access to the on-chip registers and have registers to point to addresses in memory. Also, each microprocessor can use constants via immediate addressing. Additionally, the 68000 has a pair of extremely useful addressing modes that have been omitted on not only the iAPX 286 but on Zilog and National chips – postincrement

Mode	No. of Register Options	
	68000 Family	iAPX 286*
Data register	8	4**
Address register	8	4
Address register deferred	8	4
Address deferred postincrement	8	unavailable
Address deferred predecrement	8	unavailable
Address deferred with offset	8	4
Index deferred with offset	128	4
Absolute short	N/A	N/A
Absolute long	N/A	unavailable
Relative with offset	N/A	unavailable
Relative Index with offset	16	unavailable
Immediate	N/A	N/A

* Each access typically requires an implied segment register as well.
** 4 more, if using 8-bit only. N/A Not Applicable

Fig.3 Addressing combinations for accessing data

68000 Family		iAPX 286	
000 rrr	Dn	reg	mod = 11
001 rrr	An	"	"
010 rrr	(An)	(SI/DI/BP/BX)	mod = 00
011 rrr	(An), An = An + incr	- no auto increment -	
100 rrr	(An), An = An – incr	- no auto decrement -	
	see next	(SI/DI/BP/BX) + dB	mod = 01
101 rrr	(An) + d16	(SI/DI/BP/BX) – d16	mod = 10
110 rrr	(An) + (Xn) + d8	(BX/BP) + (SI/DI) + d8	mod = 01
		(BX/BP) + (SI/DI) + d16	mod = 10
111 000	absolute 16	absolute 16 only	mod = 00 & r/m = 110
111 001	absolute 32	- no 24-bit absolute !!	
111 010	(PC) + d16	- no IP relative -	
111 011	(PC) + (Xn) + d8	- no IP relative-	
111 100	#Immediate	#Immediate	special
Conditional Program Transfers			
	8-bit displacement	8-bit displacement	
	16-bit displacement	- no long jumps -	

Fig.4 Comparison of addressing modes

and predecrement. Using these two modes the programmer can easily:

- scan through a data string
- push data on to a user-defined stack
- pull data from a user-defined stack
- push data on to a user-defined queue
- pull data from a user-defined queue.

These instructions are available to almost any instruction using any of the address registers and so eliminate specialized stack instructions that work on a dedicated stack. The instructions allow the 68000 user to define as many stacks as required, with 8 stacks directly available.

The only stack instructions available on the iAPX 286 are POP and PUSH and these are limited to one specific stack. They place data on or remove it from the top of the stack but are restricted to 16-bit values. Moreover, there is no way around the single stack limitation and no other instruction can operate on the stack without manipulation.

A further stack problem with the iAPX 286 is that to access data from within the stack, a frame pointer must be defined. This takes up the base register that is most easily accessed, adding to an already congested register set.

Indexed and direct addressing

Both architectures provide indexed addressing that accesses a location defined by the sum of one or two registers and an offset. In addition to the 8 and 16-bit offsets that are also provided by the iAPX 286, the 68000 can index 32-bit values. The iAPX 286 can only do this as two 16-bit addresses; thus the 68000 is an order of magnitude faster. A further advantage of the 68000's indexed addressing is its flexible instructions that allow any appropriate register to be used. Compared with the iAPX 286, which allows only 4 register options with each instruction, there is significantly less chance of register congestion using the 68000.

Because of its segmented architecture, the iAPX 286 does not allow direct addressing but requires that each address has a selector (stored in one of two segment registers) and an offset. In many operations, the code saving that this is supposed to give is outweighed by the extra instruction code required to handle the segments. The iAPX 286 architecture also uses segment registers for other dedicated purposes, adding to the overhead by repeatedly having to swap register contents in and out of memory. Segment overhead is increased even further each time a segment register is loaded because a complete segment check is necessary.

Program counter relative addressing

Writing position-independent object code for the 68000 is eased greatly by the two program counter relative addressing modes. These modes allow data to be placed relative to the location of the instruction that accesses the data. So, if the instruction is in ROM #17/18 and the data is in ROM #21/22 in one system then those ROMs can be moved to another system and will execute identically as long as they remain four chips apart in the instruction ROMs.

The iAPX 286, in contrast, has no program counter relative instructions and prevents shifting ROMs from one system to another since the MMU requires all programs to have segment IDs. Segment IDs only have meaning to a specific operating system running on a particular hardware system. This shows up another drawback of the segmented iAPX 286 architecture, and like many others, results from the programmer having to deal with the MMU system directly.

The 32-bit linear address space of the 68000 can be accessed directly, avoiding the difficulties that result from segmentation. Overall, 68000 family microprocessors have more flexible and useful addressing modes than the iAPX 286, making accessing data simple, convenient and functional.

INSTRUCTION SET

One of the most important attributes of a microprocessor is its instruction set. This provides a good idea of the functional flexibility of a processor. The more functional the instructions, the more flexible the microprocessor. An important part of the instruction set is formed by the data types and operations discussed in the previous section.

68000 processors were designed to use an entirely new instruction set. No previous microprocessor instruction set was used as a mold. However, years of experience in 8-bit microprocessor design, together with the experience of minicomputer designers and users, has contributed significantly to the concepts involved in the 68000's instruction set. Speed was one of the main requirements of the 68000 and the instruction set was designed with this in mind. Execution speed is linked to the frequency of instruction use, frequently used instructions, such as register data movement, being designed to execute the quickest.

Powerful and yet flexible general-purpose instructions are included for data movement, BCD arithmetic, logic functions, integer arithmetic, shift and rotate operations, program control, and high-level language and operating-system support. Special-purpose instructions, appealing only to a small group of users, were avoided because these limit the functional flexibility of the processor.

The instruction set of the iAPX 286 is basically that of its predecessor the 8086 with two major additions. Instructions were included in an attempt to duplicate some of the 68000's features, and control instructions for the MMU were also added. The iAPX 286 retains all 8086 instructions, together with their shortcomings. Stack operations discussed in the previous section exemplify where the iAPX 286 instruction set falls short.

String manipulation

Complex string operations are handled with ease by the 68000 using its auto increment and auto decrement instructions. The DBcc (decrement and branch on condition) instruction is a key looping instruction. When placed after one or a series of instructions, DBcc will repeat the series until a specified condition (cc) is met or until the counter overflows.

If DBcc is placed after a MOVE instruction, many simple string operations can be formed. DBcc, in combination with a few other instructions, can be used to build many flexible and extremely powerful string and other repeated functions. Without the 68000's auto increment and decrement instructions these functions would not be possible.

With the 68010, a special routine allows some repeated functions to execute even faster. When the processor fetches most single-word op codes followed by the DBcc instruction that immediately branches back to the op code, the processor stores both instructions internally and then executes the loop without further re-fetch of op codes. The routine speeds up these loops by as much as 70%.

The iAPX 286, again like its predecessor, provides a few dedicated simple string instructions for block moves and fills, string comparisons and scans, and string input and output operations. However, as these instructions use dedicated registers, their flexibility is limited. If a programmer requires more complex, yet common, string operations than normal, relatively slow instructions must be used with a looping control (LOOPxx) instruction.

Branches and program jumps

Conditional branching enables programs to make decisions based on a set of conditions. From these decisions, one of a number of actions can ensue. Branches and program jumps are fundamental concepts in programming. The ability of a processor to make decisions, and the ease by which it takes alternative action determines the processor's functional flexibility.

The 68000 processors provide both conditional branching and program jumps using any addressing mode to define the destination. Branches direct the program to the next routine relative to the current instruction and can be specified up to ±32 Kbytes. The large displacement range allows highly flexible programming. The 68020's branch destinations can be specified over the whole 32-bit memory addressing range.

Program control operations in the iAPX 286 allow conditional branches only over a ±128 byte range. This short range will cause great difficulty in even modest tasks that are greater than the branching range. Multiple branches are necessary to compensate for such a short range.

Memory-to-memory arithmetic

Memory-to-memory operations allow arithmetic-intensive programs, such as array functions, to run efficiently. Unlike the 68000, the iAPX 286 has no such operations. Another application of memory-to-memory architecture is in multiple precision calculations. Here, the 68000 places no limit on the data size. Because the iAPX 286 does not support multiple precision calculations directly, very time-consuming manipulations are required.

The iAPX 286 single accumulator

The origins of the iAPX 286 are evident in the large number of operations that require dedicated use of the AX (accumulator) register. In the past, it was quite justifiable to implement only one accumulator because it took up considerable chip area and the gating required to provide multiple accumulators was prohibitive. Today, however, with all the advances that have been made in micro-technology there is no reason for such a restriction.

Every multiply and divide must use this one AX accumulator, most don't even permit an immediate operand. In addition, there are many other functions that also require dedicated use of the accumulator. The result is an awesome bottleneck that builds up at the AX register. A further reason why the accumulator is so overloaded is that many special instructions work more efficiently with the AX register than with any other register.

In almost every instance, program code for the iAPX 286 has significantly more lines than for the 68000. This is due mainly to the dedicated register set of the iAPX 286 architecture compared with the 68000's general-purpose register set. Benchmarks such as EDN's Quicksort are a stark revelation of how much more program code the iAPX 286 requires.

FAMILY COMPATIBILITY

Extending a microprocessor architecture beyond its original design can only be justified if there is full software compatibility between the updated and original processors. Where software compatibility is not maintained, each program must be checked to determine the required modifications – a time-consuming task that may take longer than completely rewriting the software. Program modification for an updated architecture can be difficult unless the original architecture was designed with future extension in mind.

68000 family compatibility

The 68000 architecture is an original design, trading backward compatibility with previous 8-bit microprocessors for a powerful architecture designed for today's and tomorrow's programming needs. In designing the 68000, space was left to cater for future requirements. Not all operations that may be desired could be produced in a microprocessor.

However, the fundamental functions were designed to allow desirable extensions to be implemented as improving technology permitted. The instruction set was also designed to allow possible enhancement to fulfill the requirements of future programmers. Fruits of this earlier planning are now visible in the 68010 and 68020 processors and in the industry-wide acceptance of the 68000.

100% code compatibility is maintained between the 68000 and the 8-bit derivative 68008. These microprocessors have the same instruction set which is used as the basis for later processors. All programs written for the 68000 will execute identically on the 68008, and vice-versa. To provide more capabilities than the 68000, some new instructions were added in the 68010. Two compromises that could affect supervisor-level programs had to be made to implement the additional capabilities. Apart from these two minor differences, requiring a few instructions to be changed in affected supervisor-level programs, all 68000 code will execute identically on the 68010. Further additions have been made to the 68010 instruction set for its enhanced version, the 68020 – a true 32-bit microprocessor. The 68020 also has a co-processor interface and an on-chip instruction cache.

All 68000 user (application) programs will run identically on the 68008, 68010, 68020 and all future 68000 family microprocessors. Therefore, any media containing 68000 family application code will execute exactly the same whether it is run on a 68020 or a 68000.

iAPX 286 compatibility

The iAPX 286, as a descendant of the 8086/8088, is promoted as being upward compatible with its predecessors. However, although it retains most of the registers, addressing modes and instructions of the 8086, the exact execution of some instructions is not the same and even some concepts introduced make them completely incompatible. To use these new concepts with 8086 code will require the code to be rewritten comprehensively, destroying the intent of code compatibility.

Operating modes

There are two operating modes for the iAPX 286: compatibility and native. The compatibility mode is designed to run 8086 code exactly, with a few new instructions. Software written for the 8086 could run correctly on the iAPX 286 (probably faster) with possible differences arising close to the 64 K segment boundaries.

In the native mode, the user gets all the iAPX 286's facilities. Unfortunately, most of the new features are contrary to the basis on which 8086 software is written. It is not that the program, will not run but that in so doing they will violate the protection and privilege principles of the iAPX 286.

Segment wrap

The 8086 can access a data or program word using an offset or $FFFF, getting high and low bytes from $FFFF and $0000, respectively (the segment wraps around instead of flagging an overrun fault). With the iAPX 286 this is an illegal access, causing a special trap to occur. Segment wrapping happens infrequently and the trap provides a correction mechanism.

Bus locking

Another difference between iAPX 286 and 8086 systems results from the automatic assertion of the bus lock signal upon executing the XCHG (exchange) instruction in the iAPX 286 architecture. This could cause a system fault if it were not decoded in hardware. The LOCK prefix may precede any instruction in the 8086 but it is privileged in the iAPX 286, causing a trap if executed at a lower level.

8086 application program compatibility

Programs will run unaltered when the iAPX 286 is in the compatibility mode but then the system has no advantage over the 8086. In contrast, when the iAPX 286 is in the native mode, almost every 8086 application program would require significant rewriting before it could be used.

Segment base address

For many routines, the fundamental change in segmentation between the iAPX 286 and 8086 will result is severe problems. The 8086 segment registers directly form a 20-bit base address used to access code and data, and the programmer can accurately predict the base address. With the iAPX 286, however, the segment registers indirectly specify a 24-bit base address and its value, determined by the operating system, and this base address cannot be predicted by the programmer.

Operating system compatibility

Unless the iAPX 286 is run in the compatibility mode, operating system compatibility between iAPX 286 and 8086 is completely lacking. Every 8086 operating system will have to be rewritten entirely to be used on the iAPX 286.

The new privilege levels will have to be assigned and handled by the operating system and application programs. To prevent invalid pointers being passed by lower level programs, the operating system must verify that each pointer is valid. Any instruction that accesses an interrupt vector must be changed because the vectors may only be accessed by the operating system.

These are just a few of the necessary changes to run an 8086 operating system on the iAPX 286. In general, an entirely new operating system will be necessary. Additionally, most application programs will have to be rewritten to prevent illegal accesses to the now privileged operating system and I/O operations.

MEMORY MANAGEMENT

There are many reasons why systems require memory management. Since every system will require different aspects of memory management, fixed schemes with a specific set of criteria are undesirable.

The memory management unit can be used to translate addresses. Microprocessors generate logical addresses based on internal address calculations associated with instruction processing. These logical addresses, however, may not correspond to the physical memory or I/O locations due to hardware restrictions. The memory manager is required to control these logical addresses and translate them to physical addresses.

However, translation is not the only purpose of memory management. It is also used to protect various memory spaces from illegal access by other users, whether accidental or by design. Also, should the operating system decide that it is dangerous for a low-level application program to have access to a set of resources, the operating system will prevent such accesses. Only the operating system can then access the set of resources.

The more isolation between memory management and the program and data that the processor operates on, the better. Ideally, only the highest level in the operating system kernel should know of the memory management scheme, handling the memory accordingly. However, some systems such as the iAPX 286, function according to the opposite philosophy, requiring that every program knows of the memory management. This results in increased memory management overhead and is therefore undesirable and unnecessary.

Smalltalk or graphics strings

Single-user computers, that handle data structures of several megabytes with ease, are now being introduced. Innovations, such as the Smalltalk language, need a processor that can handle huge linear memory elements to provide the powerful concepts they offer. The 64 K segments of the iAPX 286 architecture makes the processor unsuitable for such systems.

Consider the simple task of scanning a string in a Smalltalk or large graphics environment (see Fig.5). The 68000's auto-increment addressing mode gives no overhead whatsoever. The overhead is zero because the powerful 68000 processor can increment address registers by the correct data type size in parallel with the instruction execution.

Contrast this with the inordinate manipulations required on the iAPX 286, even assuming the operating system can assign consecutive segment descriptors for large contiguous data structures (no mean feat on the iAPX 286). The problems caused by such trivial operations on the iAPX 286 indicate the scale of problems that will result from more complex manipulations.

iAPX 286		68000 processors
INC BX	2 cycles	(An) + 0 cycles
JNE LBL	16 cycles	Auto-increment addressing
or		
INC BX	2 cycles	
JNE LBL	4 cycles	
MOV DS,AX	2 cycles	
INC AX	2 cycles	
MOV DS,AX	20 cycles	
LBL		
Result: 10 bytes, 18 or 30 cycles		0 bytes 0 cycles
1 extra register		no extra registers

Fig.5 String scan operation

Dynamic storage areas and sophisticated software systems

Many of the benchmarks used to compare processors don't even stretch the architecture of the simplest 8-bit microprocessor. In the real world, however, systems have to execute large and sophisticated program modules and their associated linking, and so, many weaknesses of processors are only revealed when they are installed in a working environment. The limited segment size of the iAPX 286 architecture shows several of these weaknesses. A simple example is enough to demonstrate that significant problems result.

In a sophisticated system, tens or even hundreds of modules interact and when called each module demands access to new and unique dynamic storage areas for such things as large local data structures, stacks for temporary working areas, dynamic data allocation and subroutine return linking. A dynamic area cannot be static (assigned when loading the modules) because it can be requested directly by itself, indirectly by its subroutines, or by a completely separate task. Dynamic areas are also used in recursive programming techniques – all modules in Ada, Pascal, and Lisp languages are recursive by definition.

Dynamic areas are handled with ease in all 68000 microprocessors because of the large linear address space. When a routine is called, the 68000 allocates as much dynamic memory as the routine requires on the current stack and continues processing. The LINK instruction then assigns the fixed stack requirements for local dynamic data and any of the address registers as a frame pointer. If fixed stack requirements increase, the operating system can easily extend the stack. As a result, there is no wasted space or memory management overhead. At the end of a routine, the data is simply freed with the UNLK instruction.

68000 Method

```
* ALLOCATE DYNAMIC AND LOCAL MEMORY
  LINK      #SIZE,An                    16 Cycles
    .
    .       PROGRAM BODY
    .
* FREE DYNAMIC AND LOCAL MEMORY
  UNLINK    An                          12 Cycles
```

iAPX 286 Method

```
; ALLOCATE OUR OWN STACK FOR TEMPORARIES AND SUBROUTINE CALLS
  CALL      NEWSTACK                    Hundreds to Thousands of Cycles
; ALLOCATE LOCAL DATA STRUCTURE SEGMENT
  MOV       AX,SIZE                     3 Cycles
  CALL      ALLOCSEG                    Hundreds to Thousands of Cycles
; REPEAT ABOVE FOR EACH LARGE STRUCTURE REQUIRED
    .
    .       PROGRAM BODY
    .
; FREE LARGE STRUCTURES
  MOV       BX,DESCRIPTORADDRESS        7 Cycles
  CALL      FREESEG                     Hundreds to Thousands of Cycles
; REPEAT ABOVE FOR EACH STRUCTURE OBTAINED

; FREE DYNAMIC STACK AND RETURN TO CALLER
  JMP       FREESTACK                   Hundreds to Thousands of Cycles
```

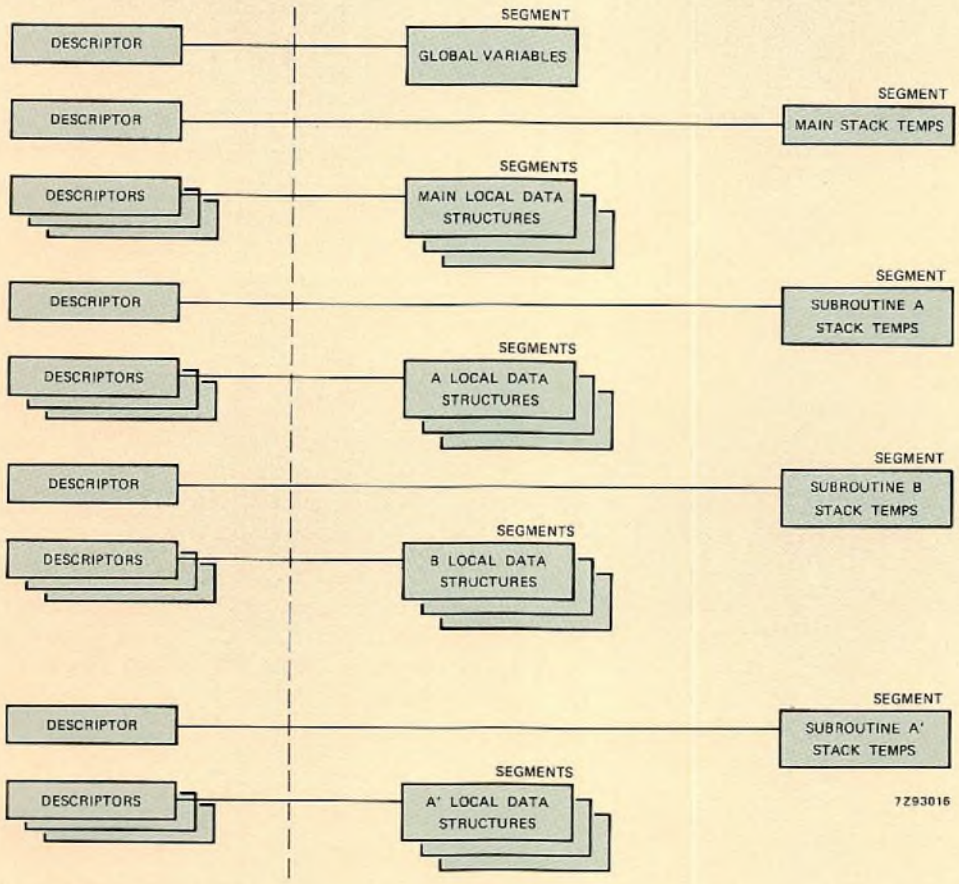


Fig.6 Dynamic storage

With the iAPX 286 there is no obvious method of handling dynamic areas. The ENTER instruction tries to extend the stack but will fail whenever the total dynamic area used by all modules exceeds 64 K. Moreover, with the limited stack size, even local dynamic data will overflow the 64 K boundary. As a result, very cumbersome routines must be used to support dynamic areas on the iAPX 286. Figure 6 gives an indication of the extra descriptors and overhead resulting from dynamic area management with the iAPX 286.

In addition, when using high-level languages, the iAPX 286 must often interface with the operating system since descriptors pointing to blocks of memory segments can only be assigned and modified by higher priority protected routines. The result of all these restrictions is that processor time is increased by several orders of magnitude even when handling only simple dynamic stack/variable allocation.

Excessive descriptor overhead in native mode

The programmer who is forced to use the iAPX 286 MMU descriptor control method will run into other problems, besides having to rewrite most of any current operating systems. One such problem is that before building a descriptor, it must first be addressed. Therefore, another temporary descriptor must be assigned for addressing to allow the other descriptor to be built. Temporary descriptors are required even to allow other descriptors to be read. The resulting increased overhead from these restrictions is a constant reminder of the inadequacies of the iAPX 286 architecture.

REFERENCE

KLEWER, M. and VERNOOIJ, B., 'The 68000 microprocessor design philosophy', Electronic Components and Applications, Vol. 6 No. 2, 1984, pp. 79 to 82.

Interfacing different local area networks

JEFF SELTZER and NASEER SIDDIQUE

The increasing use of local area networks in offices and factories is creating a growing demand for inter-network interface controller systems that can interconnect them. However, since these networks usually operate in different time domains, few of the gateway controllers currently available can meet this demand. So networks such as Ethernet, used in office applications where real-time response is not a requirement, cannot communicate with token-passing networks that offer real-time response to critical manufacturing processes and robotics control operations.

The answer can be found in a gateway based on a high-performance bit-stream manipulation scheme (Fig.1) incorporating interface controllers for each network, a message buffer memory and a gateway management controller. This article describes a simple gateway based on this scheme in which the interface controllers and gateway management controller all use identical LSI chips, namely: the 8X305 bipolar microcontroller, the 8X310 interrupt control coprocessor (ICC), the 8X360 memory-address director (MAD) and the 8X350 high-speed dynamic RAM.

The configuration assumes that the two network media can be interconnected locally without the need for long-distance communication. However, the same general hardware design and software structures can be applied to many network types, as well as to gateway halves that interconnect remotely-located networks.

The gateway controller has stringent requirements. It must be able to receive back-to-back messages on either network interface and store the messages that are addressed

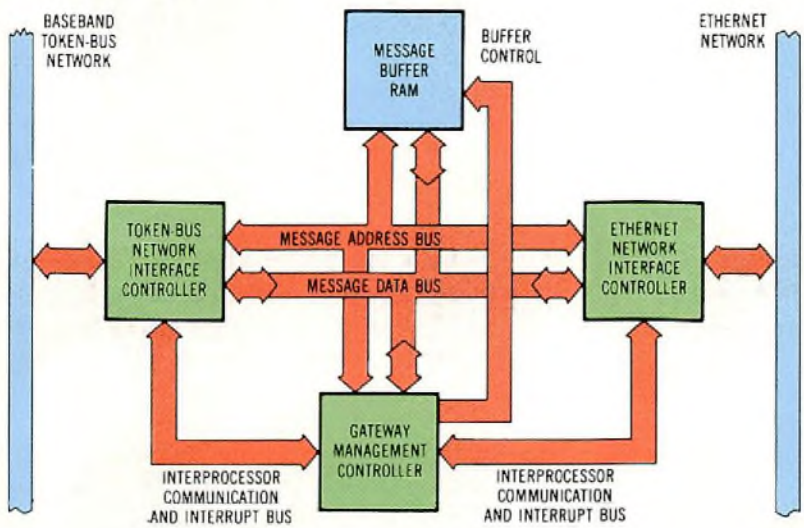


Fig.1 Two local area networks, one CSMA/CD Ethernet, and the other based on a token-passing access method, can communicate via a gateway consisting of interface controllers for both networks, a message buffer memory, and a management controller. This configuration is independent of the distance between the networks

to the gateway for later transmission to the destination network. It must also be able to receive or transmit on one network while simultaneously receiving or transmitting on the other. To accomplish this, processing power is usually distributed among three controller subsections; two network interface controllers (Ethernet and token-bus), and one centralized gateway management controller. The functions of the Ethernet and token-bus interface controllers are shown in the Table.

The gateway controller should receive and transmit messages in the same manner as any other master/slave station. Each of the two network interface controllers has to handle all message traffic situations defined under its corresponding protocol. The controllers operate at network data rates up to 10 Mbit/s, as prescribed by IEEE 802 standards for both carrier sense multiple access/collision detection (CSMA/CD) and token-bus networks.

FUNCTIONAL DISTRIBUTION OF THE GATEWAY NETWORK CONTROLLER SECTIONS

Ethernet and token-bus controller section
Receiving and transmitting message packets on the network media.
Media access control functions (e.g., Ethernet collision recovery, token passing, etc.).
Serialization, coding, and synchronization of the data stream.
Packet destination address matching.
CRC generation and error detection.
DMA to central message buffer.
Interrupt handling and interprocessor communication with the gateway management controller.
Gateway management controller
Allocation of buffer areas to incoming and outgoing messages.
Arbitration of the central message buffer between the gateway manager's processor and the two network controllers.
Interrupt handling and interprocessor communication with the network interface controllers.
Translation of the packet format into the data-link protocol of the destination network.
Any additional higher level protocol translation and management functions.

DESIGN CONSIDERATIONS

These requirements can be met by storing the messages received by the gateway in a central buffer memory. These messages can then be processed to some extent by the gateway manager before they are ready to be retransmitted. This processing includes translation between each network's data-link protocols, so, the demands on the buffer memory are significant.

The message packets on Ethernet and token-bus networks (and most other network protocols) are variable in length. In the case of the token bus, the packet length can vary from about 20 bytes up to over 8000 bytes. Moreover, messages addressed to the gateway are received from each network at random times. After protocol translation is completed, the message is held in queue until its destination's network controller gains access to the media for transmission.

For Ethernet communications, the buffer must be ready either to supply a stored message for transmission, or to receive a new incoming message at short notice. The latter usually occurs when another station's transmission precedes or collides with the outgoing message.

At a 10-Mbit/s data rate, a new data byte has to be accessed by a controller every 800 ns. Within that time interval, the message buffer is able to access a data byte for each of the three controller subsections. If necessary, it will do so in different memory regions. In addition to handling all these varied requirements, one important objective is that the system should make the most efficient use of available buffer memory space to prevent a buffer overflow condition. This requires a robust memory management scheme implemented in the gateway management controller.

The 8X305 bipolar microcontroller meets all these requirements. Designed for bit-manipulative I/O operations, it can execute 5 million instructions per second, including data I/O instructions. Its Harvard architecture separates the program address and instruction buses from its local data (IV) bus. This keeps system control and message data flowing at peak rates. The 8X305 architecture also allows the system to be configured into separate independent buses for optimal data throughput. This allows network message data transfer operations to gain exclusive access to the central message buffer.

In contrast, conventional MOS microprocessor architectures allow the message buffer to be located within the processor's main memory and attached to its common address and data buses. In addition to message traffic, these buses are used by the microprocessor for instruction fetching and local working storage. Since the network controllers may not be able to gain immediate access to the processor's memory bus, first in, first out (FIFO) buffers are usually required to keep the message data flowing. Use of the bus by the network interfaces may also result in an overall reduction in the processor's data throughput rate.

However, with the 8X305, the local program buses, local data bus, and system message buses can be separated, allowing multiple independent operations to be performed in parallel. Yet, the I/O performance capabilities of the 8X305 let it access the message buffer as quickly as most MOS processors access their own memory.

The controller, coupled with the 8X310 interrupt control coprocessor, adds priority-set, multi-level interrupt handling capability. This gives the 8X305 the power to deliver real-time interrupt responses within about 400 ns.

The 8X360 MAD completes the critical link for each of the gateway's controllers to access the central message buffer management.

CONTROLLING THE NETWORK INTERFACE

The gateway system's essential activity centres around the data bus. This bus interconnects the gateway management and the two network interface subsections (Fig.2). All message traffic flows over this data bus.

Messages on the data bus are passed to and from a network medium (Ethernet or token bus) through physical interface circuitry and serial interface control logic. Standard line driver and receiver circuitry form the physical interface to a baseband network medium. They fulfill the electrical requirements defined by the network's physical layer protocol, and convert between the higher voltage signals on the medium and TTL logic levels used by the controller hardware.

Data on the network medium is encoded so that the high/low logic levels convey both data and synchronization information. Several standard coding formats are currently used in both data communication and mass storage applications. The most common include "Manchester" and fm.

The serial interface control logic consists of an encoder/decoder circuit, a cyclic redundancy check (CRC) generator/checker, and a serializer/deserializer circuit. The encoder/decoder circuit translates between the encoded data on the medium and raw serial data. The decoder section must lock onto an incoming message's synchronization pattern and provide the data bits extracted from the message, as well as a clock signal by which the adjacent circuitry can strobe in the serial data. The decoder also detects coding violations that may be caused by line noise, but are also used intentionally by the taken-bus protocol for certain message delimiters. The decoder typically employs a phase-locked-loop (PPL) to establish synchronization. In the transmit mode, the encoder section receives raw serial data and generates the encoded pattern at its own fixed clock rate.

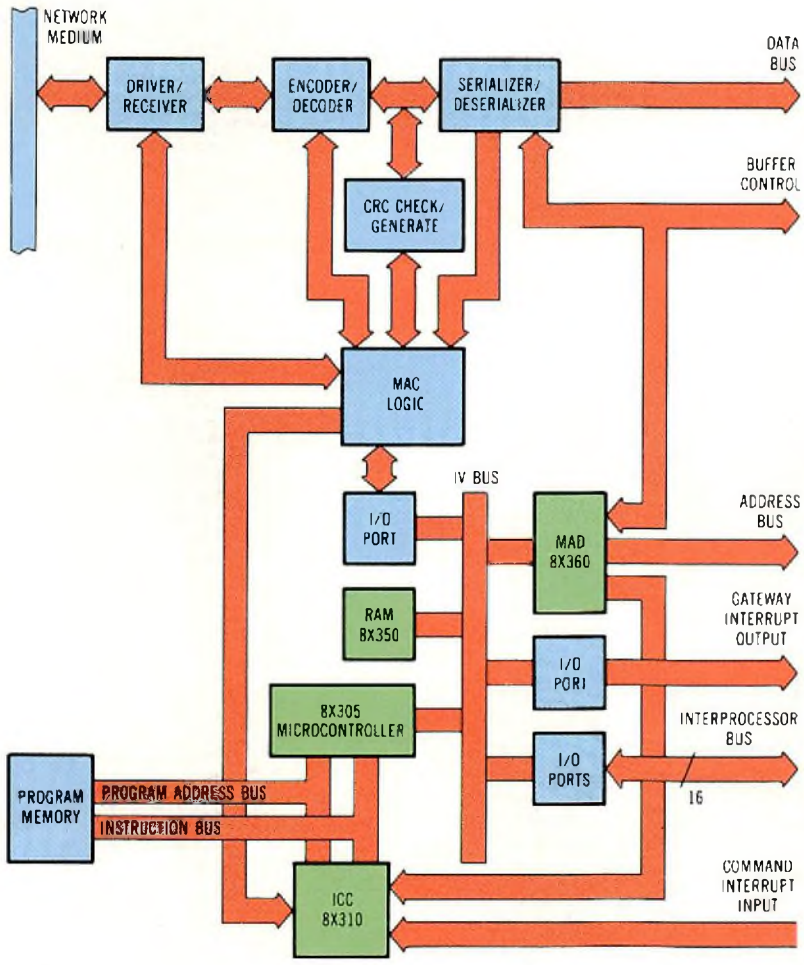


Fig.2 The network interface controller handles all message traffic as defined by the network protocol. In this configuration, the 8X305 microcontroller, in conjunction with a memory address director (MAD), controls the messages from one medium to the gateway. Each network's interface controller has unique software operating the appropriate data-link protocol

The gateway controller services the message-packet routing from one network to another, fragmenting packets for a specific destination network, and embedding internetwork packets in the format of the destination network protocol. Gateway controllers can be subcategorized as media translators, protocol translators, and application translator gateways. Media translators connect two networks that differ from each other only at the physical and the data-link layers. Media conversion gateways can be thought of as devices that bridge an otherwise incompatible gap.

Protocol translator gateways interconnect networks that differ from the physical to the presentation layer. Such gateways convert the media as well as the higher level network protocols.

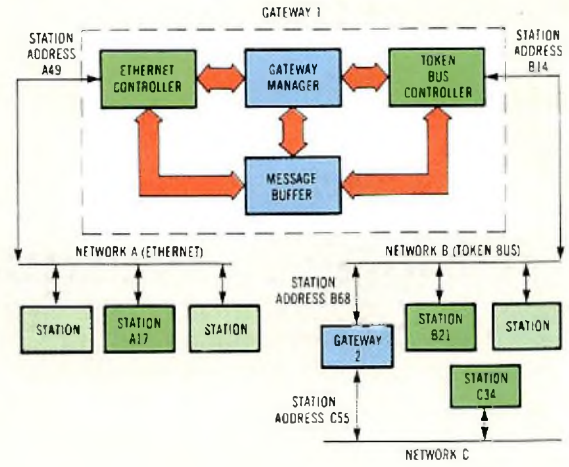
Application translator gateways manage the higher level communication layers and handle application-specific tasks. The gateway has to receive a packet from a network, unwrap the network package, identify the destination address, calculate the optimal routing path, wrap it again into the next network's package, and transmit it on the other network.

Translating these operations into design requires critical design considerations. For example, translation of a longer address to a shorter address and vice versa, translation of bit-oriented protocols (HDLC) to character-oriented protocols (Bisync), and translation of higher level protocols are typical problems.

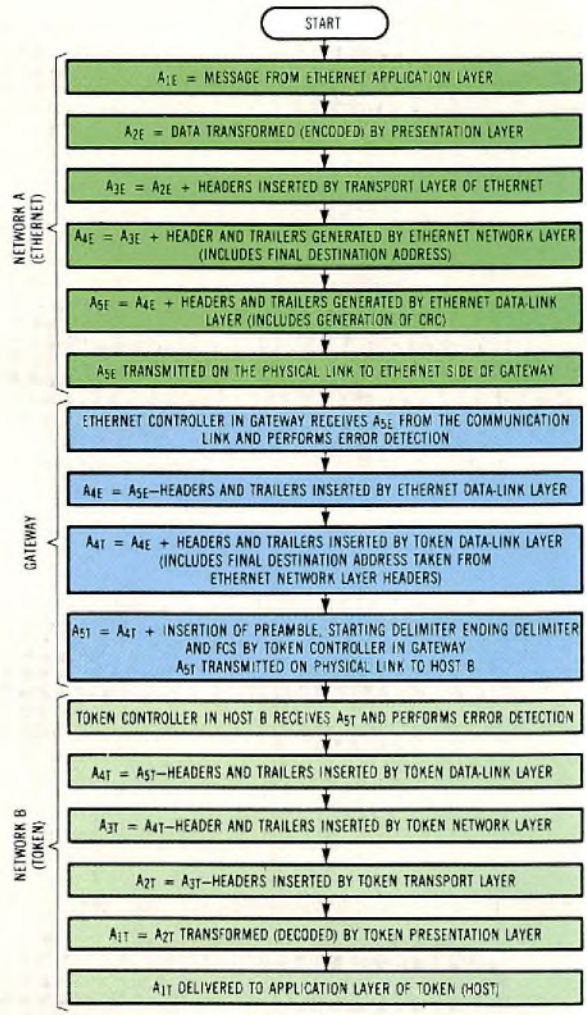
The functions performed by gateways vary considerably, depending upon different interconnection strategies. For instance, a gateway interconnecting a token network to an Ethernet network will have to perform a different set of functions to a gateway interconnecting an x.25 network to Ethernet. Therefore, interconnecting heterogenous networks raises various questions. Are there any standards available? Should the protocols of one network be directly translated to the protocols of another, or should both of them be converted to a standard protocol? What kind of performance will be achieved?

A case in point is an Ethernet that has to communicate with a token-passing network (see the figure). Suppose a station with address A17 on the Ethetnet (network A) has a packet to be transmitted to a station with address B21 on the token bus (network B). The actual message will be formed by station A17's application layer and passed to its presentation layer, which will transform (encode) the message. The next level, the session layer, handles functions such as recovery from abnormal situations and establishment of the connections across the network. Thus, it is not really in the way of the data flow.

The encoded packet is therefore passed directly to the transport layer. This layer encapsulates the entire packets into its data field, and generates and adds new headers to the packet. As the packet flows down to the network layer, new headers and trailers are added. This includes the final destination address on the token network (in this case, station B21).



(a)



(b)

The actual header and trailer formats generated at each layer depend upon the higher level protocols used to implement the corresponding layer. The packet format generated by the network layer will be passed to the data-link layer which will generate the standard Ethernet packet at the link level, including the CRC field. The packet's destination address will be A49, the station address of gateway 1 on the Ethernet. This packet will then be passed on to the physical layer for transmission on the communication channel to the Ethernet side of gateway 1.

Upon receiving the packet, the Ethernet controller in the gateway will unwrap it by peeling off all the necessary headers and trailers inserted by various layers (the number of layers to be unwrapped will depend upon the type of gateway being used). In the case of a media translator gateway, it will only unwrap up to the data-link layer of Ethernet, which will also include error checking.

By eliminating the fields added to the packet by the data-link layer, the message will be restored to the way it was in the network layer of station A17. The gateway controller will now convert the packet to a standard token-packet format by generating another set of headers and trailers. The source address will be B14 (the address of gateway 1 on token network B) and the destination address will be B21 (as indicated in the network layer header). As the message passes through the token controller in the gateway, it will be appended with a new frame check sequence, starting and ending delimiter, and a preamble. This reformatted packet will be transmitted by the token controller on network B. Upon reception, station 21 will have to unwrap all the layers up to the application layer to get the actual message.

Several different networks can be interconnected through multiple gateways. In such situations, a message that has to be transmitted from one network to another may need to be routed via a third network. The possibility of errors is of course increased, but one can assume that every gateway involved in transmission, reception, and translation of packets can detect and correct the errors. Suppose the packet transmitted by station A17 is destined for station C34 on network C, but the packet has to go through network B. The packet will first be reformatted for network B by gateway 1 and then again for network C by gateway 2.

Station A17 defines the immediate destination address (A49) in its destination field and the final address (station C34) in its network layer header, then forwards the packet to gateway 1. Upon reception, gateway 1 reformats the packet by replacing the source address with its own address (B14) and the destination address with the intermediate address for gateway 2 (B68). Gateway 1 also generates a new CRC, then forwards the packet to gateway 2. Upon reception, gateway 2 reformats the packet again, replacing the destination address field with the address of station C34 and the source address field with its own address (C55). Gateway 2 also calculates a new CRC, then forwards the packet to the final destination.

Each network interface controller checks for errors and generates CRC values (referred to as the frame check sequence in the token-bus protocol). It is necessary to regenerate a new CRC for each outgoing message because the packet header information is altered during protocol translation. Depending on its software capabilities, the gateway manager may correct detected errors in received messages. Otherwise, the message is tagged to inform the destination host of the error condition.

The serializer/deserializer logic converts between the raw serial data and parallel data bytes that can be transferred on the gateway's data bus. The encoder/decoder circuit synchronizes the serial interface. Every eight clock cycles, a parallel data byte is transferred between the shift register (which performs the conversion) and a holding register. Before the next byte has finished shifting (i.e., within the next eight clock cycles), the holding register must be serviced (contents read or replaced). This is the point at which the network data stream synchronizes with the message buffer access cycles.

The serial interface control logic performs functions that are fairly common to various data communication and mass storage controller applications. Serial interface control logic implemented with discrete logic functions, or more advanced field-programmable arrays, or even gate arrays, typifies the current solution to management of the data stream. Inevitably, single-chip solutions will appear to replace discrete ones.

The 8X305 microcontroller coordinates all the activities within the network interface control section. It controls data movement through the serial interface control logic and to and from the message buffer without actually standing in the data path. Instead, it sets up conditions that allow the buffer to be accessed directly by hardware. Arbitration logic in the gateway management section controls the actual real-time access to the buffer.

Message transfers on the data bus are very similar to conventional DMA in computer systems, except that the data bus is used exclusively for the message data traffic, and there is no host processor on the bus to contend with. These message transfers require that the controller generates a sequence of buffer addresses to accompany the data. The MAD performs that function.

After the microcontroller initializes the various registers and counters, the MAD chip can count through a sequence of addresses independently with respect to the 8X305's processing. In the gateway design, the MAD output connects with the system address bus, which runs parallel to the data bus. The MAD units in each of the three controller sections supply address information to the message buffer via the address bus.

A collection of I/O ports provides a multipurpose interface between the microcontroller and other parts of the system for passing data and control/status signals. All I/O ports and data-oriented peripherals, such as the 8X360, are

connected to the microcontroller on a special local data bus called the IV bus. A 256-byte high speed bipolar RAM (8X350) is also attached to the IV bus, and provides local working storage for the microcontroller.

Media access control (MAC) functions are distributed between the MAC logic (specialized hardware) and microcontroller software in each network interface controller. The 8X305's instruction execution and data transfer speeds allow the microcontroller to handle many of the packet-header processing and MAC functions in software. Consequently, the size of the MAC hardware can be minimized. The primary duty of the MAC logic is to handle the operations that require an immediate response on the network.

The network controller activities are coordinated with the gateway management controller through the exchange of interrupts between the respective microcontrollers. Interrupt-request signals from the gateway manager are handled by the 8X310 ICC. The ICC attaches to the 8X305's program address and instruction buses to control interrupt calling and return transfers. The microcontroller is also interrupted to respond to certain network control activity requested by the MAC logic, and to occasionally service the MAD.

THE GATEWAY MANAGER

Interrupt requests to the gateway manager are software generated. A pair of I/O ports passes information directly between the controllers via a separate interprocessor bus. These exchanges occur in conjunction with interrupt requests and include such information as interrupt type, buffer address, and message length. Since these interrupts often occur during real-time message transfers, the system data bus cannot conveniently be used to exchange the information.

The main functions of the gateway manager are to maintain the message buffer and to perform all necessary message-packet translations and higher level protocol processing. As in the network interface controllers, the heart of the gateway manager is an 8X305 microcontroller interfaced with the message buffer RAM via an 8X360 MAD and an I/O port. Also included are the 8X310 ICC and I/O ports required to perform the interprocessor communication with the network controllers (Fig.3).

The message buffer itself consists of 32 Kbytes of RAM. With a maximum token-bus message size of 8 Kbytes, this buffer size accommodates one incoming message on each network interface plus storage for other message being translated or awaiting transmission. The buffer is segmented by the controller into 128 pages of 256 bytes. Although this should be sufficient, the buffer size can be doubled if extremely heavy traffic is anticipated or lengthy protocol translation routines are to be implemented. With an access cycle time of 200 ns, the buffer memory can serve all three controller sections within the required time interval of 800 ns (derived from the network transmission data rate up to 10 Mbits/s).

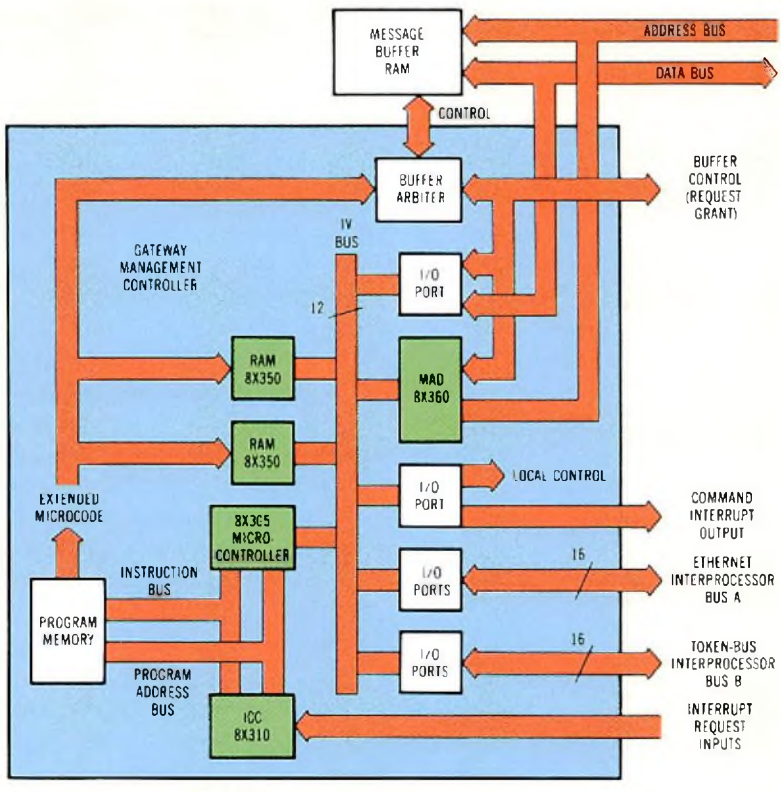
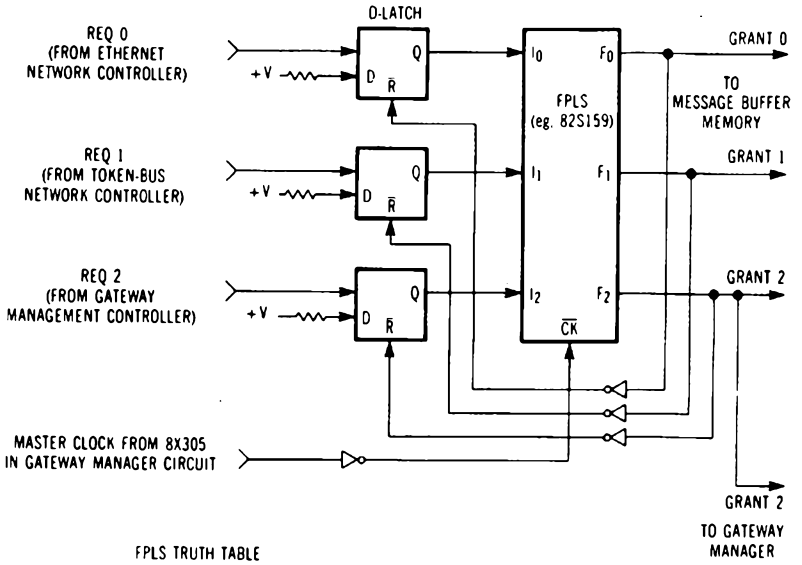


Fig.3 The same chips that made up the heart of the network interface controller can be used for implementing the functions of the gateway management controller. The message buffer memory should be able to accommodate data from both the management controller and the two interface controllers within the 800-ns access time that is available in a network operating at up to a 10-Mbit/s rate



FPLS TRUTH TABLE

GRANT 0	= REQ 0 • $\overline{\text{GRANT 0}}$ • $\overline{\text{GRANT 1}}$ + REQ 0 • $\overline{\text{REQ 2}}$ • GRANT 1 + REQ 0 • $\overline{\text{REQ 1}}$ • $\overline{\text{REQ 2}}$
GRANT 1	= REQ 1 • GRANT 0 + REQ 1 • $\overline{\text{REQ 0}}$ • $\overline{\text{GRANT 1}}$ + REQ 1 • $\overline{\text{REQ 0}}$ • $\overline{\text{REQ 2}}$
GRANT 2	= REQ 2 • GRANT 1 + REQ 2 • $\overline{\text{REQ 1}}$ • GRANT 0 + REQ 2 • $\overline{\text{REQ 0}}$ • $\overline{\text{REQ 1}}$

Fig.4 The buffer arbitration logic acts as a special-purpose stand-alone DMA controller. Using a few latches and a field-programmable logic sequencer (FPLS), this logic circuit samples the request lines from the three interface controllers every 200 ns and arbitrates a memory cycle to the message buffer memory according to the round-robin sequence specified by the FPLS truth table

The buffer arbitration logic acts as a special-purpose stand-alone DMA controller (Fig.4). At the end of every 200-ns memory access cycle, the DMA controller samples the buffer access request lines from each of the three controllers and arbitrates a memory cycle on a round-robin priority basis. In the worst-case situation, when all three requests are active during consecutive cycles, the arbiter rotates among the three controllers and grants each an access window every 600 ns. The logic required to perform this function is easily implemented using a few latches and a field-programmable logic sequencer (FPLS) circuit.

The buffer access-request signal for a network controller is taken from the parallel data strobes of the serial interface control logic. This signal also triggers the local MAD to sequence to the next address in its assigned buffer space. When the arbiter grants the memory cycle, the corresponding MAD is enabled and a data byte is transferred to or from the buffer over the data bus. Since all of the gateway manager's buffer accesses are software-controlled, the memory cycle requests for this controller are generated by extended microcode bits programmed in the local microcontroller's program memory. Extended microcode is a block of extra bits added to the program memory's width to provide, with each instruction executed, signals for general control functions and for fast I/O port selection.

The software running on all three microcontrollers exists as a collection of real-time tasks. Except for initialization, all tasks in the network interface controller are invoked in response to interrupts received either from the gateway management controller or from local operational units. Most tasks in the gateway management controller are invoked by interrupts from the network controllers, with the exception of the packet translation routines. These run in the main program under the control of a primitive executive.

As mentioned earlier, the message buffer is divided into 128 pages with 256 bytes/page. The gateway management software maintains a table of 128 pointers in its local working storage that correspond to the 128 pages. When a message received from a network controller runs over into more than one buffer page, the pointer for the first page will contain the address of the pointer for the second page which, in turn, contains the next pointer address, etc. Thus, a linked list is formed. The software includes several lists for packets currently being received, those waiting for protocol translation, those waiting to be transmitted, and free pages ready to be allocated to new messages. For each of the above processes (except free pages), separate lists are maintained for each of the two networks from which messages originate.

The first and last pointer of each list are tracked. As a process is completed on a buffer page or packet (packets may consist of one or more pages), the corresponding pointers are removed from that process' list and appended to the list for the subsequent process. For example, when a packet has finished being translated from token-bus proto-

col to Ethernet, the pointers for those message pages are removed from the "translate token-bus" list and added to the packet list awaiting transmission on the Ethernet.

Initializing the gateway manager

When the gateway manager is initialized, it links all the pointers to the free page list. Then, it allocates one page to each network receiver and sends each network interface controller an ALLOCATE-RECEIVE-PAGE (ARP) command. The command is sent by placing the ARP command code and the allocated page address on the interprocessor bus to the desired controller and generating an interrupt request.

When interrupted, the network controller prepares its MAD (8X360) with the received page address and awaits an incoming message. As a message addressed to the gateway is detected, the controller begins filling the designated buffer page and immediately requests another page to use in case the first one fills up. The network controller sends a REQUEST-RECEIVE-PAGE (RRP) interrupt to the gateway manager, which responds by sending another ARP command with a new page address. Each page allocated to that receiver is added to the receive Ethernet list or to the receive token-bus list.

Should a page become filled during an incoming message, the network controller is interrupted by its MAD. The MAD is then reloaded so that the incoming message begins to fill the new buffer page. At this time, another RRP interrupt is issued to set up for the next page.

When an incoming message terminates, the network controller sends a RECEIVE-MESSAGE-COMPLETE interrupt with the length of the last buffer page on the interprocessor bus. The gateway manager responds by transferring all page pointers associated with the completed message from the receive list and links them onto the appropriate translate list. A flag is then set to notify the operating system kernel to execute the protocol translation routines.

For basic media-translation operation, where only the data-link fields of the packet are altered, the network controllers are set up to leave sufficient padding at the beginning of each packet so that translation can be performed in place (i.e., without copying the message data to another buffer location). However, if additional higher level protocol translation routines require message copying, another pointer list (for each network) could be added to keep track of the intermediate pages.

After the message is translated, the associated pointers are transferred to the transmit list for the opposite network. A TRANSMIT-PAGE command with the first page address is sent to the network controller. The MAC logic is armed and awaits the appropriate conditions on the network medium to begin (or attempt to begin) transmission of the outgoing message.

Similar to the receiving process, whenever the network controller begins to transmit a new page of the outgoing message, it sends a REQUEST-TRANSMIT-PAGE (RTP) interrupt to the gateway manager so that it can continue to transmit when the current page becomes exhausted. The gateway manager responds to RTP interrupts with TRANSMIT-PAGE commands.

Upon completion of a packet transmission, the network controller sends a TRANSMIT-MESSAGE-COMPLETE interrupt that lets the gateway manager return all page pointers associated with the completed message back to the free page list, thereby making them available for new incoming messages. Thus, communications between different local area networks is assured through a gateway configuration that has LSI circuits and sophisticated software.

ACKNOWLEDGEMENT

This article originally appeared in the February issue of Computer Design, copyright 1984, PennWell Publishing Co.; permission to reprint is gratefully acknowledged.

The KP100A monolithic pressure sensor

GERD KEITEL

The trend toward integration in electronic control and measuring systems has created a growing demand for fast, accurate sensors. Among these the KP100A silicon pressure sensor stands in a leading position. Not only is it both very fast and accurate, but with its monolithic silicon construction, it's also highly compatible with the electronic systems that it's meant to serve.

The KP100A (Fig.1) is essentially an aneroid gauge in that it relies for its pressure signals on the movement of a diaphragm closing an evacuated chamber. But unlike the conventional aneroid gauge, the KP100A has no need of mechanical or optical means to detect diaphragm movement. In the KP100A, diaphragm movement is detected directly by a series of piezoresistive strain gauges implanted in the diaphragm in a Wheatstone bridge configuration.

The advantages of an all silicon device include highly stable characteristics, thanks principally to the well-defined behaviour of silicon, plus high reliability and low cost, thanks to the vast fund of experience in semiconductor manufacture that can be brought to bear in its production.

This article describes the construction and operation of this new device, and the control circuitry for using it in a wide range of applications: from simple pressure switches in domestic appliances, to altimeters and depth gauges, from domestic and professional barometers to automotive control systems.

The article also describes a new signal-conditioning IC that for many applications replaces much of the external control circuitry and provides a linear output signal, temperature compensation and facilities for offset and sensitivity trimming.

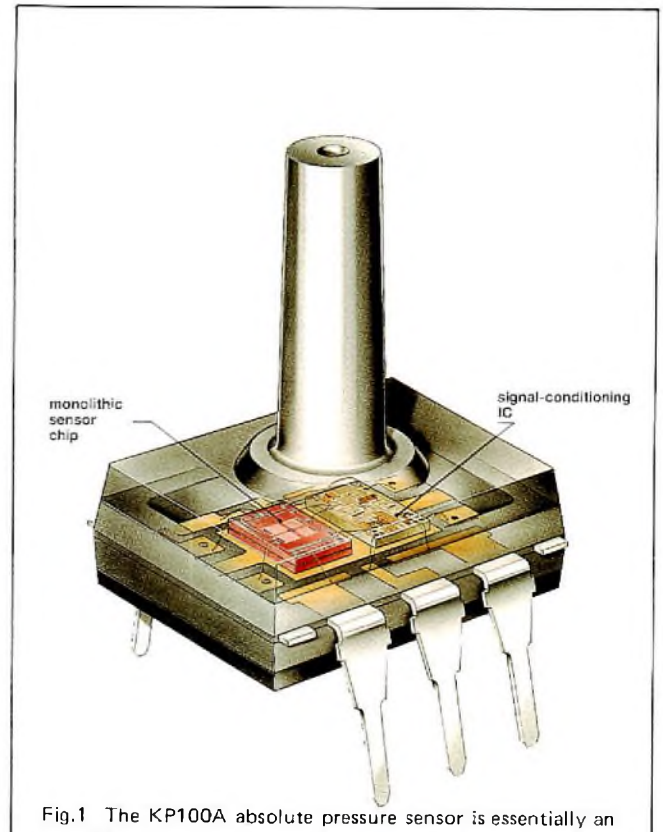


Fig.1 The KP100A absolute pressure sensor is essentially an aneroid gauge in which diaphragm movement is detected directly by a series of piezoresistive strain gauges implanted in the diaphragm in a Wheatstone-bridge configuration

PIEZORESISTIVE STRAIN GAUGES IN THE KP100A

In metals and semiconductors, both the mobility and concentration of carriers may change with strain. In semiconductors, concentration changes come from a change in energy gap with strain. Depending on the type of doping, an increase in strain may produce either a fall or a rise in material resistivity; this is the *piezoresistive effect.*

The amount by which resistance changes with strain is given in terms of the gauge factor, defined as the fractional change of resistance ($\Delta R/R$) per unit strain ($\Delta l/l$). For semiconductors, gauge factors between 50 and 100 are common. Metals that exhibit the piezoresistive effect have gauge factors of around 2.

Figure 2 shows, in plan and elevation, the major features of the KP100A chip. Essentially the chip consists of a Wheatstone bridge located in the centre of a rectangular ($1200 \times 2400\ \mu m^2$) silicon membrane closing an evacuated chamber. Flexure of this membrane produced by external pressure, causes strain in the resistive elements of the bridge, and so (by virtue of the piezoresistive effect) imbalance of the bridge. The degree of bridge imbalance is then used as a measure of the external pressure.

Note: the device relies for its action on the fact that the membrane is non-square, and hence that adjacent arms of the bridge experience differing strains. If the membrane were square, all arms would experience exactly the same strain and no bridge imbalance would occur.

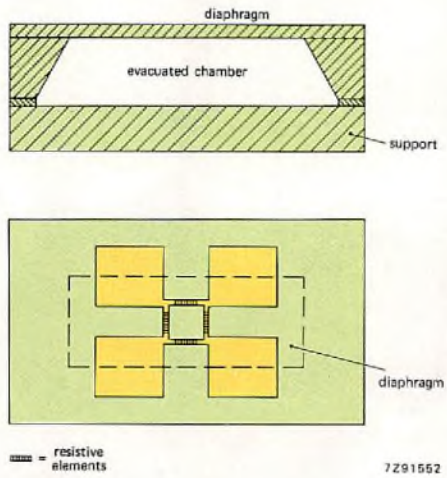


Fig.2 KP100A in plan and elevation. The strain gauges are centrally located in a rectangular silicon membrane enclosing an evacuated chamber

Figure 3 shows the basic bridge setup. The sensitivity of the bridge is around 13 mV/Vbar at 25 °C, but since the resistive elements are temperature sensitive, the sensitivity of the bridge itself varies with temperature – by about –0,2%/K. To compensate for this, and to allow its use over wide temperature ranges, the device incorporates integrated V_{BE} multipliers. These increase the voltage across the bridge as the temperature rises, and so compensate the loss in sensitivity due to rise in temperature.

With a multiplier in circuit, sensitivity of the bridge falls to around 7,5 mV/Vbar, but its temperature coefficient falls to a mere ±0,02%/K.

The KP100A chip has, in fact, five such integrated multipliers with different temperature characteristics, each optimized for a specific operating voltage.

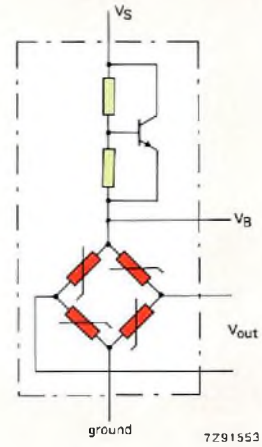


Fig.3 Bridge configuration. The device incorporates integrated V_{BE} multipliers that increase the voltage across the bridge as the temperature rises, and so compensate the loss in sensitivity due to rise in temperature

KP100A MANUFACTURE

Figure 4 (opposite) shows the essential production steps for the KP100A. The production process can be summarized as follows:

- forming a high-ohmic n-type epitaxial layer on the surface of an arsenic-doped polished silicon slice (normal to the ⟨100⟩ crystal direction)
- diffusing p-type strain gauges (and contact pads) into the epitaxial layer in a Wheatstone bridge configuration, and at the same time diffusing separate p-type base regions for the integrated V_{BE} multipliers
- diffusing n-type emitter regions for the V_{BE} multipliers
- etching the underside of the silicon chip to form the silicon membranes. First a chemical etch (potassium hydroxide) followed by an electrolytic etch (in hydrofluoric acid). The latter etch stops automatically at the high-ohmic epitaxial layer, and so allows for very well defined membranes
- attaching the etched slice to a silicon support slice (with aluminium adhesive) and evacuating the chambers beneath the membranes

– aluminium metallization, formed on a silicon dioxide insulation layer (with contact windows), followed by silicon nitride passivation.

The individual sensor elements on the slice are then measured electrically to provide a preselection on characteristics, after which they are cut into separate chips and encapsulated (in 6-pin DIL). The encapsulation bears on its upper side a 3 mm diameter tube through which the unevacuated space above the membrane opens to the atmosphere. Figure 5 shows a sensor chip before encapsulation.

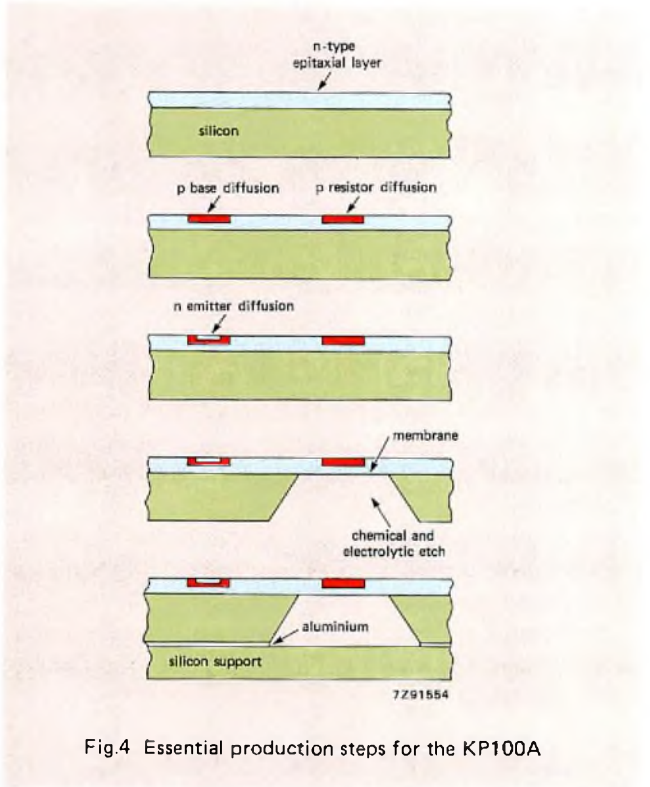


Fig.4 Essential production steps for the KP100A

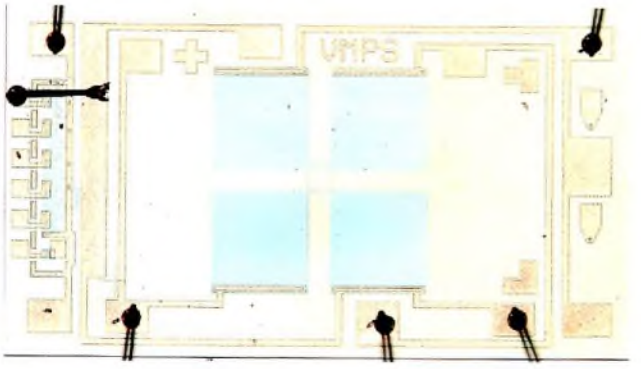


Fig.5 Sensor chip before encapsulation. On the left you can see the transistors that form the V_{BE} multipliers used for temperature compensation

CHARACTERISTICS AND PERFORMANCE

Temperature compensation

Figure 6 shows the relationship between bridge output voltage and pressure, with supply voltage as parameter. The figure illustrates two things: first it shows that the relationship is substantially linear, and second it shows how the V_{BE} multipliers function. In Fig.6(a) the multiplier in circuit is the one optimized for a supply voltage of 4 V, and here you see that the variation in sensitivity over a 25 to 100 °C temperature range is extremely small, amounting to no more that about 1,5% over the whole temperature range. Figure 6(b) shows the same relationship using the multiplier optimized for a supply voltage of 10 V.

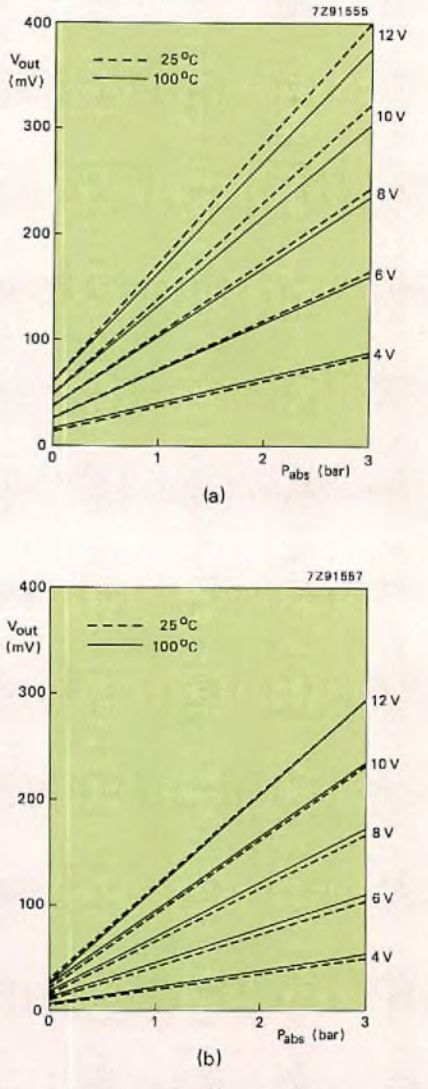


Fig.6 Bridge output voltage versus pressure with supply voltage as parameter and V_{BE} multiplier in circuit, (a) optimized for 4 V supply and (b) optimized for 10 V supply

Figure 7 gives further illustration of the benefits gained from temperature compensation. Here output voltage is plotted against temperature with pressure as parameter, Fig.7(a) without compensation, Fig.7(b) with compensation.

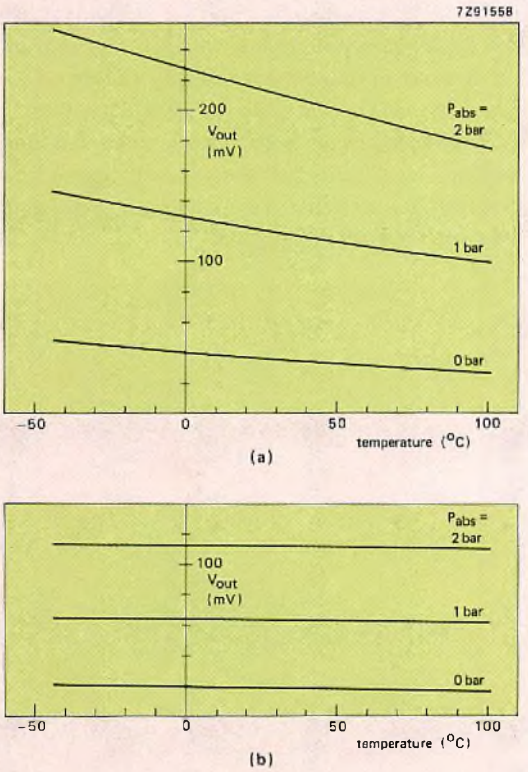


Fig.7 Output voltage versus temperature with pressure as parameter, (a) without compensation, and (b) with compensation

The table gives some important data on three versions of the sensor: the KP100A optimized for 7,5 V operation, a version optimized for 5 V operation (designated KP100A1), and a low-pressure variant (the KP101A).

CONTROL CIRCUITRY

Present versions of the KP100A require external control circuitry, but shortly, a version of the device will be available with a special signal-conditioning IC encapsulated with the sensor chip. To allow for both these future options, we'll describe here an external control circuit for the sensor, and then follow with a brief description of the signal-conditioning IC.

External control circuitry for the KP100A

Figure 8 (opposite) shows the circuit. A stabilized supply comprising opamp A_1, zener diode D_1 and transistor TR_1 provides the settable operating voltage V_S for the bridge.

Opamp A_3, wired as a differential amplifier with summing inputs, provides an output signal proportional to the bridge output voltage, and this signal is amplified by opamp A_4.

Opamp A_2 generates a reference voltage V_{ref} for A_3 equal to half the bridge supply voltage V_B. This voltage can be used to extend the linearity range down to zero absolute pressure; it can also be used to avoid common-mode errors at the input of A_3.

The final output signal from A_4 can be set to a convenient value by means of potentiometer R_5 without affecting the symmetry of the signal to A_3.

The circuit has facilities for adjusting offset voltage and sensitivity, and for adjusting the temperature coefficients of these quantities.

Sensor specifications

	KP100A	KP100A1	KP101A
bridge supply voltage			
max	12 V	12 V	12 V
typ (optimum temperature compensation)	7,5 V	5 V	5 V
operating pressure range	2 bar	2 bar	1,2 bar
sensitivity (at 25 °C)	9 – 17 mV/Vbar	9 – 17 mV/Vbar	14 – 28 mV/Vbar
offset voltage	±5 mV/V	±5 mV/V	±5 mV/V
temperature coefficient of sensitivity			
uncompensated (bridge supply ≤12 V)	−0,2%/K	−0,2%/K	−0,2%/K
compensated	±0,02%/K	±0,02%/K	±0,02%/K
temperature coefficient of offset voltage (full scale)			
uncompensated (bridge supply ≤7,5 V)	±0,04%/K	±0,04%/K	±0,04%/K
compensated	±0,06%/K	±0,06%/K	±0,06%/K
bridge resistance	1,8 kΩ	1,8 kΩ	≈1 – 2 kΩ
pressure hysteresis (full scale)	±0,6%	±0,6%	≈±0,6%

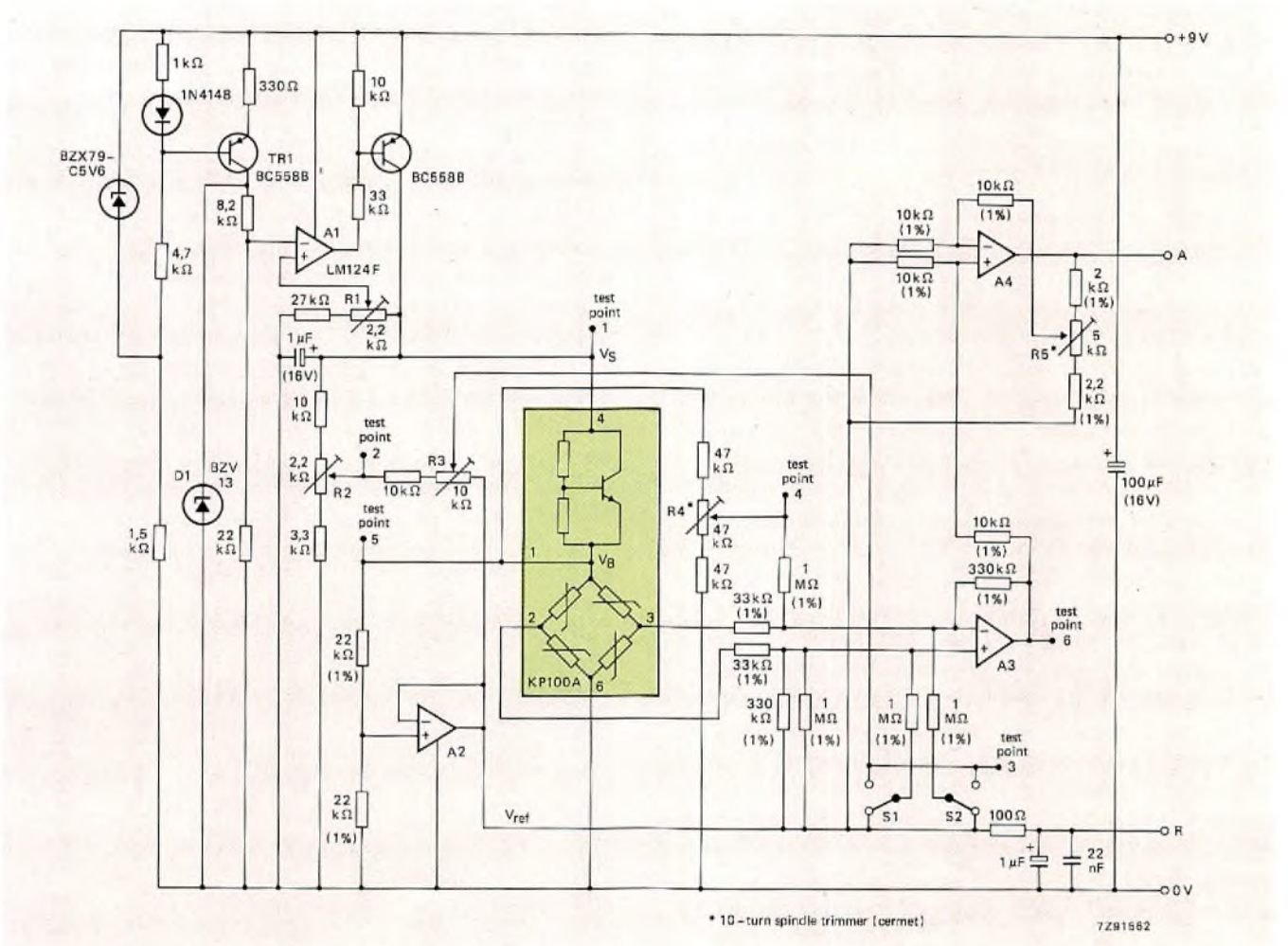


Fig.8 External control circuitry for the KP100A

Adjusting offset voltage

A finite offset voltage, i.e. non-zero bridge supply voltage even at zero pressure (due to small imbalances of the bridge resistances), would lead to common-mode errors at the input of A_3. The offset must therefore be eliminated as far as possible.

It's convenient to split the offset voltage into two parts: one that depends directly on the temperature of the bridge resistors, and one that depends on the bridge supply voltage V_B, and therefore indirectly on temperature (since V_B itself varies with temperature).

To compensate the first part i.e. the part directly dependent on the temperature of the bridge resistors, a fixed voltage, derived from the stabilized supply voltage V_S, is applied to one of the inputs of A_3 (determined by switches S_1 and S_2) and a voltage derived from V_{ref} is applied to the other input. Since V_{ref} is temperature dependent, so is the voltage difference between the two inputs of A_3. The dependence of this voltage difference can be adjusted by potentiometer R_3, and its polarity adjusted by S_1 and S_2 to precisely compensate the temperature-dependent offset of the bridge.

Potentiometer R_2 is used to set the compensating signal to zero at the reference temperature of the bridge, at which point the temperature-dependent part of the offset voltage is zero.

To compensate the second part of the offset voltage, i.e. the part dependent on V_B, an equal voltage (set by potentiometer R_4) derived from V_B is added to the negative input of A_3. The compensating voltage then effectively acts in opposition to the offset voltage, and since both are proportional to V_B, complete compensation ensues for all values of V_B (and hence for all temperatures).

Adjusting sensitivity

Although the V_{BE} multipliers considerably reduce the bridge's temperature coefficient of sensitivity (from −0,2%/K to ±0,02%/K), further reduction may still be necessary for precision applications. This can be done by finely adjusting the operating voltage V_S for optimum compensation using potentiometer R_1.

Signal-conditioning IC for the KP100A

The KP100A is also available with a dedicated signal-conditioning IC encapsulated with it. The IC requires an external power supply and provides the following functions:

- an output signal proportional to pressure ranging from 1,5 V to 5 V
- temperature compensation
- and facility for offset and sensitivity trimming.

In addition, the IC can operate from a range of supply voltages from 4,75 V to 8 V (stabilized).

The circuit is based on a p-type silicon substrate with aluminium metallization. It incorporates low-ohmic WTi and high-ohmic CrSi thin-film resistors. All active areas of the circuit are protected by silicon dioxide glassover.

Figure 9 gives the functional layout of the IC, which incorporates a preamplifier stage, a current-limiting stage and a feedback control line.

The preamplifier stage, which incorporates a series of opamps and trimmable WTi and CrSi resistors, amplifies the bridge signal by between 40 and 200. Moreover, it compensates for the bridge's offset at low operating pressures, and for its temperature coefficients of offset and sensitivity.

The analog output stage comprises 10 parallel-connected pnp transistors (one of which is utilized for current limiting), to provide an output signal of at least 90% of the supply voltage (for loads from 1 kΩ to infinity). The output stage has a low voltage drop (no more than about 0,5 V for a nominal supply of 5 V), so to prevent oscillation, a capacitance of at least 1,5 μF is connected externally across the output.

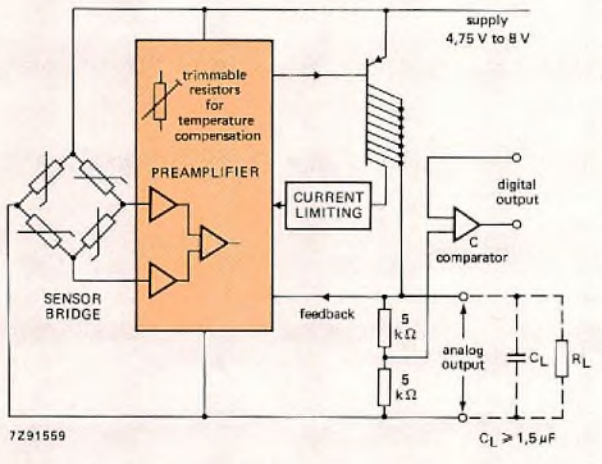


Fig.9 Signalling-conditioning IC

The feedback loop serves as a voltage divider for comparator C which provides a TTL-compatible digital output for switching functions. The loop also provides an internal load that enables the circuit to operate down to zero load resistance.

The accuracy of the circuit depends on trimming. With very fine trimming, a tolerance of ±2% can be realized, but in general, a tolerance of around ±4% may be expected.

Finally, Fig.10 shows how the output signal from the sensor/IC combination varies with pressure for a supply voltage V_{CC} of 5 V, and a load of 1 kΩ. The variation is linear from a pressure of 2 bar down to virtually zero. The residual voltage at zero pressure is shown in Fig.11. For an 8 V supply, the linear relationship extends down to 90 mbar, and gives a residual voltage (for zero load) of 0,325 V at zero pressure; but for a 5 V supply the linear relationship extends down to 9 mbar and gives a residual voltage (for a load of 1 kΩ) of no more than 0,02 V at zero pressure.

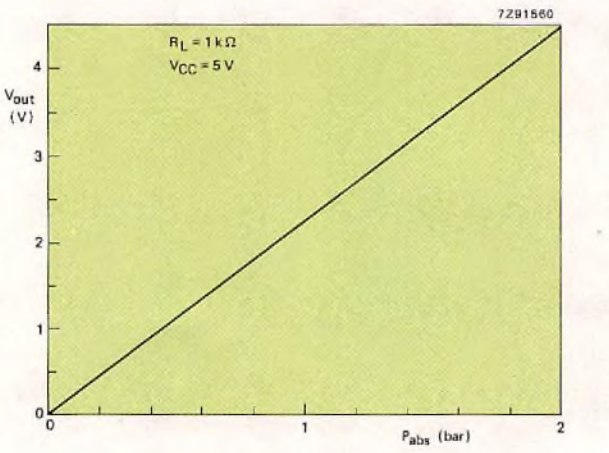


Fig.10 Output voltage of sensor/IC combination as a function of pressure, for a 5 V supply and 1 kΩ load

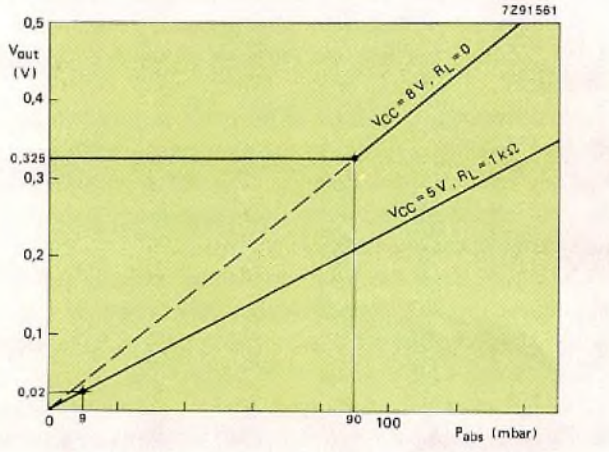


Fig.11 Residual voltage of sensor/IC combination for supply voltage of 8 V and 5 V

New implosion technology to meet explosion in diode demand

GRAHAM HINE

It's not often in the electronics industry that we get the best of both worlds. Usually advances in one area lead to penalties in another, and the overall benefits of an innovative step may not always be clear. When they are, however, the prospects can be pretty exciting. Such is the case with our new implosion diode (ID) technology.

Developed primarily to simplify the production process of our glass-bead diodes, ID technology also provides many other advantages. On the electrical side, for example, IDs have a lower forward voltage than glass-bead diodes thanks to a lower contact resistance. And on the mechanical side, they're smaller and far more regular in shape. So they allow much higher packing density on PCBs and they're ideal for use with automatic assembly equipment. What's more, IDs are just as robust as their glass-bead counterparts, and their reliability is at least as high.

To explain what IDs are, let's first look at the way glass-bead diodes are made, and then see how we've revolutionized this process in our new implosion technology.

IMPLOSION PROCESS SIMPLIFIES PRODUCTION AND CUTS COSTS

Essentially, glass-bead diodes are produced in four stages. In the first stage, the leads are soldered at high temperature to molybdenum studs. These are then alloyed (with aluminium) to the crystal in the second stage. The third stage involves individually coating each diode with glass in the form of a suspended slurry, which is then sintered in the fourth stage, the molten glass on cooling forming the glass bead.

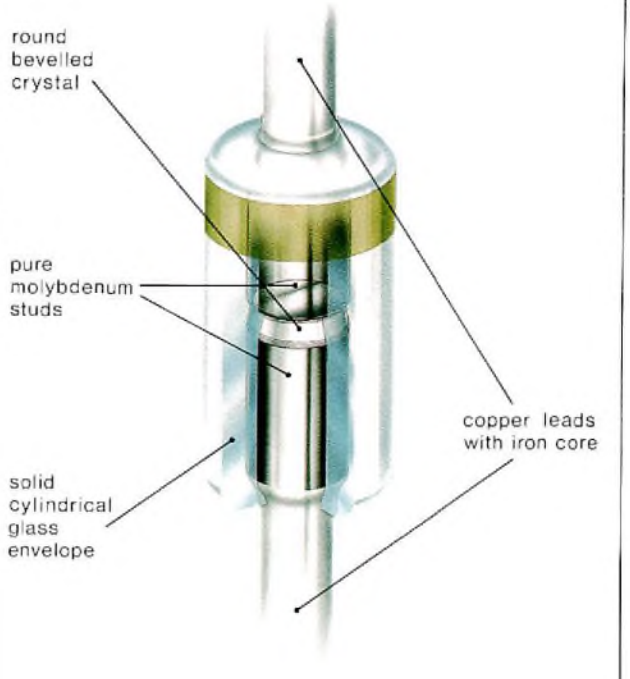


Fig.1 Round, bevelled crystals, molybdenum studs and thermally-matched cavity-free glass encapsulation are just some of the features IDs share with glass-bead diodes. They are also smaller, more regular in shape, and have a lower forward voltage. And they allow a wider range of crystal sizes and diode technologies

In ID manufacture we replace the last three stages of glass-bead production with a single implosion process. Figure 2 shows how the process works. As in glass-bead production, the leads and molybdenum studs are first soldered together. The crystal and studs (silver coated for good electrical contact) are then assembled in a glass sleeve and the whole assembly is clamped in a jig and heated under nitrogen at atmospheric pressure. The assembly is then wetted under vacuum (with nitrogen containing 1% oxygen) to oxidize the surface of the molybdenum studs, and the glass sleeve, which has become pliable in the meantime, then adheres to this oxidized surface.

Implosion then follows, a strictly controlled process in which the vacuum is rapidly removed, causing the glass sleeve to implode and tightly grip the studs and crystal. The result on cooling is a highly regular, cavity-free solid-glass diode in which the glass not only forms the encapsulation and provides passivation for the crystal, but is also the principal agent for maintaining contact between studs and crystal.

Here are the principal advantages of this new process:

- first of course there's the larger production capacity (and lower production costs), thanks to the fact that the implosion process eliminates three essential steps in the production of glass-bead diodes
- the implosion process also allows us to produce IDs in a much wider range of crystal sizes than is currently possible with glass beads. Since the diode is assembled inside the glass sleeve, the crystal/stud joint is never subjected to mechanical strain. The lower size limit for ID crystals is therefore 0,7 mm instead of the 1 mm of glass beads. And the upper limit increases too, from around 2 mm in glass beads to 3 mm, and even 4 mm should be possible in the future.
- then there's the fact that the studs and crystal maintain contact by pressure (Ag/Ag contact). This is the chief reason for the lower contact resistance, and hence the lower forward voltage of IDs, which stems from the fact that we don't need to alloy the studs and crystal together (necessary in glass-bead production). So the crystal can be around 30 μm thinner, reducing forward voltage and, in addition, allowing the use of shallow junction technology. Moreover, the pressure contact has no adverse effect on component reliability
- and finally, by carefully matching the thermal characteristics of the glass sleeves to those of the molybdenum studs, we completely eliminate the chance of failure due to thermal fatigue.

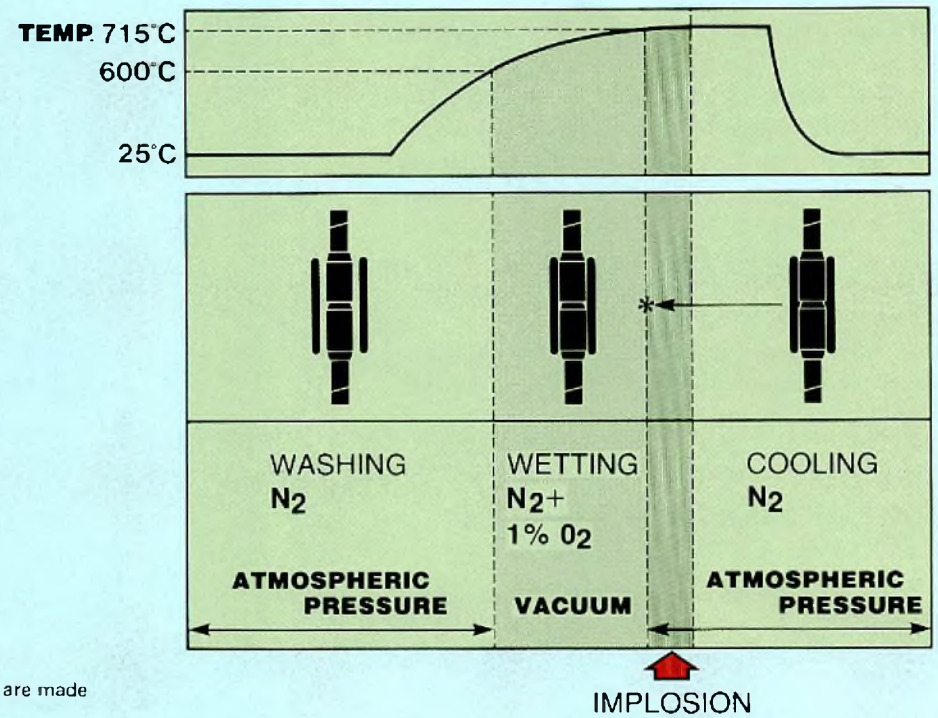


Fig.2 How IDs are made

ALL THE ADVANTAGES OF GLASS-BEAD DIODES AND MORE . . .

The major features that IDs share with our glass-bead diodes include the use of circular crystals with bevelled edges for optimum reverse blocking stability and for reduced field strength at the crystal surface (hence excellent passivation).

Fig.3 Axially-leaded IDs

In both types, the rugged solid-glass encapsulation provides a perfect hermetic seal, entirely eliminating flammability problems experienced by other types (e.g. plastic encapsulated diodes). And both glass beads and IDs have a guaranteed high tolerance to reverse-voltage transients, excellent forward-surge behaviour, and complete freedom from the 'flake problems' associated with hollow-glass (whiskerless) diodes.

TABLE 1
Advantages of IDs over glass-bead diodes

manufacturing process	simpler: lower temperatures for lower stresses, allows a much wider range of technologies and crystal sizes
encapsulation	smaller, cylindrical encapsulation with fewer voids, allows higher packing densities on PCBs, ideal for automatic insertion and surface-mounted assembly
chip-stud contacts	high-quality Ag-Ag pressure contact gives lower forward voltage and allows shallow junction technology
quality	higher
price	lower

Surface-mounted assembly

Both IDs and glass-beads are small and mechanically robust, so they are ideal for automatic assembly. In this respect, however, IDs have the edge since their regular shapes makes them perfect for surface-mounted assembly (SMA), something that's always been a problem for glass-beads. And with SMA, manufacturers can reduce the size, weight and cost of their circuits, increase reliability and improve h.f. performance. They can also speed up production, and increase their product flexibility. Our surface-mounted IDs (Fig.4) form just a part of the wide range of SMDs we now produce.

Fig.4 Surface-mounted IDs – currently still in development

A wide range of technologies for a wide range of applications

IDs also score over glass beads in their ability to incorporate a wider range of diode technologies (Table 2), thanks mainly to the elimination of the alloying stage of glass-bead production, which effectively precludes the use of shallow junction crystals. This allows us to produce our existing glass-bead types all in shallow-junction technology, as well as Schottky and breakover diodes – something that's just not possible with glass-beads. With Schottky diodes, for instance, aluminium diffusion during the alloying phase would damage the precious-metal/silicon layer that forms the Schottky junction. A thicker layer would overcome this, but besides being prohibitively expensive, it would also introduce unacceptably high thermal stresses.

Our current range of IDs (all soft-recovery types) includes general-purpose, double-diffused rectifiers for bridge networks in television and telephony equipment, fast (250 ns) double-diffused rectifiers for SMPS, inverters, converter and scan rectifiers in tv receivers, very-fast (50 ns) epitaxial diodes for h.f. applications, and epitaxial and double-diffused zener diodes for voltage regulation and transient suppression. Schottky diodes will be included in our range in the near future.

In common with our glass-bead diodes, all our IDs are platinum killed for optimum carrier recombination, and hence lower leakage current and lower dissipation. So the diodes can operate at much higher junction temperatures than gold-killed types (175 °C rather than 150 °C), which gives them a greater reserve at normal operating temperatures.

TABLE 2
Chip technology in axially-leaded ID

	availability
general-purpose double diffused	now
fast (<250 ns) double diffused	now
ultra-fast (<100 ns) double diffused	soon
epitaxial (<50 ns)	now
Schottky (<10 ns)	soon
Zener	now
transient suppression	now
break-over	soon

A NEW DEVICE WITH A PROVEN TRACK RECORD

Its close relationship with the glass-bead diode (similar structure and often identical crystals) puts the ID in a very strong and somewhat paradoxical position: that of a completely new device with a revolutionary production process, yet one that effectively has a proven track record based on our long experience of glass-bead diodes.

Just like our glass-bead diodes, IDs are manufactured under CECC-approved conditions. Quality assurance in production follows identical, established lines (Fig.5) with rigorous incoming-goods inspection, on-line testing at every stage, acceptance testing by our Quality Department (environmental, mechanical and endurance testing), and finally CECC quality assessment before packing and release for delivery.

As a final remark we can say without doubt that the ID is indeed a worthy successor to the glass-bead diode, inheriting and improving on all of its good points. This is certainly one instance where we really do get the best of both worlds.

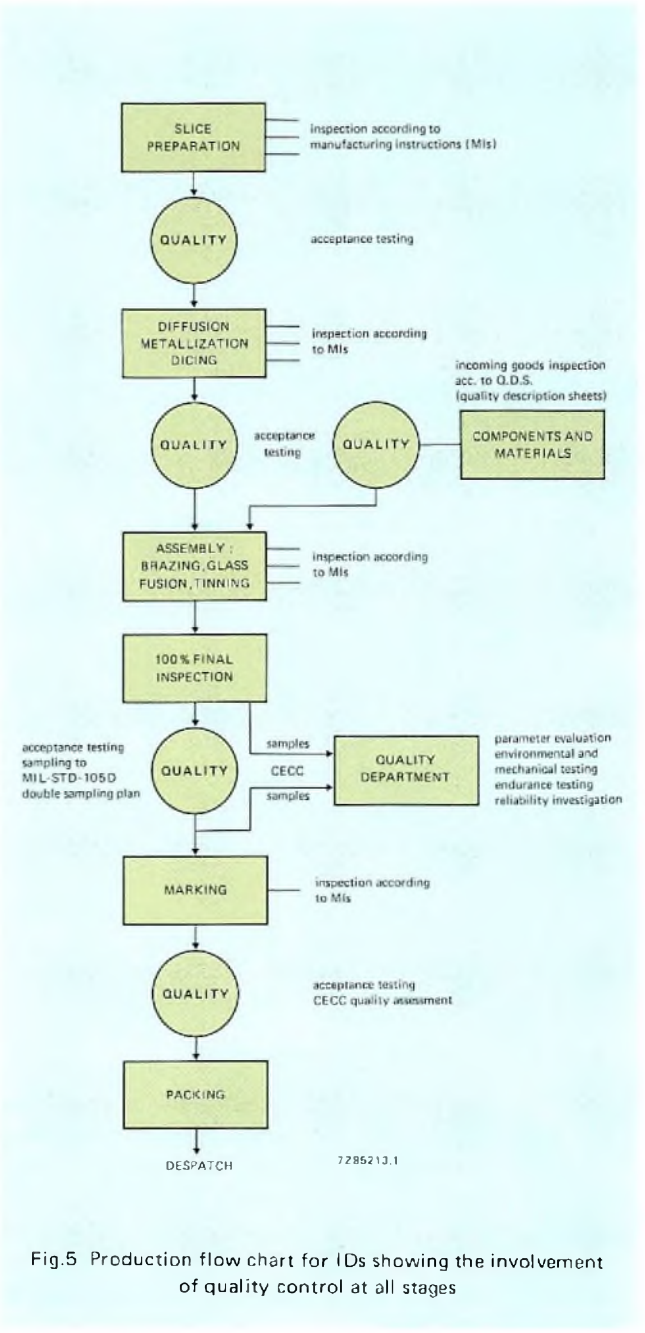


Fig.5 Production flow chart for IDs showing the involvement of quality control at all stages

HCMOS - fast but cool logic ICs

J. EXALTO

Our PC54/74HC/HCT high-speed CMOS (HCMOS) family of logic ICs have the short propagation delays (high speed), formerly only attainable with TTL logic elements, combined with the much lower quiescent power dissipation which is an inherent feature of CMOS circuits. Furthermore, PC54/74 circuits with an HCT suffix also have TTL input switching levels, operate from a 5 V ±10% supply, and are pin-compatible with the most popular LSTTL circuits which they are intended to replace.

For LSTTL circuits operating below about 10 MHz, the most significant part of the total power dissipation is the quiescent power dissipation due to the many bipolar transistors that continuously conduct. With HCMOS circuits however, the converse is true because quiescent power dissipation is only due to leakage currents through reverse-biased junctions and is so low that it's practically negligible compared with the frequency-dependent dynamic power dissipation.

Since the logic functions in most systems only change state during brief periods, the average system frequency is between one or two orders of magnitude lower than the system clock frequency and the ICs therefore only draw quiescent current for most of the time. This means that replacing LSTTL circuits with equivalent HCT circuits, with their much lower quiescent power dissipation, results in a very significant reduction of overall system power dissipation without loss of operating speed.

However, total system power dissipation, is the sum of both the quiescent and the dynamic power dissipation of all the ICs and must be determined and minimised during system design. For LSTTL, where the quiescent power dissipation is the most significant contributor to the total power dissipation, the total power dissipation can be simply derived from the product of V_{CC} and I_{CC} given in the data sheets. For HCMOS circuits however, the dynamic power dissipation which is the most significant part of the total power dissipation is influenced by circuit design. It can't be simply read from the data sheets but must be calculated from the supply voltage, average switching frequency, load capacitance, internal capacitances of the IC, and transient switching currents.

This article explains how our method of specifying HCMOS devices in the data sheets makes it very simple to calculate the quiescent, dynamic and total power dissipation of HCMOS logic circuits.

QUIESCENT POWER DISSIPATION

Quiescent power is dissipated by a device when it is not switching and $V_I = V_{CC}$ or ground. Figure 1(a) will be used to illustrate this power dissipation in HCMOS devices. In the quiescent state, either the PMOS or the NMOS transistor is fully off, and, in theory, no direct MOS transistor channel path exists between V_{CC} and ground. In practice however, thermally generated minority charge-carriers, which are present in all reverse-biased diode junctions, allow a very small leakage current to flow between V_{CC} and ground. This quiescent supply current (I_{CC}) is specified in the published data.

Three factors influence the value of I_{CC}, and therefore the quiescent power dissipation, for a particular device. They are:

Temperature:

increasing temperature causes I_{CC} to increase because the minority charge-carriers in the reverse-biased diode junctions are thermally generated

Device Complexity:

MSI devices dissipate more power than SSI devices because they have a proportionally greater reverse-biased diode junction area.

Supply voltage:

the number of minority charge-carriers is linearly related to reverse junction voltage.

Table 1 shows the JEDEC industry standard for the worst-case I_{CC} in PC54/74HC/HCT high-speed CMOS devices. It shows the effect of temperature and device complexity on I_{CC} at the maximum recommended HC supply voltage $V_{CC} = 6$ V. I_{CC} can be linearly derated for other supply voltages and would be approximately one-third of the value in Table 1 for $V_{CC} = 2$ V. Typical I_{CC} values are well below the maximum specified values.

Another factor which influences quiescent power dissipation is a steady-state input voltage level which may slightly turn-on one of the input transistors shown in Fig.1(a) and yet not fully turn-off the other. This causes a small additional quiescent supply current (I_C) to flow between V_{CC} and ground. The level of I_C depends on the size of the input transistors and is different for each device.

In a system consisting entirely of HC circuits, the additional quiescent supply current I_C is so small that it can be omitted from practical power dissipation calculations. This is because HC I/O levels are fully compatible. The worst-case output levels with $|I_O| = 20\,\mu A$ are $V_O = 0{,}1$ V max. and $V_{OH} = V_{CC} - 0{,}1$ V min., very close to ground and V_{CC} respectively. Figure 2(a) shows that I_C is negligible when these levels are applied to HC inputs because they always turn one of the input transistors fully off.

However, if HC device input levels are held close to the switching threshold (typically $V_{CC}/2$), Fig.2 shows that the additional quiescent supply current I_C becomes much greater than quiescent supply current I_{CC}. This occurs if the mistake is made of driving an HC input from a TTL output. With a minimum TTL V_{OH} of 2,4 V driving an HC input, not only will a logic "1" probably not be recognized, but several milliamps of I_C will flow. To overcome this

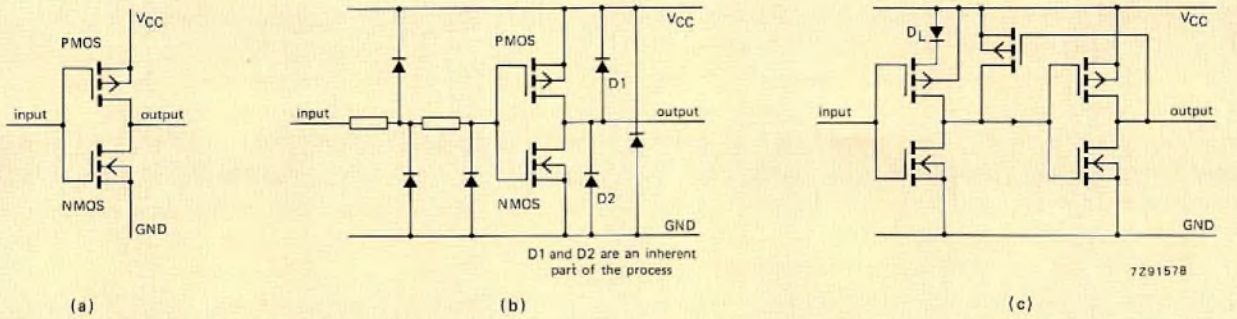


Fig.1 HC inverter. (a) input structure of HC inverter; (b) HC inverter with protection circuitry; (c) input structure of HCT inverter

TABLE 1
JEDEC industry standard for d.c. characteristics of HCMOS circuits
DC characteristics for 54/74HC/HCT

symbol	parameter	T_{amb} (°C) 54/74HC/HCT +25 min.	typ.	max.	74HC/HCT −40 to +85 min.	max.	54HC/HCT −55 to +125 min.	max.	unit	test conditions V_{CC} V *	V_I	other
	quiescent supply current											
I_{CC}	SSI	–	–	2,0		20,0	–	40,0	μA	5,5	V_{CC}	$I_O = 0$
I_{CC}	flip-flops	–	–	4,0		40,0	–	80,0	μA	5,5	or	$I_O = 0$
I_{CC}	MSI	–	–	8,0		80,0	–	160,0	μA	5,5	GND	$I_O = 0$

* for HC, $V_{CC} = 6$ V.

problem, an external pull-up resistor could be used as shown in Fig.3 but the resistor would dissipate significant power because its value would have to be low to maintain switching speed. HCT devices have TTL input switching levels and should therefore be used instead of HC devices whenever it is necessary to interface HCMOS with TTL logic.

Unlike HC devices, HCT devices can be substituted for LSTTL devices and/or mixed with LSTTL, ALSTTL, ASTTL or FAST-TTL family ICs in the same system. Under some conditions, they may dissipate somewhat more quiescent power than HC devices. For example, Fig.2(b) shows that a worst-case TTL V_{OL} of 0,5 V max. is close enough to ground to turn the input NMOS transistor fully off so that I_C is close to zero. However, a worst-case TTL V_{OH} of 2,4 V min. causes some I_C to flow. For this reason, HCT data sheets specify I_C at the worst-case input voltage of $V_{CC} - 2{,}1$ V for V_{CC} ranging from 4,5 V to 5,5 V. It is further specified on a per input pin basis to allow more accurate power dissipation calculations if all the functions within an IC are not being used, or are being driven by different input voltage levels.

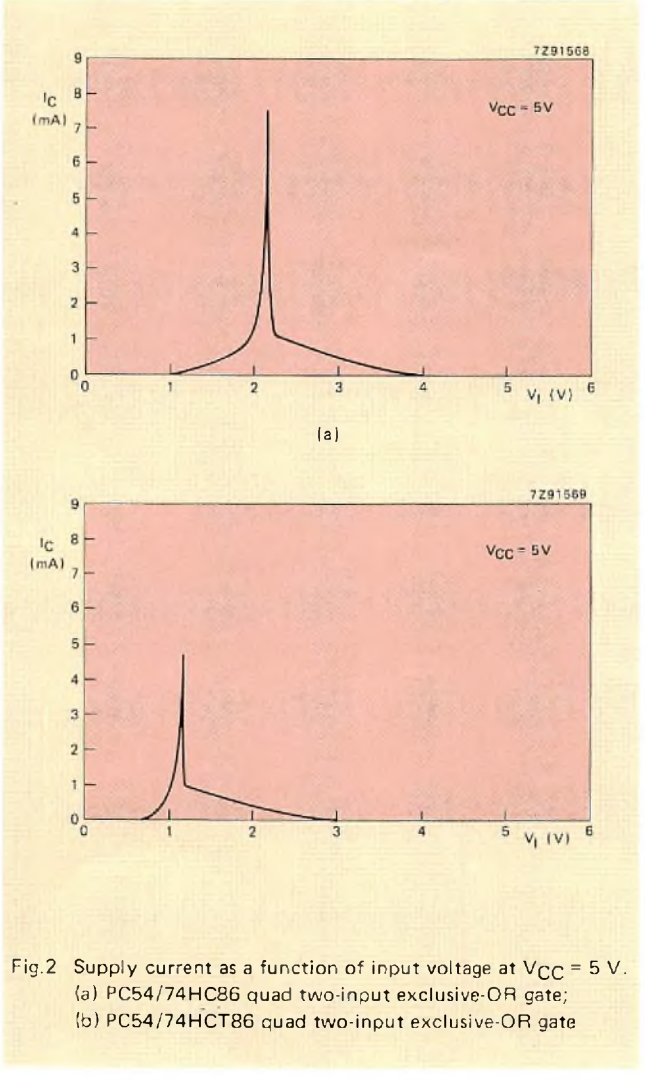


Fig.2 Supply current as a function of input voltage at V_{CC} = 5 V. (a) PC54/74HC86 quad two-input exclusive-OR gate; (b) PC54/74HCT86 quad two-input exclusive-OR gate

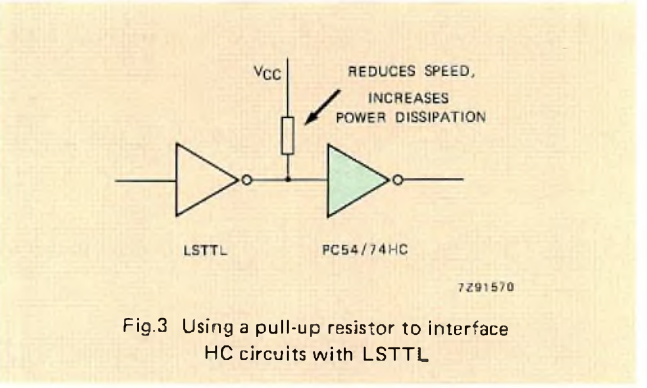


Fig.3 Using a pull-up resistor to interface HC circuits with LSTTL

Our proprietary HCT input structure shown in Fig.1(c) considerably reduces the additional quiescent supply current I_C. The structure is identical to that for HC circuits except for a level-shifting diode between the PMOS transistor and V_{CC}, and the connection of the substrate of the CMOS transistor to V_{CC}. The effect is to reduce the input level switching threshold to 1,4 V instead of $V_{CC}/2$ as is the case with HC circuits. This therefore reduces the additional quiescent current I_{CC} when a TTL minimum HIGH level of 2,4 V is applied to an HCT input by ensuring that the PMOS transistor is fully turned off. Figure 2(b) shows that I_C is negligible when an HCT input is held at a typical TTL HIGH output level (3,4 V) or LOW output level (0,25 V).

Calculating HC quiescent power dissipation

For power-critical applications such as battery-powered equipment, it may be necessary to calculate HC quiescent power dissipation as a standby value of battery drain. It is given by:

$$P_{QHC} = V_{CC}I_{CC} \qquad (1)$$

V_{CC} is dependent upon the particular application, we recommend that a ±10% variation be allowed. I_{CC} at V_{CC} = 6 V is obtained from the data sheet for the particular device. For critical battery-powered applications, the value of I_{CC} at V_{CC} = 6 V can be linearly derated for any desired V_{CC}; for example, at V_{CC} = 2 V, use one-third of the limits shown in Table 1.

Calculating HCT quiescent power dissipation

Assume that an LSTTL device with an output duty factor of 0,5 is switching one of the inputs to one of the gates in a PC54HCT11 (triple 3-input AND gate) with a 5 V supply and an ambient temperature of 25 °C. Quiescent power dissipation is calculated from:

$$P_{QHCT} = V_{CC}(I_{CC} + \delta I_C) \qquad (2)$$

where δ = switching duty factor.

$I_{C\,max}$ is calculated on a unit-load basis from the part of the data sheet reproduced in Table 2:

$I_{C\,max} = 275\,\mu A$ per pin × 1 pin × 0,5 unit-load = 136 μA

TABLE 2
Specification of I_{CC}, I_C and unit load coefficient for PC54/74HCT11 triple 3-input AND gate

symbol	parameter	T_{amb} (°C) 54/74HCT +25 min.	typ.	max.	74HCT −40 to +85 min.	max.	54HCT −55 to +125 min.	max.	unit	test conditions V_{CC} V	V_I	other
I_{CC}	quiescent supply current			2,0		20,0		40,0	μA	5,5	V_{CC} or GND	$I_O = 0$
I_C	additional quiescent supply current per input pin for unit load coefficient is 1 (note 1)		100	275		375		425	μA	4,5 to 5,5	V_{CC} −2,1 V	other inputs at V_{CC} or GND; $I_O = 0$

Note

1. The additional quiescent supply current per input is determined by the I_C unit load, which has to be multiplied by the unit load coefficient as given in the table below.

input	unit load coefficient
nA, nB, nC	0,5

Inserting this current and the values for V_{CC} (5,5 V), $I_{CC} = 2\,\mu A$ from Table 2, and δ (0,5) into equation (2) gives:

$$P_{QHCT} = 5{,}5\ V[2\,\mu A + (0{,}5 \times 136\,\mu A)] = 385\,\mu W.$$

This is only 1,5% of the 25,5 mW maximum quiescent power that would be dissipated by the equivalent LSTTL device. Furthermore, as previously stated, the I_C of 275 μA per input pin quoted in Table 2 for the HCT device is based on a worst-case HIGH input level of $V_{CC} - 2{,}1$ V. In a typical application, the TTL HIGH input level driving the device would be much higher than this, resulting in a reduction of I_C by an order of magnitude.

If all the inputs of an HCT device are driven by HC or equivalent CMOS outputs, the input levels are such that the additional quiescent supply current I_C is so small that it can be omitted from HCT power dissipation calculations. HC quiescent power dissipation equation (1) can then be used to calculate HCT quiescent power dissipation.

DYNAMIC POWER DISSIPATION

Unlike quiescent power dissipation, dynamic power dissipation is calculated in the same manner for both HC and HCT devices. All equations presented here for dynamic power dissipation are therefore applicable to both HC and HCT devices.

Three factors influence the dynamic power dissipation of HCMOS devices. They are load capacitance, internal capacitance and switching transient currents (through-currents of transistor pairs when both transistors momentarily conduct during logic level transitions).

Load capacitance

The first contribution to dynamic power dissipation is caused by the charging and discharging of external capacitive loads. Figure 4 illustrates an HCMOS inverter circuit with a capacitive load and, together with the following equations, will help to illustrate how load capacitance consumes power. The energy dissipated (joules) in charging and discharging the capacitive load is:

$$P_{CL}T = C_L V_{CC}^2 \qquad (3)$$

where $T = 1/f_O$ and C_L = total external load capacitance due to interconnections, driven inputs and any sockets that are used.

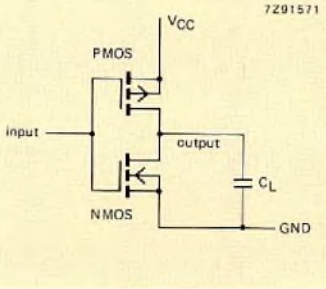


Fig.4 HC inverter driving a capacitive load

The dynamic power dissipation due to capacitive loads is therefore:

$$P_{CL} = C_L V_{CC}^2 f_o \quad (4)$$

Equation (4) is only applicable if all the outputs are switching the same load. If they are not, the equation becomes:

$$P_{CL} = \Sigma(C_L V_{CC}^2 f_o) \quad (5)$$

For multiple output devices, it is important to calculate with the appropriate output frequency. For example, at either output from a flip-flop, $f_o = f_i/2$; for a 6-stage binary ripple counter (type PC54/74HC/HCT4024), f_o is halved for each successive output stage so that $f_o = f_i/64$ for the final output stage.

Internal capacitance

All ICs have internal parasitic capacitance caused by diode junctions, MOS transistor structures, and the aluminimum and polysilicon interconnections. It has the same effect as external capacitive loads, and its magnitude depends on the complexity of the device.

HCMOS devices are manufactured with a self-aligned polysilicon gate process (3 μm gate length) and local oxidation to reduce internal capacitance by minimising gate-to-source and gate-to-drain capacitances. The junction capacitances, which are proportional to junction area, are smaller than those in HE4000B CMOS devices because the diffusions are shallower. Figure 5 shows the location of the capacitances in an HC inverter.

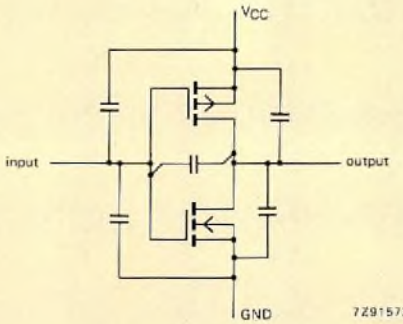


Fig.5 Location of capacitances in an HC inverter.

For power dissipation calculation purposes, the total load caused by internal capacitances and by switching transient currents is defined as a single effective internal no-load power dissipation capacitance C_{PD}. It is defined in the data sheet for each HCMOS device on a 'per function' basis and, where appropriate, it is also separately specified for each different logic function (e.g. gate or flip-flop) within a device. This allows more accurate power dissipation calculations to made if logic functions within the same device are operating at different frequencies.

The published figure for C_{PD} is valid for the worst-case operating mode under typical operating conditions. For example, in the case of a NAND gate, the state of the inputs is assumed to be such that the output is changing state; for a shift register or D-type flip-flop, it is assumed that alternately HIGH/LOW data is being clocked in. The specified value for C_{PD} however is a typical one; nevertheless, some protection will already be built-in to dynamic power dissipation calculations because the assumed worst-case operating modes don't always occur. Although we are not yet prepared to officially publish a maximum value for C_{PD}, a rough guide would be to increase the published figure by 50% for worst-case calculations.

Switching transient currents

The final factor that contributes to the dynamic power dissipation of HCMOS is internal switching transient currents. When the output of a basic HCMOS inverter as shown in Fig.6(a) changes state, either from a logic "1" to a logic "0" or vice-versa, there is a brief period during which both transistors conduct. This creates a temporary low-resistance path between V_{CC} and ground as shown in Fig.6(b). In this transitory state, additional quiescent supply current (I_C) flows and power is dissipated, so input rise and fall times should be kept short. The average value of this transient current increases linearly with increasing switching frequency. In other words, power dissipation due to device switching (like power dissipation due to internal capacitance) increases linearly with increasing switching frequency. However, since it is small compared to the power dissipation due to internal capacitance, its effect is included in the published value of power dissipation capacitance (C_{PD}) which was discussed under the previous heading.

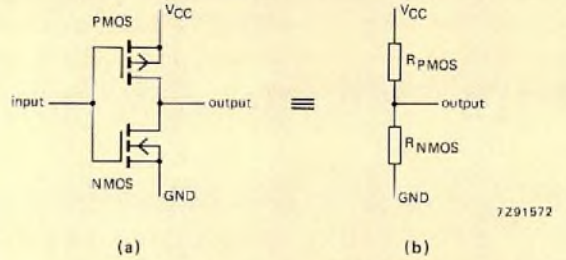


Fig.6 When an HCMOS inverter changes state, both input transistors momentarily conduct.
(a) inverter input structure
(b) equivalent circuit when the input level is between logic levels

Total dynamic power dissipation

Since C_{PD} represents the load imposed by both internal capacitance and switching transient currents, the total dynamic power dissipation due to these factors is:

$$P_{CS} = C_{PD}V_{CC}^2 f_i \quad (6)$$

The total dynamic power dissipation of HCMOS devices is obtained by adding equation (6) to the power dissipation due to the total external capacitive load (equation 5) and is given by:

$$P_D = C_{PD}V_{CC}^2 f_i + \Sigma(C_L V_{CC}^2 f_o) \quad (7)$$

CALCULATING TOTAL POWER DISSIPATION FOR HC AND HCT CIRCUITS

Total HCMOS power dissipation is a summation of the appropriate quiescent and dynamic power dissipation formulae previously described.

For HC devices and HCT devices driven by CMOS levels:

$$P_{tot} = V_{CC}I_{CC} + C_{PD}V_{CC}^2 f_i + \Sigma(C_L V_{CC}^2 f_o) \quad (8)$$

For HCT devices driven by TTL:

$$P_{tot} = V_{CC}(I_{CC} + \delta I_C) + C_{PD}V_{CC}^2 f_i + \Sigma(C_L V_{CC}^2 f_o) \quad (9)$$

POWER DISSIPATION IN OSCILLATORS AND ONE-SHOTS

The information presented so far is only valid for devices switching rapidly between logic levels. Additional quiescent supply current I_C is greater for one-shots, oscillators and gates arranged as oscillators because, in these applications, the input slowly passes through the switching level (typically $V_{CC}/2$) causing flow-through current as shown in Fig.2. Refer to the individual data sheets for the power dissipation of these applications.

POWER DISSIPATION COMPARISON BETWEEN HCMOS, LSTTL AND ALSTTL

In any integrated circuit, there is a balance between speed and power dissipation. LSTTL logic is relatively fast but the quiescent power dissipated by its bipolar circuitry is considerable. ALSTTL improves upon LSTTL by using advanced wafer fabrication techniques and smaller geometries. These improvements increase speed and approximately halve the quiescent power dissipation.

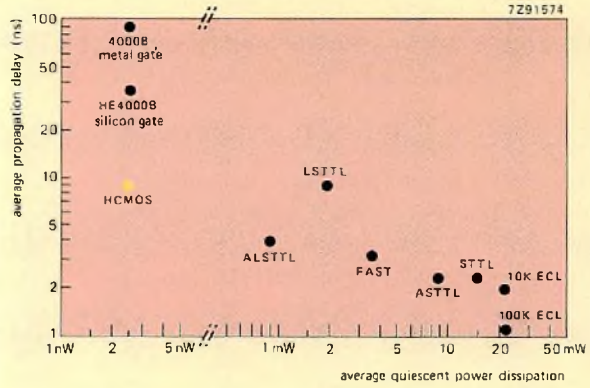


Fig.7 Speed/power products for the most popular logic IC technologies

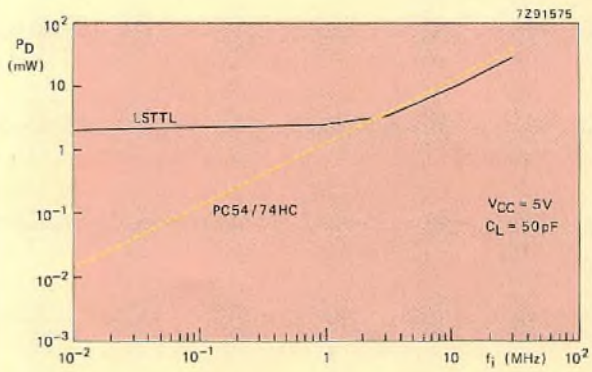


Fig.8 Typical dynamic power dissipation as a function of switching frequency for gates

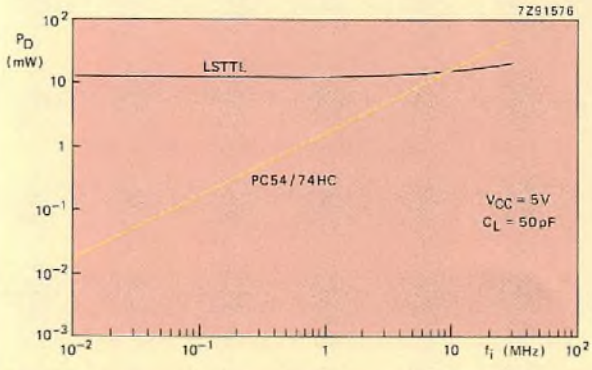


Fig.9 Typical dynamic power dissipation as a function of for a dual flip-flop with one of the flip-flop toggling

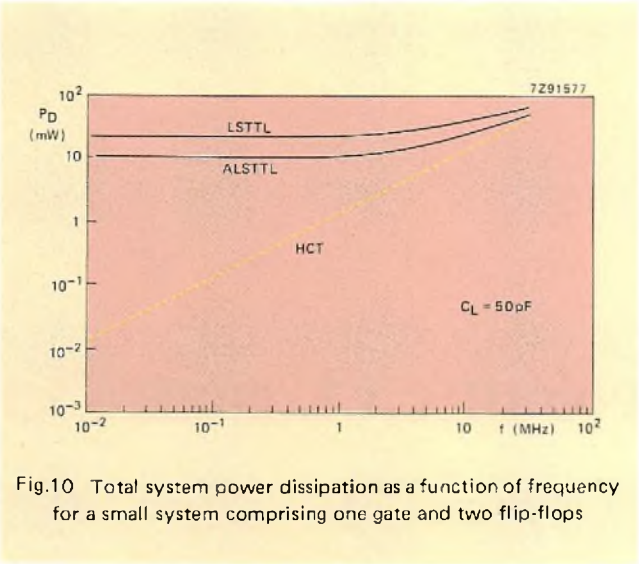


Fig.10 Total system power dissipation as a function of frequency for a small system comprising one gate and two flip-flops

CMOS devices dissipate negligible quiescent power compared with all bipolar TTL logic devices but, until the development of the HCMOS family, CMOS devices were relatively slow. Use of advanced wafer fabrication techniques and smaller geometries has now made it possible for HCMOS to match the speed of LSTTL and yet retain the substantial power savings afforded by CMOS. Figure 7 shows the speed-power products for today's most popular logic IC technologies.

Figures 8 and 9 compare the typical dynamic power dissipation of SSI and MSI for HC, and LSTTL devices. These graphs show that HC devices maintain their power dissipation advantages for switching frequencies up to several MHz. This is because power is only dissipated during switching. The constant, frequency-independent power dissipation exhibited by LSTTL devices is caused by the many bipolar transistors that continuously conduct.

Figures 8 and 9 also show that, as device complexity increases, the frequency at which HCMOS devices dissipate the same amount of power as LSTTL devices also increases. This is because, as LSTTL device complexity increases, there are more resistive paths between V_{CC} and ground which carry more quiescent bias current and thus cause more quiescent power dissipation. HCMOS devices also dissipate more quiescent power as device complexity increases, but the leakage currents which cause it are so small that it can be ignored.

The power dissipation of the different logic IC technologies is translated into total system power as a function of frequency in Fig.10 which is for a small system consisting of one gate and two flip-flops. The graph shows that HCMOS also dissipates substantially less power than LSTTL at the system level.

INFLUENCE OF HCMOS ICs ON APPLICATIONS

The significantly lower power dissipation in an HCMOS logic system, compared with its LSTTL or ALSTTL equivalent, is *the* primary reason why HCMOS devices should be used for new system designs and to replace LSTTL or ALSTTL ICs in many existing designs where power consumption and/or dissipation is a problem.

For new designs, HCMOS is the only suitable family of logic ICs for battery-powered portable personal computers. The use of HCMOS is *the* major trend in personal computers using all CMOS microprocessors, RAMS, ROMS, and peripherals. All CMOS designs can be powered-down to 2 V standby to extend battery life.

For non-portable equipment, the use of HCMOS logic and CMOS LSI is also preferred because it not only reduces power dissipation, but also significantly reduces, in order of priority, cost, size, and weight. Cost reductions stem from major reductions of power supply current and regulation, cooling fans, heatsinks, and copper buses.

An equally powerful motivating force for using HCMOS logic ICs with their lower power dissipation is the inherent and proven increase of component and equipment reliability. Equipment life is considerably extended because IC junction temperatures are much reduced and other components are exposed to lower ambient temperatures.

EUROM - a single-chip colour c.r.t. controller

R. E. F. BUGG

To meet the requirements of CEPT, including the level of the A4 reference model terminal*, we now have available a new single-chip integrated c.r.t. controller type SAA5350 (known as EUROM) to supplement our already extensive and successful range of Prestel and Télétel compatible microcircuits. The SAA5350 is a 40-pin NMOS IC which not only meets all the requirements for the CEPT A4 terminal but also offers a host of additional features that will be appreciated by designers of other types of terminals such as those used for personal computers. Typical of these additional features are 80 characters per row option (colour), multi-page memory, full-field DRCS to allow full-screen colour graphics, broadcast sync, on-screen status row, serial (stack) and parallel attribute storage, cursor and smooth scroll. Nevertheless, as shown in Fig.1, only minimal hardware is required to construct an inexpensive terminal; in the simplest configuration, just a microcontroller and 4 Kbytes of memory (2 Kbytes for page memory and 2 Kbytes for DRCS).

The main features of EUROM are:

- 40/80 character by 1-to-25 row display.
- 512 alphanumeric or graphical characters on-chip.
- Dynamically Redefinable Character Set (DRCS).
- Interfaces with 8/16-bit microprocessors with optional direct memory access.
- On-chip scroll map eliminates the need for massive data transfer when scrolling.
- On-chip colour map RAM, and three gamma-corrected D/A converters which generate RGB outputs, compensated for c.r.t. non-linearity.
- Memory interface capable of supporting multi-page terminals. EUROM can access up to 128 Kbytes of display memory.
- Row 25 may be used for local status messages.
- Zoom to allow the height of any group of rows to be increased for improved legibility.
- Programmable cursor.
- On-chip timing with composite sync output.
- Three synchronisation modes:

  Stand-alone – using an external 6 MHz crystal with on-chip oscillator. This would be used, for example, in stand-alone terminals.

  Simple slave externally synchronised from a source of text such as a teletext IC or another EUROM.

  Phase-locked slave allows synchronisation of the text with video displays, i.e. VCR/VLP video with text overlay (picture in text) as might be used, for example, in the travel industry.

These features, together with the range of attributes defined by the CEPT, give the information provider extensive editorial flexibility. In particular, the dynamically redefinable character set (DRCS)** is a powerful tool which allows each pixel of a character cell to be individually set to permit almost unlimited expansion of the character repertoire and the display of more complex alphabets (Cyrillic, Arabic, Katakana, etc.), simple pictures, company logos, and other symbols (Fig.2).

* see facing panel.

** see panel on page 171.

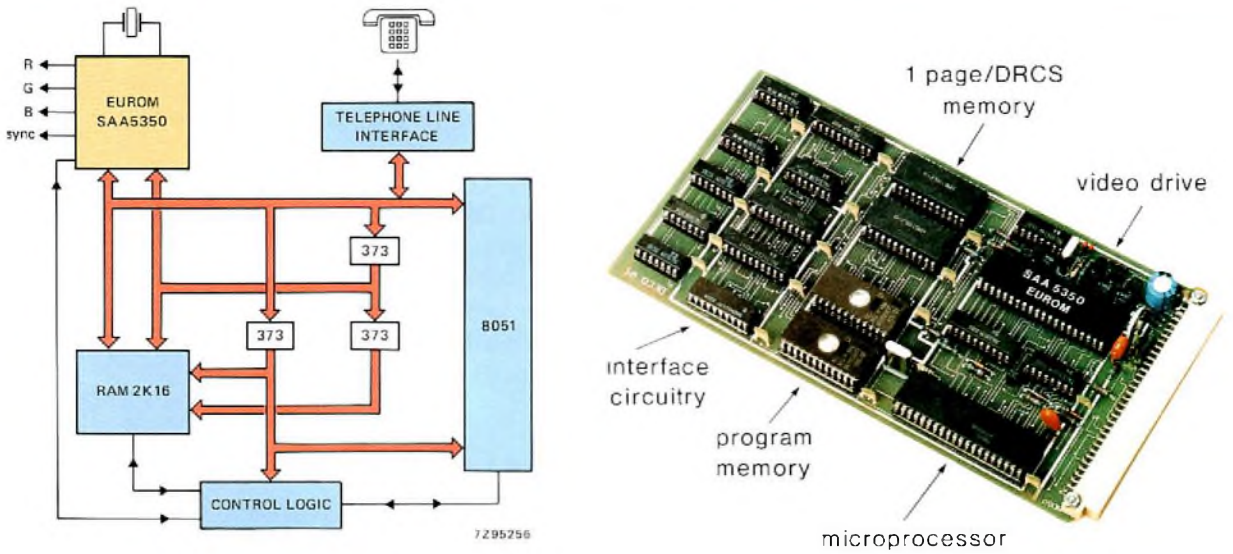


Fig.1 Practical basic videotex decoder using EUROM

PRESENTATION LEVELS DEFINED BY THE CEPT SPECIFICATION FOR VIDEOTEX

Twenty-six countries throughout Europe have now accepted the standard for videotex displays promulgated by the Conférence Européenne des Postes et des Télécommunications (CEPT). The CEPT standard embodies the requirements of the two original videotex systems (UK Prestel and French Télétel), permits full use of existing databases, offers many additional features, and has a sufficiently 'open' coding structure to allow for future expansion. The most advanced videotex terminal reference model defined by the current CEPT specification (Ref.1) is model A4, as used, for example, by the Bildschirmtext service in West Germany. This terminal provides powerful display attributes and has a dynamically redefinable character set (DRCS) to allow almost unlimited extension of the character repertoire, and dynamically redefinable colours to improve graphics and pictorial presentation.

The CEPT standard for videotex incorporates the ISO repertoire of 335 characters which meet the requirements of the 40 Latin-based European languages. It also incorporates a wide range of attributes which give great editorial flexibility to the information provider and result in a high order of legibility for the viewer.

The Videotex standards define four downward-compatible levels of presentation protocol requiring progressively more display memory. The four levels are basic alphamosaic, alpha-DRCS, alphageometric and alphaphotographic.

Basic alphamosaic presentation level

This relatively simple system offers a limited range of characters (94 or 96) and block mosaics, eight standard colours, and a small selection of other attributes. It is typified by the UK Prestel videotex service and by the broadcast teletext services now in operation in twelve European countries. Its simplicity and economy have contributed greatly to the initial success of these services.

Alpha-DRCS presentation level

The 'alpha-DRCS' and alphamosaic presentation levels are compatible in both display standard and transmission coding, but the alpha DRCS level offers non-spacing attributes at the presentation level. It includes the full ISO repertoire of 335 characters, including mathematical symbols. In addition, 151 'smooth' mosaics and other line drawing characters supplement the block mosaics available in the alphamosaic system. Colours are redefinable, with up to 32 on-screen from a repertoire of 4096. The range of attributes includes underline, double-width, and double-size, which allow much greater editorial flexibility.

This level also introduces Dynamically Redefinable Character Sets (DRCS), potentially the most powerful of the new techniques defined by CEPT. DRCS extends the character repertoire and permits sophisticated on-screen graphics to be displayed.

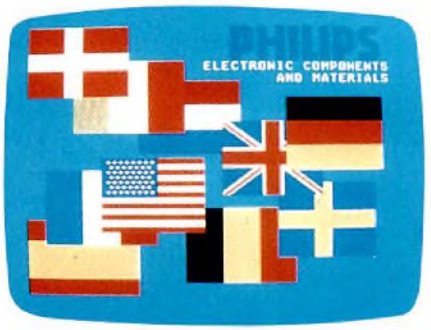
Additional features include the ability to specify full screen colours, extended flash (colour table flash and three-phase flash), freely intermixable double-height and double-width characters, and scrolling to display more information.

Alphageometric presentation level

This system of coding is pixel- rather than character-oriented and allows the transmission not only of alphanumerics but also of geometric primitives — vectors, arcs, polygons, etc.

Alphaphotographic presentation level

Alphaphotographic coding allows an analog picture of arbitrarily good quality to be transmitted using YUV coding (for example, from a video camera). Picture quality is limited only by the terminal resolution, transmission time, and amount of memory available. A practical terminal implementation would include, say, alphamosaics coded to the 'alpha-DRCS' standard.



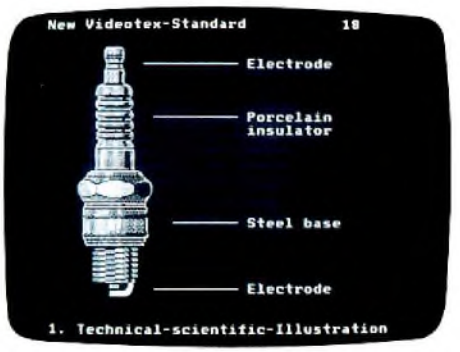


Fig.2 Examples of displays using DRCS

EUROM ARCHITECTURE SIMPLIFIES HARDWARE CONFIGURATION (Fig.3)

For full implementation of all the required functions in EUROM, 32 bits per character location are necessary. This means that to display a full screen of 25 rows of 40 characters, 4 Kbytes of external memory are required – four times the capacity of a basic alphamosaic decoder memory, excluding any DRCS requirements. Attributes in EUROM are therefore coded using a stack architecture which exploits the natural redundancy of normal text by allocating memory dynamically. This effectively halves the required capacity for the external memory as explained in the box on page 174.

Character and attribute data are fetched from the external memory, processed by the row buffer fill logic according to the stack coding scheme (when in Stack Mode), and then fed into one half of the dual display row buffer. Data is fetched from the external memory during one line flyback period (per row), and since time is required to complete the fill, the other half of the dual row buffer is used for display. The row buffers exchange functions on alternate rows; each holds the 40 columns of 32 bits required to define explicitly every character in a row.

The addresser is used for row buffer filling and for fetching screen colours; during the display time it is also used for addressing DRCS characters.

Timing

The timing chain operates from an external 6 MHz clock or the on-chip crystal oscillator. The basic format is 40 characters per row, 24/25 rows per page, 10 video lines per row, but EUROM will also operate with 20/21 rows per page, 12 video lines per row. (This meets the requirements of Ref.2 which embodies the DIN standard on terminal ergonomics, reducing operator fatigue by improving the legibility of displayed text.) The two extra lines per row are added symmetrically, and contain background colour only for ROM-based alphanumeric characters. DRCS characters, block and smooth mosaics, and line drawing characters, however, occupy all 12 lines.

The display is generated according to the 625-line/50 Hz scanning standard, interlaced or non-interlaced. In addition to composite sync for display IC or monitor timebases, a clock output at 1 MHz or 6 MHz is available for driving other videotex devices, and a 12 MHz clock is available for hard-copy dot synchronisation. A Defined Display Area timing signal simplifies the use of peripherals such as a light pen. This signal is nominally coincident with the character dot information. Eurom can also be synchronised from external sources such as a teletext IC or another EUROM. A phase-locked slave sync mode of operation allows EUROM to be included in the phase locked loop of an external sync generating device such as Video Input Processor SAA5230.

Associated with the timing chain is the scroll map, an area of on-chip RAM of 26 bytes. It maps the scan row on to the fetched memory row, allowing the stored page to be displayed in any row order. For each row, a one-byte pointer to the display memory row is stored in the scroll map. This allows scrolling without the need for data transfer to or from side storage. Additional control bits are stored, allowing 1 to 25 rows to be displayed at any location on the screen.

Character generation

EUROM supports eight character tables (Tables 0 to 7), each of nominally 128 characters. Tables 0 to 3 are in on-chip ROM and contain the fixed characters shown in Fig.4.

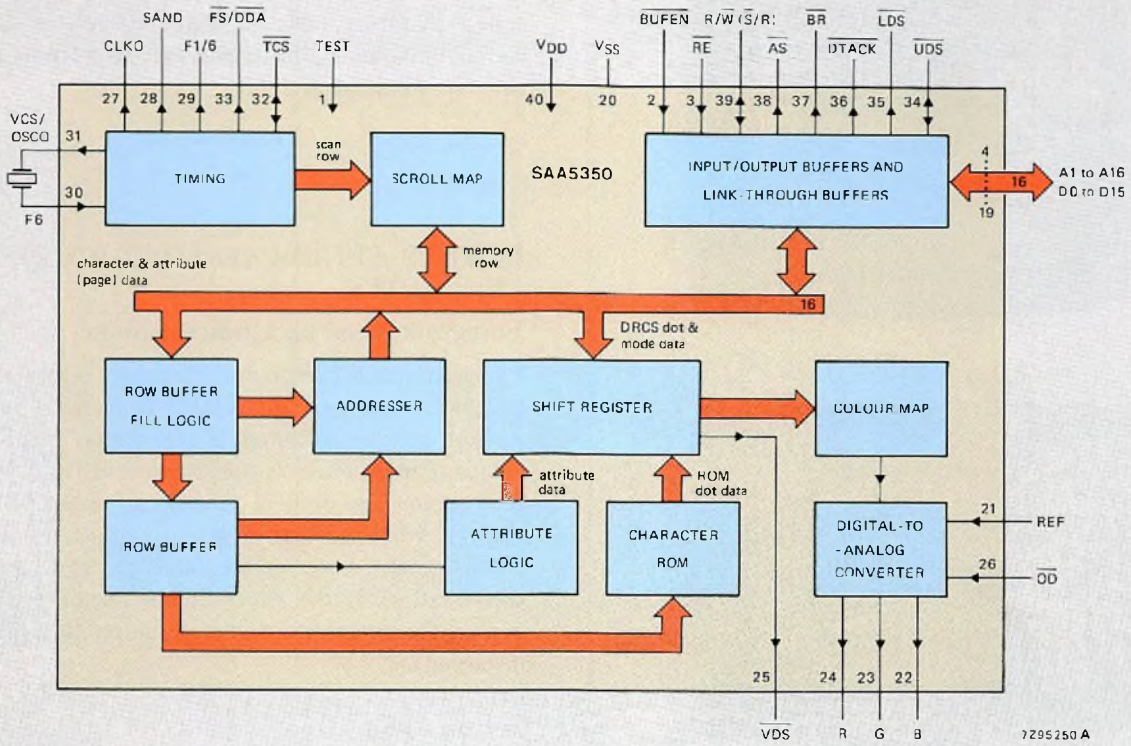


Fig.3 EUROM block diagram

Fig.4 On chip characters

Fig.5 Examples of displays using smooth mosaics and line drawing characters

Tables 4 to 7 are for DRCS and are stored in external RAM. Table 0 contains the 128 most commonly used characters: standard upper- and lower-case letters of the Roman alphabet, numerals, punctuation marks, and the more common accented characters. Table 1 contains lesser used accented characters. Table 2 contains mathematical symbols, a line drawing character set, and accents. Table 3 contains a set of block mosaics for the basic alphamosaic service, together with the new smooth mosaics set. The two sets are complementary and can be combined to create attractive graphic displays such as maps, some examples of which are shown in Fig.5. Although the editorial need for these characters has decreased somewhat with the availability of DRCS, being pre-defined and resident on-chip, they do save transmission time and avoid the delays associated with the downloading of DRCS characters.

Colour map and D/A converters

The colour map RAM contains thirty-two 12-bit colour defining words that are loaded by the microcontroller and read out in three 4-bit groups at pixel rate. Each group is fed to a non-linear (gamma-corrected) D/A converter. The resulting R, G, and B outputs are low impedance with amplitudes controlled by an external reference voltage.

Cursor

A cursor is available in Stack Mode. Its position, character code, character table, foreground colour, background colour, lining and flash attributes are all software programmable via internal register bits.

DISPLAY ATTRIBUTES FOR EACH DISPLAYED CHARACTER

Foreground and background colour

Foreground and background colour are separately coded in five bits, implying a total of 32 colours, 31 of which represent specific locations in the colour map. The 32nd colour is interpreted as transparent but its location is also used during line or field flyback to output blanking level (black). When a pixel is set to transparent, the display colour pointer is set to the value of the screen colour at this location. If the screen colour attribute is also transparent, the underlying tv picture, if any, is unblanked (i.e. displayed).

Screen colour

The screen is notionally divided into 27 areas corresponding to the 25 character rows and the upper and lower border areas. Each of these 27 areas can be set to one of the 31 colour map locations, or transparent.

Flash

Reference 2 defines normal, inverted or colour table flash and six flash rates. Other combinations such as 3-phase flash at 1 Hz are also available with EUROM. If the image of an object is given sequential phases in adjacent character cells, three-phase flash gives the impression of movement along a row. Three-phase flash can also be used with DRCS to produce dynamic displays without the need for continuous transmission. Incremented or decremented flash causes objects to appear to move right or left respectively, in the same way as with 3-phase flash. It avoids the need to specify flash phase explicitly on a per-character basis. In Stack Mode, EUROM automatically supplies the correct phase. This method of specifying object motion reduces transmission time and serial attribute memory utilisation.

Character size

Double height is available in the basic alphamosaic service, with certain restrictions. For example, a single row can contain only top or bottom halves of characters, not both, and so double-height characters cannot be interleaved. In accordance with Ref.1, EUROM provides, double-width and double-size in addition to double-height, with no restrictions on horizontal or vertical interleaving.

DYNAMICALLY REDEFINABLE CHARACTER SETS (DRCS)

In a basic alphamosiac system, the shape of each character is stored as a dot (pixel) pattern within a defined matrix. Since the repertoire of possible characters within the matrix is finite, simple and inexpensive decoders can be designed. The use of DRCS, however, enormously extends the display repertoire. Using DRCS, additional characters can be defined by the information provider, and then used as part of the character set for a specific page or group of pages. The additional characters can be used singly, or as alphanumerics in a different alphabet, or as symbols in timetables, etc. They can also be used in groups to create simple designs such as company logos.

In essence DRCS requires the transmission of the dot pattern for each character matrix and the allocation of a code to that matrix. When transmitting a page containing DRCS, the DRCS data can be transmitted independently of the page information and stored in DRCS RAM. For display, both the fixed and the DRCS character tables are used, depending on the character code stored in the page memory.

The DRCS character cell is based on a 12 pixel horizontal resolution. When operating with 10 lines/row, the following modes, each representing different combinations of horizontal, vertical, and colour resolutions, are available.

10 lines/row

mode	pixel configuration (h x v)	bits/pixel	maximum number of characters/chapter*
1	12 x 10	1	102
2	12 x 10	2	51
3	6 x 10	1	2 x 102
4	6 x 10	2	102
5	6 x 10	4	51
6	6 x 5	2	2 x 102
7	6 x 5	4	102

* one chapter contains 1024 16-bit words

EUROM can also operate in a 6 x 10, 4 bits/pixel mode with a memory organisation more suited to bit-map implementation.

When operating with 12 lines/row, fewer characters are available per chapter.

12 lines/row

mode	pixel configuration	bits/pixel	maximum number of characters/chapter
8	12 x 12	1	85
9	12 x 12	2	42
10	6 x 12	1	2 x 85
11	6 x 12	2	85
12	6 x 12	4	42
13	6 x 6	2	2 x 85
14	6 x 6	4	85

All attributes apply to DRCS in the same manner as to normal characters, but for multi-colour DRCS (that is, the mode with more than one data bit per pixel) the following rules apply:

- The whole character cell is treated as foreground colour
- When the Conceal (or Flash, Invert) attribute is used, the background colour that would otherwise be pertaining is displayed
- The Underline attribute has no effect (the one bit/pixel DRCS modes are underlined as normal alphanumeric characters).

When operating with 2 bits/pixel colour DRCS, the DCLUT (DRCS Colour Look-up Table) is used. This behaves as a small RAM that maps the four combinations of two bits onto any four of the 32 locations in the colour map.

When operating with 4 bits/pixel colour DRCS, the 16 combinations can be taken either from locations 0 to 15 or from locations 16 to 31 of the colour map depending on the value of a register bit. The CEPT/Bildschirmtext application requires that locations 16 to 31 are used.

The physical organisation of EUROM's DRCS memory is 1K16 (1024 16-bit words) for Tables 4 and 5, and a further 1K16 for Tables 6 and 7. In addition to the page memory pointer, two independent memory pointers in EUROM indicate the beginning of each 1K16 'chapter' of DRCS memory.

In Stack Mode, EUROM applies 'size rules' to determine the displayed output when conflicts occur; for example, if the bottom half of a double-height character would occupy the same position as the right-hand half of a double-width character. Part-characters are never displayed.

Lining

A Lining attribute underlines alphanumeric characters or separates mosaics and line drawing characters into six blocks or sub-squares (mosaic separation).

Conceal

A Conceal attribute sets the foreground and background to the same colour until a local reveal function is activated. The local reveal function can be applied either to the whole field or on a row-by-row basis, allowing progressive reveal under the control of the user.

Invert

An Invert attribute exchanges the foreground and background colours and is included for compatibility with Télétel transmission codes. It also applies to Flash, giving anti-phase instead of normal flash.

Box/Window

If the basic frame begins in tv mode, a Box/Window attribute superimposes a box containing text (Foreground

and Background or Screen colours) on the tv picture. It is compatible with the Box function used in the basic alpha-mosaic teletext service. If the basic frame begins in text mode, the attribute provides a window by setting the screen colour to transparent at the character positions where it applies, so that the underlying tv picture is visible at pixels that are not obscured by foreground or background colours.

White button

Various attributes and combinations of attributes can cause on-screen data to be obscured – double height/double width, conceal, foreground and background colours the same, etc. It is a requirement of Ref.2 that this effect can be negated by a user function, colloquially known as the 'white button', which sets all the attributes to their default values without affecting the display memory contents. This function is implemented in EUROM by a microcontroller-defined register bit which is active in Stack, Explicit Fill, and 80 Characters/Row modes.

EUROM ALSO HAS FEATURES NOT SPECIFIED BY THE CEPT STANDARD

Explicit Fill

In Explicit Fill mode, the page memory is not stack coded, and no processing is carried out during the Row Buffer Fill operation. Data from the memory is transferred directly to the row buffer. Since there is then an explicit representation of all the attributes at every character location, there is no limit to the number of attribute-changes on a given row. However, this mode requires a larger amount of external RAM (6 Kbytes/page including DRCS memory). Also, enlarged characters are not checked, so the rules concerning the size attributes must be implemented in software.

80-characters/row

The 80-character mode is also an explicit fill mode without stack coding. No additional circuitry is required; the row buffer is effectively rearranged as eighty 16-bit words, each containing 8 character bits, 3 foreground colour bits, 3 background colour bits, 1 underline bit and 1 flash bit. Dot data is fetched from external memory in the same way that DRCS data is retrieved. All characters are displayed as a 6 x 10 dot matrix, with both 1 and 2 bits/dot modes available. In the 1 bit/dot mode, the external dot memory need only be eight bits wide. When using 10 lines/row, 204 different character matrices may be stored in a 2K8 memory.

The flash mode incorporates colour table flash. For maximum flexibility of display, the foreground and background colours are applied to different areas of the colour map.

Full-field DRCS

For alphageometric and similar applications, a bit-map display is desirable, where each pixel on the screen corresponds to a location in the memory. EUROM implements this indirectly by expanding the DRCS character repertoire so that the entire defined display area can be covered with fully random data.

One chapter (1K16) of DRCS memory can contain data for 51 6 x 10 x 4 (6 pixels wide, 10 pixels high, 4 bits per pixel) characters, sufficient for two complete character columns. If after these two columns have been scanned, the DRCS chapter is incremented to a new area of memory, a further two columns can be covered with different random data.

This method of using 20 contiguous chapters of display memory and incrementing the DRCS chapter latch in synchronism with the horizontal scan forms the basis of the full-field DRCS mode. All DRCS modes, on-chip ROM-based characters, and attributes are still available.

If, for example, a less memory-intensive DRCS mode, such as 12 x 10 x 1, is desired, then the necessary 10 chapters can be addressed by omitting the least significant chapter bit (A11) from the memory address.

MICROPROCESSOR AND RAM INTERFACE

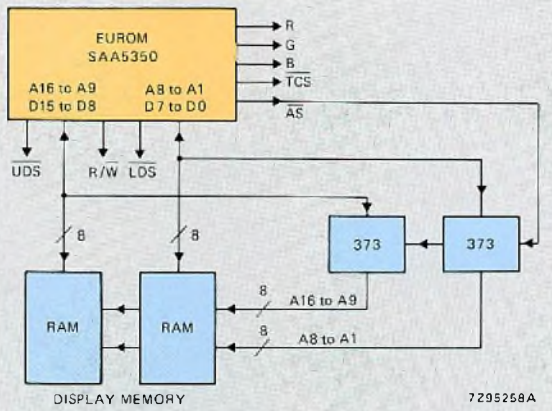
Three types of data transfer take place at the bus interface:

- EUROM fetches data from the display memory
- The microprocessor reads from, or writes to, EUROM's internal register map
- The microprocessor accesses the display memory.

EUROM access to display memory

EUROM accesses the external display memory via a 16-bit multiplexed address and data bus with a 500 ns cycle time. Figure 6 shows a rudimentary RAM interface circuit and bus timing diagram. When EUROM accesses the display memory, its Address Strobe signal $\overline{AS}$ flags the bus cycle and writes the address into the '373 latches. The display RAMs, shown in Fig.9 as two 8-bit blocks, are enabled with Upper Data Strobe, $\overline{UDS}$, and Lower Data Strobe, $\overline{LDS}$, respectively. (EUROM never actually fetches a single byte from memory; $\overline{UDS}$ and $\overline{LDS}$ are always asserted together to fetch a 16-bit word.) The Read/Write control signal, R/$\overline{W}$ is included for completeness although EUROM only reads the display memory.

Although the EUROM data bus is 16 bits wide, the data fetched is often considered to exist in terms of bytes and so the byte addressing convention is important. The standard adopted is that of the 68000 microprocessor where the even-numbered bytes exist on the left or upper (most significant) part of the bus, as shown in Fig.7. The word addresses are numerically the same as the upper byte they contain -- there are no odd-numbered word addresses.



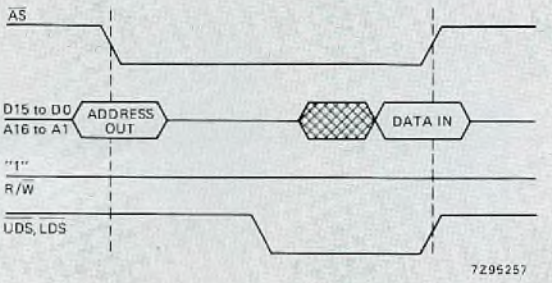


Fig.6 EUROM access to display memory

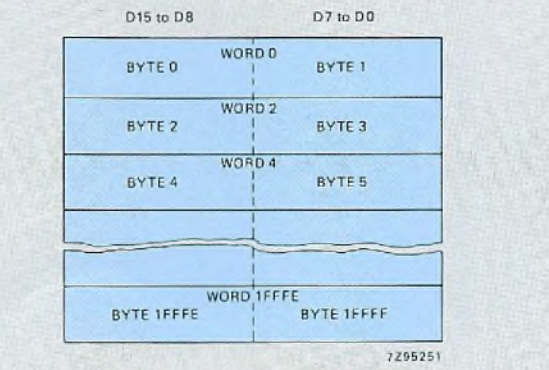


Fig.7 EUROM's display memory word/byte organisation

Warning time

Because EUROM is a real-time display device, it must have direct access to the display memory with priority over the controlling microprocessor or other peripheral devices. To achieve this, EUROM issues a Bus Request ($\overline{BR}$) signal for the duration of the memory access plus a programmable advance warning time to allow the microprocessor to complete its current bus cycle.

In systems where the microprocessor's bus and EUROM's bus are intimately connected, (a 'connected' system), $\overline{BR}$ may be used to suspend all microprocessor activity so that EUROM acts as a dedicated DMA controller. In systems where the two buses are separated by buffers ('disconnected' systems), the $\overline{BR}$ signal may be used either to generate an interrupt or as a directly testable signal. To these ends, the warning time between the assertion of $\overline{BR}$ and the beginning of EUROM's bus activity is programmable from 0 to 23 μs.

Microprocessor access to EUROM's register map

The set of internal registers, when memory-mapped, behave as an 8-bit wide RAM connected to the upper part of EUROM's bus (Fig.8). The control signals $\overline{UDS}$ and R/$\overline{W}$ are reversed to become inputs and the register map is enabled with Register Enable, ($\overline{RE}$). Addresses are input via the lower portion of the bus. A Data Transfer Acknowledge ($\overline{DTACK}$) is also generated to indicate to the microprocessor that data transfer is complete.

Figure 9 shows the main data and address paths used in a 'connected' 68000 interface. The output of the '373 latches are only enabled when the 68000 has yielded the bus in response to Bus Request $\overline{BR}$. When the register map is accessed, data is transferred via the upper part of the bus, and the microprocessor's low-order address is passed to EUROM via the '244 buffer. Simultaneously, the '245 bi-directional buffer disables the signals from the low-order data bus of the 68000.

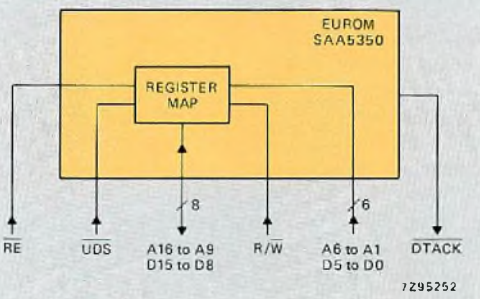


Fig.8 Processor access to EUROM's internal register map

STACK CODING

For full implementation of all the the required functions in EUROM, 32 bits per character location are necessary. This means that to display a full screen of 25 rows of 40 characters, 4 Kbytes of external memory are required – four times the capacity of a basic alphamosaic decoder memory, excluding any DRCS requirements.

7 Bits Character Code
3 Bits Character Table
5 Bits Foreground Colour
5 Bits Background Colour
5 Bits Flash
3 Bits Size (D.Ht., Top/Bot., D. Width)
1 Bit Lining (Underline or Mosaic Separation)
1 Bit Conceal
1 Bit Invert
1 Bit Window/Box
32 Bits per character position

To reduce the amount of memory required, attributes in EUROM are coded using a Stack architecture. Such a system exploits the natural redundancy of normal text by allocating memory dynamically. It allows the external memory to be reduced to 2 Kbytes per screen. This has beneficial side effects; for example, it reduces the memory bandwidth for a given display, reducing the memory speed required or increasing the time available for microprocessor operations.

In the stack coding system used in EUROM, the page memory is divided into character and attribute sections, each organised as 40-byte groups. The 40 character bytes and 40 attribute bytes together make up one displayed row.

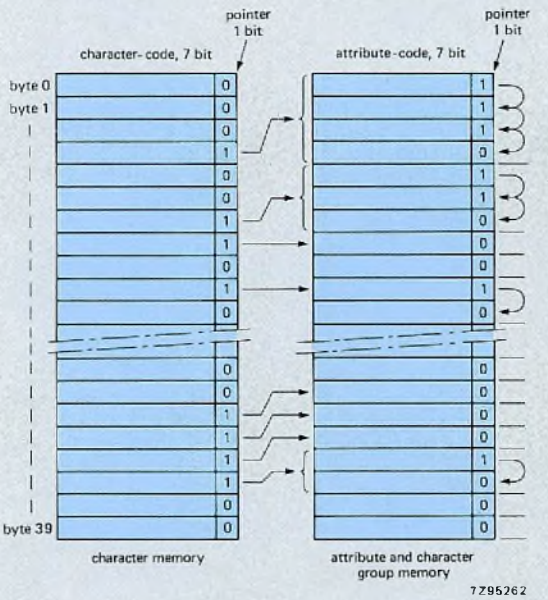
Each 8-bit byte includes a pointer bit. When the pointer bit of a character byte is set, it indicates the presence of one or more attributes set at the same character position. When the pointer bit of an attribute byte is set, it indicates that there are further attributes in that group. At the beginning of a row, default attributes are set, which are then updated by the attribute bytes fetched from the stack.

An example of stack coding is given in the Figure. The first three characters of the row have clear pointer bits. These characters will be taken from the default group of 128 (on-chip) characters, and will be displayed white on black, normal size, not underlined, etc. The fourth character in the row has its pointer bit set, and so the first (or, generally, the next) attribute byte is fetched from memory. This byte also has its pointer bit set, and so the next attribute byte is alse fetched, and so on.

The fourth attribute byte has a clear pointer bit indicating that it is the last in the group. The next character byte is now fetched. The pointer being clear, this character is displayed with the same attributes as those set for the previous one.

The stack system records only the position in a row where attribute-changes occur, with no restriction upon how many attribute-changes apply to any one character. The restriction to 40 attribute-changes in a row has been carefully studied, and not found in practice to be an editorial limitation; it is built in to Ref.2 as a transmission requirement.

The actual coding of attributes, a form of Huffman coding, is shown below.



Stack coding used in EUROM

B7	B6	B5	B4	B3	B2	B1	B0	comments
P	0	0	F4	F3	F2	F1	F0	foreground colour (PIBGR) transparent = 00000
P	0	1	B4	B3	B2	B1	B0	background colour (PIBGR) transparent = 00000
P	1	0	H4	H3	H2	H1	H0	flash
P	1	1	0	L	T2	T1	T0	character table and lock bit
P	1	1	1	0	0	G	D	size. double height and width
P	1	1	1	0	1	0	U	underline (lining)
P	1	1	1	0	1	1	1	invert
P	1	1	1	1	0	0	C	conceal
P	1	1	1	1	0	1	W	window/box
P	1	1	1	1	1	0	H	marked area (not a display attribute)
P	1	1	1	1	1	1	P	protected area (not a display attribute)

The '244 and '245 buffers may be omitted in a 16-bit write-only configuration where the least-significant data byte is interpreted by EUROM as an address. Here it will generally be necessary for the microprocessor to hold a (readable) 'master copy' of EUROM's scroll map contents at some location in main memory.

8-bit microprocessors

Although the control bus is optimised for the 68000, EUROM will operate with a number of widely different industry-standard 8- and 16- (or more) bit microprocessors such as 80(1)88, 68008, 8051, etc. The interfacing of 8-bit microprocessors to the 16-bit wide display memory is simplified by EUROM's on-chip link-through buffer which provides the microprocessor with bidirectional access to the lower (odd byte) half of the display RAM. The link-through buffer is enabled with Buffer Enable, $\overline{BUFEN}$, and its Send/Receive direction is controlled by $\overline{S}/R$ which is physically the same EUROM pin as $R/\overline{W}$.

Figure 10 shows the main data and address paths used in a 'connected' 8-bit microprocessor interface. This is very similar to the 68000 interface but it should be noted that the display memory does not receive A0 as an address, rather A0 (when high) is used as the major enabling signal for $\overline{BUFEN}$.

Disconnected systems

For many applications it may be desirable to 'disconnect' EUROM and the display RAM from the microprocessor and its ROM, RAM, and peripherals. The two parts of the system then operate independently and communicate only when the microprocessor accesses EUROM's register map or the display memory. Figure 11 shows the rudiments of such an 8-bit system; it can be seen that the main data and address paths are essentially the same as those described above, the only difference being the addition of a set of isolating buffers.

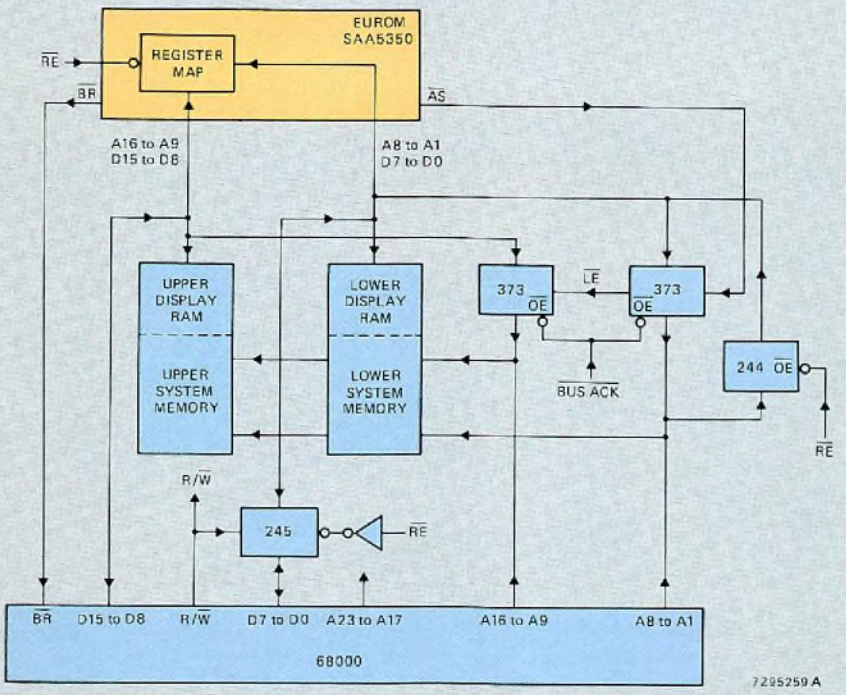


Fig.9 'Connected' 68000 interface

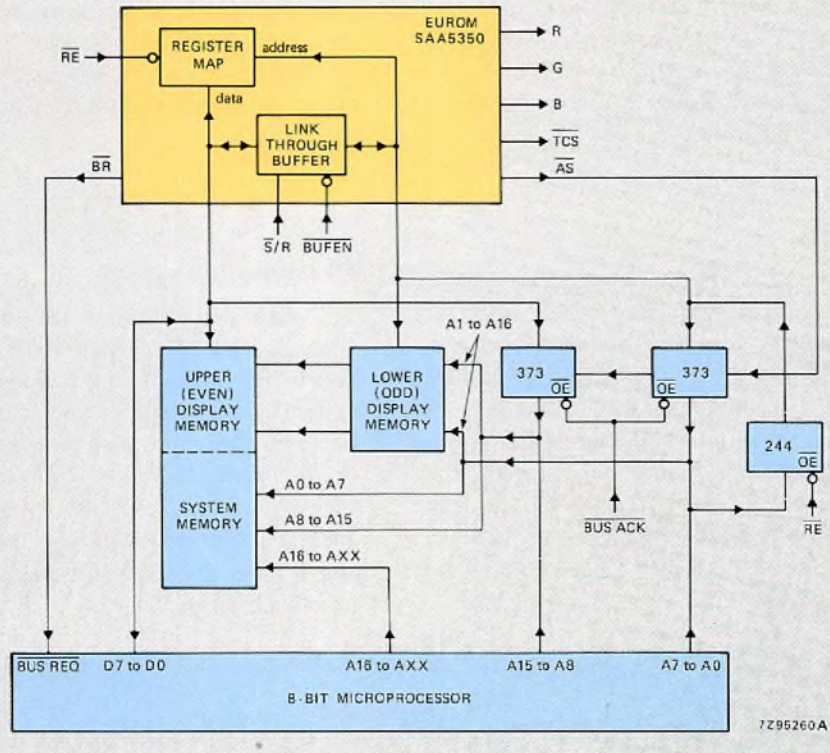


Fig.10 'Connected' 8-bit microprocessor

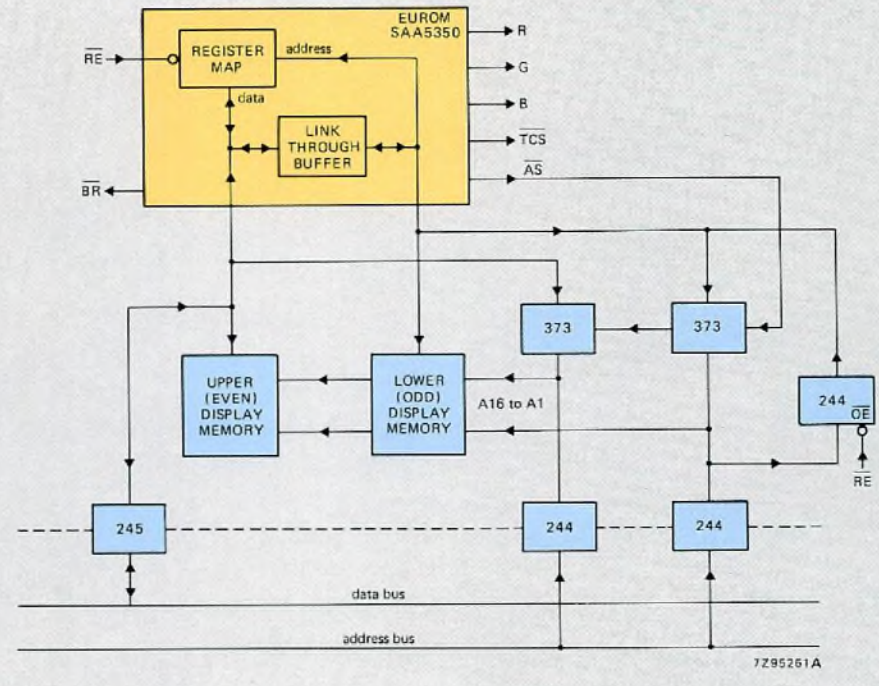


Fig.11 'Disconnected' 8-bit system

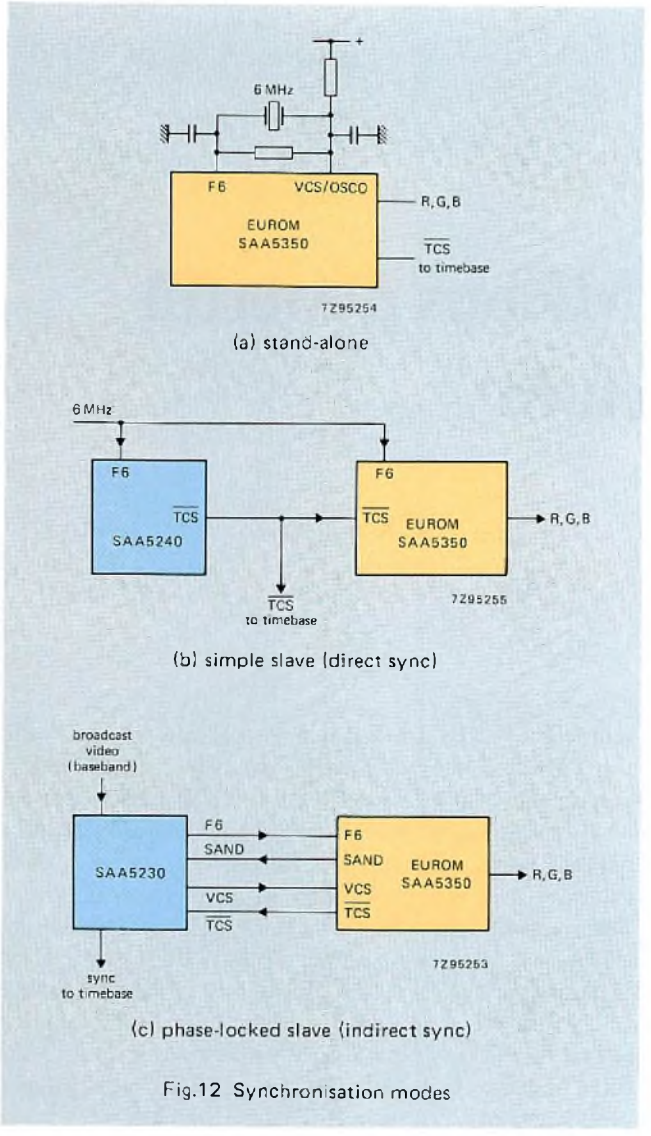


Fig.12 Synchronisation modes

Synchronisation

EUROM has three synchronisation modes. As a stand-alone device (in terminal applications for example), it can output a composite sync signal $\overline{TCS}$ to the display timebase IC or to a monitor. Timing is derived from a 6 MHz on-chip oscillator with an external crystal as shown in Fig.12(a).

EUROM can also sync directly to another device, such as the $\overline{TCS}$ signal from the SAA5240 teletext IC or another EUROM, as shown in Fig.12(b). EUROM's horizontal counter is reset by the falling edge of $\overline{TCS}$. A dead time of 250 ns is built-in to avoid resetting the counter on every tv line, so that screen jitter does not occur. Field synchronisation is achieved with EUROM's internal field sync separator.

The third mode is phase-locked slave operation, as in Fig.12(c). In the SAA5230 video input processor IC, an internal phase-locked VCO provides a 6 MHz clock. When EUROM is active, its horizontal counter is part of the phase control loop; a horizontal reference is fed back to the SAA5230 via the SAND pin and the vertical reference is generated by feeding separated composite sync via the VCS pin into EUROM's field sync separator. In this mode, the display derived from EUROM can sync with that from a tv source or a local VLP (Laservision disc) player, to allow picture-in-text displays, as might be used, for example, in the travel industry.

REFERENCES

1. CEPT Videotex Presentation Layer Data Syntax, Recommendation T/CD 6-2, September 1983.
2. Rahmen bedingungen für Bildschirmtext-Terminals (functional objectives), Deutsche Bundespost, FTZ, February 1983.

Backup support gives VMEbus powerful multiprocessing architecture

CRAIG MACKENNA, RICK MAIN and JOHN BLACK*

Advances in microprocessor design have had a dramatic effect on system performance. As densities have climbed and clocking frequencies increased, designers have been drawn toward implementing multiprocessor systems to take full advantage of computing resources. But the delays associated with shared memory in a multiprocessing environment have limited system performance. To overcome the accessing bottleneck, designers have opted for the 32-bit-bus architecture with multiple paths between boards. The VMEbus with its recently added supplementary buses is an excellent candidate for use by these designers.

The VME specification (a joint effort of Mostek, Motorola, and Signetics/Philips) now defines three separate buses: the VMEbus, the VMXbus, and the VMSbus. Each may be used independently of the others or all three may be used to produce a highly capable multiprocessor system architecture. The VMEbus provides parallel non-multiplexed 8- and 16-bit data and 16- and 24-bit addressing paths on a single connector. It can be increased to full 32-bit data and addressing by using part of a second card connector. The seven interrupt lines and four arbitration lines can be configured according to design requirements.

The VMXbus uses the balance of the second connector to provide another parallel path for data transfer and addressing. Unlike the other two buses, the VMSbus uses message frames on a self-arbitrating serial bus to pass messages among functional modules within a backplane or in separate card racks. Together, these three buses can be combined to maximize multiprocessor throughput.

Figure 1 shows the VMEbus, VMXbus, and VMSbus in a typical multiprocessor configuration. Global resources such as hard disks or local-network interfaces can be accessed over the VMEbus. Dedicated resources such as memory, terminals, and data-acquisition devices can interface with the VMXbus. The VMSbus provides the critical message passing and shared resource arbitration between processors.

The VMEbus provides the global data path and has a backplane that allows up to 20 master and slave boards to exchange data at speeds up to 40 megabytes per second. But even with these high data rates, it is not desirable for all the central processing units on the VMEbus to fetch all their instructions over the global bus. Today's pipelined CPU architectures can drive any bus at near 100% capacity, so getting the best performance with current memory sizes often requires that each CPU be provided with its own private memory bus, the role filled by the VMXbus.

The VMXbus extends the local bus from the processor board to several adjacent card slots where the processor can access additional memory without the arbitration overhead typically encountered over the global VMEbus. It permits a processor board to access random-access memory at speeds in excess of 48 megabytes/s. RAM can also be accessed by one direct-memory-access controller on each VMXbus. Thus a functional module such as a high-speed graphics controller can share RAM with the processor board. Dual-ported memory boards can interface with both the VMXbus and VMEbus.

In addition to the type of data and program transfer usually associated with microcomputer activity, multiple-processor systems require another type of information transfer, often called message routing. The VMSbus handles the special *message routing* traffic of a multiprocessor system. In tightly coupled multiprocessor systems, such as the backplane of the VMEbus, the processors typically share an address space and therefore can pass large amounts of information to one another through their shared memory.

* Motorola Inc. Tempe, Arizona.

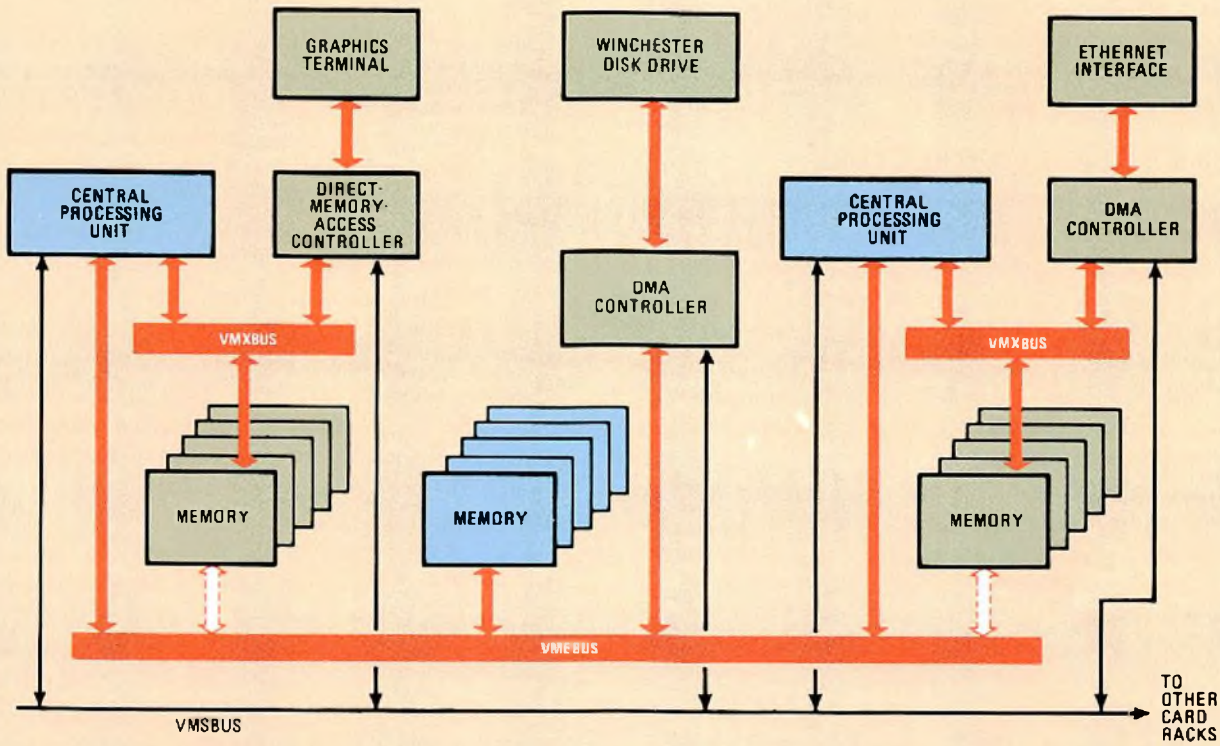


Fig.1 Three buses share the work. The serial VMSbus and the memory expansion VMXbus provide the global VMEbus with the message passing and access to extended local memory required in a 32-bit multiprocessor environment

In loosely coupled systems, processors typically have their own memories. In both types, applications are split into several related processes that are run by several processors simultaneously. Critical points in system operation that must be communicated among processes are often called *events*.

The VMSbus is capable of carrying messages up to 32 bytes long, and is designed to handle the short, urgent event messages typical of both tightly and loosely coupled systems. Since loosely coupled processors typically do not share a common memory, the VMSbus provides a message path that can be used to switch longer messages in packets between processors within a card rack as well as between those in separate card racks.

THE VMEbus

The VMEbus's functional characteristics have not changed since the initial release of Version B in October 1981. The bus provides full 32-bit address and data lines in an asynchronous environment. Because address and data lines are not multiplexed, 8-bit byte, 16-bit word, and 32-bit longword transfers can be completed at rates in excess of 10 MHz.

In addition to byte, word, and longword transfers, a block-transfer feature is implemented in the VMEbus. With the block-transfer mode, the master initiating a transfer can send a single address to a slave. The two functional modules then transfer a sequence of bytes, words, or longwords starting with the specified address, with both modules operating in a pipelined fashion. Whichever method is selected, all data transfers must either be acknowledged or rejected by the slave to ensure that the data was actually transferred.

Concurrent with data transfers, the VMEbus handles arbitration among competing system resources. While a master is using the VMEbus to transfer data, bus requests can be sampled and arbitrated for the next cycle to ensure that the bus is not left idle during arbitration to select the next master.

Different systems require different bus-arbitration algorithms. For example, one system may require that all masters be given equal opportunity to use the bus, while another may require that some masters be given priority over others. The VMEbus arbitration protocol accommodates both approaches with its two methods for bus arbitration: round-robin and priority. When round-robin arbitration is used, each bus request is sampled in a rotating

fashion so that a master on each request level is given an opportunity to use the bus before the previous master regains it. When priority arbitration is used, each bus request is assigned a priority level. Whenever the bus becomes available, all requests are sampled and the bus is granted to the highest-priority request level.

VMEbus INTERRUPT STRUCTURE

Interrupts can also be tailored to meet the requirements of the system designer. The VMEbus has seven interrupt request lines, each of which may be assigned to any processor. Each interrupt request line may be assigned to a separate processor, allowing any interrupt requester to direct its interrupt to the desired processor by choosing the appropriate line. On the other hand, all seven interrupt lines could be assigned to a single processor, which acknowledges and services all interrupts.

Within the card cage, all the elements conform to Eurocard size and connection specifications (Fig.2). Two types of card can be installed in a common rack – the single-height 100-by-160-millimeter card using a single 96-pin DIN connector, and the double-height 233,35-by-160-mm card using one or two DIN connectors. The upper portion of the card cage is called the P1 backplane. The slots reserved for this backplane can accommodate both the single-height cards and the P1 connectors on the double-height cards. The P1 connector provides 24-bit addressing and 16-bit data transfers as well as the supervisory and control functions of the VMEbus; it also includes the VMSbus.

The lower portion of the card rack, called the P2 backplane, can be used only by double-height boards. It must be used when a full 32-bit address or data path is required, or when VMXbus or rear-facing input/output connections are needed. Since only 32 of a P2 connector's 96 pins are needed to expand the VMEbus to 32-bit address and data, the remaining 64 pins can be used for I/O or can be connected to form a VMXbus.

The VMXbus, then, is a subsystem bus formed by the interconnection of up to six boards using the outer rows of the P2 connectors. Mechanically, the subsystem is formed by using the outer two rows of pins on the P2 DIN connectors. Functionally, it is an auxiliary data path used by a master processor to access memory at a slave board through the P2 connector in a fashion and at a speed similar to on-board memory.

The VMXbus has a 32-bit data path and can provide a 16-megabyte address space with its 24 address lines. Two data strobes enable upper or lower byte transfer. The address bus is multiplexed on 12 lines, and the two strobes indicate whether the lower or upper 12 address bits are present on the bus. A line for data-error indication or data-transfer acknowledgement signals completion of a handshake.

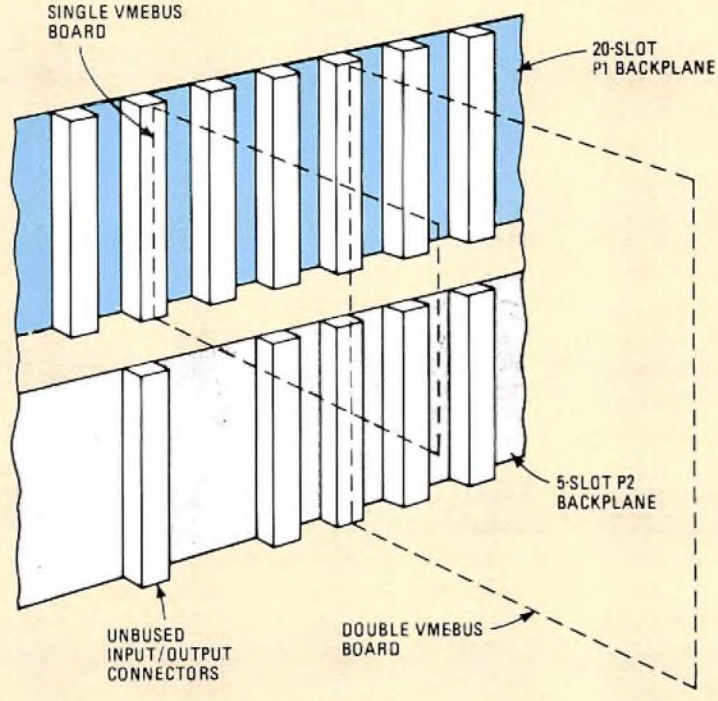


Fig.2 Two backplanes. The P1 backplane supports 24-bit addressing and 16-bit data transfers on the VMEbus. For full 32-bit addressing, the centre row of each P2 connector is bused on the plane. The P2 connectors' outer rows can be used for the VMXbus.

MASTERS AND SLAVES

At most, six boards, occupying six adjacent P2 slots, can be simultaneously attached to one VMXbus. Cards may be designated primary master, secondary master, and slave. A VMXbus must include at least one primary master and one slave. The primary master is usually a CPU that requires more high-speed local memory than can be accommodated on its own board. The primary master controls the VMXbus and manages the secondary master's access to slaves, as well.

To use the VMXbus most effectively, the primary master initiates a bus cycle and places lower address and control signals on the bus before deciding whether the data transfer will take place on the VMXbus or on the VMEbus or will use on-board memory. Before the upper address has been multiplexed onto the bus, the master may decide to abort the VMXbus cycle (called a withdrawn cycle) and instead access on-board or VMEbus resources. If the master does complete the VMXbus cycle, it benefits from the advanced use of the address and control signals.

Secondary master

An optional secondary master can access the slave memory with slightly greater latency times. Slave memory is controlled by the primary master, so the secondary master must first request the use of the VMXbus and receive the grant from the primary master before it is allowed to access a slave. Designating another master as secondary reduces bus-arbitration requirements.

The slaves contain the expansion memory for use by the primary or secondary master. Given the addressing and data-path widths of the VMXbus, it is possible to address 16 million bytes of memory on the slave cards as bytes, words, and longwords.

The transfer timing of the VMXbus fosters close cooperation between masters and slave memory by using both advanced acknowledgement and late data-error timing schemes. Anticipating that the data on the bus will be valid, an acknowledge may be asserted before all checking is completed to counteract the acknowledge overhead in typical CPUs. A data-error signal (DERR*) may follow the acknowledge signal (ACK*) if an error is discovered by the error-detection circuitry. To ensure compatibility between all VMXbus modules, every slave must support ACK*-to-data-valid and SCK*-to-DERR* times of less than a nanosecond.

To improve internal efficiency, a slave uses a technique similar to that employed by the master to improve its VMXbus efficiency. If the time taken to transfer from slave to the master is significant (125 ns) a slave will pass the lower address information through to on-board circuitry (the row address strobe of a dynamic RAM) as soon as VMXbus lower address strobe is asserted. If the upper address is asserted, the RAM can respond quickly because it only needs to strobe the column address; if the cycle is withdrawn, no harm is done.

ATTRIBUTES OF THE VMSbus

The VMSbus provides the event message routing required in both tightly and loosely coupled multiprocessing environments. It also provides a path for transferring the packets of longer messages in a loosely coupled system and can be an efficient alternative to the VMEbus. Under certain conditions, it is useful in configuring a fault-tolerant system, and furnishes such functions as intelligent semaphores, broadcasting, and simultaneous polling.

In a typical VMEbus backplane environment, the VMSbus has a data rate of 3,2 Mbits/s, but the clock rate can be adjusted if the bus length exceeds 50 cm or if the signal path is particularly prone to reflective ringing. To allow for just this type of implementation, the VMSbus's serial protocol includes more error checking and resynchronization provisions than the VMEbus backplane environment requires.

Another accommodation to designers who want to implement the VMSbus outside the VMEbus environment is found in the content and length of message frames. In tightly coupled systems, processors share memory, and the VMSbus frame need contain only the address of the data or parameter block in the shared memory. Restricting the length of VMSbus messages in this way minimizes the access and latency times for very urgent messages. In a loosely coupled system without shared memory, processors can switch 32 bytes of packaged data per frame.

The VMSbus also lends itself to building fault tolerance into the VME system architecture by providing redundancy in the fault-isolation path. When several processors observing each other's actions register a fault on a given board, the faulty board should be either reset or disconnected from the backplane. If the fault disables the system bus, however, then an alternative route must be used to disconnect it. The VMSbus represents an alternative to the system bus; it can be used to reset or disable a board that threatens the system bus or, conversely, the system bus can be used to reset or disable a board that threatens the VMSbus.

VMSbus signals include a clock line, SERCLK, and a data line, SERDAT*. SERCLK is driven by a high-current totem-pole driver at one end of the bus. SERDAT* is an open-collector, low-true signal, which is both driven and sampled by serial-bus modules under SERCLK's control. This permits several functional modules to place data on SERDAT* in the same bit time, with the result being the logical OR of their data.

The SERCLK and SERDAT* waveforms are shown in Fig.3. SERCLK has an asymmetric waveform with four transitions per data bit on SERDAT*. This is done so that the start bit that begins each message frame can be distinguished from *1s* that occur within frames and *0s* that occur within and between frames.

A VMSbus message frame always begins with a start bit, followed by a 25-bit header subframe. These first 26 bits

are sent by a functional module called a header sender. The combination of a header sender and a frame-monitor module is used to initiate a serial-bus operation involving modules that can be elsewhere on the bus.

The header subframe includes two 10-bit address fields called S and R. Either or both of these fields selects modules called header receivers, which may be paired with a data-sender module, a data-receiver module, or both. Alternatively, the header receiver can simply control a status flip-flop. The header subframe does not contain an operation code indicating the function of the frame; that is determined by the S and R codes and the particular header receiver (or receivers) they select.

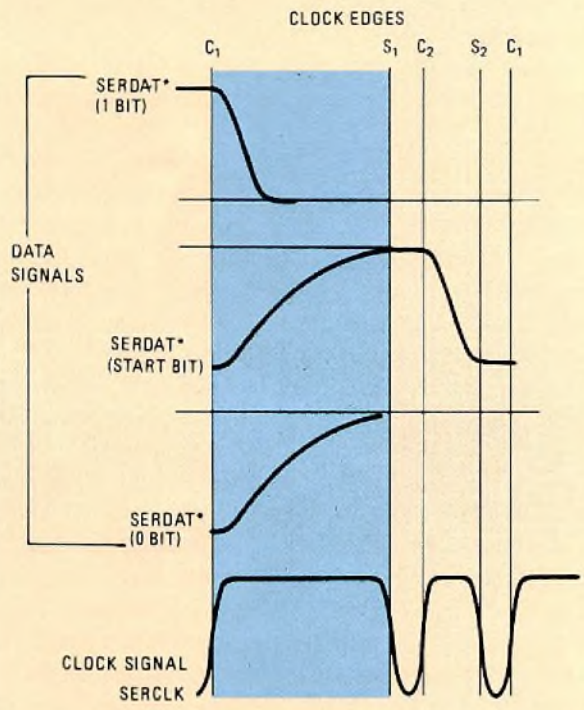


Fig.3 VMSbus timing. SERCLK provides the clock for all modules and edges C_1, S_1, C_2, and S_2. Modules can change data on SERDAT* at C_1 and C_2 and sample data at S_1 and S_2 in the same bit time because the clock's asymmetric waveform distinguishes between bits

CONTROLLING THE BUS

Arbitration takes place as the frame is transmitted. There are no separate signal lines or time periods in which serial-bus modules acquire control of the VMSbus. If a frame is already in progress when on-board logic signals a header sender to initiate a frame, the sender waits until the end of the frame. When a frame is not in progress, several header senders can simultaneously place a start bit on the bus. As they send the following header subframe, each drives SERDAT* low for a 1 (or releases it for a 0) on the C_1 edge of SERCLK in Fig.3, and then samples SERDAT* at the subsequent S_1 edge. The three possible cases are:

- if SERDAT* is released (to high) at C_1 and sampled high at S_1, then the header sender module proceeds to the next bit
- if SERDAT* is driven low at C_1 and sampled low at S_1, then the module proceeds to the next bit
- if SERDAT* is released (to high) at C_1 and sampled low at S_1, then the header sender has lost the arbitration for the serial bus. It then stops sending, waits for the frame to complete, and tries again.

Arbitration handled in this manner ensures that one of several contending messages will always be transmitted; no situation exists in which all contenders release the bus. If contending header subframes are viewed in terms of their binary values, arbitration is won by the one with the highest binary value. Should two header senders send out exactly the same header subframe, both will win the arbitration, since both are attempting to command identical operations. Operations that must be initiated by only one header sender are handled by ensuring that the header subframes are unique.

Following the header is a 3-bit frame-type code. Since the function of a message frame is determined by the S and R addresses in the header, the frame type indicates the general function that will result. Frame-type codes fall into three categories:

- 000: no data sender is selected. Frames with this code are used only to set or reset a flip-flop paired with header receivers that are selected by the S or R fields or both. This type of operation may change simple status outputs or semaphores.
- 001-110: a data sender is selected by the S field. The function of the frame is thus to send data to a data receiver selected by the R field. The value sent by the data sender indicates whether 1, 2, 4, 8, 16, or 32 bytes will be sent.
- 111: one or more of the selected bus modules is not ready for the frame. Examples include an attempt to set a semaphore that is already set, to send data to a data receiver that still contains data from a previous frame, or to send data from a data sender that has nothing to send.

Figure 4 shows the three general types of frame. In type 111, there is no frame status field. In type 001-110, the frame type is followed by the data field, which is followed by the frame status. In type 000, the frame status directly follows the frame type.

The 3-bit frame status is used to indicate exceptional conditions resulting from S and R module selection. It can indicate, for example, that a data sender has been selected without a corresponding receiver (or vice versa), or that a

module has been selected by the S and R fields. It can also indicate that the data sender has sent a data field longer than the data receiver can handle, or that two or more senders transmitting identical data (in fault-tolerant systems for example) disagree on the amount of data to be sent.

The jam bit ends every frame and guards against frame desynchronization in VMSbus operation. Usually it is high (0). If any frame-monitor module detects a start bit within a frame, it immediately jams, or stops, SERDAT* with 512 low bits *(1s)*. Since the longest possible frame is 286 bits, 512 consecutive *1s* are certain to affect the jam bit of the frame in progress. All modules ignore a frame that ends in a jam bit of 1. The jam sequence also resynchronizes all header senders, which restart their transmissions after the sequence is over.

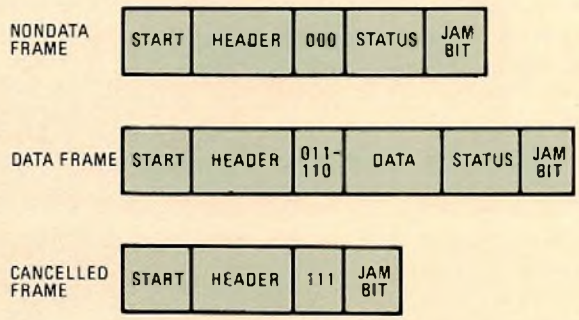


Fig.4 VMSbus frame structures. The three kinds of frame on the VMSbus handle all protocol requirements. All frames start with a 26-bit header subframe followed by a 3-bit frame-type code and end with a jam bit. The frame-type code determines the final frame format

The first 3 bits of every header subframe are the message-priority field and serve to subdivide messages into eight priorities for serial-bus arbitration. The field can be programmed to indicate the initial or incoming priority of a message into the serial-bus subsystem. For example, the highest level (111) can be reserved for communicating events and conditions that endanger system operation, integrity, or human safety. It can also be used to guarantee "fairness" of the serial-bus arbitration among competing frames. For example, if a header sender loses a bus arbitration, it can increment the priority field by 1 for the next retry, perhaps up to a defined upper limit, say 110. On the other hand, if a competing frame wins the arbitration but cannot transmit because one of the receiving modules is not ready, its priority can be dropped (that is, to 000). While less effective than message-level handshaking described below, this feature can minimize the interference of repetitively cancelled frames with more productive bus traffic.

The basic modules included in the VMSbus specification can be combined in numerous ways to implement various functions and operations on the serial bus. Figure 5 shows the simplest VMSbus configuration used to control a physical signal on board. When the header sender sends a header subframe with the header receiver's selection code in the S field, the associated flip-flop is set. When the header sender sends the same code in the R field, the flip-flop is reset. If the header sender is on a card with a processor in a fault-tolerant system, this configuration can be used to disconnect a malfunctioning board from the VMEbus. Generally, this configuration can be regarded as implementing a virtual signal line, whereby an input signal to the header sender controls the flip-flop output on the other board.

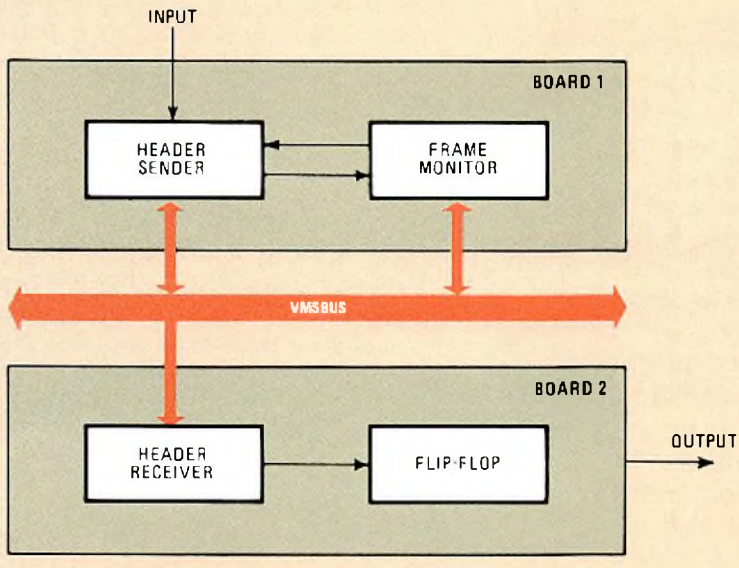


Fig.5 Signal control on the VMSbus. The VMEbus can be used to control a physical signal on another board. A virtual signal line can be established between boards without the constraints that are encountered when dedicated lines are used in a backplane

VMSbus CONFIGURATIONS

Another VMSbus configuration implements an intelligent semaphore on a processor board (Fig.6). A processor's executive software signals the on-board serial-bus hardware when it needs the system resource associated with the semaphore. If the semaphore is already set, the on-board copy inhibits the header sender from sending the header until the semaphore is cleared. The header calls out the semaphore address in the S field and a unique requester code in the R field. The requester code ensures that only one processor at a time will set the semaphore.

When the hardware has set the semaphore, it signals the processor by an interrupt or status read. The processor then uses the associated system resource. When it is finished, the processor signals the header sender, which sends a frame to clear all copies of the semaphore. This allows other processors to contend for the semaphore once again.

Multilevel addressing is useful in systems using the VMSbus's broadcast and simultaneous poll facilities. In the configuration shown in Fig.7, processors can be addressed in any of three ways: message to all processors (broadcast mode); a particular set of processors (group mode); or an individual processor (specific mode).

POLLING OPTIONS

When using multilevel addressing with the simultaneous-polling feature, say to find out the status of a set of sensors, it is possible to poll all sensors, the sensors in a particular zone, or a specific sensor. The concept can also be used with non-data functions such as flip-flop set and clear signals. For example, in a fault-tolerant system, if a VMEbus malfunction is detected, a frame could be sent to disconnect all VMEbus masters, a particular master, or all boards from a particular supplier.

Since data receivers can vary greatly in the speed with which they process, dispose, and become ready for new data, the VMSbus provides for message-level handshaking to prevent cancelled frames from wasting bus time. For example, one data receiver may always be ready for data; another may dispose of data relatively rapidly, say in the time required to transmit a few bits on the VMEbus; while a third may have extensive processing to do before a new frame can be accepted.

The VMSbus uses frame cancellation (frame type 111) to avoid overrun problems at relatively slow receivers. Yet by itself this feature is insufficient since the typical action of a header sender, when a frame is cancelled, is

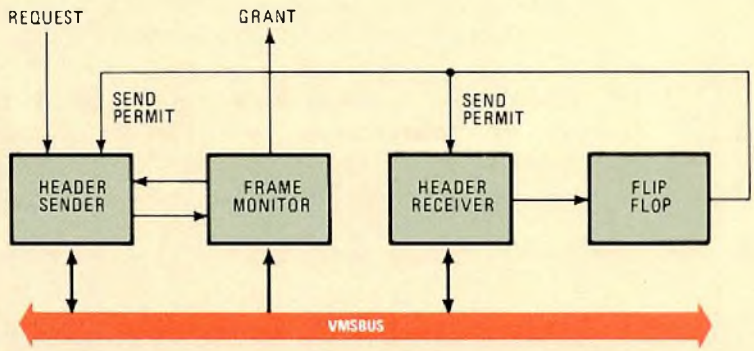


Fig.6 Semaphores using the VMSbus. In a multiprocessor environment, VMEbus semaphores can prevent two processors from attempting to simultaneously access a shared hardware device, such as a printer, or a shared resource, such as a data file

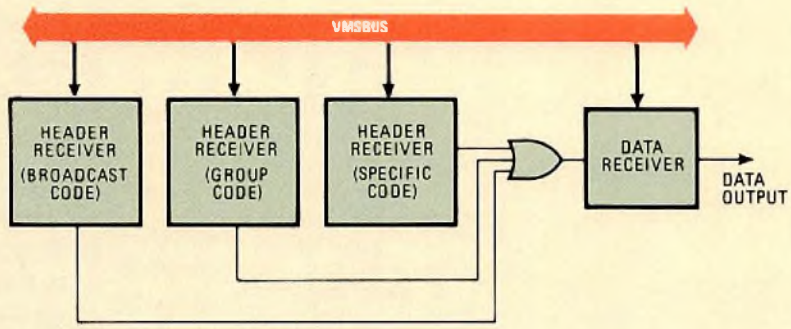


Fig.7 Multilevel VMSbus addressing. Multiple address codes for the same module permit selection of a particular module or a group of modules. Here, data can be sent to a single receiver, a group of receivers, or all receivers along the bus

simply to retry the transmission. If the receiver takes a long time to dispose of data, the VMSbus can become heavily loaded with useless cancelled frames. To avoid this overburdening, a response frame is used to indicate when the listener has disposed of data.

Figure 8 shows a system that uses a response frame. Here, a controller function is included with the listener function, and a status flip-flop is included with the originating controller and talker. The same frame that transmits data to the listener also sets the flip-flop. The controller-talker then refrains from sending further data frames. When the listener is ready for new data, it uses its controller to send a frame that clears the flip-flop and thus enables the next data frame. The response frame could itself be a data frame that includes result or status information.

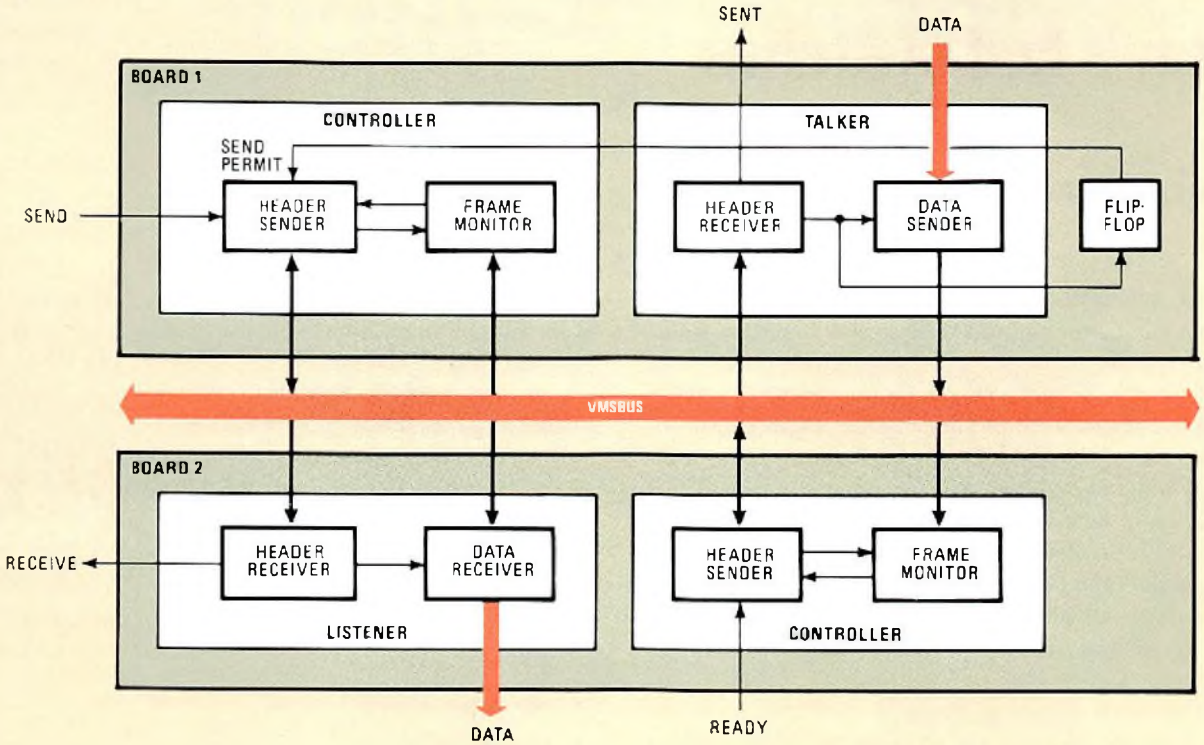


Fig.8 Message-level handshaking on the VMSbus. The controller pair at the upper left initiates a data transfer from the talker to the listener and also sets the flip-flop at the upper-right. When the listener's on-board hardware is ready for new data, it uses its controller (lower right) to send a response frame to clear the flip-flop, which allows the next data transfer and prevents loading the bus with cancelled frames

ACKNOWLEDGEMENT

This article originally appeared in the March 22 issue of Electronics, copyright 1984, McGraw-Hill Inc.; permission to reprint is gratefully acknowledged.

Epitaxial diodes - rectifiers compatible with today's fast switches

ARTHUR WOODWORTH

For many applications, for example, switching inverters, converters and motor controllers, designers are interested in increasing switching frequency. This improves efficiency and reduces the volume and weight of equipment. To help them achieve this, there are a number of excellent fast active switching devices available today including bipolar transistors, darlingtons, GTOs, and power MOSFETs. Until now, however, there have been no really suitable rectifiers for use with these switches, and often in practice it has not been possible to utilise their full switching performance. Our new range of ultra-fast epitaxial diodes now enables full use to be made of fast switches, even at medium and high voltage.

The new range of diodes, shown in the Table, incorporates the principal advantages of epitaxial technology, that is fast switching with low forward voltage drop and minimal r.f.i. Furthermore, improvements in design have extended the operating voltage range up to 800 V, and with careful production control, excessive on-state losses have been avoided.

In short, the performance of our epitaxial diodes is now compatible with the fastest switches. The following features are particularly noteworthy:

- up to 800 V V_{RRM}
- reverse recovery times of 25 ns at 200 V and 100 ns at 800 V
- low, stable leakage current
- minimal r.f.i.

Ultra-fast epitaxial rectifier range

$I_{F(AV)}$ (A)	V_{RRM} 50/200 V (t_{rr} 25–50 ns)	V_{RRM} 300/500 V (t_{rr} 50–60 ns)	V_{RRM} 600/800 V (t_{rr} 100 ns)	outline	style
2	BYV27			SOD-57	single
3,5	BYV28			SOD-64	single
2 × 5	BYQ28	BYT28	BYR28	TO-220	dual
7,5	BYW29	BYV29	BYR29	TO-220	single
14	BYV79	BYT79	BYT79*	TO-220	single
2 × 10	BYV32	BYV34	BYR34*	TO-220	dual
2 × 15	BYV42			TO-220	dual
2 × 15	BYV72	BYV74*		SOT-93	dual
14	BYW30	BYV30		DO-4	single
28	BYW31	BYV31*		DO-4	single
40	BYW92	BYV92		DO-5	single
60	BYW93	BYV93*		DO-5	single
80	BYW94			DO-5	single

* in development.

CIRCUIT CONSIDERATIONS

Some of the problems generated by rectifier imperfections can be better understood by considering the case of a fast switch in series with an inductive load (which could be a motor) with an energy recovery rectifier connected across the load. This is a very common configuration, occuring in d.c. choppers (Fig.1(a)), switching regulators (Fig.1(b)), and many pulse-width modulated a.c. and d.c. controllers (Fig.1(c)).

DATA PARAMETERS

Reverse recovery

The speed of a rectifier can be measured by several different parameters. These are indicated in Fig.A, which shows reverse recovery current as a function of time. One parameter is the reverse recovery time t_{rr}. This is the time from the instant the reverse current starts, to the instant it falls (recovers) to a fixed proportion (usually 10%) of its peak value (I_{RRM}). Alternatively, stored charge Q_s, represented by the area under the current/time curve, can be used as a parameter. Some manufacturers use peak reverse current I_{RRM} as a measure of switching speed.

Whichever parameter is used, it is most important, when comparing one manufacturer's device with another, to ensure that the conditions of measurement are the same. The conditions which need to be specified are:

- steady-state forward current (I_F); high current increases both recovery time and stored charge
- reverse-bias voltage (V_R); low reverse voltage increases both recovery time and stored charge
- rate of fall of forward current ($-dI_F/dt$); high rates of fall reduce recovery time but increase stored charge
- junction temperature (T_j); high temperatures increase both recovery time and stored charge.

The shape of the reverse-recovery current waveform is also important. If the decaying current contains a fast-changing portion, (the 'snap-off' effect shown in Fig.B), this can have serious consequences in the circuit, for example generating conducted and radiated r.f.i., as well as voltage spikes across circuit inductances. With snap-off devices, sufficient voltage can be generated across even stray inductances to damage the rectifier and associated components. This is a problem often encountered with other types of diode, notably PIN diodes, and it can only be alleviated by the addition of snubber components.

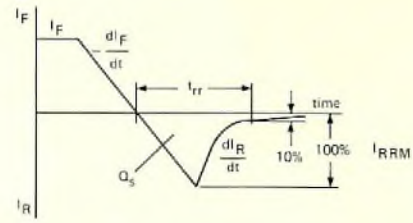


Fig.A Reverse recovery

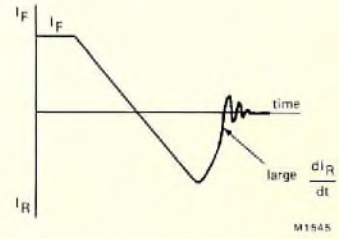


Fig.B Snap-off effect

Forward recovery

Forward recovery (Fig.C), caused by lack of minority carriers, occurs in p-n rectifiers at turn-on. At the instant forward bias is applied, there are no carriers present at the junction, hence the initial forward voltage drop may be large. As the charge builds up and conductivity modulation takes place, the forward voltage drop rapidly falls to the steady-state value. The peak value of forward voltage drop is known as the forward recovery voltage (V_{fr}). The time from the instant the current reaches 10% of its steady-state value to the time the forward voltage drop falls to within 10% of its final steady-state value is known as the forward recovery time (t_{fr}).

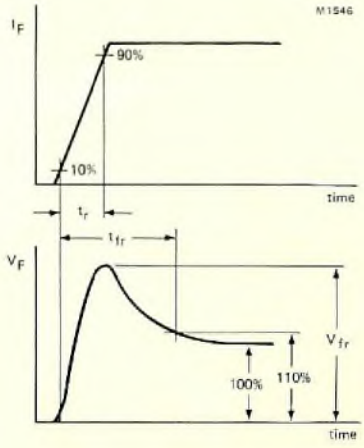


Fig.C Forward recovery

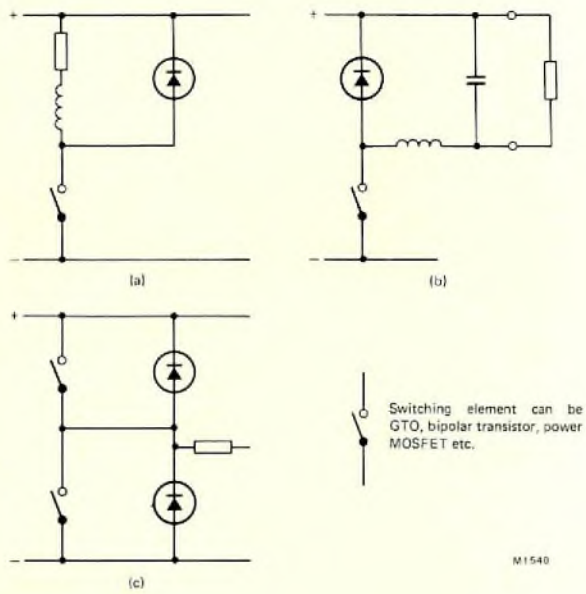


Fig.1 Common circuit configurations containing both fast switches and diodes:
(a) d.c. chopper;
(b) switching regulator
(c) PWM a.c. controller (part)

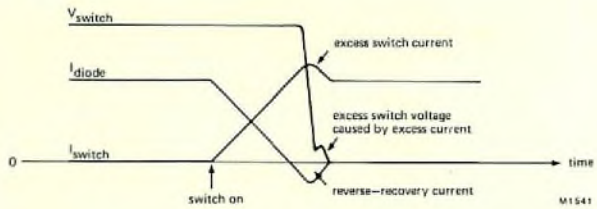


Fig.2 Current commutation from rectifier to switch with an inductive load

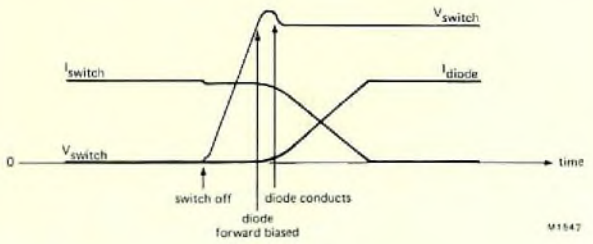


Fig.3 Current commutation from switch to rectifier

Current commutation from rectifier to switch

As the switch is turned on, the rectifier current falls, switch current rises, and the voltage across the switch rapidly falls at a rate determined by circuit capacitance (Fig.2). As the rectifier goes into reverse recovery, the switch current overshoots its steady-state value producing a switch voltage 'spike'. This results in increased turn-on dissipation in the switch which may, in an extreme case, cause switch failure (by exceeding the maximum current rating or dissipation), failure of the rectifier, or even both.

One way of reducing the reverse recovery current is to slow down the rate of change of current, either by slowing down the switch (for example at the gate of a power MOSFET) or by adding series inductance. In the latter case, an additional rectifier and resistor are needed to prevent voltage transients on switch off; sometimes the inductor is made saturable to minimise energy storage. However, all this represents additional expense, complication and loss of efficiency. The new range of epitaxial diodes allows a more elegant solution to this problem by reducing the stored charge until its effects are negligible.

Current commutation from switch to rectifier

Secondary problems can occur during this phase, as illustrated in Fig.3. When the switch is turned off, the voltage across it rises and the diode reverse bias falls. However, there is a delay between the time the diode becomes forward-biased and conduction begins, due to the diode forward recovery behaviour. During this time, the switch voltage overshoots the d.c. supply voltage whilst the switch is still conducting virtually full current. This can result in failure of the switch if its voltage rating is exceeded; if not, it simply adds to the dissipation. Therefore it is important to choose a rectifier with a sufficiently low forward recovery voltage.

ULTRA-FAST EPI-RECTIFIERS

The structure of an epi-diode is shown in Fig.4(a). It consists of an n-type epitaxial layer on an n^+ substrate with a diffused p-type layer above it. Both the epitaxial layer and the p-type diffusion are shallow and well-controlled. The result is a well-defined narrow base that gives very high carrier injection efficiency. This allows lower forward voltage drops and tighter control of the stored charge than can be achieved with double-diffused technology. Recovery times for epitaxial rectifiers are in fact about an order of magnitude lower than their double-diffused counterparts. Because of their very low I_{RRM}, the recovery is free from r.f.i. and transient generation; moreover there is no need for external snubbers.

Planar epitaxial construction, shown in Fig.4(b) is unsuitable for applications above 200 V. This is because the high field at the edges of the p-diffusions may lead to premature breakdown and despite the presence of a channel stopper it is difficult to achieve stability with oxide passivation. The voltage range may be extended to 500 V by using a half-mesa construction (Fig.4(c)), as in our previous product range.

To extend the operating voltage range still further, full-mesa construction is used (Fig.4(d)). This has no sharp edges to produce high fields but passivation must be improved for stable reverse blocking characteristics at high voltage. This is done by depositing a layer of semi-insulating material over the junction to prevent charge build-up. A special high-purity hard glass with controlled charge is then deposited over this layer to ensure excellent stability either in plastic or in metal encapsulations. The n^+ substrate then forms a channel stopper without the need for an extra diffusion.

Another requirement for high-voltage operation is control of the stored charge. As the base width increases (to permit higher voltage operation), so does the on-state voltage-drop due to the heavy metal doping. The amount of doping must therefore be controlled to within a very close tolerance. Enough must be introduced to keep the stored charge down to an acceptable level, but not so much that the on-state voltage drop becomes too large. A further feature of epitaxial rectifiers with close tolerance base widths is that the forward recovery voltage is the minimum theoretically attainable for that structure.

One advantage of a full-mesa construction is that it has a strong wafer edge which makes surface damage much less likely. If damage does occur, it is unlikely to reach the junction and affect device operation. This means that the rectifier is rugged and reliable, and also that wafers are fully testable whilst still 'on slice' so only perfect devices need be assembled.

Special attention has been paid to all these requirements in our range of 500 V and 800 V epitaxial diodes. In particular, the 800 V range is suitable for off-line direct switching without series-connection. Details of the complete range are shown in the Table.

Finally, the range includes recently developed monolithic double epitaxial rectifiers, which feature perfect matching of forward voltage drop and recovery characteristics. These are ideal for switched-mode power supplies, where a pair of rectifiers with a common cathode connection is often required. This occurs both in forward converters, Fig.5(a) and in push-pull output stages, Fig.5(b). The use of a double diode in a push-pull converter has the added advantage of avoiding one possible cause of transformer saturation: unmatched output rectifiers.

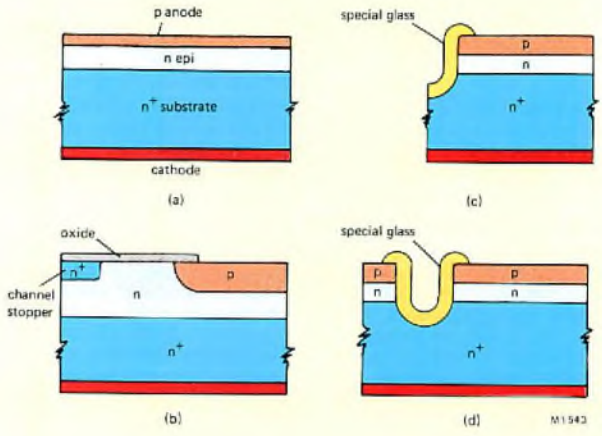


Fig.4 Epitaxial rectifier:
(a) structure;
(b) planar construction;
(c) half-mesa construction;
(d) full-mesa construction

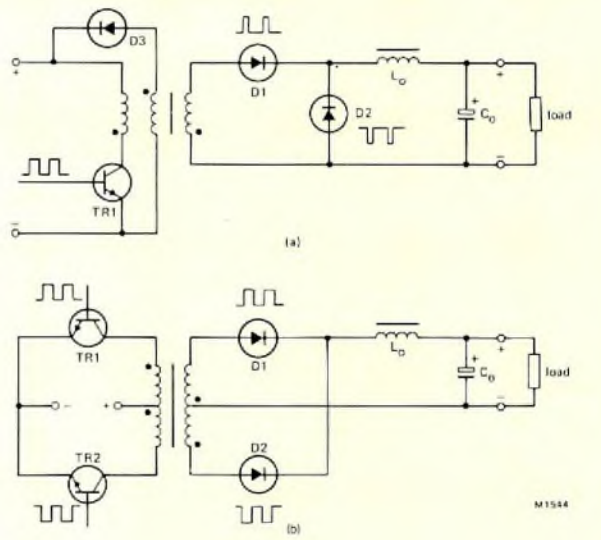


Fig.5 Switched-mode power supply configurations:
(a) forward converter;
(b) push-pull converter

Abstracts

An architectural contrast – 68000 versus iAPX 286

By comparing the architecture of the 68000 with that of another of today's advanced 16-bit microprocessor architectures (the iAPX 286), this article shows why the 68000 is one of the most powerful and versatile microprocessors available. Areas where the 68000 excels are in its 32-bit true general-purpose registers, its data handling and virtual memory capabilities, its revolutionary instruction set and its flexibility which permits a wide choice of memory-management schemes.

Interfacing different local area networks

Two local area networks, one CSMA/CD Ethernet and the other based on a token-passing access method, can communicate via a gateway consisting of interface controllers for both networks, a message buffer memory and a management controller. Both interface controllers and the management controller can be produced from identical LSI chips: the 8X305 bipolar microcontroller, the 8X360 memory address director, the 8X310 interrupt control coprocessor and the 8X350 high-speed bipolar RAM.

The KP100A monolithic pressure sensor

The KP100A monolithic pressure sensor is a silicon-based aneroid gauge in which diaphragm movement is detected by a series of piezoresistive strain gauges implanted in the diaphragm in a Wheatstone bridge configuration. It has highly-stable, linear characteristics and can be used in for example pressure switches, altimeters, barometers and automotive control systems.

New implosion technology to meet explosion in diode demand

Implosion diodes – solid glass encapsulated diodes produced by a new implosion process – have many advantages over conventional glass-bead diodes. They have a lower forward voltage for example, and they are smaller and far more regular in shape. So they're ideal for use with automatic assembly equipment. What's more, IDs are just as robust as their glass-bead counterparts and their reliability is at least as high.

HCMOS – fast but cool logic ICs

Our high-speed CMOS (HCMOS) family of logic ICs combine the high speed of LSTTL circuits with the low power dissipation of other well-known CMOS families. However, in TTL circuits the static or quiescent power dissipation is far greater than the dynamic power dissipation. For HCMOS circuits, exactly the opposite is true, and this can cause confusion when making power dissipation calculations. This article dispels the confusion by explaining the various factors that contribute to the dynamic power dissipation of HCMOS circuits.

EUROM – a single-chip colour c.r.t. controller

Using a new single-chip CRT controller, known as EUROM, it is possible to build a videotex terminal that exceeds the requirements of the most recent CEPT specification yet uses a minimum of peripheral components and board area.

Backup support gives VMEbus powerful multiprocessor architecture

The VMEbus can provide full 32-bit data transfer and addressing. A powerful multiprocessing system can be produced by using the VMEbus to provide the global data paths, VMXbuses for interfacing dedicated resources such as memory, terminals and data aquisition devices, and a VMSbus for critical message passing and shared resource arbitration between processors.

Epitaxial diodes – rectifiers compatible with today's fast switches

The latest ultra-fast epitaxial diodes are comparable in switching speed with modern fast, active switches. Improved design has increased V_{RRM} to 800 V and careful production control ensures minimal on-state losses. Reverse recovery times range from 25 ns at a V_{RRM} of 200 V to 100 ns at 800 V.

Vergleich der unterschiedlichen Architektur der Mikroprozessoren 68000 und iAPX 286

Anhand eines Architektur-Vergleichs zwischen dem Mikroprozessor 68000 und einem anderen hochentwickelden Mikroprozessor (iAPX 286) zeigt dieser Artikel, weshalb der 68000 einer der leistungsfähigsten und vielseitigsten zur Zeit verfügbaren Mikroprozessoren ist. Mermale, in denen sich der Mikroprozessor 68000 besonders ausgezeichnet sind die 32 bit-Register für universellen Einsatz, sein leistungsfähiger Befehlsvorrat sowie seine Flexibilität in Hinblick auf eine Vielzahl unterschiedlicher Speicherverwaltungskonzepte.

Anpassung unterschiedlicher Lokaler Netze

Zwei unterschiedliche Lokale Netze, Ethernet GSMA/CD und ein auf einem Token-Passing-Zugriffsverfahren basierendes Netz, können über einen Gateway (Kommunikationsrechen) miteinander kommunizieren, der aus Interface Kontrollern für beide Netze, einem Datenpufferspeicher und einem Steuerprozessor besteht. Die Interface-Controller und der Steuerprozessor lassen sich mit LSI-Schaltungen desselben Typs realisieren: dem bipolaren Steuerprozessor 8X305, dem Speicheradressverwalter 8X360, dem Interrupt-Controller 8X310 und dem schnellen Bipolar-RAM 8X350.

Der monolitische Drucksensor KP100A

Der monolitische Drucksensor KP100A ist ein auf Siliziumbasis arbeitendes Aneroidbarometer, in dem die Diaphragma-Bewegung durch mehrere piezoresistive Dehnungsmessstreifen detektiert wird, die in das Diaphragma in Form einer Wheatstone-Brücke eingebaut sind. Der Sensor hat eine sehr stabile, lineare Charakteristik und kann beispielsweise als Druckschalter, Höhenmesser, Barometer und für Kontrollsysteme in Kraftfahrzeugen verwendet werden.

Neue Implosions–Technologie zur Deckung Nachfrage explosion von Dioden

Implosions-Dioden, mit der neuen Implosions-Technologie hergestellte glasgekapselte Dioden, besitzen gegenüber den herkömmlichen glasbead-Dioden viele Vorteile. Beispielsweise haben sie eine niedrigere Durchlassspannung, sind kleiner und haben eine gleichmässigere Gehäuseform. Sie sind daher ideal für die Verwendung in Bestückungsautomaten. Ausserdem sind Implosions-Dioden ebenso robust wie die glasbead-Dioden, und ihre Zuverlässigkeit ist mindestens genauso hoch.

Verlustleistungsbetrachtungen über HCMOS-Logik-ICs

Unsere sehr schnellen Logik-ICs der CMOS (HCMOS) -Familie vereinen die hohe Geschwindigkeit von LSTTL-Schaltungen mit der niedrigen Verlustleistung anderer bekannter CMOS-Familien. Jedoch ist die statische oder Ruheverlustleistung in TTL-Schaltungen weit grösser als die dynamische Verlustleistung. Für HCMOS-Schaltungen gilt genau das Gegenteil, was bei Verlustleistungsbetrachtungen zu Fehlern führen kann. Dieser Artikel beseitigt diese Verwirrung indem er die verschiedenen Faktoren behandelt, welche zur dynamischen Verlustleistung von HCMOS-Schaltungen beitragen.

Neuer CRT-Controller erfüllt die CEPT-Videotex-Anforderungen

Unter Verwendung eines neuen Einchip-CRT-Controllers mit der Bezeichnung EUROM lässt sich ein Bildschirmtext-Terminal aufbauen, das trotz sehr geringen Bedarfs an Peripheriebauelementen und Platinenfläche die neueste CEPT-Spezifikation erfüllt und mit einigen Leistungsmerkmalen sogar übertrifft.

Vielfältige Unterstützung verleiht dem VMEbus eine leistungsfähige Multiprozessor-Architektur

Der VMEbus erlaubt eine vollständige 32-bit-Datenübertragung und -adressierung. Ein leistungsfähiges Multiprocessing-System lässt sich realisieren, in dem folgende Aufgabenteilung vorgenommen wird: der VMEbus für den globaler Datenverkehr (z.b. mit Speicher, Termin und Datenakquisitionsschaltungen) und der VMEbus für die Koordinierung des gesamten Datenverkehrs im Multiprozessorsystem.

Epitaxial–Dioden – Gleichrichter mit kompatiblem Verhalten zu modernen, schnellen Schaltern

Die neuesten, sehr schnellen Epitaxial-Dioden sind bezüglich ihrer Schaltgeschwindigkeit mit den modernen, schnellen, aktiven Schaltern vergleichbar. Durch einen verbesserten Entwurf hat sich V_{RRM} auf 800 V erhöht, und durch aufwendigere Fertigungskontrollen konnten minimale Durchlassverluste gesichert werden. Die Sperrverzugszeiten liegen im Bereich von 25 ns bei V_{RRM} = 200 V bis 100 ns bei V_{RRM} = 800 V.

Le contraste de deux architectures: 68000 contre iAPX 286

En comparant l'architecture du 68000 à celle d'un autre microprocesseur 16 bits très évolué (l'iAPX 286), cet article démontre que le 68000 est l'un des microprocesseurs les plus puissants et les plus polyvalents qui soient. Le 68000 se distingue notamment par ses registres 32 bits réellement universels, ses possibilités de traitement de données et de mémoire virtuelle, son jeu d'instructions révolutionnaire, et sa souplesse qui autorise de nombreux systèmes de gestion de la mémoire.

Connexion de différents réseaux urbains

Deux réseaux urbains différents, l'un un Ethernet CSMA/CD, l'autre fondé sur une méthode d'abord de type "à jeton", peuvent communiquer par une voie d'accès constituée par des contrôleurs d'interface pour les deux réseaux, une mémoire tampon pour les messages et un contrôleur de gestion. Les deux contrôleurs d'interface et le contrôleur de gestion peuvent être réalisés à partir des mêmes circuits LSI: le microcontrôleur bipolaire 8X305, le sélecteur d'adresses en mémoire 8X360, le contrôleur d'interruption 8X310 et la mémoire RAM bipolaire ultra-rapide 8X350.

Le capteur de pression monolithique KP100A

Le capteur de pression monolithique KP100A est une jauge anéroide en silicium dans laquelle le mouvement du diaphragme est detecté par une série de jauges d'extensometrie piézorésistives, implantées dans le diaphragme, dans la configuration d'un pont de Wheatstone. Le dispositif possède une caractéristique linéaire de grande stabilite. Il peut être utilisé, par exemple, dans des manocontacts, des altimètres, des baromètres et des systèmes de commande pour véhicules automobiles.

Nouvelle technologie d'encapsulation par implosion pour faire face à l'explosion de la demande de diodes

Les diodes à encapsulation par implosion – diodes en boitier de verre massif – obtenues d'un nouveau procédé par implosion de verre – présentent de nombreux avantages par rapport aux diodes classiques sous perle de verre. Elles ont, en outre une tension directe plus faible, un boitier plus petit et de forme bien plus régulière. Elles autorisent donc des densités d'assemblage bien plus élevées sur les circuits imprimés et elles sont bien adaptées à l'emploi avec des équipements d'assemblage automatique. De plus, les diodes "implosion" sont tout aussi robustes et fiables que les diodes "perle de verre".

Remarques sur la dissipation de puissance des circuits intégrés logiques HCMOS

Les circuits intégrés logiques ultra-rapides de notre famille CMOS (HCMOS) associent la grande rapidité des circuits TTL-LS à la faible dissipation de puissance des familles CMOS standard. Toutefois, la dissipation statique ou au repos des circuits TTL est bien plus élevée que la dissipation dynamique, et exactement l'inverse pour les circuits HCMOS, ce qui peut être une source d'erreurs lors des calculs de dissipation de puissance. Pour y remédier, cet article explique les divers facteurs qui contribuent à la dissipation dynamique de puissance des circuits HCMOS.

Un nouveau contrôleur de visualisation EUROM répond largement aux recommandations pour le vidéotex

Il est possible, à l'aide du contrôleur de visualisation monopuce, appelé EUROM, de réaliser un terminal vidéotex qui surpasse les recommandations CEPT les plus recentes, tout en occupant une superficie minimale et en utilisant un minimum de composants périphériques.

Des moyens de support donnent au bus VME une puissante architecture de multitraitement

Le bus VME possède la capacité nécessaire pour le transfert de données et l'adressage en 32 bits. Un puissant système de multitraitement peut être réalisé en utilisant le bus VME pour l'acheminement global des données, des bus VMX pour la connexion de sources spécialisées tel les que mémoire, terminaux et dispositifs d'acquisition, et un bus VMS pour le passage des messages critiques, et l'arbitrage des sources partagées entre processeurs.

Des redresseurs à diodes épitaxiales compatibles avec les commutateurs rapides actuels

Les diodes épitaxiales ultra-rapides les plus récentes ont des temps de commutation comparables à ceux des commutateurs actifs modernes. La valeur de V_{RRM} a pu être portée à 800 V par des améliorations de conception. Un contrôle rigoureux en fabrication assure des pertes minimales à l'etat conducteur. Les temps de récupération inverses s'échelonnent de 25 ns pour une valeur V_{RRM} de 200 V à 100 ns pour 800 V.

Un contraste arquitectónico – 68.000 respecto al iAPX 286

Comparando la arquitectura del 68.000 con otras arquitecturas avanzadas de microprocesador de 16 bits (el iAPX 286), este artículo muestra por qué el 68.000 es uno de los microprocesadores disponibles más versátil y de mayor potencia. Las áreas donde sobresale el 68.000 son en sus auténticos registros de aplicación general de 32 bits, su capacidad de memoria virtual y manejo de datos, su revolucionario juego de instrucciones y su flexibilidad que permite una amplia elección de memoria gobierno.

Interconexion de diferentes redes de área local

Dos redes de área local, una Ethernet CSMA/CD y otra basada en un metodo de acceso de paso con prenda, se pueden comunicar mediante una vía con puerta que consta de controladores de interconexión para ambas redes, una memoria intermedia de mensaje y un controlador de gobierno. Ambos controladores de interconexión y el controlador de gobierno pueden producirse a partir de chips LSI identicos: el microcontrolador bipolar 8X305, director de dirección de memoria, el procesador de control interrupción 8X310 y la memoria bipolar RAM de alta velocidad.

Sensor monolítico de presión KP100A

El sensor monolítico de presión KP100A es un indicador de presión aneroide en el cual se detecta el movimiento de un diafragma mediante una serie de indicadores de esfuerzos piezoresistivos implantados en el diafragma en una configuración de puente de Wheatstone. Tiene caracteristicas lineales, altamente estables. El dispositivo se puede usar, por ejemplo, en conmutadores de presión, altímetros, barómetros y sistemas de control de automóvil.

Nueva tecnología de implosión hace explosión en la demanda de diodos

Los diodos de implosión – diodos encapsulados en vidrio sólido, producidos mediante un nuevo proceso de implosión – tienen muchas ventajas sobre los diodos perla de cristal convencionales. Por ejemplo, tienen una tensión directa más baja, y son más pequeños y bastante más regulares en forma. Así, pues, son ideales para ser usados con equipos de montaje automático. Además, los diodos de implosión son tan robustos como sus equivalentes de perla de cristal y su fiabilidad es por lo menos igual de alta.

Consideraciones de disipación de potencia para circuitos integrados lógicos HCMOS

Nuestra familia CMOS (HCMOS) de circuitos integrados lógicos combina la alta velocidad de los circuitos LSTTL con la baja disipación de potencia de las otras familias CMOS bien conocidas. Sin embargo, en los circuitos TTL la disipación de potencia en reposo o estática es mucho mayor que la disipación de potencia dinámica. En los circuitos HCMOS ocurre exactamente lo contrario y esto puede ser motivo de confusión al hacer los cálculos de disipación de potencia. Este artículo disipa la confusión explicando los diversos factores que contribuyen a la disipación de potencia dinámica de los circuitos HCMOS.

Un nuevo controlador CTR supera los requisitos cept para Videotex

Usando un nuevo controlador CRT de un chip, conocido como EUROM, es posible construir un terminal de videotex que supere los requisitos de la más reciente especificación cept a la vez que utiliza un mínimo de componentes periféricos y area de la placa.

El soporte de base da al bus VME una potente arquitectura de multiprocesador

Direccionamiento de 32 bits. Puede obtenerse un potente sistema de multiproceso usando el bus VME para el manejo global de datos, el bus VMX para el acoplamineto a recursos específicos tales como memoria, terminales, . . . y el bus VMS para el envio de mensajes críticos y arbitraje de recursos compartidos entre procesadores.

Diodo epitaxial – rectificador compatible con los conmutadores rápidos actuales

Los últimos diodos epitaxiales ultra rápidos son comparables en velocidad de conmutación con los modernos conmutadores activos rápidos. El diseño mejorado ha incrementado la V_{RRM} a 800 V y el cuidadoso control de producción asegura unas pérdidas mínimas en estado de conducción. El margen de tiempos de recuperación inversa se extiende desde 25 ns con una V_{RRM} de 200 V a 100 ns a 800 V.

Authors

R. E. F. Bugg was born at Aylesbury, Buckinghamshire, in 1949 and took his B.Sc in electrical engineering at Imperial College, London, in 1970. After graduation he joined Mullard Ltd, where he initially worked in the tv section and, later, in the text-handling group where he now is. He has been associated with teletext since its beginning and holds several patents related to it.

Jeff Seltzer received in his B.A. in physics from the University of Pennsylvania in 1977 and is now a candidate for a M.Sc. in computer science at California State University. He joined Signetics in 1980 as bipolar LSI applications engineer and, from August 1981, he was a Semi-custom Program administrator for the bipolar LSI division. He has now left Signetics and works for National Semiconductor, Santa Clara.

Jan Exalto, born at Eindhoven in 1942, graduated from Eindhoven Polytechnic in 1964. After military service he joined the Central Application Laboratory of Philips Electronic Components and Materials Division in 1966. Since then he has been principally engaged in digital electronics and is at present responsible for applications of standard and customised CMOS.

Naseer Siddique was born in Rabwah, Pakistan in 1957. He earned his BSEE at the University of Engineering and Technology in Lahore, Pakistan in 1980. In 1982, he was awarded an MS, Computer Engineering, by Wayne State University, Detroit, Michigan. Naseer joined Signetics LSI Division in September 1983, where he continues to serve as an Applications Engineer for standard products.

Graham Hine, born in 1954 in Cheshire, attended Darwin College, Cambridge, and in 1980 gained his doctorate there in experimental physics with a dissertation on radio astrophysics. The same year he joined Mullard Ltd. at Hazel Grove where, as a member of the development team for discrete power semiconductor devices, he specialized in fast Schottky diodes. In 1983 he moved to Philips Electronic Components and Materials Division in Eindhoven and is now in commercial product management for glass diodes. Graham has previously published in Nature and the Monthly Notice of the Royal Astronomical Society.

Bart J. M. Vernooij was born in Bennebroek, The Netherlands in 1955. He studied Physics at the Technical University of Delft and after military service joined Philips Electronic Components and Materials Division where he is engaged in the marketing of the 68000 microprocessor family.

Gerd Keitel was born in Gumbinnen, Germany in 1943. He studied solid-state physics at the University of Hamburg, receiving a doctorate there in 1971. The following year he joined Valvo, Hamburg, and since 1982 he has been product manager for semiconductor sensor devices, based at Philips Electronic Components and Materials Divison, Eindhoven.

Arthur Woodworth was born in Manchester in 1945. He took a B.Sc. in electrical Engineering at Salford University in 1968 and later that year joined Mullard, Stockport, where he is now head of the Electrical Development Department responsible for the specification and evaluation of a wide range of rectifiers, thyristors, triacs, and GTOs.

Philips Electronic Components & Materials

Elcoma - Philips Electronic Components and Materials Division - embraces a world-wide group of companies operating under the following names:

IBRAPE
PHILIPS
MBLE
RTC
Miniwatt
signetics
Mullard
VALVO

Elcoma offers you a technological partnership in developing your systems to the full. A partnership to which we can bring

- world-wide production and marketing
- know-how
- systems approach
- continuity
- broad product line
- fundamental research
- leading technologies
- applications support
- quality

DISPLAY SYSTEMS

B & W tv display systems
Colour tv display systems
Data graphic display systems

INTEGRATED CIRCUITS

Bipolar analogue
Consumer circuits
- video & radio/audio circuits

Industrial circuits
- opamps • voltage regulators
- comparators • D/A & A/D converters
- amplifiers • interface circuits

Bipolar digital
Standard logic families
- TTL/STTL • ECL 10K/100K

LSI circuits
- gate arrays • interface circuits

8-bit Microprocessors

Memories
- RAMs/PROMs • Fuse logic

NMOS
8 and 16-bit Microprocessors & peripherals

Logic systems
- video & radio/audio circuits
- text decoders

ROM memories

CMOS
Standard logic families

LSI circuits
- gate arrays • clock circuits

8-bit Microprocessors

Hybrid integrated circuits
LF, VHF, UHF & microwave circuits
D/A converters
Proximity switches
Custom-designed circuits

ELECTRO-OPTICAL DEVICES

Image intensifier devices
Infra-red image detectors
Camera tubes
Imaging devices

SEMICONDUCTORS

Small-signal diodes
Medium and high-power diodes
Controlled rectifiers
LF & HF small-signal transistors
LF & HF power transistors
Microwave semiconductors
Opto-electronic semiconductors
Semiconductors for hybrid ICs
Semiconductor sensors

PROFESSIONAL TUBES

Industrial cathode-ray tubes
Photomultiplier tubes
Geiger Müller tubes
Transmitting tubes
Microwave devices
Reed switches

MATERIALS

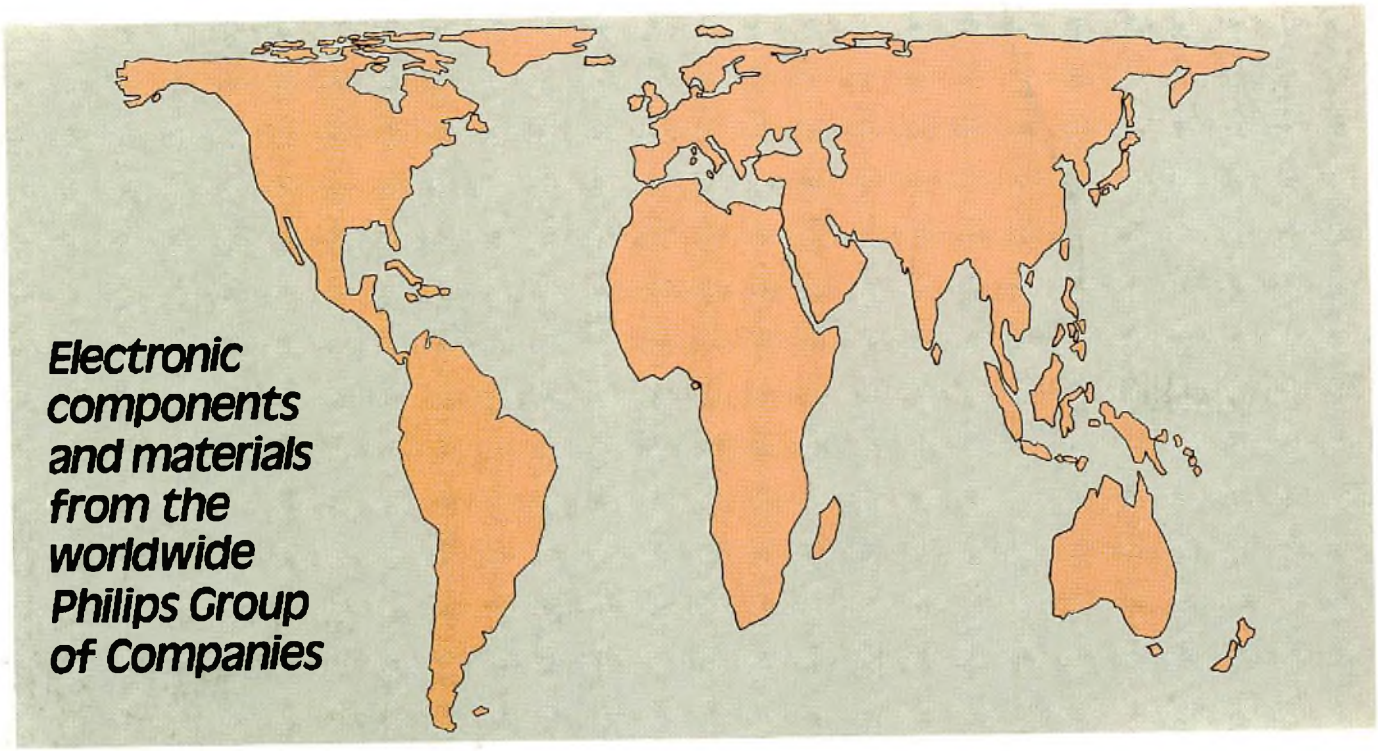
Ferroxcube products
Permanent magnets
Piezoelectric products
White ceramic products

PASSIVE COMPONENTS

Ceramic, film and foil capacitors
Electrolytic capacitors
Fixed resistors
Variable capacitors
Potentiometers and resistor trimmers
Non-linear resistors
Delay lines
Piezoelectric quartz devices

ASSEMBLIES

Electric motors
Loudspeakers
Tuners
Printed circuit boards
Variable transformers
Thumbwheel switches
Industrial microcomputer systems
Microwave sub-assemblies



Argentina: PHILIPS ARGENTINA S.A., Div. Elcoma, Vedia 3892, 1430 BUENOS AIRES, Tel. 541-7141/7242/7343/7444/7545.
Australia: PHILIPS INDUSTRIES HOLDINGS LTD., Elcoma Division, 67 Mars Road, LANE COVE, 2066, N.S.W., Tel. 427 0888.
Austria: ÖSTERREICHISCHE PHILIPS BAUELEMENTE INDUSTRIE G.m.b.H., Triester Str. 64, A-1101 WIEN, Tel. 62 91 11.
Belgium: N.V. PHILIPS & MBLE ASSOCIATED, 9 rue du Pavillon, B-1030 BRUXELLES, Tel. (02) 242 74 00.
Brazil: IBRAPE, Caixa Postal 7383, Av. Brigadeiro Faria Lima, 1735 SAO PAULO, SP, Tel. (011) 211-2600.
Canada: PHILIPS ELECTRONICS LTD., Electron Devices Div., 601 Milner Ave., SCARBOROUGH, Ontario, M1B 1M8, Tel. 292-5161.
Chile: PHILIPS CHILENA S.A., Av. Santa Maria 0760, SANTIAGO, Tel. 39-4001.
Colombia: IND. PHILIPS DE COLOMBIA S.A., c/o IPRELENSO LTD., Calle 17, No. 9-21, Of. 202, BOGOTA, D.E., Tel. 57-2347493.
Denmark: MINIWATT A/S, Strandlodsvej 2, P.O. Box 1919, DK 2300 COPENHAGEN S, Tel. (01) 54 11 33.
Finland: OY PHILIPS AB, Elcoma Division, Kaivokatu 8, SF-00100 HELSINKI 10, Tel. 17271.
France: R.T.C. LA RADIOTECHNIQUE-COMPELEC, 130 Avenue Ledru Rollin, F-75540 PARIS 11, Tel. 338 80-00.
Germany (Fed. Republic): VALVO, UB Bauelemente der Philips G.m.b.H., Valvo Haus, Burchardstrasse 19, D-2 HAMBURG 1, Tel. (040) 3296-0.
Greece: PHILIPS S.A. HELLENIQUE, Elcoma Division, 52, Av. Syngrou, ATHENS, Tel. 9215111.
Hong Kong: PHILIPS HONG KONG LTD., Elcoma Div., 15/F Philips Ind. Bldg., 24-28 Kung Yip St., KWAI CHUNG, Tel. (0)-245121.
India: PEICO ELECTRONICS & ELECTRICALS LTD., Elcoma Div., Ramon House, 169 Backbay Reclamation, BOMBAY 400020, Tel. 221012.
Indonesia: P.T. PHILIPS-RALIN ELECTRONICS, Elcoma Div., Panim Bank Building, 2nd Fl., Jl. Jend. Sudirman, P O. Box 223, JAKARTA, Tel. 716131.
Ireland: PHILIPS ELECTRICAL (IRELAND) LTD., Newstead, Clonskeagh, DUBLIN 14, Tel. 693355.
Italy: PHILIPS S.p.A., Sezione Elcoma, Piazza IV Novembre 3, I-20124 MILANO, Tel. 2-6752.1.
Japan: NIHON PHILIPS CORP., Shuwa Shinagawa Bldg., 26-33 Takanawa 3-chome, Minato-ku, TOKYO (108), Tel. 448-5611.
(IC Products) SIGNETICS JAPAN LTD., 8-7 Sanbancho Chiyoda-ku, TOKYO 102, Tel. (03) 230-1521.
Korea (Republic of): PHILIPS ELECTRONICS (KOREA) LTD., Elcoma Div., Philips House, 260-199 Itaewon-dong, Yongsan-ku, SEOUL, Tel. 794-4202.
Malaysia: PHILIPS MALAYSIA SDN. BERHAD, No. 4 Persiaran Barat, Petaling Jaya, P.O.B. 2163, KUALA LUMPUR, Selangor, Tel. 774411.
Mexico: ELECTRONICA, S.A de C.V., Carr. México-Toluca km. 62.5, TOLUCA, Edo. de México 50140, Tel. Toluca 91 (721) 613-00.
Netherlands: PHILIPS NEDERLAND, Marktgroep Elonco, Postbus 90050, 5600 PB EINDHOVEN, Tel. (040) 793333.
New Zealand: PHILIPS ELECTRICAL IND. LTD., Elcoma Division, 110 Mt. Eden Road, C.P.O. Box 1041, AUCKLAND, Tel. 605-914.
Norway: NORSK A/S PHILIPS, Electronica Dept., Sandstuveien 70, OSLO 6, Tel. 68 02 00.
Peru: CADESA, Av. Alfonso Ugarte 1268, LIMA 5, Tel. 326070.
Philippines: PHILIPS INDUSTRIAL DEV. INC., 2246 Pasong Tamo, P.O. Box 911, Makati Comm. Centre, MAKATI-RIZAL 3116, Tel. 86-89-51 to 59.
Portugal: PHILIPS PORTUGUESA S.A.R.L., Av. Eng. Duarte Pacheco 6, 1009 LISBOA Codex, Tel. 683121.
Singapore: PHILIPS PROJECT DEV. (Singapore) PTE LTD., Elcoma Div., Lorong 1, Toa Payoh, SINGAPORE 1231, Tel. 2538811.
South Africa: EDAC (PTY.) LTD., 3rd Floor Rainer House, Upper Railway Rd. & Ove St., New Doornfontein, JOHANNESBURG 2001, Tel. 614-2362/9.
Spain: MINIWATT S.A., Balmes 22, BARCELONA 7, Tel. 301 63 12.
Sweden: PHILIPS KOMPONENTER A.B., Lidingövägen 50, S-11584 STOCKHOLM 27, Tel. 08/7821000.
Switzerland: PHILIPS A.G., Elcoma Dept., Allmendstrasse 140-142, CH-8027 ZÜRICH, Tel. 01-4882211.
Taiwan: PHILIPS TAIWAN LTD., 3rd Fl., San Min Building, 57-1, Chung Shan N. Rd, Section 2, P.O. Box 22978, TAIPEI, Tel. (02)-5631717.
Thailand: PHILIPS ELECTRICAL CO. OF THAILAND LTD., 283 Silom Road, P.O. Box 961, BANGKOK, Tel. 233-6330-9.
Turkey: TÜRK PHILIPS TICARET A.S., Elcoma Department, Inönü Cad. No. 78-80, ISTANBUL, Tel. 435910.
United Kingdom: MULLARD LTD., Mullard House, Torrington Place, LONDON WC1E 7HD, Tel. 01-5806633.
United States: (Active Devices & Materials) AMPEREX SALES CORP., Providence Pike, SLATERSVILLE, R.I. 02876, Tel. (401) 762-9000.
(Passive Devices) MEPCO/ELECTRA INC., Columbia Rd., MORRISTOWN, N.J. 07960, Tel. (201) 539-2000.
(Passive Devices & Electromechanical Devices) CENTRALAB INC., 5855 N. Glen Park Rd., MILWAUKEE, WI 53201, Tel. (414)228-7380.
(IC Products) SIGNETICS CORPORATION, 811 East Arques Avenue, SUNNYVALE, California 94086, Tel. (408) 739-7700.
Uruguay: LUZILECTRON S.A., Avda Uruguay 1287, P.O. Box 907, MONTEVIDEO, Tel. 91 43 21.
Venezuela: IND. VENEZOLANAS PHILIPS S.A., Elcoma Dept., A. Ppal de los Ruices, Edif. Centro Colgate, CARACAS, Tel. 360511

For all other countries apply to: Philips Electronic Components and Materials Division, International Business Relations, Building BAE, P.O. Box 218, 5600 MD EINDHOVEN, The Netherlands, Tel. +31 40 72 33 04, Telex 35000 phtcnl

A42

Printed in The Netherlands

9398 704 20011