

EMS 4.0 pulls together

In search of supernumerary memory for DOS applications, EMS 4.0 unites something old, something borrowed, and something new, says Ted Mirecki.

Until applications arrive that fulfill the promise of OS/2, many users are searching for alternatives that can run widely available DOS applications, yet overcome the limitations of that operating system. Expanded memory is the technology that gives such alternatives a large part of their power. The Lotus/Intel/Microsoft Expanded Memory Specification (EMS) version 4.0 unites features of earlier EMS versions with strengths of the AST Research/Quadram/Ashton-Tate Enhanced Expanded Memory Specification (EEMS), and it adds some traits of its own to create a new and improved EMS. The new version may open all kinds of doors for DOS applications.

Lotus, Intel and Microsoft originally developed EMS to relieve the spreadsheet power user's memory crunch. It was intended to provide 8Mbytes of memory beyond the 640k limit as a repository for data such as spreadsheet cells or a RAM disk. Because most programs of this type already provided their own data space management, expanded-memory management was also left at the application level. Therefore, expanded memory benefited applications only to the extent that they were written specifically to take advantage of it. Such applications and utilities soon proliferated.

However, DOS systems needed additional memory for purposes other than data storage. Programs were growing in size, and users were discovering multi-tasking in the form of terminate-and-stay-resident (TSR) utilities and task-switching environments that loaded more than one program at a time. Although the original EMS offered the possibility of expanding the memory space by an order of magnitude, the design of its hardware interface with the rest of the system made it inefficient for running multiple programs.

To provide for this further use of expanded memory, AST, Quadram, and Ashton-Tate developed EEMS. For applications and small TSR utilities, EEMS provides the same capabilities, in the same way as EMS. For system-level control programs that create and manage task-switching or multi-tasking environments, it efficiently expands memory available for running programs.

The most notable EEMS success is DESQview from Quarterdeck Office Systems. DESQview and Microsoft Windows run on top of DOS and rely on a hardware solution to expanded memory. Either of these, in conjunction with underlying hardware, can provide two features lacking in DOS: a large address space and multi-tasking.

Quarterdeck's windowing multi-tasking

environment is much like Microsoft Windows, but it is text-based and menu-driven instead of graphics- and icon-oriented. It can simultaneously run — not just hold in memory for task switching but actually execute — as many programs as will fit in available expanded memory. By comparison, Windows versions prior to 2.0 used EMS memory for data storage only, so it could multi-task only as many programs as fit in the 640k of conventional memory space at one time.

DESQview demonstrated that expanded memory could go beyond merely providing data storage space. In recognition of this fact, Lotus, Intel, and Microsoft revised their specification extensively to create EMS 4.0. The major changes are increasing the maximum expanded memory space from 8Mbytes to 32Mbytes, incorporating the same hardware interface as EEMS, and adding multi-tasking support in software.

Part of the motivation for this revision was undoubtedly the release of Windows 2.0, which takes advantage of the multi-tasking support provided by EMS 4.0 software running on appropriate hardware. Another reason could have been the design of IBM's 80286 Memory Expanded board for the PS/2 Models 50 and 60, which supports a mapping scheme functionally equivalent to EEMS

MEMORY

(but for reasons other than providing expanded memory).

Something old

Understanding expanded memory requires clear definitions of the terms used in describing the memory architecture of a PC system. The 'system address space' is memory addressable by the CPU. In real mode (the only mode available to 8088 processors), the address space is 1Mbyte. In protected mode, it is 16Mbytes for 80286-based computers and 4Gbytes for 80386-based machines. 'Conventional memory' is the portion of system address space available to DOS. On most PC-compatible systems, this is limited to 640k, but this number is a consequence of hardware design, not an inherent limitation of DOS.

'Extended memory' is memory above 1Mbyte that lies within the system address space of a '286 or '386. It is accessible only after switching into protected mode (for example, under OS/2 or UNIX), whereupon it becomes part of the larger system address space. DOS cannot use this memory, but the BIOS of '286 and '386-based systems has procedures for transferring data between conventional and extended memory by temporarily switching into protected mode. This facility is of use to device drivers (such as VDISK) that implement a RAM disk in extended memory. EMS is not concerned in any way with extended memory.

'Expanded memory' resides outside system address space and therefore cannot be accessed directly in any processor mode. It is applicable in real mode only to provide memory in excess of the conventional memory limit. Expanded memory is divided into 'pages' of 16k and accessed by mapping some number of these pages into unused portions of this address space. Each 16k block of the system address space that can hold an expanded memory page is called a 'page frame'.

The implementation of expanded memory requires both hardware and software components. The hardware must allow changing of the effective address of memory under software control. This capability is built into the '386 processor, but for '286 and 8088-based systems, it must be provided by hardware on the memory board. On such boards, each page frame requires a page register whose contents determine which expanded-memory page appears at the address of that frame.

Software needed to control mapping takes the form of a device driver called the Expanded Memory Manager (EMM).

FUNCTION

DESCRIPTION

| | |
|-------|------------------------------------------------------------------|
| 1 | Get EMM status. |
| 2 | Get page-frame base address. |
| 3 | Get total number of pages and free-page count |
| 4 | Assign an EMS handle ID and allocate pages to it |
| 5 | Map a single page into a page frame |
| 6 | Close an EMS handle and deallocate all its pages |
| 7 | Get the EMM version number |
| 8 | Save the EMM status in an internal buffer |
| 9 | Restore the EMM status from the internal buffer |
| 10-11 | Reserved |
| 12 | Get the number of active EMS handles |
| 13 | Get the number of pages allocated to a specific handle |
| 14 | Get an array of page counts per handle |
| 15 | Save, restore, or swap the EMM status to/from an external buffer |

Table 1 EMM functions common to all versions: the basic set of EMM functions defined by EMS 3.2 is a proper subset of the functions supported by both EEMS 3.2 and EMS 4.0. Programs using only this set of functions are compatible with all EMS hardware and software

However, programs that use expanded memory communicate with the EMM not through DOS calls or IOCTL functions (the normal means for applications to talk to device drivers) but through interrupt 67h. Typically, interrupt handlers are installed as TSR programs, not device drivers.

The EMS and EEMS implementation of the EMM as a device driver turns out to be an unfortunate choice because it makes expanded memory unusable in the DOS compatibility mode of OS/2. If the EMM were a TSR, it could be installed after switching to OS/2 real mode, and DOS applications running in the compatibility box could fully use expanded memory. But a DOS device driver, even if it is designed for real mode only, cannot be installed under OS/2. Using EMS memory in the DOS box will be possible when and if EMM drivers are rewritten to allow installation under OS/2.

Table 1 lists the basic set of functions common to all implementations of EMS.

Something borrowed

The differences introduced by EEMS (and now adopted by EMS 4.0) have to do with the number and location of page frames in the system address space. Incorporation of these features into EMS acknowledges the role that expanded memory can play in support of multi-tasking operating environments.

EMS 3.2 specifies four contiguous page frames above conventional

memory, creating a 64k window into expanded memory. EEMS allows a page frame in every unused 16k block in the system address space, both above and within conventional memory.

In version 3.2 of each specification, EEMS is a proper superset of EMS, meaning that EEMS performs the same functions in the same way — and adds more. All software written to take advantage of EMS runs flawlessly on EEMS hardware. Little practical difference exists at the applications level, and no widely marketed applications take advantage of features specific to EEMS. These features are meant to be used by systems-level software such as multi-tasking operating environments.

To illustrate the advantages of mapping expanded memory into the conventional address space, consider a system with 256k of memory on the system board, an EGA, and a 2Mbyte expanded memory board with 2Mbyte of memory (see Fig 1). The memory board is configured to backfill conventional memory to 640k. If this board conforms to EMS 3.2, it provides a window of four page frames. By default, these are located at segments C800H, CC00H, D000H, and D400H; the user also can configure the window to begin at CC00H or D000H. In any case, two of the six available 16k blocks remain unused. The 384k used to backfill conventional memory are lost from the expanded memory pool, leaving 1664k (104 pages) of expanded memory available to the EMM.

If the memory expansion board is con-

MEMORY

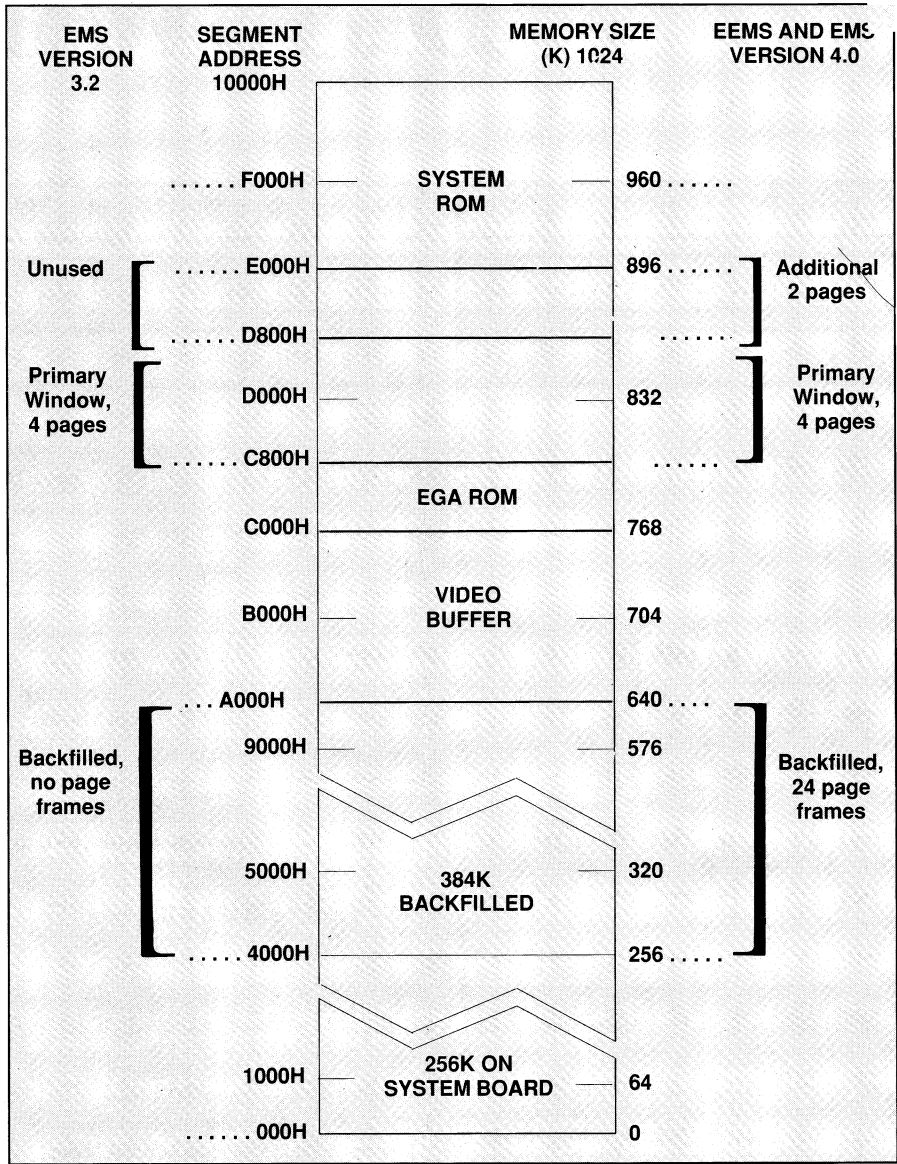


Fig 1 Example memory configuration: previous versions of EMS were limited to providing four page frames above conventional memory. Version 4.0, like previous versions of EEMS, allows a page frame in any 16k block, above or below 640k, not populated by RAM or ROM

structured to the standards of EEMS 3.2 or EMS 4.0, the primary window at segment C800H can extend over six page frames instead of four, allowing an application to manipulate 96k of expanded memory at a time instead of 64k. In addition, the backfilled memory below 640k supports page frames, allowing the mapping of expanded memory pages into conventional memory. Memory used for backfilling remains in the expanded memory pool. While the primary window is meant to be controlled by applications, only an operating system or system-level tasking environment should manage page frames within conventional memory.

If an operating environment running

several co-resident applications consumes the entire conventional memory space, and the user requests start-up of yet another application, a program must be unloaded from memory to make room for the new one. In the absence of expanded memory, one of the applications must be written out to disk before a new one is loaded. With EMS memory, the operating environment could copy an application into expanded memory with this sequence:

1. Allocate a sufficient number of expanded memory pages to hold the unloaded application.
2. Map four of those pages into the window at segment C800H.

MEMORY

3. Copy 64k from conventional memory to the window.

4. Loop back to step 2 until the system has copied sufficient memory to hold the new application.

With EEMS's paged memory below 640k, an application in the upper 384k of conventional memory already resides in expanded memory, so it does not need to be copied out to make room for a new program. Instead, the operating system merely allocates pages for the new application and maps them into page frames in conventional memory. Pages formerly in these frames become inaccessible but retain their contents. This process requires the same number of calls to the EMM (one for each page of the new application) but avoids copying each byte of data through the primary window.

The advantage of EEMS and EMS 4.0 over EMS 3.2 is even greater when the system needs to switch between two previously loaded applications — one in conventional memory, the other in expanded. With previous EMS versions, the system must perform two copies: one to roll out the application being suspended, another to roll in the application being activated. With EEMS and EMS 4.0, it performs both functions merely by mapping the pages holding the incoming application. The pages holding the suspended application are mapped out but not overwritten.

An EEMS or EMS 4.0 board can contain two sets of page registers, with each set containing mapping information for a distinct set of expanded memory pages. By specifying one or the other as the active register set, an entire set of pages can be mapped in with one call to the EMM.

Another advantage of EEMS and EMS 4.0 is that they provide more memory for simultaneously holding applications. With EMS 3.2, the example configuration provides 104 pages of expanded memory for holding inactive applications; active applications are replicated in the conventional memory space. With EEMS and EMS 4.0, 128 pages are available because an active application does not appear twice.

Something new

Besides incorporating EEMS features, EMS 4.0 adds others that provide even more support for multi-tasking. These features are described below in functional groups; EMM functions that implement them are listed in Table 2.

Named handles. The set of expanded memory pages a program obtains by one call to the EMM is identified by a

number called the 'EMM handle'. To map a page into the system address space, the program identifies it by specifying the handle number and a logical page number ranging from zero to one less than the number of pages allocated to that handle. Function 20 allows the assignment of an eight-character name of the user's choice to any handle. The EMM keeps a directory of handle names, and function 21 searches this directory and returns the handle number for a given name. This feature can be used for communication between co-operating programs.

For example, a database program could allocate some pages and store data in them. If the program gives a name to the handle, another program that knows the name (say a word processor or spreadsheet) can find out from the EMM the handle number associated with the name. Knowing the handle, the second program can access the data stored in these pages. The handle number cannot be hard-coded into programs because this number can change at every execution, depending on the order in which various co-resident programs are loaded.

Physical page addressing. In previous EMS versions, page frames in the system address space are identified by frame number. For the configuration shown in Fig 1, the four frames in the primary window are numbered from 0 for the frame at segment C800H, to 3 for the one at D400H. In the EEMS im-

plementation, the first four frames are numbered the same way; frames at D800H and DC00H are designated as 4 and 5, respectively, and the ones beginning at 4000H in conventional memory are numbered from 6 upward. When mapping an expanded memory page, a program specifies its location in the system address space by supplying the frame number.

For compatibility, version 4.0 accepts the same frame numbering, but function 17 also allows specifying the location of the frame by its physical segment address. When calling this function to map a page into the lowest frame in conventional memory, a program could pass either 4000H or 6 as the frame argument. A subfunction code specifies the argument type.

Function 25 constructs an array of all page-frame addresses in the system; it is similar, but not identical, to EEMS function 41. Entries in the array returned by function 25 are ordered by physical address and consist of two words: the segment address and the frame number. Function 41 returns an array of bytes containing the high-order six bits of each frame's address; entries are ordered by frame number, with the address for frame *N* in the *N*th byte of the array. The new method is more convenient for physical addressing of page frames, the old one for addressing them by number.

Raw page sizes. The division of expanded memory into pages of 16k is not

| FUNCTION | DESCRIPTION |
|----------|-----------------------------------------------------------|
| 16 | Save/Restore partial page map |
| 17 | Map/Unmap multiple pages into page frames |
| 18 | Change number of pages allocated to a handle |
| 19 | Get/Set volatility attribute of a handle |
| 20 | Get/Set handle name |
| 21 | Get handle number for handle name |
| 22 | Alter page map and jump to a far address |
| 23 | Alter page map, call a far address, then restore page map |
| 24 | Move/Exchange contents of memory block |
| 25 | Get array of page-frame addresses |
| 26 | Get expanded memory hardware information |
| 27 | Allocate pages in sizes other than 16k blocks |
| 28 | Control alternate mapping registers |
| 29 | Prepare expanded memory hardware for warm reboot. |
| 30 | Disable/Enable system-level EMM functions |

Table 2 EMM functions added in EMS 4.0: the new EMS 4.0 functions that support the hardware features defined earlier by EEMS are inoperative on existing EMS boards. All EMS boards, however, can benefit from the functions that add software support for multi-tasking control programs

MEMORY

convenient when the hardware implements a paging scheme with a different page size. For example, the '386 processor performs paging in increments of 4k. In version 4.0, an EMM can be written to allocate expanded memory in units that are convenient to the hardware, provided that the size of such a unit, called a 'raw page', is a submultiple of 16k. Function 27 creates a raw handle and allocates to it a specified number of raw pages. All subsequent page operations for this handle, such as mapping pages or changing the allocation, must be specified in raw pages.

Multiple register sets. EEMS memory boards are constructed with two sets of page-mapping registers, so that a multi-tasking operating environment can switch rapidly between mapping contexts for two applications. Version 4.0 extends this to allow memory boards to provide any number of register sets. For boards with a single set of registers, function 28 simulates multiple sets by storing inactive mapping contexts in memory provided by the calling program. The effect of the simulation is the same as if the caller saved and restored contexts with function 15, but the advantage is that the context-switching method ap-

propriate to the capabilities of the hardware resides in the EMM; the calling program need not incorporate logic for both switching register sets and saving contexts.

An even more significant extension is the capability to dedicate specific register sets for use by the direct memory access (DMA) controller for high-speed transfers to and from expanded memory. A program can allocate a set of mapping registers for use by DMA, initiate a DMA request to or from the memory mapped by that set, and then switch the context by means of function 15 or 28. The DMA process continues to use the memory mapped by the DMA set of registers, while the CPU can access a different set of pages mapped by a different register set.

Version 4.0 does not require that EMS boards provide hardware for multiple register sets for either CPU or DMA access; it merely specifies that the EMM can take advantage of this capability if available. In contrast, an EEMS memory board must have exactly two sets of mapping registers for the CPU to use; a version 4.0 EMM also can use these registers. No boards currently on the market support any DMA registers or

more than two sets of CPU registers. Function 26 provides information about the EMS hardware: the raw page size, the number of alternative register sets (after the first), and the number of DMA register set.

Manipulating register sets should be done only by the operating system or task-switching executive. The system program can use function 30 to disable or enable functions 26 (get EMS hardware information), 28 (manipulate alternate register sets), and 30. Once disabled, these functions return error codes to all callers, even the system.

When function 30 is disabled, how does the system re-enable it? Immediately after installation of the EMM, the system-level functions are enabled by default. The first call to function 30 returns a random 32-bit access key that must be used on all subsequent calls to the function. Thereafter, function 30 can be called, even when disabled, if the key value is given as a parameter in the call. During its installation, the operating environment is presumably the first program to call function 30, so it is the only one to obtain the key.

Control and data transfers. In a multi-tasking environment, transfers of control

MEMORY

to various programs often involve changes in the mapping context. For example, when a task-switching executive activates a dormant program, it must map that program's expanded memory into the system address space before branching to the program's code. This capability is also useful within an application — for example, when a small TSR kernel in conventional memory maps in and branches to its main code in expanded memory.

Function 22 automates establishing a new mapping context and jumping to a far address. Logically, it is equivalent to a call to function 27 (map multiple pages), followed by a far jump to an absolute address. The target can be in expanded or conventional memory. As is the case for function 27, the page frames in the new mapping context can be specified either as frame numbers or absolute segment addresses.

Function 23 performs the equivalent of a far call. It saves the current mapping context in a caller-supplied memory area, establishes a new context (in the same way as function 27), and transfers control to a far address. When a far return is subsequently executed, the EMM restores the saved context before returning control to the instruction following the call to function 23.

Function 24 provides a means of efficiently moving or exchanging large blocks of data between expanded and conventional memory or between two areas of expanded memory. In the latter case, the two blocks might belong to different handles. Blocks need not be aligned on page or segment boundaries. Overlapping source and destination blocks are handled properly for a move but generate an error for an exchange. This function is useful for moving data between blocks that span several pages, especially when the blocks are larger than any available contiguous window. The calling program need not save and restore the context before calling this function, nor provide a save area for the EMM to use.

Nonvolatile handles. An EMS board can provide hardware features to maintain the contents of expanded memory through a warm boot created by a keyboard reset or other event that can be trapped by software. Function 19 applies the 'nonvolatile' attribute to a handle; only the data in pages belonging to nonvolatile handles survive a reboot.

Function 29 called just prior to reboot saves data on the EMS board; its specific action is not specified in EMS documentation because that depends on hardware implementation. Typically, function 29 writes information into EMS

| EEMS 3.2 FUNCTION, SUBFUNCTION | CLOSEST EMS 4.0 EQUIVALENT | DESCRIPTION |
|--------------------------------|----------------------------|----------------------------------------------------------------------------|
| 33 | None | Get addresses of page frames outside of conventional memory |
| 34 | None | Generic accelerator card support |
| 41 | 25 | Get addresses of all page frames in system |
| 42 | 5 | Map a page into any frame |
| 43, 0 | 16, 0 | Save partial page map |
| 43, 1 | 16, 1 | Restore partial page map |
| 43, 2 | None | Save and restore partial page map |
| 43, 3 | 16, 2 | Get size of save array |
| 43, 4, 5 | 28, 1, 2 | Switch to another set of map registers |
| 43, 6 | 18 | Deallocate pages mapped at initialisation to frames in conventional memory |

Table 3 Support for EEMS functions: version 4.0 of EEMS incorporates most of the enhancements introduced by EEMS version 3.2. Although these services are functionally similar in both versions, they require different calling sequences and produce output in a different format

hardware registers that notifies the EMM driver (when it subsequently reinstalls itself) which data are nonvolatile and therefore are not to be erased during initialisation.

Calling function 29 requires the detection of an imminent reboot, and this is not possible in all cases. A replacement interrupt 9 handler can detect a keyboard reset, but two other events can cause a reboot. One is to execute a direct jump to the boot code in ROM, the other is to toggle the processor's reset pin with a hardware reset button (this generates a hardware branch to the boot location). Software cannot detect either event, and the boot code, being in ROM, cannot be hooked by overwriting its entry point. Therefore, a keyboard reset is the only event that can be expected to preserve nonvolatile data.

Efficiency improvements. Function 16, 17, and 18 do not add major functionality to the EMM, but they provide services in a more efficient or convenient manner than is available by other means. Function 16 (save partial mapping context) is similar to 15 (save, restore, or swap EMM status to/from an external buffer), but it saves or restores only a specified subset of page-mapping registers, thereby consuming less conventional memory by storing only the essential portion of any context-switching information.

Function 17 (besides supporting physical frame addressing) can map more than one page at a time. It accepts an array of page numbers and page frames, so pages to be mapped need not be consecutive, nor page frames contiguous. As a result, a single call to function 17 can replace many calls to function 5.

Function 18 allows changing the number of pages allocated to a given handle. In previous versions, the page count per handle is fixed; if a process needs more pages, it has to obtain another handle. Handles are a limited resource (the absolute upper limit is 256 and the EMM typically defaults to a lower limit at installation), and obtaining and releasing them involves some overhead.

Still an individual

Although version 4.0 brings to EMS the major features of EEMS and adds others, it is not a proper superset of EEMS because it does not support all EEMS features (see Table 3). The missing ones do not diminish the utility of EMS 4.0; they merely indicate different design philosophies.

One of the primary design goals of EEMS is to maintain compatibility with EMS. Therefore, most extensions are implemented by new EMM function calls, not extensions to EMS-defined functions. The standard set of EMM function calls, shared with EMS (see Table 1), handles page frames only above conventional memory. The difference is that EEMS can handle more than four page frames in this area.

All other enhancements are supported by a set of functions with distinct numbers. EEMS deals with two sets of page frames: one containing only those above conventional memory, the other containing all frames in the address space. Each set has a distinct numbering scheme and separate mapping function.

The advantage of this EEMS approach is greater integrity. The upper page frames are for use by applications, the

lower by the operating system. Each program has its own set of functions, and although an application is not prevented from accessing any frame, mapping lower memory requires the conscious effort of a different function call. By contrast, version 4.0 considers all the page frames as one set — it has no EMM service equivalent to function 33, which returns an array of addresses of only those page frames above conventional memory. Both applications and the system use the same functions (5 and 27) to map upper and lower memory, so an errant program can inadvertently map out a page in conventional memory by supplying a wrong frame number or address.

Although EEMS provides slightly better security than EMS 4.0, all versions of EMS are generally weak in memory protection. This reflects the original purpose of EMS: expanding data storage memory in a single-tasking environment. A program is in no way prevented from accessing expanded memory pages belonging to another program. Access requires only a numeric handle, and any program can obtain a list of active handles. Handle numbers are assigned consecutively from a small range (0 through

255), so a failing program can inadvertently specify a handle belonging to another process. This is especially likely in the case of handle zero, which is owned by the operating system and consists of all pages initially mapped into the backfilled area of conventional memory.

Another feature introduced by EEMS that is not fully incorporated into EMS 4.0 is the interface with caching accelerator boards. Changing the contents of memory by remapping can invalidate the contents of an on-board cache that holds data from the mapped-out location. The caching hardware is unaware of the change because mapping changes the contents of memory by writing to an I/O port, not by direct access to affected locations.

EEMS provides function 34 for communicating with the accelerator board. Support for this function must be provided in both hardware and software by the manufacturer of the accelerator; details are available on request from AST Research. EMS 4.0 does not describe this hardware function, but in the section on EMM implementation guidelines, the documentation states, 'the support of function 34, as defined by AST, is encouraged'.

Just the beginning

Version 4.0 EMM drivers are now available for existing expanded memory boards of both the EMS and EEMS variety. Owners should contact the manufacturers of their boards for information on obtaining updated software.

For this article, four different EMM drivers were tested on five memory boards: Intel's for the Above Board AT, AST's for the RAMpage and RAMpage AT, Quarterdeck's for the IBM 80236 Memory Expansion board for PS/2 Models 50 and 60, and Intel's for the Above Board 2 for Models 50 and 60. The boards for the PC and AT were older models originally supplied with version 3.2 drivers. Version 4.0 drivers for the PC and AT ran only with the boards from their manufacturers. The Intel PS/2 driver refused to install itself without an Above Board 2 but could combine memory from the Above Board 2 with memory from an IBM expansion board. The Quarterdeck PS/2 driver worked with either Intel or IBM memory boards or a combination.

All drivers worked well in the tests. The AST and Quarterdeck software supports functions of both the new version 4.0 and older EEMS version 3.2, giving them full functionality with current versions of both DESQview and Windows. Intel's drivers implement only 4.0, so they do

not fully support DESQview (which predates version 4.0 and thus issues only EEMS 3.2 calls). The Above Board 2 (but not Above Board models for the PC and AT) provides equivalent context-switching support for Windows 2.03.

Even with an old board, running a new driver provides many innovations, such as named handles, physical addressing, transfer control, and other improvements in efficiency of managing paged memory. These improvements will lie dormant until programs are written to take advantage of them. However, these software features are not 4.0's primary advantages.

Achieving a spectacular improvement in the performance of DESQview or Windows requires hardware support for mapping conventional memory and multiple register sets; these are not implemented on current EMS boards. Version 4.0 does very little for existing EMS boards because they cannot use the most significant new features. For the most part, the effect of installing a new EMM driver on an old board is that the error response to version 4.0 functions is 'not implemented' instead of the old driver's 'invalid function'.

On the other hand, EEMS boards already have the hardware to support the EMS 4.0 mapping protocol. The only two innovations not already supported by EEMS hardware are more than two sets of general mapping registers and mapping registers that are dedicated to DMA transfers. The EEMS 3.2 and EMS 4.0 drivers for EEMS boards provide the same functionality with one immediate exception: with the older driver, Windows 2.0x cannot take advantage of the multi-tasking hardware features of the board.

For those who already own expanded memory boards, EMS 4.0 does little — EMS boards cannot use the major new features, and EEMS boards already have them. Still, the creation of a single unified standard that incorporates and adds to the features of the better one is undoubtedly a step in the right direction. This unification is just in time to provide a consistent memory standard for IBM's PS/2 series — the memory-mapping capability of the Micro Channel memory expansion board no doubt played heavily in the decision to create 4.0.

The capabilities offered by the newly unified EMS to task-switching environments breathes new life into DOS-based applications, providing the opportunity to design characteristics not otherwise available in this operating system: large memory spaces for both programs and data, rapid switching between co-resident programs, and concurrent execution.

END

Christensen Protocol

The Christensen file transfer protocol (also called Modem and Xmodem) has been around for quite a few years now and like many things from the past, it's veiled in mystery and misunderstood by many.

The protocol sends data in 128 byte blocks within sequentially numbered packets. Each packet includes a packet header, the data, and a checksum of the packet (excluding the checksum itself which is sent as a separate byte after the 128 byte block). As the receiving computer stores the incoming data in its memory, it performs its own checksum of the packet, then compares it with that in the packet. A successful comparison means that all of the information was sent correctly and the receiving computer acknowledges this by sending an ACK (ASCII for Acknowledge Hex 06) to the sending computer. If the comparison fails, the receiving computer tells the sender to resend the packet by sending a NAK (ASCII for Negative Acknowledge Hex 15).

When a file transfer is initiated there is a possible problem in that the sending computer will wait for the receiving computer and vice versa, which would mean that you get nowhere at all! What happens is that the receiving computer sends a NAK to the sender to initiate the transfer; if that doesn't happen, the sending computer starts transmitting the first block after 10 seconds. This consists of a SOH (ASCII for Start of Heading, Hex 01), the block number, and the two's complement of the block number, then the 128 bytes of data followed by the checksum (132 bytes all up) — see Figure a).

This process continues with the next packet of information. What happens when the receiving computer sends a NAK? Simply, the sender transmits it

again (just like when someone says 'Beg your pardon?' when you're talking — see Figure b). What happens if the sending computer doesn't 'see' the ACK/NAK that the receiver sends? Well, after 10 seconds it will resend the packet; the receiving computer knows it already has that packet and just sends an ACK when the packet ends — all fixed!

How does the receiver know what packet it's getting? The block number is incremented by one for each packet sent and the receiver also adds the SOH, Block Number, and the two's complement of the block number and it should get zero (for example 01 [SOH] + 01 + FE is 00) — so there is a double check of the block num-

ber, in addition to the check on the whole packet.

Normal completion of a file transfer is when the sending computer transmits an EOT (ASCII for End of Transmission Hex 04), the receiving computer then sends a confirmatory ACK — see Figure c), if the EOT is lost then the receiver will continue to wait for the next block, sending NAKs every 10 seconds; after 10 NAKs, it'll give up and time out.

One thing that is often overlooked is that the receiving computer must be capable of receiving the entire packet at the full speed of the communication link; if it can't, this can cause problems at high speeds. □

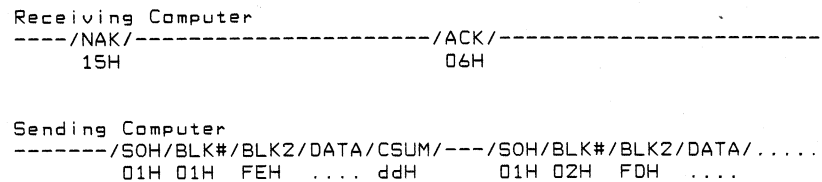


Figure a) Packet Organisation and Block Numbering.

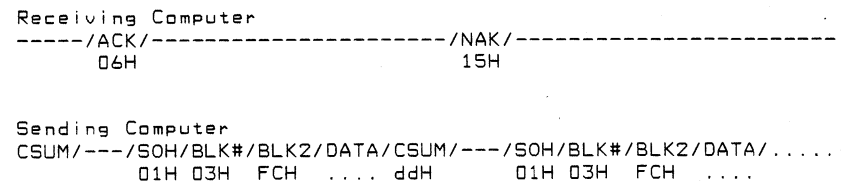


Figure b) 'Beg your pardon?'

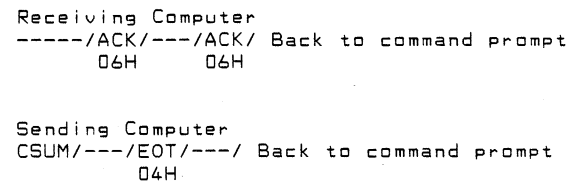


Figure c) End of file sequence.

NEW SOUTH WALES

NSW Contact:
 Prophet TBBS (02) 6285222
 FidoNet [712/606]
 *** AMENDED ***
 ABCOM
 Sysop: Ben Sharif
 FIDONet: [712/304]
 Phone: (047) 364165
 Baud: V21.V22.V22bis.V23
 Access: Mem.VA
 Computer: IBM XT
 DOS: PCDOS
 BBSSoftware: Fido
 ACE (NSW) BBS
 Sysop: Larry O'Keefe
 Phone: (02) 5292059
 Baud: V21

Access: Mem.Reg.LVA
 Computer: Atari
 DOS: Atari
 BBSSoftware: Michtron
 *** AMENDED ***
 Alpha Juno BBS
 Sysop: Kevin Withnall
 FIDONet: [620/701]
 Phone: (02) 7741543
 Baud: V21.V22.V22bis
 *** AMENDED ***
 Amstrad ABBS
 Sysop: Riccay Schmahl
 FIDONet: [711/903]
 Phone: (02) 9812966
 Baud: V21.V22.V22bis.V23
 Access: Reg.VA
 Computer: Amstrad PC1512
 DOS: MSDOS 3.2

BBSSoftware: Opus
 Apple Users Group BBS
 Sysop: Matthew Barnes
 & Andrew Riley
 Phone: (02) 4987084
 Baud: V21.V22.V22bis.V23
 Access: Mem.VA
 *** AMENDED ***
 ArcoTel BBS
 Sysop: Alex Szx
 FIDONet: [620/601]
 Phone: (02) 6833956
 Baud: V21.V22.V22bis.V23
 Access: Mem.VA
 BBSSoftware: Opus
 *** AMENDED ***
 AUGUR TBBS
 Sysop: Mark James
 Phone: (02) 6614739

Baud: V21.V22.V22bis.V23
 Access: Reg.VA
 Computer: PC Clone
 DOS: PC
 BBSSoftware: TBBS 2.0m
 Ausborne (Osborne) RCPM
 Sysop: Daniel Moran
 Phone: (02) 4397072
 Access: Mem.VA
 Australian Pick User's BBS
 Sysop: Kurt Johannessen
 Phone: (02) 6318603
 Baud: V21.V22.V22bis.V23
 Access: Reg.VA
 BeeHive BBS
 Sysop: Paul Pinches
 Phone: (02) 5205181
 Baud: V21.V22
 Access: Mem.Reg

ETI data sheet

INTERSIL ICL 8038

Waveform Generator/V.C.O.

The 8038 has been around for about 5 years — which is a long time in electronics. In fact it has reached the position of becoming an 'Industry Standard' on a par with the 741. An inherently versatile device it has its drawbacks like most chips — but overall has a lot going for it. Intersil even produced a very honest application bulletin (A013) called 'Everything you always wanted to know about the 8038', which explained how to get the best out of this device and admitted its defects — an uncommon event with most manufacturers! Some of the data from A013 has been included in this data sheet, but for more information ask for application bulletins A012, A013, and the latest information sheet. Intersil are distributed in Australia by R & D Electronics Pty. Ltd., 23 Burwood Rd., Burwood, 3125.

Description

The 8038 Waveform Generator is a monolithic integrated circuit, capable of producing sine, square, triangular, sawtooth and pulse waveforms of high accuracy. The frequency (or repetition rate) can be selected externally over a range of less than 1/1000 Hz to more than 1 MHz and is highly stable over a wide temperature and supply voltage range. Frequency modulation and sweeping can be accomplished with an external voltage and the frequency can be programmed digitally through the use of either resistors or capacitors. The Waveform Generator utilizes advanced monolithic technology, such as thin film resistors and Schottky-barrier diodes.

Theory of operation

A block-diagram of the waveform generator is shown in Figure 1. An external capacitor C is charged and discharged by two current sources. Current source #2 is switched on and off by a flip-flop, while current source #1 is on continuously. Assuming that the flip-flop is in a state such that current source #2 is off, then the capacitor is charged with a current I. Thus the voltage across the capacitor rises linearly with time. When this voltage reaches the level of comparator #1 (set at 2/3 of the supply voltage), the flip-flop is triggered, changes states, and releases current source #2. This current source normally carries a current 2I, thus the capacitor is discharged with a net-current I and the voltage across it drops linearly with time. When it has reached the level of comparator #2 (set at 1/3 of the supply voltage), the flip-flop is triggered into its original state and the cycle starts anew.

Four waveforms are readily obtainable from this basic generator circuit. With the current sources set at I and 2I respectively, the charge and discharge times are equal.

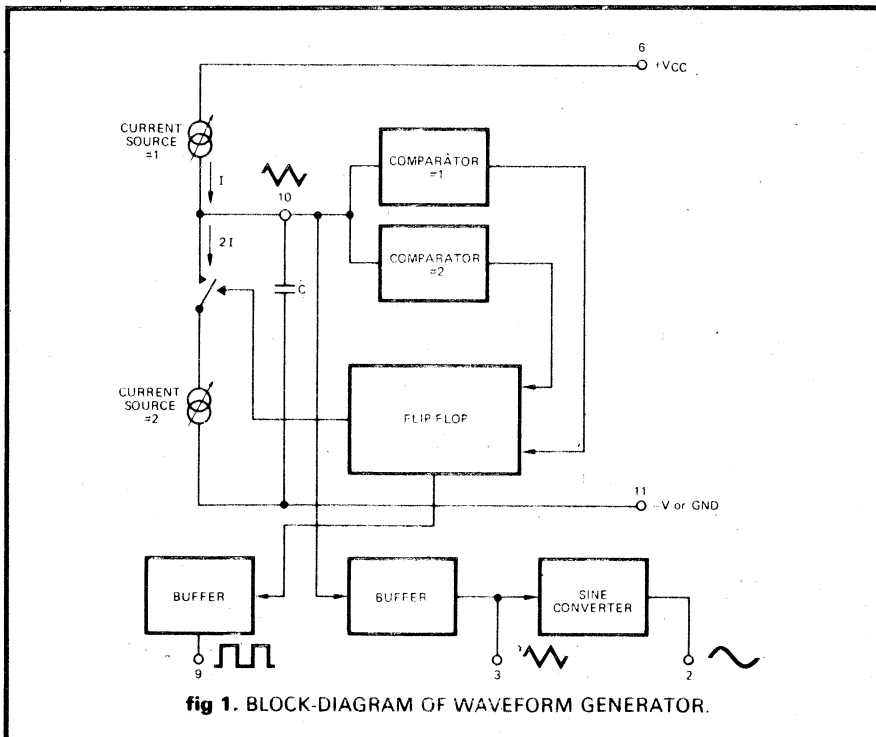
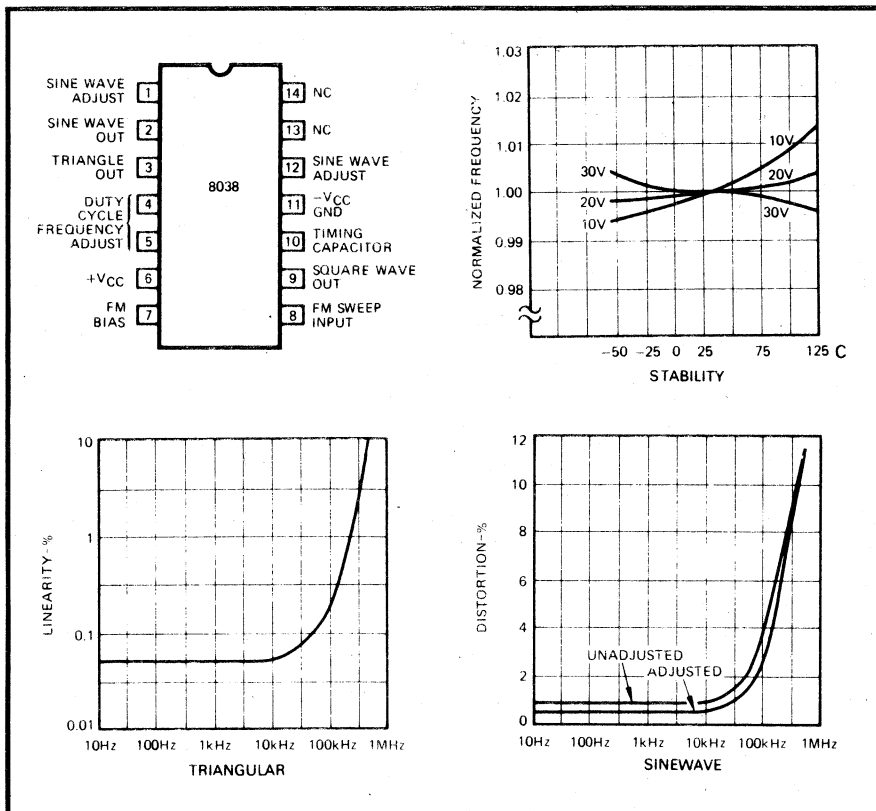


fig 1. BLOCK-DIAGRAM OF WAVEFORM GENERATOR.

Thus a triangle waveform is created across the capacitor and the flip-flop produces a square-wave. Both waveforms are fed to buffer stages and are available at pins 3 and 9.

The levels of the current sources can, however, be selected over a wide range with two external resistors. Therefore, with the two currents set at values different from 1 and 21, an asymmetrical sawtooth appears at terminal 3 and pulses with a duty cycle from less than 1% to greater than 99% are available at terminal 9.

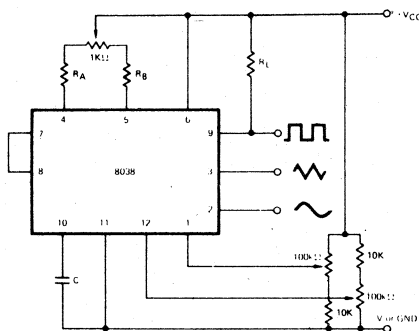
The sine-wave is created by feeding the triangle-wave into a non-linear network (sine-converter). This network provides a decreasing shunt-impedance as the potential of the triangle moves toward the two extremes.

Power Supply

The waveform generator can be operated either from a single power supply (10 to 30 Volts) or a dual power supply (± 5 to ± 15 Volts). With a single power supply the average levels of the triangle and sine-wave are at exactly one-half of the supply voltage, while the square wave alternates between +V and ground. A split power supply has the advantage that all waveforms move symmetrically about ground.

Also notice that the square wave output is not committed. The load resistor can be connected to a different power supply, as long as the applied voltage remains within the breakdown capability of the waveform generator (30 V). In this way, for example, the square-wave output can be made TTL compatible (load resistor connected to +5 Volts) while the waveform generator itself is powered from a much higher voltage.

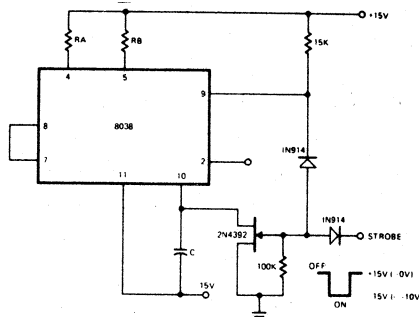
Purity



The symmetry of all waveforms can be adjusted with the external timing resistors. To minimize sine-wave distortion the resistors between pins 11 and 12 are best made variable ones. With this arrangement distortion of less than 1% is achievable. To reduce this even further, two potentiometers can be connected as shown. This configuration allows a reduction of sinewave distortion close to 0.5%.

Both the sine-wave and triangular outputs, are only useful up to about 20kHz if a reasonably pure signal is required. A perusal of the graphs will show why.

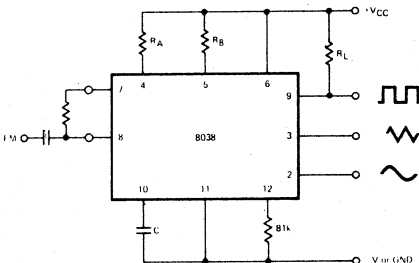
Strobe



With a dual supply voltage (e.g., ± 15 V) the external capacitor (pin 10) can be shorted to ground so that the sine wave and triangle wave always begin at a zero crossing point. Random switching has a 50/50 chance of starting on a positive or negative slope. A simple AND gate using pin 9 will allow the strobe to act only on one slope or the other.

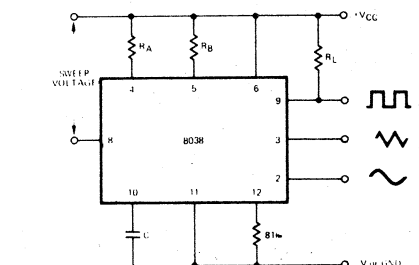
Using only a single supply, the capacitor (pin 10) can be switched either to V+ or ground to force the comparator to set in either the charge or discharge mode. The disadvantage of this technique is that the beginning cycle of the next burst will be 30% longer than the normal cycle.

F.M. and Sweeping



The frequency of the waveform generator is a direct function of the DC voltage at terminal 8 (measured from +VCC). Thus by altering this voltage, frequency modulation is achieved.

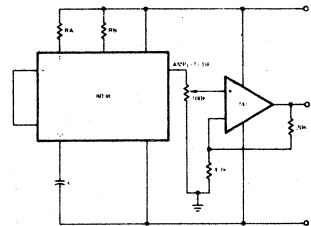
For small deviations (i.e. $\pm 10\%$) the modulating signal can be applied directly to pin 8, merely providing dc decoupling with a capacitor. An external resistor between pins 7 and 8 is not necessary, but it can be used to increase input impedance. Without it (i.e. terminals 7 and 8 connected together), the input impedance is 8k, with it, this impedance increases to $(R+8k)$.



For larger FM deviations or for frequency sweeping, the modulating signal is applied between the positive supply voltage and pin

8. In this way the entire bias for the current sources is created by the modulating signal and a very large (e.g. 1000:1) sweep range is created ($f = 0$ at $V_{\text{sweep}} = 0$). Care must be taken, however, to regulate the supply voltage; in this configuration the charge current is no longer a function of the supply voltage (yet the trigger thresholds still are) and thus the frequency becomes dependent on the supply voltage. The potential on pin 8 may be swept from V_{cc} to about $2/3 V_{\text{cc}}$.

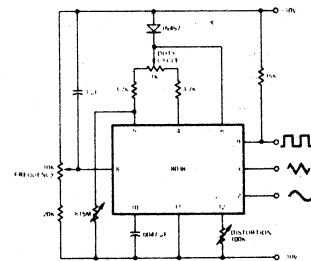
Buffering



The sine wave output has a relatively high output impedance (1K Typ). The circuit provides buffering, gain and amplitude adjustment. A simple op amp follower could also be used.

If the available outputs are all fed through a buffer, extra resistors can be inserted in series with the signal before a switch. Values of 47k (square wave), 15k (triangular) and 10k (sine wave) will ensure equal amplitude signals.

Audio Oscillator



To obtain a 1000:1 Sweep Range on the 8038 the voltage across external resistors RA and RB must decrease to nearly zero. This requires that the highest voltage on control Pin 8 exceed the voltage at the top of RA and RB by a few hundred millivolts.

The Circuit achieves this by using a diode to lower the effective supply voltage on the 8038. The large resistor on pin 5 helps reduce duty cycle variations with sweep. The range of this circuit is 20Hz to 20 kHz, output buffer can be added to make a general purpose bench unit.

Points to Note!

The 8038 runs hot to touch, this is normal, and is due to the resistive nature of the sinewave shaping network.

The optimum supply voltage, for minimum temperature drift is 20V, this can be seen in the stability graph.

ETI data sheet

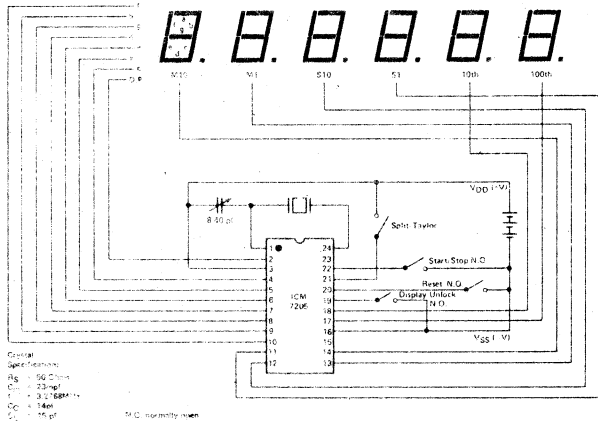
ICM 7205 Stopwatch Chip

The Intersil ICM 7205 is a relatively new device, the main points of interest being: on-chip display drivers, full protection against static — no special handling precautions are required —, and an average current when in operation (including display!) of only 10 mA.

The ICM 7205 is a fully-integrated CMOS six digit stopwatch circuit. The circuit interfaces directly with a six digit/seven segment common cathode LED display. The low battery indicator can be connected directly to the decimal point anode or to a separate LED lamp. The only components required for a complete stopwatch besides the display are: three SPST switches, a 3.2768 MHz crystal, a trimming capacitor, three AA batteries, and an on/off switch. For a two function stopwatch, one additional switch would be required.

The circuit divides the oscillator frequency by 2^{15} to obtain 100 Hz which is fed to the fractional seconds, seconds and minutes counters. An intermediate frequency is used to obtain the 1/6 duty cycle 1.07 kHz multiplex waveforms. The blanking logic provides leading zero blanking for seconds and minutes independently of the clock. The ICM7205 is packaged in a 24 lead plastic DIP.

Stopwatch Circuit



Switch Characteristics

The ICM 7205 is designed for use with SPST switches throughout. On the display unlock and reset the characteristics of the switches are unimportant, since the circuit responds to a logic level for any length of time, however short. Switch bounce on these inputs does not need to be specified. The Start/Stop input, however, responds to an edge, and it requires a switch with less than 15 ms of switch bounce. The bounce protection circuitry has been specifically designed to let the circuit respond to the first edge of the signal, so as to preserve the full accuracy of the system.

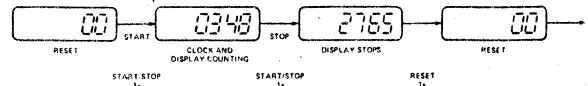
Low Battery Indicator

The on-chip low battery indicator is intended for use with a small LED lamp or with the decimal points on a standard LED display. The output is the drain of a P-channel transistor of approximately half the size of one of the segment drivers. The LBI circuitry is designed always to provide a voltage difference between the LBI trigger voltage and the minimum operating voltage, i.e., the lower the LBI trigger voltage the lower the minimum operating voltage. In this way a stopwatch using three AA batteries will provide at least 15 minutes of accurate timekeeping after the LBI comes on.

Functional Operation

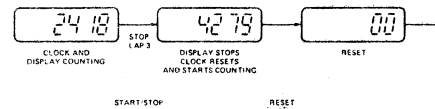
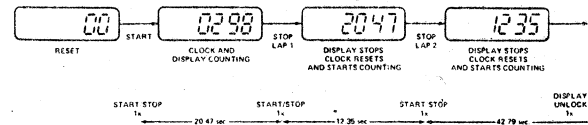
Turning on the stopwatch will bring up the reset state where the fractional seconds are on displaying 00 and the other digits are blanked. This display always indicates that the stopwatch is ready to go.

Start/Stop



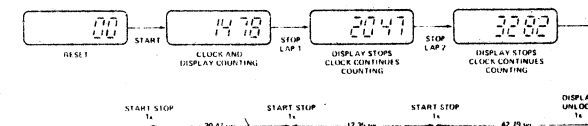
The Start/Stop modes can be used for a single event timing with the Split/Taylor input in either state. The illustration indicates the operations and the results. To time another event the reset switch must be used prior to the start of the event. Seconds will be displayed after one second, minutes after one minute. The range of the stopwatch is 59 minutes 59.99 seconds. If an event exceeds one hour, the number of hours must be remembered by the user. Leading zeros are not blanked after one hour.

Taylor



When the Split/Taylor input is left open circuit or is connected to Vss, the stopwatch can be used in the Taylor or sequential mode. As depicted graphically above, each split time is measured from zero in the Taylor mode, i.e., after stopping the watch, the counters reset to zero momentarily and start counting the next interval. The time displayed is the time elapsed since the last activation of Start/Stop. The display is stationary after the first interval unless the display unlock is used to show the running clock. Reset can be used at any time.

Split



When the Split/Taylor input is connected to VDD the stopwatch is in the Split mode. The Split mode differs from the Taylor in that the lap times are cumulative in the Split mode. The counters do not reset or stop after the first start until reset is activated. Any time displayed is the cumulative time elapsed since the first start after reset. Display unlock can be used to let the display 'catch up' with the clock. Reset can be used at any time.

Points to Note!

Absolute maximum supply voltage is 5V5. Never short outputs to earth or low impedance power supply as this will destroy the device.

ETI data sheet

INTERSIL ICM 7208

FEATURES:

- Useful for:
 - Unit counter
 - Frequency counter
 - Period counter
- Low operating power dissipation < 10mW
- Low quiescent power dissipation < 5mW
- Counts and displays 7 decades
- Wide operating supply voltage range
 $2V \leq IV_{DD} - V_{SS} \leq 6V$
- Drives directly 7 decade multiplexed common cathode LED display
- Internal store capability
- Internal inhibit to counter input
- Test speedup point
- All terminals protected against static discharge

DESCRIPTION

The ICM 7208 is a fully integrated seven decade counter-decoder-driver and is manufactured using the Intersil low voltage metal gate C-MOS process. As such it has applications as either a unit, frequency or period counter. For unit counter applications the only additional components are a 7 digit common cathode display, 3 resistors and a capacitor to generate the multiplex frequency reference, and the control switches.

Specifically the ICM 7208 provides the following on chip functions: a 7 decade counter, multiplexer, 7 segment decoder, digit & segment drivers, plus additional logic for display blanking reset, input inhibit, and display on/off.

The ICM 7208 is intended to operate over a supply voltage of 2 to 6 volts as a medium speed counter or over a more restricted voltage range for high frequency applications.

As frequency counter it is recommended that the ICM 7208 be used in conjunction with the ICM 7207 Oscillator Controller which provides a stable HF oscillator, and output signal gating.

TESTING PROCEDURES

The ICM 7208 is provided with three input terminals: 7,23,27 which may be used to accelerate testing. The least two significant decade counters may be tested by applying an input to the 'COUNTER INPUT' terminal 12. 'TEST POINT'

terminal 23 provides an input which bypasses the 2 least significant decade counter. Similarly terminals 7 and 27 permit rapid counter advancing at two points further along the string of decade counters.

COUNTER INPUT DEFINITION

The internal counters of the ICM 7208 index on the negative edge of the input signal at terminal #12.

Format of Signal to be Counted

The noise immunity of the Signal Input Terminal is approximately 1/3 the supply voltage. Consequently, the input signal should be at least 50% of the supply in peak to peak amplitude and preferably equal to the supply. **NOTE: The amplitude of the input signal should not exceed the supply; otherwise, damage**

Fig. 1. Pinout.

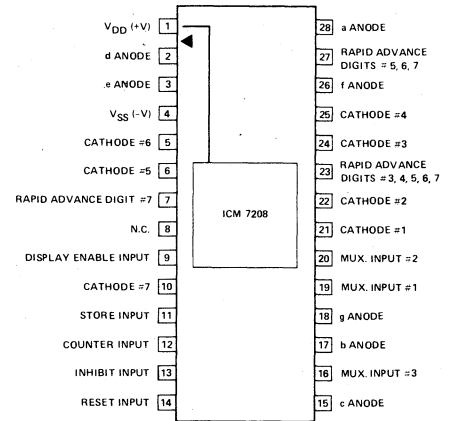


Fig. 2. Absolute maximum ratings.

| | |
|---------------------------------------------|----------------------------------|
| Power Dissipation (Note 1) | 1 watt |
| Supply voltage $ V_{DD} - V_{SS} $ (Note 2) | 6 V |
| Output digit drive current (Note 3) | 150 mA |
| Output segment drive current | 30 mA |
| Input voltage range (any input terminal) | Not to exceed the supply voltage |
| Operating temperature range | -20°C to +70°C |
| Storage temperature range | -55°C to +125°C |

*Absolute maximum rating define parameter limits that if exceeded may permanently damage the device.

Fig. 3. Typical operating characteristics.

($V_{DD} - V_{SS} = 5V$, $T_A = 25^\circ C$, TEST CIRCUIT, display off, unless otherwise specified)

| PARAMETER | SYMBOL | CONDITIONS | MIN | TYP | MAX | UNITS |
|--------------------------------------------|-----------|---------------------------------------------------------------------------------------------------------|-----|-----|-----|---------|
| Quiescent Current | I_{DD1} | All controls plus terminal 20 connected to V_{DD} . No multiplex oscillator | | 30 | 100 | μA |
| Quiescent Current | I_{DD2} | All control inputs plus terminal 20 connected to V_{DD} except store which is connected to V_{SS} . | | 70 | 150 | μA |
| Operating Supply Current | I_{DDS} | All inputs connected to V_{DD} . RC multiplexer osc operating $f_{in} < 25KHz$ | | 210 | 500 | μA |
| Operating Supply Current | | $f_{in} = 2MHz$ | | | 700 | μA |
| Supply Voltage Range | V_{DD} | $f_{in} = 2MHz$ | 3.5 | | 5.5 | V |
| Digit Driver On Resistance | R_D | | | 4 | 12 | ohm |
| Digit Driver Leakage Current | I_D | | | | 500 | μA |
| Segment Driver On Resistance | R_S | | | 40 | | ohm |
| Segment Driver Leakage Current | I_S | | | | 500 | μA |
| Pullup Resistance of Reset or Store Inputs | R_p | | 100 | 400 | | Kohms |
| Counter Input Resistance | R_{IN} | Terminal 12 either at V_{DD} or V_{SS} potentials | | | 100 | Kohms |
| Counter Input Hysteresis Voltage | V_{HIN} | | | 25 | 50 | mV |

- NOTE 1 This value of power dissipation refers to that of the package and will not be obtained under normal operating conditions.
- NOTE 2 The supply voltage must be applied before or at the same time as any input voltage. This poses no problems with a single power supply system. If a multiple power supply system is used, it is mandatory that the supply for the ICM 7208 is not switched on after the other supplies otherwise the device may be permanently damaged.
- NOTE 3 The output digit drive current must be limited to 150 mA or less under steady state conditions. (Short term transients up to 250 mA will not damage the device.) Therefore, depending upon the LED display and the supply voltage to be used it may be necessary to include additional segment series resistors to limit the digit currents.

may be done to the circuit.

The optimum input signal is a 50% duty cycle square wave equal in amplitude to the supply. However, as long as the rate of change of voltage is not less than approximately $10^{-4}V/\mu\text{sec}$ at 50% of the power supply voltage, the input waveshape can be inusoidal, triangular, etc.

Display Considerations

Any common cathode multiplexable LED display may be used. However, if the peak digit currents exceeds 150 mA for any prolonged time, it is recommended that resistors be included in series with the segment outputs (terminals 2, 3, 15, 17, 18, 26, 28) to limit current to 150 mA.

The ICM 7208 is specified with $500\mu\text{A}$ of possible digit leakage current. With certain new LED displays that are extremely efficient at low currents, it may be necessary to include resistors between the cathode outputs and the positive supply V_{DD} to bleed off this leakage current.

Display Multiplex Rate

The multiplex frequency reference is divided by eight to generate an 8 bit sequencer. Thus the display multiplex rate is one eighth of the multiplex frequency reference.

The ICM 7208 has approximately $0.5\mu\text{s}$ overlap between output drive signals. Therefore, if the multiplex rate is very fast, digit ghosting will occur. The ghosting determines the upper limit for the multiplex frequency reference. At very low multiplex rates flicker becomes visible.

It is recommended that the display multiplex rate be within the range of 50 Hz to 200 Hz which corresponds to 400 Hz to 1600 Hz for the reference frequency.

CONTROL INPUT DEFINITIONS

| INPUT | TMNL | VLTG | FUNCTION |
|------------|------|----------------------|--------------------------------------------------------------|
| 1. Display | 9 | V_{DD} V_{SS} | Display on Display off |
| 2. Store | 11 | V_{DD} V_{SS} | Counter Inform. Stored Counter Inform. Transferring |
| 3. Inhibit | 13 | V_{DD} V_{SS} | Input to Counter Blocked Normal Opertn. |
| 4. Reset | 14 | V_{DD} V_{SS} | Normal Opertn. Counters Reset |

Fig. 4. Typical performance characteristics.

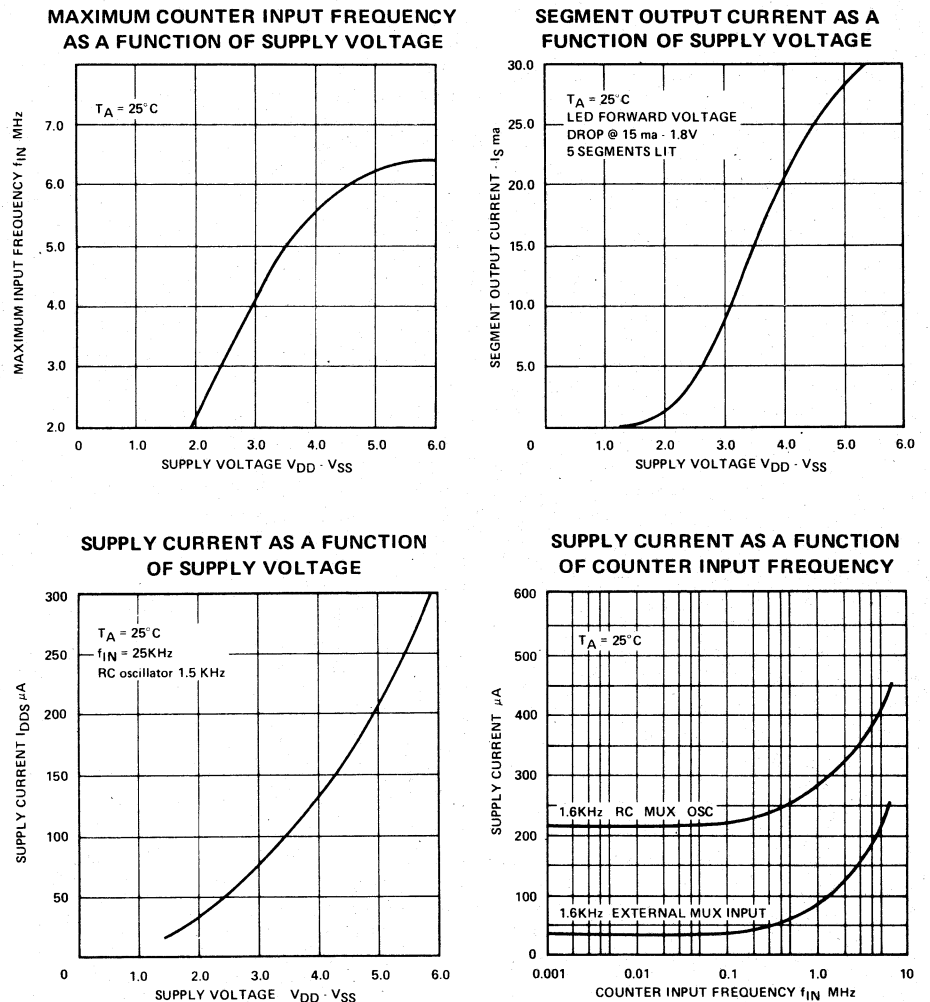
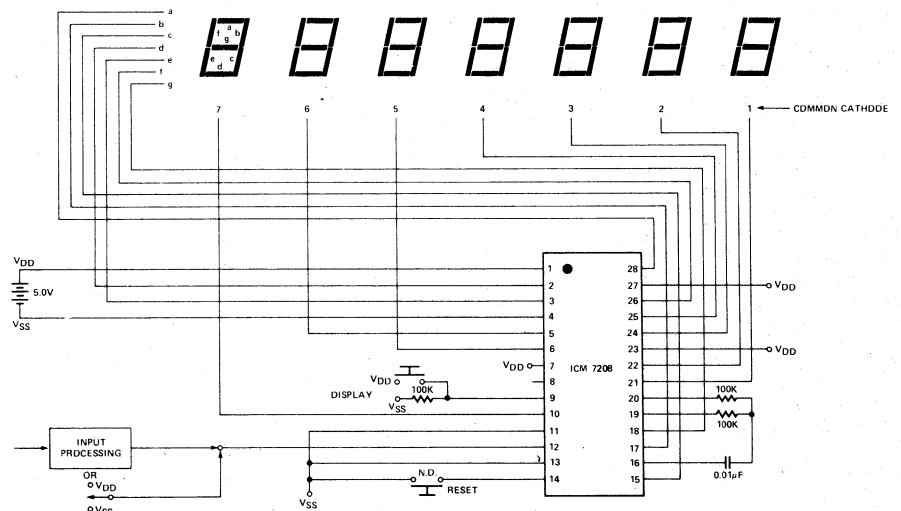


Fig. 5. Unit counter schematic.



ETI data sheet

Unit Counter

The unit counter updates the display for each negative transition of the input signal. The information on the display will count after reset from 00 to 9,999,999 and then will reset to 0000000 and will begin to count up again. To blank leading zeros actuate reset at the beginning of a count. Leading zero blanking affects two digits at a time.

For battery operated systems the display may be switched off to conserve power.

An external generator may be used to provide the multiplex frequency reference (input terminal 20). The signal applied to terminal 19 (terminals 16 and 20 open circuit) should be approximately equal to the supply voltage and for minimum power dissipation should be a square wave.

For stand alone systems two inverters are provided so that a simple but stable RC oscillator may be built using only 2 resistors and a capacitor.

Figure 5 shows the schematic of an extremely simple unit counter that can be used for remote traffic counting, to name one application. The power cell stack should consist of 3 or 4 nickel cadmium rechargeable cells (nominal 3.6 or 4.8 volts). If 4 x 1.5 volt cells are used it is recommended that a diode be placed in series with the stack to guarantee that the supply voltage does not exceed 6 volts.

The input switch is shown to be a single pole double throw switch (SPDT). A single pole single throw switch (SPST) could also be used with a pullup resistor. However, anti-bounce circuitry must be included in series with the counter input. In order to avoid all contact bounce problems due to the SPDT switch the ICM 7208 contains an input latch on chip.

Fig. 8. Frequency counter input waveforms.

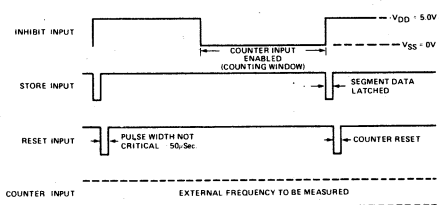
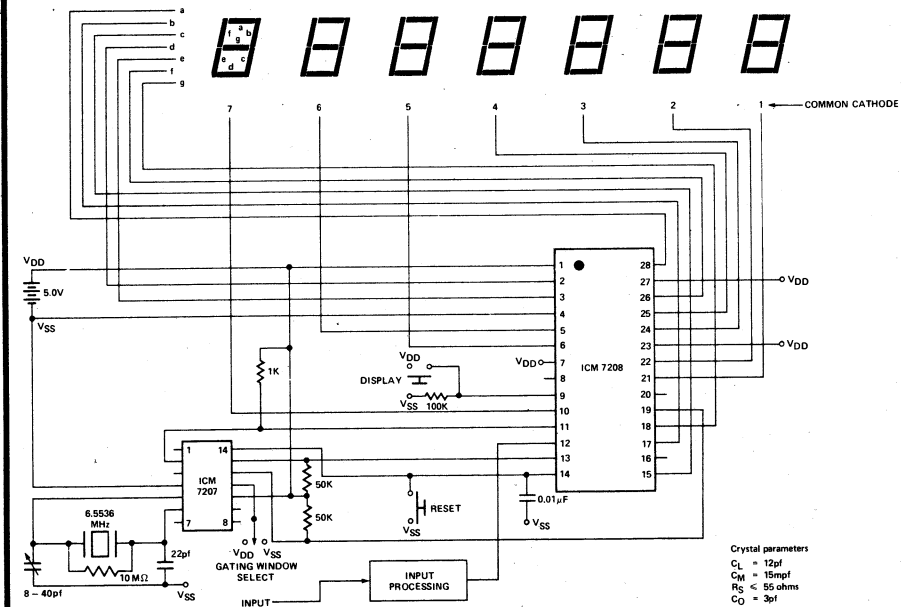
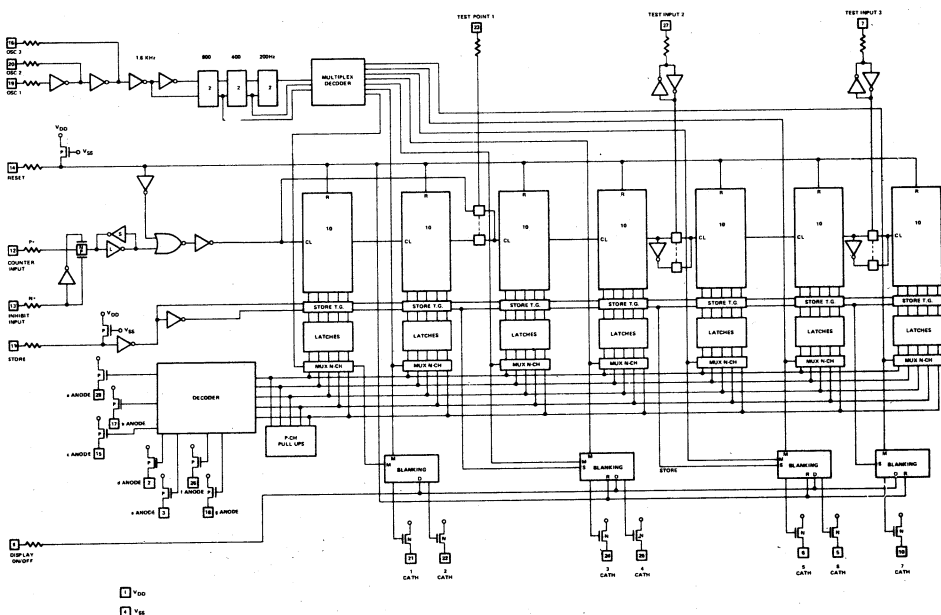


Fig. 6. Frequency counter schematic.



Note: For a 1 sec count window which allows all 7 digits to be used with a resolution of 1 Hz, the ICM7207 can be replaced with the ICM7207A. Circuit details are given on the 7207A data sheet.

Fig. 7. Internal block diagram.



USING THE LM 3900N

Four amplifiers on a single chip

ONE of the most noticeable trends in modern electronics is for more and more components to be packed into smaller and smaller spaces. One example of this is the fairly new LM3900 device manufactured by the National Semiconductor Company. It contains four separate internally compensated amplifiers in a single 14 pin dual-in-line encapsulation.

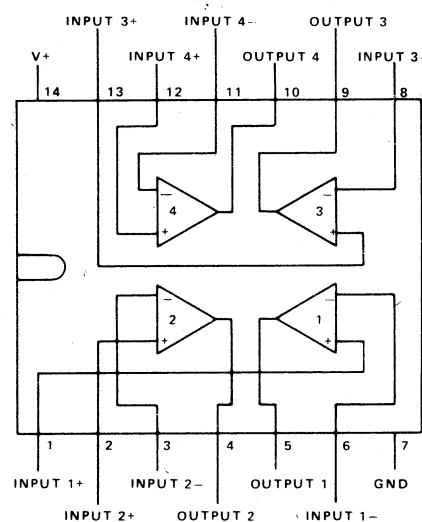
All four amplifiers are fabricated on a single silicon chip. Each amplifier contains seven transistors, a diode and a capacitor, whilst other internal components are used in the bias and power supplies.

One might expect that new devices of this type would be quite expensive, but the LM3900 is available at just over \$2.00 each — or even less in quantities of ten or more.

two voltages (as in a conventional amplifier).

The type of amplifier used in the LM3900N may be referred to as a 'Norton' amplifier, since Norton is the name of the person who developed a theorem relating the *current* flowing in a circuit to the equivalent current generator and shunt impedance.

Fig.1. The connections of the LM3900N.



CONNECTIONS

The connections of the four separate amplifiers are shown in Fig.1. Each amplifier has a non-inverting input (marked +), an inverting input (marked -) and an output connection.

In addition, there is a single common positive supply connection and a common ground connection (negative supply line) for the whole device.

INTERNAL CIRCUIT

Conventional high gain amplifiers employ a differential input stage to provide inverting and non-inverting inputs, but a rather different approach is employed in the LM3900N. A 'current mirror' is employed in the non-inverting input circuit, the current 'reflected' in this mirror being subtracted from that which enters the inverting input.

This type of amplifier therefore acts as a differential stage by amplifying the difference between two *currents* rather than the difference between

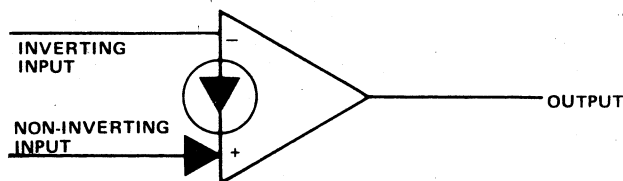


Fig.2. The symbol for one of the Norton amplifiers of the LM3900N.

SYMBOL

The symbol recommended for each of the four Norton amplifier stages in the device is shown in Fig.2. This symbol distinguishes this type of amplifier from the standard operational amplifier symbol and avoids confusion in circuits.

The symbol of Fig. 2 contains an indication that there is a current source between the inverting and non-inverting inputs and implies that the amplifier uses a current mode of operation. In addition, the circuit symbol indicates that current is removed from the inverting input, whilst the arrow on the non-inverting input shows that this functions as a current input.

PERFORMANCE

The LM3900N has the advantage that it can operate from a single supply voltage over the range of four volts to 36 volts. Most conventional operational amplifiers require supplies symmetrical with respect to ground (typically ± 15 V); the LM3900N can be used with such supply lines if desired.

The maximum peak to peak output amplitude of an LM3900N amplifier is only 1 V less than the supply voltage employed. The current consumed from the power supply is typically 6.2 mA (maximum 10 mA).

The typical voltage gain of each amplifier is 2800 or nearly 70 dB. The