Design Engineer's Reference Series

microNOVA® INTEGRATED CIRCUITS DATA MANUAL

Data General

Data General Corporation, Westboro, Massachusetts 01581

Notice

Ordering No. 014-000074

Printed in the United States of America
Rev. 04, September 1981

CONTENTS

Data General

Data General Corporation, Westboro, Massachusetts 01581

1

OVERVIEW OF microNOVA LINE INTEGRATED CIRCUITS

INTRODUCTION

iis data manual describes Data General's full microcomputer chip set. The microcomputer consists of three principal tegrated circuits: a 16-bit microprocessor, an I/O controller, and a 4Kx1 dynamic RAM. As part of the set, each integrated rcuit is supported by the necessary system buffer elements. The principal chips are fabricated with N-channel, silicon gate, etal oxide semiconductor (NMOS) technology. The system buffer chips are small and medium scale integration, bi-polar rcuits.

ie chip set is designed to implement high-performance microcomputer or controller systems. It incorporates Data General's owerful NOVA line minicomputer instruction set and 16-bit central processing unit (CPU) architecture.

icroNOVA line integrated circuits are available on an individual basis as well as in full microcomputer chip sets. In addition, the icroNOVA family includes the following supporting hardware.

- CPU boards
- RAM memory boards
- programmable read-only memory boards
- asynchronous interface boards
- general purpose I/O boards
- power supply
- hand-held console
- card cages
- a variety of peripheral devices; such as:
  - video displays
  - hard copy data terminals
  - single and dual diskette sub-systems
  - line printers
  - paper tape reader

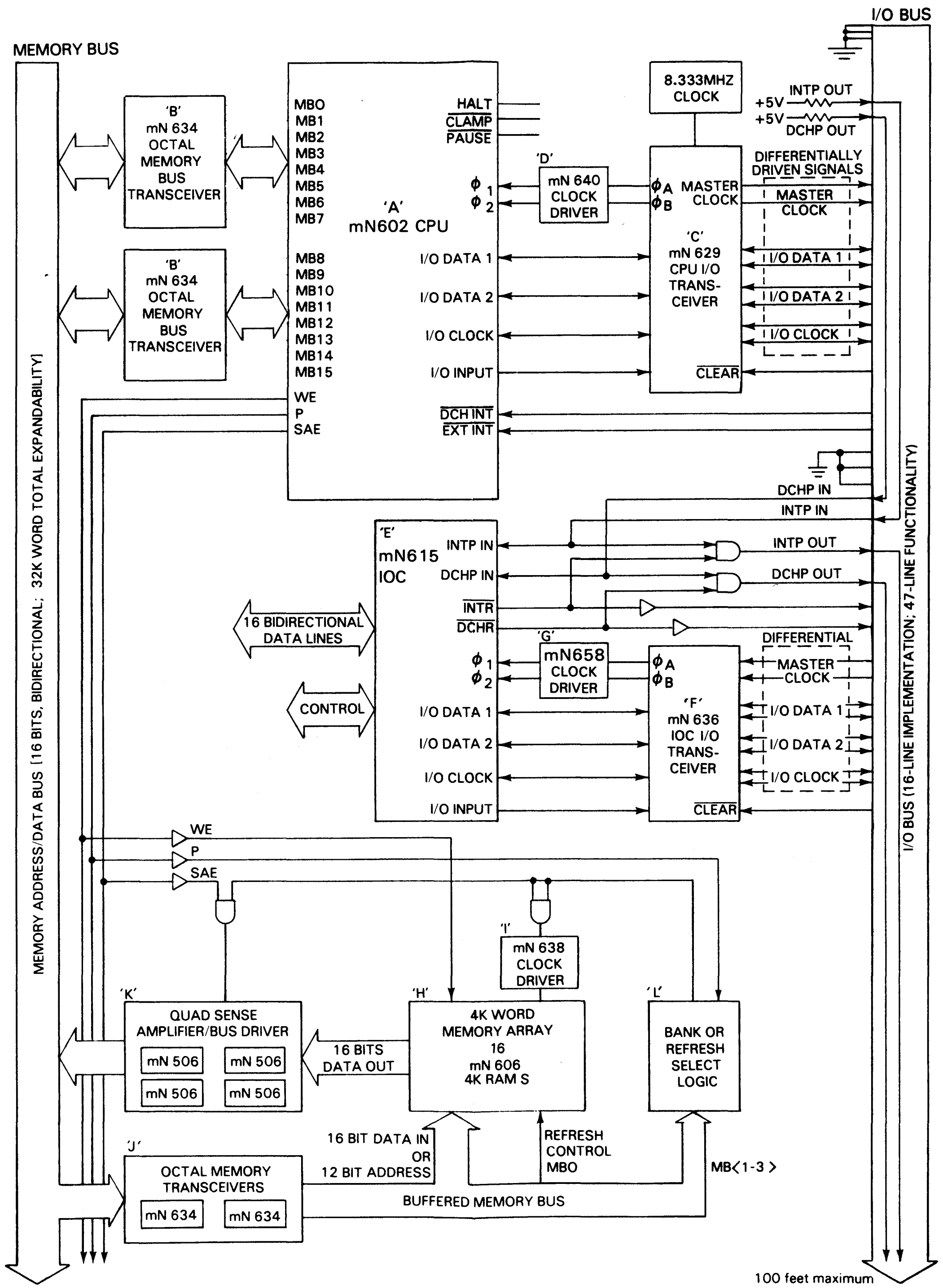
he microNOVA family comes with proven Data General development and runtime software that includes three operating ystems, the diskette-based Disc Operating System (DOS), the Real-Time Operating System (RTOS), and the icroproducts Operating System (MP/OS). Software developed for Data General's NOVA line minicomputers can be asily reassembled for use with microNOVA systems. The operating systems support two high level language compilers, ORTRAN IV and extended BASIC, as well as various utilities, including: a Command Line Interpreter, text editor, macro ssembler, relocatable loader, a library file editor, and a symbolic debugger.

icroNOVA development software is based on the Disc Operating System. The diskette-based DOS provides a smooth flow rough varying development phases to completed production software. OEMs who need a larger development system can utilize e NOVA 3 line of minicomputers. NOVA 3 is software-compatible with microNOVA and supports up to 128K words of memory, a ll peripheral range, and the Real-time Disc Operating System (RDOS).

ata General's Real-Time Operating System is a DOS-compatible runtime executive. It is compact and memory-resident, and as a real-time multitask capability for controlling real-time applications. RTOS provides standard interrupt servicing, device andling and execution scheduling functions.

ll of these features make the microNOVA family ideal for applications like complex instrumentation, industrial automation, ommunications, and data acquisition, where 16-bit NOVA power and precision are required.

microNOVA FUNCTIONAL BLOCK DIAGRAM



DG-04347

Data General
Data General Corporation, Westboro, Massachusetts 01581

THE CHIP SET

Data General's microcomputer chip set consists of the following dual-in-line packaged (DIP), integrated circuits:

Integrated Circuit	Packaging
mN601 16-bit microprocessor	40-pin ceramic
mN602 16-bit microprocessor	40-pin ceramic
mN603/613/615 I/O controller (IOC)	40-pin ceramic
mN606 4Kx1 dynamic RAM	20-pin plastic
mN640 CPU and I/O clock driver	16-pin cerdip
mN638 memory clock driver	16-pin cerdip
mN629 CPU I/O transceiver	20-pin ceramic
mN633 octal driver	20-pin ceramic
mN636 IOC I/O transceiver	20-pin ceramic
mN634 octal memory transceiver	20-pin ceramic
mN506 quad sense amp/memory bus driver	14-pin cerdip
mN658 I/O clock driver	16 pin cerdip

THE MICROCOMPUTER

The block diagram on the opposite page shows a recommended mN602 microcomputer chip set configuration. It is organized around two data buses: memory and I/O (see diagram).

microNOVA MEMORY BUS

Memory addresses and data are passed between the mN601 or mN602 microprocessor and an array of mN606 RAMs via the microNOVA memory bus. This is a 16-bit parallel, bidirectional, high-speed bus.

The microprocessor is interfaced to the bus via two mN634 octal memory transceivers. These transceivers drive data between the CPU's memory address/data lines and the memory bus.

A RAM grouping is interfaced to the bus via two mN634 octal memory transceivers and four mN506 quad sense amp/bus drivers. The memory transceivers drive both addresses and data sent to memory via the bus onto the RAM input lines while the sense amp/bus drivers drive data from the RAM output lines onto the bus.

Three memory control lines, P, SAE and WE, allow the mN601 to perform four kinds of memory operations: read, write, read-modify-write and refresh.

The mN602 contains the same 3 control lines, plus 2 additional ones, WAIT and MAPON. WAIT allows the CPU to extend memory operations to accomodate slow memories and MAPON allows the CPU to access a second 32K-word address space. The mN602 performs three kinds of memory operations: read, write, and refresh.

microNOVA I/O BUS

Information is passed between the microprocessor and the I/O controllers via the bidirectional, microNOVA I/O bus. This bus combines the advantages of high speed with high noise immunity (due to its differential nature, described below), while permitting the use of low cost ribbon cable. The I/O bus can be extended to a maximum distance of 100 feet from the microprocessor.

The microNOVA I/O bus contains 16 lines. The most critical lines (two serial data lines and two clock lines) are differentially driven and received by an mN629 CPU I/O transceiver and one mN636 IOC transceiver per I/O controller.

At the microprocessor end of the bus, the CPU I/O transceiver receives serial data from the CPU and differentially drives it to the IOC I/O transceiver(s). It also receives the differentially driven data signals from the IOC I/O transceiver(s) and passes the data to the CPU. At the I/O controller end of the bus, each IOC I/O transceiver performs a similar function for its I/O controller. Up to twenty IOC I/O transceivers can be interfaced to the microNOVA I/O bus.

The differentially driven I/O CLOCK (see diagram) is transmitted with the data; it strobes the data into the receiving circuit. The CPU I/O transceiver differentially drives the MASTER CLOCK to all IOC I/O transceivers. Each IOC transceiver passes the MASTER CLOCK to its IOC as two non-overlapping clocks via an mN640 clock driver. The MASTER CLOCK synchronizes the operation of the I/O controllers with the CPU.

A system reset line ($\overline{\text{CLEAR}}$), two device request lines, two types of device priority lines, and three ground lines comprise the remainder of the bus.

The $\overline{\text{CLEAR}}$ line allows the system to be reset by external hardware. When this line is pulled to a low logic level for a period of 10 usecs, the microcomputer is reset. This time period ensures that the CPU and IOC I/O transceivers are in receive mode; refer to the mN601, mN602, mN603/613/615, mN629 and mN636 circuit descriptions.

The device request lines ($\overline{\text{EXTINT}}$ and $\overline{\text{DCHINT}}$ for mN601, $\overline{\text{INTR}}$ and $\overline{\text{DCHR}}$ for mN602) allow I/O devices to request microprocessor service for the purpose of transferring data in either of two modes: programmed I/O or data channel break. These are explained later in the section entitled Data Transfers.

Hardware Device Priority Network

The priority lines (INTP OUT, DCHP OUT, INTP IN and DCHP IN) comprise an interdevice daisy chain. When more than one I/O device is requesting service, this network gives priority to the device requesting service that is physically closest on the I/O bus to the microprocessor. The INTP lines control priority between devices requesting interrupt service. Similarly, the DCHP lines control priority between devices requesting data channel service.

The manner in which the priority system is implemented is the same for both types of requests. When a device requests service, that device's INTP/DCHP OUT line goes to the low logic level. This removes priority from devices further away from the microprocessor, by pulling their INTP/DCHP IN lines to a low logic level. When the microprocessor acknowledges a request for service, a device can respond only if it is requesting service and its INTP/DCHP IN line is at a high logic level. If it is at a low logic level, it indicates that a higher priority device in the chain is requesting service. The priority lines previously disabled by the service request are reenabled when the responding device clears its request line.

mN601 16-BIT MICROPROCESSOR

This is a 16-bit central processing unit (CPU) that incorporates the NOVA architecture. Its internal structure and operations are described in the mN601 section of this manual.

The mN601 is designed to operate with the following system buffer elements: two mN634 octal memory transceivers, an mN629 CPU I/O transceiver and an mN640 clock driver. The primary functions performed by these circuits for the CPU are described below. Detailed information is found in the individual circuit descriptions.

The CPU (designated 'A' in the diagram) has two ports: memory and I/O. Two mN634 octal memory transceivers (designated 'B') drive data between the CPU's memory address/data lines MB(<0-15>) and the microNOVA memory bus. The memory control signals, P, SAE and WE, allow the CPU to perform four kinds of memory operations: read, write, read-modify-write and refresh.

The mN629 CPU I/O transceiver (designated 'C') performs two functions for the CPU. First, it interfaces the CPU's serial data (I/O DATA <1,2>) and data clock (I/O CLOCK) lines with the **microNOVA I/O bus. The I/O INPUT signal is used by the CPU to control the transceiver's operating mode, either transmit or receive. Second, the CPU I/O transceiver generates the two non-overlapping system clocks (ϕA and ϕB) from a single 8.333MHz clock input.**

The ∅A and ∅B clocks are shaped by external circuitry (see Design Notes) and the outputs are supplied to the mN640 clock driver (designated 'D'). The clock driver receives these signals and provides signal characteristics that are compatible with the MOS clock input circuits of the CPU, ∅1 and ∅2.

The remaining mN601 pins perform the following functions. The HALT pin reflects the state of the CPU, halted (idling) or performing tasks. The $\overline{\text{PAUSE}}$ pin indicates when the CPU is performing a memory operation. By monitoring the output of this pin, **multiporting of memory can be implemented.** The $\overline{\text{CLAMP}}$ pin is an input pin used during the power-up and initialization sequence.

mN602 16-BIT MICROPROCESSOR

The mN602 incorporates all the features of the mN601 CPU plus five new features.

The mN602 requires only standard TTL and MOS voltage levels of -5V, +5V and 12V. The CPU can use the mN603, the mN613, or the mN615 as an IOC.

The CPU can access memories with slow cycle times (EPROM memories) by sending additional SAE or WE signals to memory when $\overline{\text{WAIT}}$ goes low. A memory can cause the mN602 to wait a maximum of 50 CPU cycles, or 24μsec.

The mN602 can access up to 64K words of memory. This physical address space divides into two 32K-word logical address spaces that are addressable by the user. The MAPON flag determines which 32K-word bank of memory is addressed. When MAPON is 0, the lower (unmapped) bank is addressed. This is the address space available on mN601. When MAPON is 1, the upper (mapped) bank is addressed.

On memory refresh, the mN602 reads a System Status Word on the memory bus, during the data transfer phase. This word provides the CPU an external real-time clock, a set of non-maskable interrupts, and the state of the system the mN602 is part of (powerfail, etc.).

The last main feature of the mN602 is the integrated fast data channel. The FDCH makes direct memory accesses (DMAs) at 2 Mbytes/sec. The three control lines $\overline{\alpha 2}$, $\overline{\text{FDCHR}}$/$\overline{\text{FDCHI}}$, and $\overline{\text{FDCHE}}$, handle all transactions between the CPU and the device. The clock signal $\overline{\alpha 2}$ synchronizes high speed channel requests when $\overline{\text{FDCHR}}$ goes low. During the transfer, the CPU sends the needed memory control signals. Because of the transfer rate of the high speed data channel, the mN602 will not allow DMAs to slow memories.

mN603/613/615 I/O CONTROLLER (IOC)

The IOC is the interface between a peripheral device and the microNOVA I/O bus. Details of its internal architecture and operations are contained in the mN603/613/615 section of this manual.

It is designed to operate with the following system buffer elements: an mN636 IOC I/O transceiver and a clock driver. The mN603 and mN613 operate with a mN640 clock driver. The mN615 uses a mN658 clock driver. The primary functions performed by these circuits for the IOC are described below. Detailed information is found in the individual circuit descriptions.

The IOC (designated 'E' in the diagram) has two ports: I/O bus and peripheral.

The mN636 IOC I/O transceiver (designated 'F') performs two functions for the IOC. First, it interfaces the IOC's serial data (I/O DATA<1,2>) and data clock (I/O CLOCK) lines with the microNOVA I/O bus. The I/O INPUT signal is used by the IOC to control the transceiver's operating mode, transmit or receive. Second, the mN636 transceiver receives the MASTER CLOCK from the I/O bus and generates two non-overlapping clock signals, ϕA and ϕB.

The ϕ A and ϕ B non-overlapping clocks are shaped by external circuitry (see Design Notes) and the outputs are supplied to the mN640 clock driver (designated 'G'). The clock driver receives these signals and provides signal characteristics that are compatible with the MOS clock input circuits of the IOC, ϕ1 and ϕ2.

The IOC communicates with the device via a 16-bit parallel data bus and a group of control lines. The data and control lines together create the functional equivalent of the 47-line NOVA I/O bus.

mN606 4Kx1 DYNAMIC RAM

This is a 4096 by one bit memory. Details of its internal architecture and operations are contained in the mN606 section of this manual.

Sixteen mN606 RAMs comprise a 4Kx16-bit memory array. This array is designed to operate with the following system buffer elements: one mN638 clock driver, two mN634 octal memory transceivers and four mN506 quad sense amp/bus drivers. For an 8Kx16-bit array, it is necessary to add only one mN638 clock driver, since the two mN634 transceivers and four mN506 sense amp/bus drivers support 8K of memory. The primary functions performed by these system buffer elements for the mN606 RAMs are described below. Detailed information is found in the individual circuit descriptions.

The 4Kx16-bit dynamic RAM array (designated 'H' in the diagram) has two ports: input (address and data) and output.

The two mN634 octal memory transceivers (designated 'J') drive addresses and data from the microNOVA memory bus onto the RAM address and data input lines. The four mN506 quad sense amp/bus drivers (designated 'K') drive data from the RAM output lines onto the microNOVA memory bus.

The mN601 addresses up to 32,768 memory locations; thus, eight 4Kx16-bit memory arrays can be interfaced to the microNOVA memory bus. The mN602 addresses up to 65,536 memory locations; thus, 16 4Kx16-bit memory arrays can be interfaced to the microNOVA memory bus. Each 4Kx16-bit array has a 12-bit memory address (MB<4-15>). The external bank select logic (designated 'L') selects the appropriate array from the address appearing on the MB<1-3> lines.

When both the memory control signal P is asserted and the bank select logic is enabled, the mN638 clock driver (designated 'I') provides the signal characteristics required for the MOS clock input circuit of the RAM.

At the beginning of a memory operation, the RAM samples the address and tests the logic level, high or low, of MB0. If MB0 is at a high level, the RAM initiates a refresh operation; if it is at a low level, the RAM initiates either a read, write, or read-modify-write operation.

When mN606 dynamic RAMs are used in a microNOVA-based system, the mN601 microprocessor automatically refreshes them without external circuitry.

DATA TRANSFERS

Addresses and data are transferred between the CPU and memory via the microNOVA memory bus. Control information and data words are transferred between the CPU and I/O devices via the microNOVA I/O bus.

MEMORY

The microprocessors perform three kinds of memory operations: read, write, and refresh. All memory operations are initiated on the rising edge of P. As described above, the address on the bus is sampled by the memories and the state of MB0 is tested. If this signal is high, a refresh operation is performed; if it is low, either a read, write, or read-modify-write operation is performed. Detailed information is found in the Memory Operations section of the mN601 and mN602 circuit description.

When non-volatile memory is required, programmable read-only memory (PROM) can be used. Refer to the microNOVA Computer Systems Technical Reference (DGC ordering number 014-000073). At the board level, PROM can overlay RAM, and PROM priority control lines resolve any memory conflicts in favor of the PROM.

INPUT/OUTPUT

Information is passed between the CPU and I/O controllers via the two serial data lines of the microNOVA I/O bus. The bus connections are made through the appropriate I/O transceivers. The data consists of both control information and 16-bit data words.

The CPU performs I/O data transfers under two operating modes: programmed I/O and data channel break. Under programmed I/O, the CPU moves data betweeen one of its internal registers (namely, accumulators; see the mN601 and mN602 sections) and an I/O device. Under data channel break, the CPU moves data between memory and an I/O device. Data channel breaks allow blocks of information to be transferred without altering program flow. In addition, data channel breaks are transparent to the executing program.

OVERVIEW

An I/O device requests microprocessor service for the purpose of transferring data by asserting either the program interrupt or data channel break device request lines. These requests are synchronized by a CPU-generated signal called Request Enable. It ensures that the device request and priority lines are at a steady state when they are examined by the CPU. Request Enable is sent to all I/O controllers connected to the I/O bus as a two-bit code (11), one bit carried on each serial data line.

Detailed information concerning I/O data transfers is found in the I/O Operations sections of the mN601 and mN602 circuit descriptions and in the mN603/613/615, mN629 and mN636 circuit descriptions. Timing diagrams of I/O bus operations can be seen in Appendix A. These diagrams depict the transmission and reception of control information and data words as they pass between the CPU, the CPU transceiver, the IOC transceiver and the I/O controller, under both programmed I/O and data channel. (The I/O transactions appearing in the diagrams apply when the system is configured as shown in the block diagram at the beginning of this section.)

Programmed I/O

Under programmed I/O, the CPU sends the I/O instruction, exactly as fetched from memory, to all controllers connected to the microNOVA I/O bus. The instruction includes information that identifies the controller to which it is directed. Following the instruction, the CPU performs either a data out operation or a data in operation, as required by the instruction.

When either data out or data in instructions are performed, both the instruction and the 16-bit data word are transmitted as 18-bits, 9 bits on each line. The first bit on each line comprises a two-bit code that identifies the type of transfer; 00 indicates an I/O command and 01 indicates a data word.

Data Channel Break (Standard Data Channel)

When the CPU performs a data channel break, it sends an address request to the microNOVA I/O bus. This request consists of a two-bit code (10). The highest priority device requesting data channel service responds to this code. The device responds first by sending a 16-bit word (plus the two-bit data word prefix code, 01, described above) containing a 15-bit memory address and a direction of transfer bit (data out or data in) to the CPU. Following the receipt of this information, the CPU initiates a memory operation at the address specified; then, it examines the direction of transfer bit.

If it is set for a data out, the CPU transmits the 16-bit word received during the read portion of the memory operation together with the two-bit data word prefix code. The memory operation is terminated without altering the contents of the addressed location.

If it is set for a data in, the device transmits the 16-bit word to be entered in memory together with the two-bit data word prefix code. After the data word is received by the CPU, it is written into memory during the write portion of the memory operation in progress.

RELATED DOCUMENTS

The following supporting documents are available from Data General Corporation.

Document	Ordering No.
microNOVA Programmer's Reference	015-000050
microNOVA DTOS	015-000059
microNOVA Computer Systems	014-000073
Microproducts Hardware Reference Series	
MP/100 Computer System	014-000668
MP/200 Computer System	014-000667
Input/Output and Interfacing	014-000665
Communications Interfaces and Character Device Controllers	014-000660
Disk and Diskette Drives	014-000661
Magnetic Tape Transport	014-000664
Sensor I/O	014-000659

mN601
16-BIT MICROPROCESSOR
CONTENTS

Data General

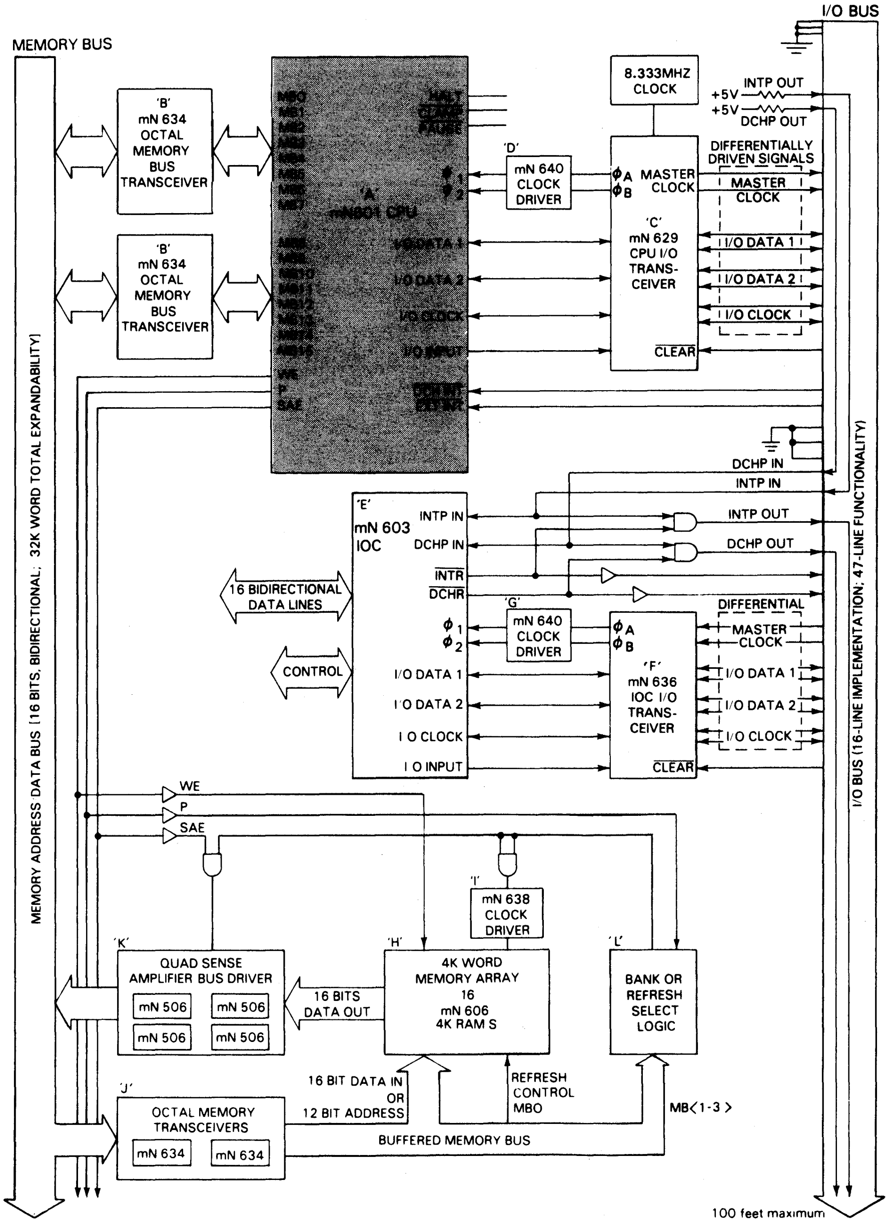
Data General Corporation, Westboro, Massachusetts 01581

mN601
CONTENTS

CONTENTS

2

mN601
16-BIT MICROPROCESSOR



DG-04335

Data General

Data General Corporation, Westboro, Massachusetts 01581

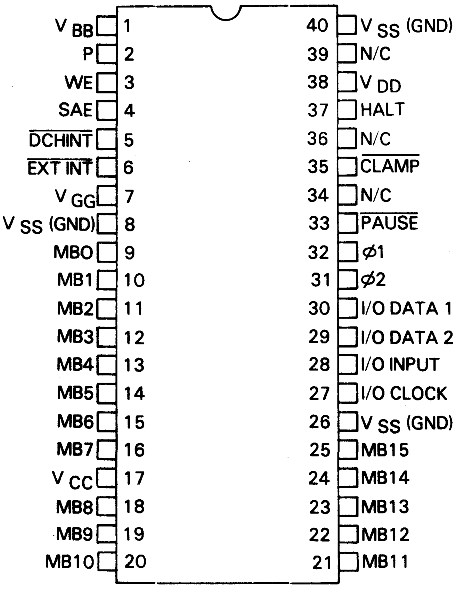
mN601

16-BIT MICROPROCESSOR (CPU) NMOS INTEGRATED CIRCUIT

FEATURES

- FULL NOVA 16-BIT ARCHITECTURE AND INSTRUCTION SET IN A SINGLE 40-PIN NMOS SILICON-GATE CHIP
- INTEGRAL DATA CHANNEL, REAL-TIME CLOCK, AND MULTIPLY/DIVIDE
- INTEGRAL HIDDEN REFRESH AND CONTROL LOGIC FOR DYNAMIC RAMs
- COMPREHENSIVE STACK ARCHITECTURE WITH HARDWARE STACK AND FRAME POINTER REGISTERS
- POWERFUL SAVE AND RETURN INSTRUCTIONS FOR SUBROUTINE CALLS
- MULTIPLE ACCUMULATORS
- SYSTEM MEMORY SUPPORT UP TO 32K WORDS OF RAM/PROM
- SINGLE WORD INSTRUCTION FORMAT IN A VARIETY OF ADDRESSING MODES - ABSOLUTE, RELATIVE, INDEXED, DEFERRED AND AUTO-INCREMENT/ DECREMENT
- SEPARATE MEMORY AND I/O BUSES
- I/O BUS PROVIDES FUNCTIONAL EQUIVALENT OF 47-LINE NOVA I/O BUS
- 16-LEVEL PROGRAMMED PRIORITY INTERRUPT

PACKAGE



DG-04026

GENERAL DESCRIPTION

The mN601 is a full 16-bit NMOS microprocessor that provides the central processing function for Data General's microNOVA family. The mN601 features NOVA 16-bit architecture, including hardware multiply/divide; multiple addressing modes, including **absolute, relative, indexed, deferred, and auto-increment/decrement; multiple accumulators, including two that can be used as** index registers; hardware stack and frame pointers with stack overflow protection; programmed priority interrupt to 16 levels; and separate memory and I/O buses.

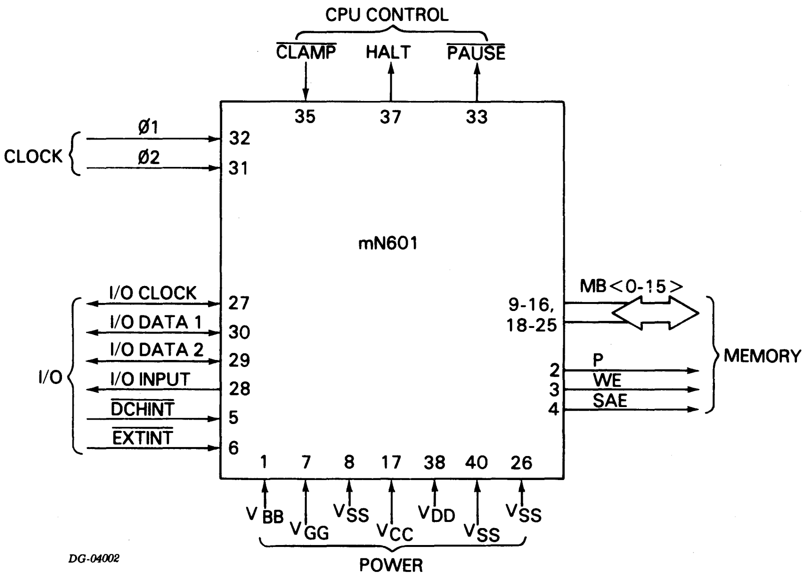
Data General's microNOVA family of integrated circuits also includes memories, peripheral controllers, and supporting circuits which can be used to develop a complete, high-performance, microcomputer system.

The mN601 is supported by the mN606 4K RAM, the mN603 I/O Controller (IOC), and system buffer elements. The 4096-bit mN606 uses cost- effective dynamic NMOS RAM technology to support microNOVA's large memory orientation. The mN603 IOC delivers the full functional capability of the 47-line NOVA I/O bus, including a parallel 16-bit I/O port. System buffer elements allow microNOVA support up to 32K words of memory and a full complement of peripherals.

PIN DESCRIPTIONS

The diagram below shows the pin connections, and the table describes the function of each pin shown in the diagram.

FUNCTIONAL PIN CONNECTION DIAGRAM



DG-04002

PIN FUNCTIONS

MNEMONIC	PIN NO.	I/O	FUNCTION
			CLOCK
$\phi 1$ $\phi 2$	32 31	IN IN	Two-phase system clock. Operates between 0 and 14V amplitude. Generates 4-phase internal clock, providing internal control timing.
			MEMORY BUS
MB0-MB15	9-16 18-25	I/O	Bi-directional, data and address bus. When used as a data bus, MB0 contains the MSB (most significant bit) and MB15, the LSB (least significant bit). As an address bus, MB<1-15>contains the address and MB0 defines the memory operation as either refresh or read/write. This bus is precharged internally by the CPU. It is held high by individual internal pull-up resistors tied to each line. During address or data transfers, each line transmitting a 0 is discharged; each line transmitting a 1 is undischarged. See the Memory Operations Section for detailed information.
			MEMORY CONTROL
P	2	OUT	Active high. Initiates all memory cycles. When P goes high, indicates a valid address is on the memory bus.
SAE	4	OUT	Active high. Enables sense amplifier outputs on the memory bus during a read operation.
WE	3	OUT	Active high. Enables write operation. On the falling edge of WE, information on the memory bus is written into memory.

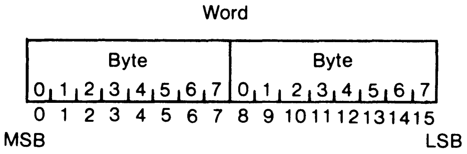
PIN FUNCTIONS

MNEMONIC	PIN NO.	I/O	FUNCTION
			I/O BUS
I/O DATA 1 I/O DATA 2	30 29	I/O I/O	Two-line, bi-directional bus. All data and I/O command information is transmitted serially between the mN601 and I/O devices at the MASTER CLOCK rate. During a 16-bit transfer, I/O DATA1 carries the most significant 8 bits (0-7); I/O DATA2, the least significant bits (8-15).
I/O INPUT	28	OUT	Indicates direction of I/O transfer. When low, indicates a transfer from the mN601; when high, a transfer to the mN601.
I/O CLOCK	27	I/O	Synchronizes all I/O transfers to and from the mN601. Holding I/O CLOCK low for 10μ secs resets the microcomputer. I/O CLOCK should not be pulled low when I/O INPUT is in the low state.
$\overline{\text{EXTINT}}$	6	IN	Active low. When asserted low by a device, initiates a program interrupt, providing CPU interrupts are enabled. When the mN601 is in the HALT state (see below), it is started by generating a program interrupt.
$\overline{\text{DCHINT}}$	5	IN	Active low. When asserted low by a device, the CPU will execute a data channel break. The mN601 responds to data channel break requests while it is in the HALT state as well as during operations.
			CPU CONTROL
HALT	37	OUT	Active high. When the CPU is halted (idling), as the result of either resetting the CPU by pulling I/O CLOCK low or the execution of a HALT instruction, HALT generates a sequence of .417MHz pulses. In the HALT state, the mN601 performs the following functions: ● It generates Request Enable signals (see mN601 I/O Operations section) that synchronize program interrupt and data channel requests. ● It responds to program interrupts and data channel requests. (The normal way to start the mN601 is to generate a program interrupt.) ● It generates dynamic RAM refresh signals to prevent the loss of information stored in RAM.
$\overline{\text{CLAMP}}$	35	IN	Active low. When the mN601 is first powered up, $\overline{\text{CLAMP}}$ should be held low for a minimum of 100μsec. Then, when it goes high, the CPU is properly initialized and enters the HALT state.
$\overline{\text{PAUSE}}$	33	OUT	Active low. $\overline{\text{PAUSE}}$ goes to the low level for a period of 240ns when the CPU is not using memory. During this period it is permissible to disable the $\phi 1$ and $\phi 2$ clocks, thus allowing multiporting of memory.
			POWER
V_{BB}	1		-4.25V ± .5V
V_{CC}	17		+5V ± .25V
V_{DD}	38		+10V ± 1V
V_{GG}	7		+14V ± 1V
V_{SS}	8,26,40		Ground
	34		Reserved for future use
	36		Reserved for future use
	39		Reserved for future use

ARCHITECTURE

The mN601 is a 16-bit parallel word central processing unit which executes the full NOVA line minicomputer instruction set, including an extended instruction set for stack and trap routines.

The instruction set addresses 32,768 words (65,536 bytes) of memory. Each memory address is 15-bits long. Each word is 16-bits long, consisting of two 8-bit bytes. Both words and bytes can be used as operands.



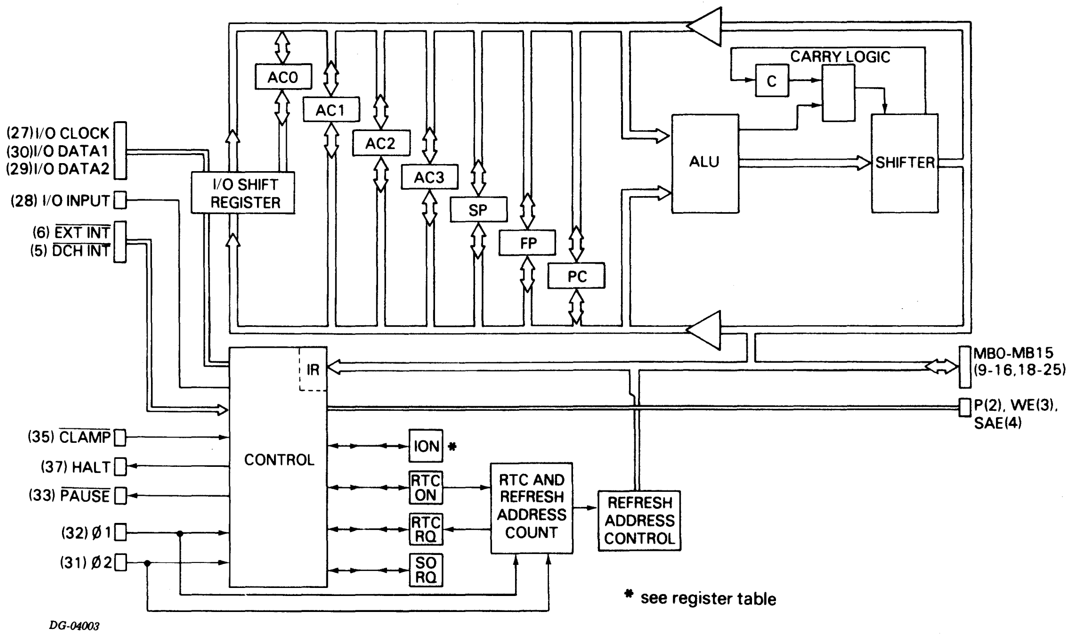
One word containing two bytes

The mN601 communicates with memory via a bi-directional, 16-bit parallel address and data bus. Three control lines direct the four memory operations: read, write, read-modify-write and refresh.

The mN601 communicates with I/O controller interfaces via two, bi-directional, serial lines that carry data and I/O commands. One control line indicates the direction of transfer, to or from the CPU, and a clock line synchronizes the transfer of information. Two interrupt lines, programmed I/O and data channel, allow I/O interfaces to request processor time.

The Expanded Block Diagram shows the internal architecture of the mN601, together with the memory bus, the I/O bus, and their respective control lines.

EXPANDED BLOCK DIAGRAM



* see register table

Within the mN601, information is stored in the thirteen registers listed below.

CPU INTERNAL REGISTERS

REGISTERS	USER ACCESSIBLE	SIZE
accumulator 0	yes	16 bits
accumulator 1	yes	16 bits
accumulator 2	yes	16 bits
accumulator 3	yes	16 bits
program counter	yes	15 bits
stack pointer	yes	15 bits
frame pointer	yes	15 bits
carry bit	yes	1 bit
interrupt enable bit (ION)	yes	1 bit
real-time-clock enable bit (RTC ON)	yes	1 bit
real-time-clock request bit (RTC RQ)	no	1 bit
stack overflow request bit (SO RQ)	no	1 bit
refresh address counter	no	6 bits

- The four accumulators provide scratchpad memory, or work space, during program execution.

- The program counter contains a 15-bit memory address that is used by the processor when fetching the next instruction from memory during program execution. When the instruction is completed, the program counter is either automatically incremented by one or is modified by the program, before the next instruction is fetched.

- The stack pointer contains the memory address of the highest location in memory used by the stack. As information is pushed onto the stack, this address is appropriately incremented; as information is popped off the stack, the address is decremented.

- The frame pointer contains the memory address of a location within the stack. When a RETURN instruction is executed, this address is used as a reference for retrieving information from the stack.

- The carry bit reflects the results of certain arithmetic computations performed by the arithmetic logic unit (ALU), e.g., addition overflow.

- The interrupt enable bit can be controlled by the program. When enabled (set to 1), the processor responds to program interrupt requests.

- The real-time-clock enable bit is set by the program. When enabled (set to 1), the real-time clock contained in the processor generates an interrupt at 1.8432ms intervals.

- The real-time-clock request bit generates a program interrupt at fixed intervals of 1.8432ms when both the real- time clock and interrupts are enabled.

- The stack overflow request bit generates a program interrupt when the stack overflows a 256-word memory address boundary, e.g., 377_8 to 400_8.

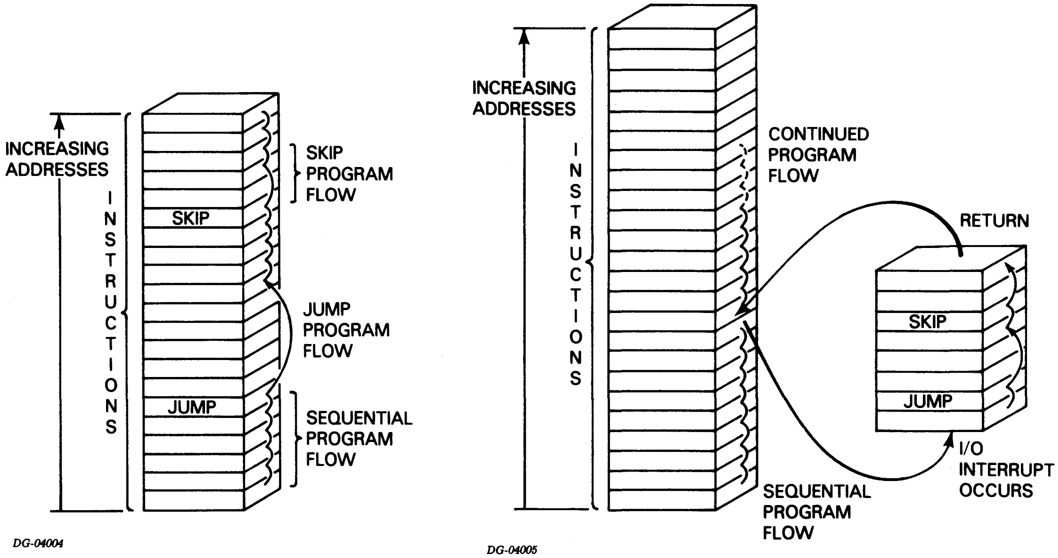
- The refresh address counter generates a 6-bit address specifying 64 numbers used to address 1/64 of all the available memory to be refreshed. All 32,768 RAM memory locations are refreshed at least once every 1.8432ms.

INTRODUCTION TO INSTRUCTION SET

Each mN601 (microNOVA) instruction is contained in one 16-bit word. Programs for the mN601 consist of sequences of instructions which are stored in external memory. The order in which these instructions are executed depends on the 15-bit memory address in the program counter register. During program execution, the processor sends this address to memory and fetches the contents of the specified memory location, via its memory address/data pins. During the execution of an instruction, information is moved between the processor's internal registers, memory or I/O data pins. When the instruction is completed, the program counter is updated. The CPU uses the contents of the program counter to address the next location.

Sequential operation can be altered by a jump or conditional skip instruction or by an interrupt. Jump instructions load a new address in the program counter; conditional skip instructions increment the program counter address by two if a specified test is true. In both cases, sequential operation is resumed from the updated address in the program counter. When the program is interrupted (e.g., I/O external interrupts, real-time clock interrupts, fault conditions of various kinds), the interrupt facility stores the next sequential program address in a specified location in memory so that the interrupt handler can return control to the interrupted program at the proper entry point. Then, the program counter is loaded with the memory address of the interrupt handler program and sequential operation resumes.

PROGRAM FLOW



The microNOVA (mN601) instruction set can be divided into five groups of operations:

- Memory reference
- Fixed point arithmetic and logical operations
- Stack manipulation
- I/O functions
- CPU I/O functions

Each instruction is formatted in a 16-bit (one word) field. In the instruction descriptions that follow, the mnemonics are those recognized by Data General's Assembler. See the microNOVA Programmer's Reference (DGC No. 015-000050) for detailed programming information, including the use of instruction set options, such as conditional skips and flag commands.

NOTE: When the instructions described below are executed, the mN601 pins follow the protocol specified by the instructions. It is assumed that the mN601 is interfaced with mN603 I/O controllers and some amount of memory.

MEMORY REFERENCE INSTRUCTIONS

This instruction group moves data between accumulators and memory, modifies the contents of memory, and alters the program flow.

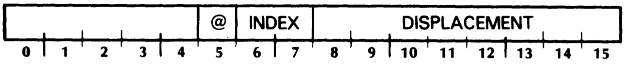
Locations in memory are sequentially numbered from 00000 through 77777_8, so that each memory address is 15 bits long.

ADDRESSING MODES

Memory reference instructions provide a variety of addressing modes for accessing 32,768 memory loctions: absolute, PC relative, and indexed (via accumulator 2 or 3). In addition, the processor supports eight levels of conventional indirect (deferred) addressing and five levels of indirect addressing through auto-incrementing/decrementing locations.

When instructions that reference memory are executed, the processor calculates an effective 15-bit address using bits 5 through 15 of the memory reference instruction field shown below.

MEMORY REFERENCE INSTRUCTION FORMAT



Bit 5 = Indirect bit; Bits 6 and 7 = Address mode: 00 = Absolute
01 = PC relative
10 = AC2 indexed
11 = AC3 indexed

In all but the absolute mode, the displacement is a signed number within the range -128 through +127. Signed numbers utilize the two's complement representation for negatives.

EFFECTIVE ADDRESS CALCULATION

Absolute	Displacement is an unsigned number within the range 00000-00377$_8$ that addresses one of the first 256 locations in memory, referred to as Page Zero. If the indirect bit is 0, the displacement is the effective address; if it is 1, the displacement is an indirect address (see the description of indirect addressing below).
PC relative	Displacement is a signed number that is added to the contents of the program counter. If the indirect bit is 0,the results of the addition are the effective address; if it is 1, the results are an indirect address.
AC2 indexed	Displacement is a signed number that is added to the contents of accumulator 2. If the indirect bit is 0, the results of the addition are the effective address; if it is 1, the results are an indirect address.
AC3 indexed	Displacement is a signed number that is added to the contents of accumulator 3. If the indirect bit is 0, the results of the addition are the effective address; if it is 1, the results are an indirect address.

NOTE: A memory address is always 15 bits long. When the results of an addition overflow 15 bits, the overflow is ignored.

Example: Program Counter = 77774_8
Displacement = $\underline{+012_8}$
Result = 000006_8 ; not 100006_8

Indirect Addressing

When the indirect bit (bit 5) contains a 1, the calculated address is an intermediate (indirect) address that points to an effective address stored in memory. In this case, the processor fetches the contents of the calculated address and tests bit 0 (indirect sign bit when stored in memory). If it is 0, bits 1 through 15 of the contents of the indirect address are used as the effective address; if it is 1, bits 1 through 15 are used as an indirect address. In the latter case, the indirection chain continues until the processor either fetches a word in which bit 0 equals 0 (in this case, the processor has the effective address) or has executed eight levels of indirection (or five levels if the indirection chain accesses auto-incrementing/decrementing locations, explained below).

An internal counter is used to control the number of indirect addresses through which the processor searches for an effective address. This counter is initialized when the indirect bit is set to 1. Each time a word in the indirection chain is retrieved from memory the counter is incremented by two (or by three if an auto-increment or decrement location is indirectly addressed). If the counter becomes greater than 17 before an effective address is retrieved, the processor halts.

Auto-Incrementing / Decrementing

Memory locations with addresses within the range 020_8 through 027_8 are called auto-incrementing locations. When one of these locations is indirectly addressed, the processor automatically increments bits 0 through 15 of the contents of the indirect address before using these bits as either an effective or indirect address. The updated contents are written back into the auto-incrementing location.

Memory locations with addresses within the range 030_8 through 037_8 are called auto-decrementing locations. When one of these locations is indirectly addressed, the processor automatically decrements bits 0 through 15 of the contents of the indirect address before using these bits as either an effective or indirect address. The updated contents are written back into the auto-decrementing location.

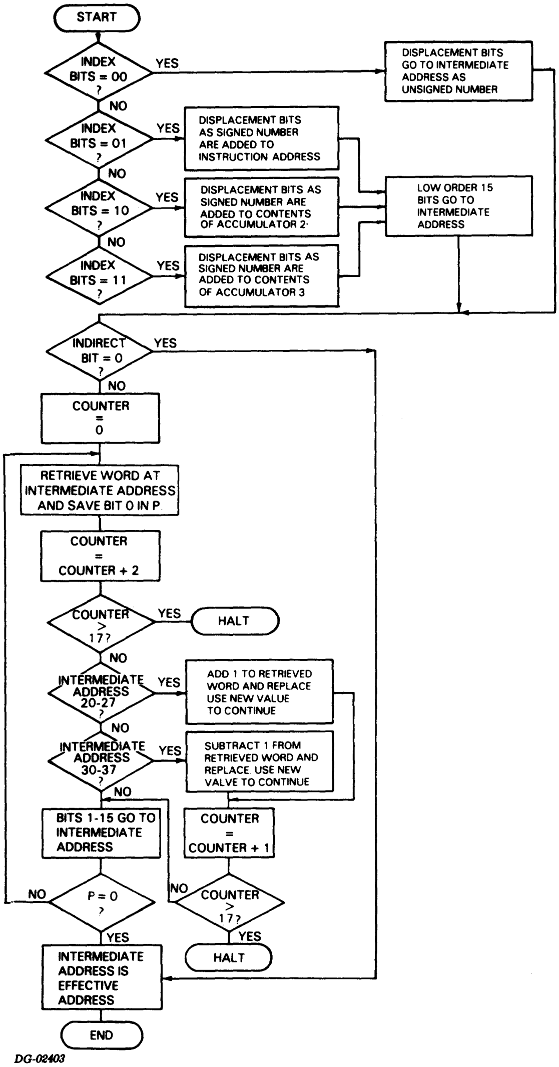
> **NOTE 1:** When referencing auto-increment/decrement locations, the state of bit 0 before the increment or decrement is the condition upon which the continuation of the indirection chain is based. For example: if an auto-increment location contains 177777_8 and the location is referenced as part of an indirection chain, location 0 will be the next indirect address in the chain; not the effective address.

> **NOTE 2:** When a non-existing memory address is specified in a memory reference instruction, the contents will appear as all ones.

MEMORY ADDRESSING

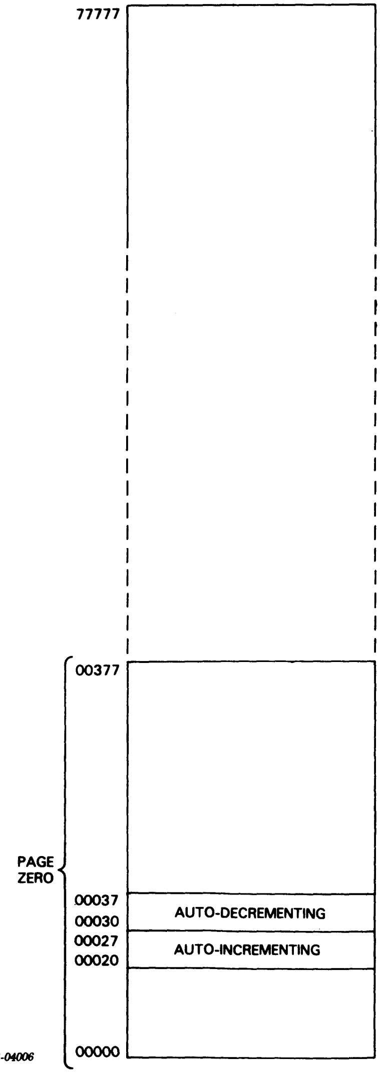
MODE	INDEX BITS	MEMORY ADDRESSING CAPABILITY (OCTAL)
Absolute	00	00000-00377
PC Relative	01	00000-77777
AC2 Indexed	10	00000-77777
AC3 Indexed	11	00000-77777

ADDRESS CALCULATION


START
INDEX BITS = 00 ?
YES
NO
DISPLACEMENT BITS GO TO INTERMEDIATE ADDRESS AS UNSIGNED NUMBER
INDEX BITS = 01 ?
YES
NO
DISPLACEMENT BITS AS SIGNED NUMBER ARE ADDED TO INSTRUCTION ADDRESS
INDEX BITS = 10 ?
YES
NO
DISPLACEMENT BITS AS SIGNED NUMBER ARE ADDED TO CONTENTS OF ACCUMULATOR 2·
LOW ORDER 15 BITS GO TO INTERMEDIATE ADDRESS
INDEX BITS = 11 ?
YES
DISPLACEMENT BITS AS SIGNED NUMBER ARE ADDED TO CONTENTS OF ACCUMULATOR 3
INDIRECT BIT = 0 ?
YES
NO
COUNTER = 0
RETRIEVE WORD AT INTERMEDIATE ADDRESS AND SAVE BIT 0 IN P
COUNTER = COUNTER + 2
COUNTER > 17?
YES
HALT
NO
INTERMEDIATE ADDRESS 20-27 ?
YES
ADD 1 TO RETRIEVED WORD AND REPLACE USE NEW VALUE TO CONTINUE
NO
INTERMEDIATE ADDRESS 30-37 ?
YES
SUBTRACT 1 FROM RETRIEVED WORD AND REPLACE USE NEW VALVE TO CONTINUE
NO
BITS 1-15 GO TO INTERMEDIATE ADDRESS
COUNTER = COUNTER + 1
NO
P = 0 ?
NO
COUNTER > 17?
YES
YES
INTERMEDIATE ADDRESS IS EFFECTIVE ADDRESS
HALT
END

DG-02403

MEMORY



DG-04006

NO ACCUMULATOR-EFFECTIVE ADDRESS INSTRUCTIONS

The general format for this class of instruction is shown below. The table shows the specific format and function for each instruction.

0	0	0	OP CODE	@	INDEX	DISPLACEMENT
0	1	2	3 4	5	6 7	8 9 10 11 12 13 14 15

INSTRUCTION FORMATS AND FUNCTIONS

MNEMONIC	MEANING	FUNCTION *	INSTRUCTION FORMAT
JMP	Jump	Compute (E) and load E into PC.	0 0 0 0 0 @ INDEX DISPLACEMENT 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
JSR	Jump To Subroutine	Compute E. Store contents of PC+1 in bits 1-15 of AC3 and set bit 0 to 0; then, load E into PC.	0 0 0 0 1 @ INDEX DISPLACEMENT 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
ISZ	Increment And Skip If Zero	Compute E. Increment contents of location E and write results back into location E. If results equal 000000, increment PC by two; otherwise increment PC by one.	0 0 0 1 0 @ INDEX DISPLACEMENT 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
DSZ	Decrement And Skip If Zero	Compute E. Decrement contents of location E and write results back into location E. If results equal 000000, increment PC by two; otherwise increment PC by one.	0 0 0 1 1 @ INDEX DISPLACEMENT 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

*E = effective address; AC = accumulator

ONE ACCUMULATOR-EFFECTIVE ADDRESS INSTRUCTIONS

The general format and accumulator code for this class of instruction are shown below. The table shows the specific format and function for each instruction.

0	OP CODE	AC	@	INDEX	DISPLACEMENT
0	1 2	3 4	5	6 7	8 9 10 11 12 13 14 15

AC	Bits 3	4
AC0	0	0
AC1	0	1
AC2	1	0
AC3	1	1

INSTRUCTION FORMATS AND FUNCTIONS

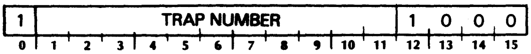
MNEMONIC	MEANING	FUNCTION *	INSTRUCTION FORMAT
LDA	Load Accumulator	Compute E. Load specified AC with contents of location E.	0 0 1 AC @ INDEX DISPLACEMENT 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
STA	Store Accumulator	Compute E. Store contents of specified AC in location E.	0 1 0 AC @ INDEX DISPLACEMENT 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

* E = effective address; AC = accumulator

TRAP INSTRUCTION

This instruction can be used as a single word call to subroutines containing extended instructions not implemented by the mN601. The format of the TRAP instruction is shown below.

INSTRUCTION FORMAT AND FUNCTION

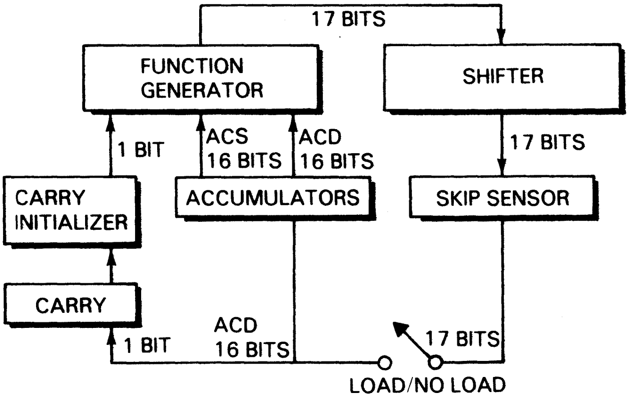
MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
TRAP	Trap	The address of this instruction is placed in memory location 46_8 and bit 0 of that location is set to 0. Then, the processor performs a jump indirect through memory location 47_8.	 1 TRAP NUMBER 1 0 0 0 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

ARITHMETIC / LOGICAL INSTRUCTIONS

This instruction group performs arithmetic and logical operations on the contents of one or two accumulators. In some cases, it changes the value of the carry bit and, if specified, manipulates and/or tests the results of the operations.

The logical organization of the mN601's internal arithmetic logic unit (ALU) is shown below.

ALU DATA PATH



DG-00927

TWO ACCUMULATOR-MULTIPLE OPERATION

1	ACS	ACD	OP CODE	SH	C	#	SKIP
0	1 2	3 4	5 6 7	8 9	10 11	12	13 14 15

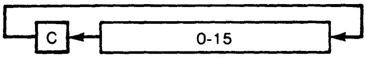
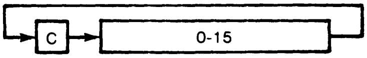
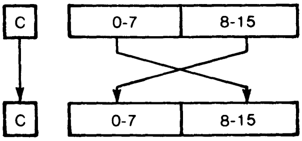
AC = accumulator
ACS = source accumulator
ACD = destination accumulator

	ACS		ACD	
Bits	1	2	3	4
AC0	0	0	0	0
AC1	0	1	0	1
AC2	1	0	1	0
AC3	1	1	1	1

Each instruction specifies two accumulators, source and destination, that supply the function generator with operands. The function generator performs the operation specified by the operation code, e.g., add, subtract, and produces a carry bit whose value depends on three quantities: an initial value specified by the instruction, the inputs, and the function performed. The initial value can be derived either from the previous value of the carry bit or a new value specified by the instruction.

The 17-bit (16 bits plus the carry bit) result of the function generator goes to the shifter. Depending upon the shift operation specified, the results are either passed undisturbed through the shifter or they are manipulated in one of the following ways: the 17-bit result is rotated one bit position, either left or right; or, the two 8-bit halves are swapped without affecting the carry bit.

ALU SHIFT OPERATION

CODED CHARACTER	SHIFTER OPERATION
L	Left rotate one place. Bit 0 is rotated into the carry position, the carry bit into bit 15. 
R	Right rotate one place. Bit 15 is rotated into the carry position, the carry bit into bit 0. 
S	Swap the halves of the 16-bit result. The carry bit is not affected. 

DG-04423

The 17-bit output of the shifter can be tested for a conditional skip. The skip sensor tests either, or both, the carry bit or the 16-bit result of the shifter, depending upon the instruction, to determine if either, or both, is equal or not equal to zero.

After the skip sensor has tested the output of the shifter, the 17- bit result of the shifter is either loaded or not loaded, depending upon the instruction, into the destination accumulator and the carry bit.

ALU CARRY, SHIFT, LOAD OPTIONS

CLASS ABBREV.	CODED CHARACTER	RESULT BITS	OPERATION
C	(option omitted)	00	Do not initialize the carry bit.
	Z	01	Initialize the carry bit to 0.
	O	10	Initialize the carry bit to 1.
	C	11	Initialize the carry bit to the complement of its present value.
SH	(option omitted)	00	Leave the result of the arithmetic or logical operation unaffected.
	L	01	Combine the carry and the 16-bit result into a 17-bit number and rotate it one bit left.
	R	10	Combine the carry and the 16-bit result into a 17-bit number and rotate it one bit right.
	S	11	Exchange the two 8-bit halves of the 16-bit result without affecting the carry.
#	(option omitted)	0	Load the result of the shift operation into ACD.
	#	1	Do not load the result of the shift operation into ACD.

DG-04424

ALU SKIP OPTIONS

CLASS ABBREV.	CODED CHARACTER	RESULT BITS	OPERATION
SKIP	(option omitted)	000	Never skip
	SKP	001	Always skip
	SZC	010	Skip if carry = 0
	SNC	011	Skip if carry ≠ 0
	SZR	100	Skip if result = 0
	SNR	101	Skip if result ≠ 0
	SEZ	110	Skip if either carry or result = 0
	SBN	111	Skip if both carry and result ≠ 0

DG-04425

NOTE: Instructions in the Two Accumulator-Multiple Operation format must NOT have both the 'No Load' and the 'Never Skip' options specified at the same time. These bit combinations are used by other instructions in the instruction set.

TWO ACCUMULATOR-MULTIPLE OPERATION INSTRUCTIONS

In the function descriptions below, it is assumed that the carry bit is set to the specified value and the shift, conditional skip and load functions are performed as stipulated in the instruction field. When a conditional skip is specified in the instruction, the program counter is incremented an additional time if the condition is true.

INSTRUCTION FORMATS AND FUNCTIONS

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
COM	Complement	The one's complement of the contents of ACS is placed in the shifter. The output of the shifter is placed in ACD and the carry bit.	1 | ACS | ACD | 0 0 0 | SH | C | # | SKIP 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
NEG	Negate	The two's complement of the contents of ACS is placed in the shifter. If the operation produces a carry of 1, carry is complemented. The output of the shifter is placed in ACD and the carry bit.	1 | ACS | ACD | 0 0 1 | SH | C | # | SKIP 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
MOV	Move	Contents of ACS are placed in the shifter. The output of the shifter is placed in ACD and the carry bit.	1 | ACS | ACD | 0 1 0 | SH | C | # | SKIP 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
INC	Increment	Contents of ACS are incremented by one. If the operation produces a carry of 1, carry is complemented. The output of the shifter is placed in ACD and the carry bit.	1 | ACS | ACD | 0 1 1 | SH | C | # | SKIP 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
ADC	Add Complement	The one's complement of the contents of ACS is added to the contents of ACD. If the operation produces a carry of 1, carry is complemented. The output of the shifter is placed in ACD and the carry bit.	1 | ACS | ACD | 1 0 0 | SH | C | # | SKIP 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

INSTRUCTION FORMATS AND FUNCTIONS (CONT)

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
SUB	Subtract	The two's complement of the contents of ACS is added to the contents of ACD. If the operation produces a carry of 1, carry is complemented. The output of the shifter is placed in ACD and the carry bit.	1 | ACS | ACD | 1 0 1 | SH | C | # | SKIP 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
ADD	Add	The contents of ACS are added to the contents of ACD. If the operation produces a carry of 1, the carry is complemented. The output of the shifter is placed in ACD and the carry bit.	1 | ACS | ACD | 1 1 0 | SH | C | # | SKIP 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
AND	And	The contents of ACS and ACD are logically AND'd. The output of the shifter is placed in ACD and the carry bit.	1 | ACS | ACD | 1 1 1 | SH | C | # | SKIP 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

MULTIPLY/DIVIDE INSTRUCTIONS

The following instructions utilize an I/O instruction format and bits 10 to 15 specify device code 1. For this reason, device code 1 is an illegal device code and should not be assigned to any I/O interface device.

INSTRUCTION FORMATS AND FUNCTIONS

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
MUL	Multiply	The contents of AC1 are multiplied by the contents of AC2, yielding a 32-bit intermediate result. The intermediate result is added to the contents of AC0, yielding a final 32-bit result. The high order bits (0-15) of the result are loaded into AC0 and the low order bits are loaded into AC1. Bit 0 of AC0 contains the high-order bit of the result; bit 15 of AC1 contains the low-order bit. The contents of AC2 are unchanged.	0 1 1 1 0 1 1 0 1 1 0 0 0 0 0 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
DIV	Divide	The 32-bit contents of AC0 and AC1 are divided by the contents of AC2. Bit 0 of AC0 is the high-order bit of the dividend; bit 15 of AC1 is the low-order bit. The resulting quotient is placed in AC1 and the remainder in AC0. NOTE: Before the divide operation is performed, the contents of AC0 are compared to the contents of AC2. If the contents of AC0 are greater than AC2, an overflow condition is indicated. In this case, the carry bit is set to 1 and the operation is terminated; otherwise, the carry bit is set to 0.	0 1 1 1 0 1 1 0 0 1 0 0 0 0 0 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

STACK INSTRUCTIONS

The mN601 maintains a last-in/first-out stack residing in external memory. The stack facility provides an expandable area of temporary storage for variables, data, return addresses, subroutine arguments, etc. Its operation depends upon the contents of two 15-bit CPU registers, the stack pointer and the frame pointer.

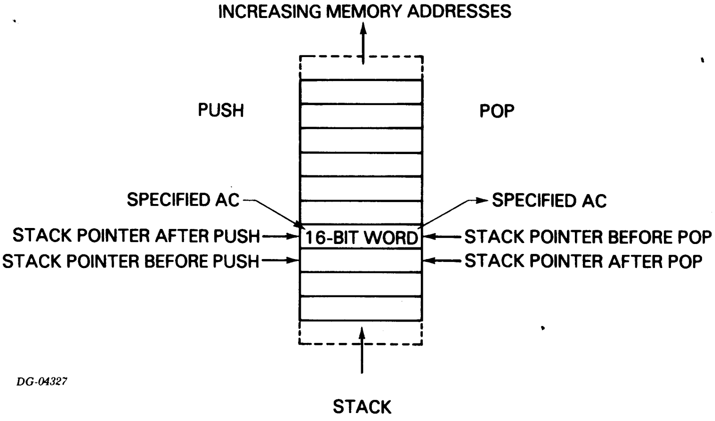
The stack pointer (SP) contains the memory address of the 'top' of the stack. It is affected by operations which either push information onto or pop information off of the stack.

The frame pointer (FP) contains a memory address that is used to reference an area within the stack, called a frame or a return block. A return block is a group of five memory locations which is used to store program reentry information. The FP usually points to the last return block pushed on the stack.

Two types of information are pushed on and off the stack, single variables and return blocks.

Two instructions, PUSH and POP, move single variables between the stack and any of the four accumulators. When a PUSH instruction is executed, the SP is incremented by one; then, the contents of the accumulator specified in the instruction are stored in the memory location addressed by the SP. When a POP instruction is executed, the contents of the memory location addressed by the SP are loaded into the accumulator specified in the instruction; then, the SP is decremented by one. See the illustration below.

PUSH/POP



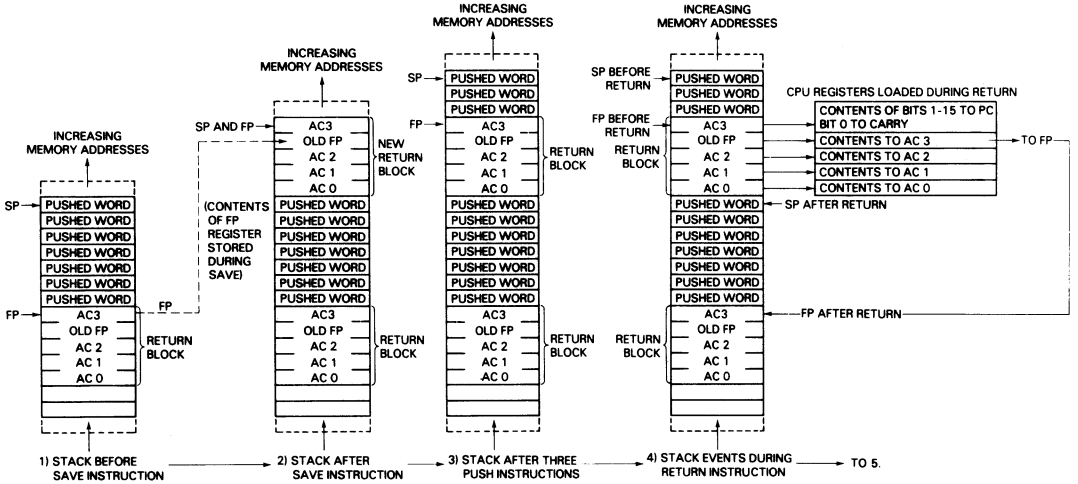
DG-04327

Two instructions, SAVE and RETURN, move information ·between a return block in the stack and the CPU registers described below. When a SAVE instruction is executed, the contents of the registers are stored in the return block addressed by the SP (see below). When a RETURN instruction is executed, the contents of the return block addressed by the FP are loaded into the appropriate registers (see below).

Return blocks are intended to be used when entering into and exiting from a subroutine. To implement this subroutine technique, the SAVE instruction is executed as the first instruction of a subroutine called by a JSR (JUMP TO SUBROUTINE) instruction (see Memory Reference Instructions). A JSR instruction stores the return address (contents of program counter plus one) in AC3. Then, the SAVE instruction stores the contents of AC3 plus the carry bit in the last word pushed on the stack. Also, SAVE stores the contents of AC0, AC1, AC2, and FP in the stack. After this information is pushed on the stack, the SP and the FP are updated to their new values, and AC3 contains the new FP.

The RETURN instruction is the last instruction of the called subroutine. When RETURN is executed, the return address is automatically loaded into the program counter (PC); the carry bit, AC0, AC1, AC2, FP and SP are restored to their original values before the execution of the SAVE instruction; and the restored value of FP is left in AC3. See the illustration below.

SAVE/RETURN


INCREASING MEMORY ADDRESSES
SP
FP
PUSHED WORD
AC3
OLD FP
AC 2
AC 1
AC 0
RETURN BLOCK
SP AND FP
NEW RETURN BLOCK
(CONTENTS OF FP REGISTER STORED DURING SAVE)
SP BEFORE RETURN
FP BEFORE RETURN
CPU REGISTERS LOADED DURING RETURN
CONTENTS OF BITS 1-15 TO PC
BIT 0 TO CARRY
CONTENTS TO AC 3
CONTENTS TO AC 2
CONTENTS TO AC 1
CONTENTS TO AC 0
TO FP
SP AFTER RETURN
FP AFTER RETURN
1) STACK BEFORE SAVE INSTRUCTION
2) STACK AFTER SAVE INSTRUCTION
3) STACK AFTER THREE PUSH INSTRUCTIONS
4) STACK EVENTS DURING RETURN INSTRUCTION
TO 5.
5) STACK AFTER RETURN INSTRUCTION

Typical stack events occurring when SAVE, PUSH and RETURN instructions are executed in sequence.

DG-04328

NOTE: The SP register contains the address of the top memory location in the stack. Thus, the three single variables pushed on the stack after the last return block (see Figure 4) are effectively pushed off the stack when the RETURN instruction is performed. These variables could be retained elsewhere in memory by executing a series of three POP instructions, each followed by a STA (STORE ACCUMULATOR, see Memory Reference Instructions) instruction prior to executing the RETURN instruction. Then, if desired, the variables can be pushed back on the stack after the execution of the RETURN instruction.

See Figures 4 and 5. After the RETURN instruction was executed, the contents of AC0, AC1, AC2, PC, carry and the SP and FP registers were restored to their original values before the SAVE instruction was executed. The new value of the FP register is left in AC3 after the RETURN instruction.

For ease of programming, there are four additional instructions which affect the contents of the stack and frame pointers: MTSP, MTFP, MFSP, and MFFP. When an MTSP (MOVE TO STACK POINTER) instruction is executed, the contents of the accumulator specified in the instruction are loaded into the SP; and the contents of the accumulator are unchanged. When an MFSP (MOVE FROM STACK POINTER) is executed, the contents of the SP are loaded into the accumulator specified in the instruction; and, the contents of the SP are unchanged. The MTFP (MOVE TO FRAME POINTER) and MFFP (MOVE FROM FRAME POINTER) instructions perform a similar function, moving information between the FP and the accumulator specified in the instruction.

During every instruction that pushes data onto the stack, a check is made for a stack overflow. A stack overflow exists when the stack pointer is incremented across a 256-word address boundary; e.g., xx377$_8$ to xx400$_8$. When this occurs, the instruction is completed and a stack overflow interrupt is generated, providing CPU interrupts are enabled. CPU interrupts are explained in the CPU I/O instructions.

Stack instructions utilize an I/O instruction format and bits 10 to 15 specify device code 1. For this reason, device code 1 is an illegal device code and should not be assigned to any I/O interface device (see I/O Operations section.)

INSTRUCTION FORMATS AND FUNCTIONS

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
PSHA	Push	The SP is incremented by 1; then, the contents of the specified AC are loaded into the memory location addressed by the contents of SP. The contents of the specified AC are unchanged. If the contents of bits 8-15 of SP are 000 (a 256-word boundary has been crossed), a stack overflow interrupt request is generated.	0 1 1 AC 0 1 1 0 0 0 0 0 0 0 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
POPA	Pop	The specified AC is loaded with the contents of the memory location addressed by SP; then, SP is decremented by 1.	0 1 1 AC 0 1 1 1 0 0 0 0 0 0 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
MTSP	Move To Stack Pointer	The contents of bits 1-15 of the specified AC are placed in SP. The contents of the specified AC are unchanged.	0 1 1 AC 0 1 0 0 0 0 0 0 0 0 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
MTFP	Move To Frame Pointer	The contents of bits 1-15 of the specified AC are placed in FP. The contents of the specified AC are unchanged.	0 1 1 AC 0 0 0 0 0 0 0 0 0 0 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
MFSP	Move From Stack Pointer	The contents of SP are placed in bits 1-15 of the specified AC and bit 0 of the specified AC is set to 0. The contents of the SP are unchanged.	0 1 1 AC 0 1 0 1 0 0 0 0 0 0 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
MFFP	Move From Frame Pointer	The contents of FP are placed in bits 1-15 of the specified AC and bit 0 of the specified AC is set to 0. The contents of FP are unchanged.	0 1 1 AC 0 0 0 1 0 0 0 0 0 0 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

INSTRUCTION FORMATS AND FUNCTIONS (CONT)

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
SAV	Save	Information is pushed on stack as follows: 1) the SP is incremented by 1; then, the contents of AC0 are stored in the memory location addressed by the contents of SP; 2) the SP is incremented by 1; then, the contents of AC1 are stored in the memory location addressed by the contents of SP; 3) the SP is incremented by 1; then, the contents of AC2 are stored in the memory location addressed by the contents of SP; 4) the SP is incremented by 1; then, the contents of FP (before the SAVE instruction) are stored in bits 1-15 of the memory location addressed by the contents of SP and 0 is written into bit 0. 5) the SP is incremented by 1; then, the content of the carry bit is stored in bit 0 and the contents of bits 1-15 of AC3 are stored in bits 1-15 of the memory location addressed by the contents of SP. If the contents of bits 8-15 of SP are 000 (a 256-word boundary was crossed) at any time during execution of this instruction, a stack overflow interrupt request is generated. At the end of the instruction, the new contents of SP are loaded in both the FP and bits 1-15 of AC3. Bit 0 of AC3 is set to 0.	0 1 1 0 0 1 0 1 0 0 0 0 0 0 0 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

INSTRUCTION FORMATS AND FUNCTIONS (CONT)

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
RET	Return	Information is popped off the stack as follows: 1) the contents of the FP are loaded into the SP; 2) bit 0 of the memory location addressed by the contents of SP is loaded into the carry bit and bits 1-15 are loaded into the PC; 3) the SP is decremented by 1; then, the contents of the location addressed by the contents of SP are loaded into AC3; 4) the SP is decremented by 1; then, the contents of the location addressed by the contents of SP are loaded into AC2; 5) the SP is decremented by 1; then, the contents of the location addressed by the contents of SP are loaded into AC1; 6) the SP is decremented by 1; then, the contents of the location addressed by the contents of SP are loaded into AC0. 7) the SP is decremented by 1. At the end of the instruction, the new contents of AC3 (bits 1-15) are loaded into FP.	0 1 1 0 0 1 0 1 1 0 0 0 0 0 0 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

INPUT / OUTPUT INSTRUCTIONS

This instruction group is used by the mN601 to communicate with I/O devices. It passes information between one of the mN601's four accumulators and the I/O bus and specifies one of three input and one of three output buffers (designated as A, B, and C) in an I/O controller connected to the bus.

I/O instructions accommodate a 6-bit device selection network which can address 64 separate device codes. When devices are interfaced to the mN601's I/O bus via an mN603 I/O Controller, each device can be connected to the bus in such a way that it will only respond to commands containing its own device code.

BUSY AND DONE FLAGS

It is assumed that each device has a set of flags, Busy and Done, that control its operation (see mN603). In general, these flags are used in the following manner. When they are both zero, the device is idle. To start a device, the program sets Busy to one and Done to zero. When an operation is finished, the device sets Done to one and Busy to zero. At this time, the device can issue a program interrupt request (see I/O Operations section.)

ADDRESSING NON-EXISTENT DEVICE CODES

If an attempt is made to issue an I/O instruction to a non-existent device, the mN601 functions as though the device exists and no indication is given that it does not exist. If an attempt is made to test the status of the Busy and Done flags (see below) of a non-existent device, the mN601 behaves as though the Busy and Done flag bits are 0. If a data in instruction is directed to a non-existent device, the specified accumulator will contain all ones. See functional description of I/O instructions.

The tables below illustrate the manner in which an I/O instruction is coded to set flags and how an I/O SKIP instruction is coded.

FLAG SETTING

CLASS ABBREV.	CODED CHARACTER	RESULT BITS	OPERATION
f	(option omitted)	00	Does not affect the Busy and Done flags.
	S	01	Start the device by setting Busy to 1 and Done to 0.
	C	10	Idle the device by setting both Busy and Done to 0.
	P	11	The effect, if any,depends on the device.

DG-04421

FLAG TESTING

CLASS ABBREV.	CODED CHARACTER	RESULT BITS	OPERATION
t	BN	00	Tests for Busy = 1
	BZ	01	Tests for Busy = 0
	DN	10	Tests for Done = 1
	DZ	11	Tests for Done = 0

DG-04422

The format of an I/O instruction is shown below. The number of data bits moved depends upon the size of the buffer and the operating mode of the device.

0	1	1	AC	OP CODE	CONTROL	DEVICE CODE
0	1	2	3 4	5 6 7	8 9	10 11 12 13 14 15

NOTE: The control bits, 8 and 9, can be f or t depending upon the type of instruction.

When the instructions described below are executed, the CPU sends the I/O instruction, exactly as it is stored in memory, to the I/O bus. Following the instruction, data is transferred either from the CPU or the I/O controller. See I/O Operations section.

INSTRUCTION FORMATS AND FUNCTIONS

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
NIO	No I/O Transfer	The Busy and Done flags of the addressed device are set according to the convention specified in bits 8-9 of the instruction. The data transferred by the CPU is meaningless and is ignored by the I/O controller.	0 1 1 | 0 0 | 0 0 0 | F | DEVICE CODE 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
DIA	Data In A	Following the receipt of this instruction, the addressed I/O controller is expected to send a 16-bit word to the I/O bus from its A buffer. This word is retrieved from the I/O bus by the CPU and placed in the accumulator specified in bits 3-4 of the instruction. After the data transfer, the Busy and Done flags are set according to the convention specified in bits 8-9 of the instruction.	0 1 1 | AC | 0 0 1 | F | DEVICE CODE 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
DIB	Data In B	Same as DIA, except B buffer.	0 1 1 | AC | 0 1 1 | F | DEVICE CODE 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
DIC	Data In C	Same as DIA, except C buffer.	0 1 1 | AC | 1 0 1 | F | DEVICE CODE 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
DOA	Data Out A	Following the receipt of this instruction, the addressed I/O controller should retrieve the next 16 bits of data appearing on the I/O bus and load them into its A buffer. The data bits are the contents of the accumulator specified in bits 3-4 of the instruction. After the data transfer, the Busy and Done flags are set according to the convention specified in bits 8-9 of the instruction.	0 1 1 | AC | 0 1 0 | F | DEVICE CODE 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
DOB	Data Out B	Same as DOA, except B buffer.	0 1 1 | AC | 1 0 0 | F | DEVICE CODE 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
DOC	Data Out C	Same as DOA, except C buffer.	0 1 1 | AC | 1 1 0 | F | DEVICE CODE 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

INSTRUCTION FORMATS AND FUNCTIONS (CONT)

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
SKP	I/O Skip	Following the receipt of this instruction, the addressed I/O controller sends the state, 0 or 1, of its Busy and Done flags to the I/O bus. This information is contained in a 16-bit data word. Data bit 0 contains the complement of the state of the Done flag. Data bit 1 contains the complement of the state of the Busy flag. The CPU retrieves this information from the I/O bus and ignores the contents of bits 2-15. It takes the appropriate action depending upon the content of the Busy and Done bits and the specified test condition contained in bits 8-9 of the instruction. If the test condition is true, the program counter is incremented an additional time.	0 1 1 | 0 0 | 1 1 1 | T | DEVICE CODE 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

PROGRAM INTERRUPTS

Program interrupts are sequences of operations during which the CPU writes the contents of the program counter into a specified memory location and loads the program counter with the address of another program. In the mN601, interrupts are performed at the request of a peripheral device, the real-time clock or the occurrence of a stack overflow condition.

The mN601 responds to program interrupts only when the interrupt enable flag contains a 1 and the CPU is either in the halted state or about to fetch an instruction. It performs an interrupt by incrementing the program counter, storing the updated contents of the program counter in memory location 0, performing a jump indirect through the appropriate memory location indicated below, and clearing the interrupt enable flag. When either a real-time clock or stack overflow interrupt request is honored, the corresponding request flag is cleared.

Type	Memory Location
External (I/O devices)	00001
CPU real-time clock	00002
Stack overflow	00003

When more than one type of interrupt is generated simultaneously, they are serviced according to the following priority scheme:

Type	Priority
Stack overflow	first
CPU real-time clock	second
External (I/O devices)	third

16-LEVEL PRIORITY INTERRUPTS

The mN601 provides sixteen levels of interrupts. These levels are established by the program, using a 16-bit word mask together with a CPU mask instruction. (See CPU I/O instructions.) Each bit in the mask (0-15) is assigned to a particular device, or devices operating at similar data transmission rates. To disable a device from generating interrupts, a 1 is placed in that device's mask bit; to enable the device, a 0 is placed in the mask bit. To use the mask, each I/O controller must contain an Interrupt Disable flag that is set by the CPU mask instruction to enable or disable interrupts.

CPU I/O INSTRUCTIONS

I/O instructions addressed to device code 77_8 are CPU I/O instructions. This instruction group performs special functions rather than controlling specific devices. Some CPU I/O instructions are global in nature and are directed to all I/O controllers; others are acted upon internally by the CPU.

In all but the I/O Skip instruction, I/O instructions addressed to device code 77_8 use bits 8-9 to control the condition of the Interrupt On flag (also referred to in the text as the interrupt enable flag). The table below shows the results of these bits when used with device code 77_8.

CPU ION FLAG SET AND TEST OPTIONS

CLASS ABBREV.	CODED CHARACTER	RESULT BITS	OPERATION
f	(option omitted)	00	Does not affect the state of the Interrupt On flag.
	S	01	Set the Interrupt On flag to 1.
	C	10	Set the Interrupt On flag to 0.
	P	11	Does not affect the state of the Interrupt On flag
t	BN	00	Tests for Interrupt On = 1.
	BZ	01	Tests for Interrupt On = 0.

DG-04420

When the instructions described below are executed, the CPU sends the instruction, exactly as it is stored in memory, to the I/O bus. In most cases, after the instruction is transmitted, data is transferred either from the CPU or the I/O controller. See I/O Operations section.

INSTRUCTION FORMATS AND FUNCTIONS

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
INTEN	Interrupt Enable	This instruction is internal to the CPU. When executed, the Interrupt On flag is set to 1 after one more instruction is executed; then interrupt requests are honored by the CPU. The data transmitted by the CPU following this instruction should be ignored by the I/O controllers.	0 1 1 | 0 0 | 0 0 0 | 0 1 | 1 1 1 1 1 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
INTDS	Interrupt Disable	This instruction is internal to the CPU. When executed, the Interrupt On flag is set to 0 and interrupt requests are not honored by the CPU. The data transmitted by the CPU following this instruction should be ignored by the I/O controllers.	0 1 1 | 0 0 | 0 0 0 | 1 0 | 1 1 1 1 1 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
INTA	Interrupt Acknowledge	This is a global instruction directed to all I/O controllers. It is sent to the I/O bus in response to a program interrupt request. The highest priority device requesting an interrupt should respond by sending its device code to the I/O bus in a 16-bit data word. The device code is contained in data bits 10-15. When the CPU retrieves this information, the device code is placed in bits 10-15 of the accumulator specified in bits 3-4 of the instruction. Data bits 0-9 are transmitted and loaded in the specified AC as 0's. If no device is requesting an interrupt, bits 0-15 of the specified accumulator are set to ones. If written in the form DIB< > ac, CPU the Interrupt On flag is set as specified by bits 8 and 9. By default, they are set to 00.	0 1 1 | AC | 0 1 1 | F | 1 1 1 1 1 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

INSTRUCTION FORMATS AND FUNCTIONS (CONT)

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
MSKO	Mask Out	This is a global instruction directed to all I/O controllers. After the CPU sends the instruction, it sends the contents of the accumulator specified in bits 3-4 of the instruction. The contents of the accumulator are interpreted by the I/O controllers as a 16-bit software priority mask. Each I/O controller responds (after one more instruction is executed) by turning its Interrupt Disable flag on or off according to the device's selected priority level. A 1 in a device's mask bit turns its Interrupt Disable flag on; a 0 turns it off. If written in the form DOB <f> ac, CPU the Interrupt On flag is set as specified by bits 8 and 9. By default, they are set to 00.	0 1 1 | AC | 1 0 0 | F | 1 1 1 1 1 1 0 1 2 | 3 4 | 5 6 7 | 8 9 | 10 11 12 13 14 15
IORST	Reset	This is a global instruction directed to all I/O controllers and the CPU. The I/O controllers respond by setting their Busy and Done flags to 0 and by turning their Interrupt Disable flags off. The CPU responds by disabling its real-time clock from generating interrupts. If written in the form DOA <f> 0, CPU bits 8 and 9 set the Interrupt On flag. By default, these bits are set to 10. The data transferred by the CPU following this instruction should be ignored by the I/O controllers.	0 1 1 0 0 | 0 1 0 | F | 1 1 1 1 1 1 0 1 2 3 4 | 5 6 7 | 8 9 | 10 11 12 13 14 15

INSTRUCTION FORMATS AND FUNCTIONS (CONT)

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
HALT	Halt	This instruction is internal to the CPU. It is used to stop the execution of instructions. The Interrupt On flag is set to 1. The CPU honors both program interrupt and data channel requests while it is in the halted state. When this instruction is executed, no data is transferred as part of the I/O transaction. See I/O Operations.	0 1 1 | 0 0 | 1 1 0 | 0 0 | 1 1 1 1 1 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
SKP	CPU Skip	This instruction is internal to the CPU. When executed, the CPU switches to I/O input mode (see I/O Operations section); but, no device should respond by placing data on the I/O bus. CPU Skip tests the state of the Interrupt On flag. If the test condition specified in bits 8-9 of the instruction is true, the program counter is incremented an additional time.	0 1 1 | 0 0 | 1 1 1 | T | 1 1 1 1 1 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
RTCEN	Real-time Clock Enable	This instruction is internal to the CPU. It enables the CPU's real-time clock to generate program interrupt requests. The Interrupt On flag is set according to the convention specified in bits 8-9 of the instruction. The data transmitted by the CPU following this instruction should be ignored by I/O controllers.	0 1 1 1 0 0 1 0 | F | 1 1 1 1 1 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

INSTRUCTION FORMATS AND FUNCTIONS (CONT)

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
RTCDS	Real-time Clock Disable	This instruction is internal to the CPU. It disables program interrupt requests by the CPU'S real-time clock. The Interrupt On flag is set according to the convention specified in bits 8-9 of the instruction. The data transmitted by the CPU following this instruction should be ignored by I/O controllers.	0 1 1 0 1 0 1 0 F 1 1 1 1 1 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

MEMORY OPERATIONS

The mN601 communicates with memory elements and I/O interfaces via its memory and I/O data buses and their associated control signals. The protocol used by the mN601 in transmitting and receiving information via its pins is described below.

The mN601 communicates with memory via the address/data pins, MB < 0-15 >, and three memory control signals, P, SAE and WE. Using these pins, it controls four kinds of memory operations, read, write, read-modify-write and refresh.

During read, write and read-modify-write operations, addresses and data are transferred between the pins and memory. During refresh operations, the CPU specifies 1/64 of the available memory locations to be refreshed by transferring a 6-bit refresh address to memory together with the appropriate control signals. MB <10-15> carry the refresh block address and MB <0-9> contain 1's.

ADDRESS / DATA PINS

The address/data pins, referred to as the memory address/data bus, are precharged to a logic high state for a period of 120ns every 480ns, regardless of the operation being performed by the CPU. Information is placed on the bus by discharging those lines which carry zeroes and leaving undischarged those lines which carry ones. Once the lines are discharged during a read or write operation, or by external devices, they cannot be used again until they are precharged.

CONTROL

Prior to initiating a memory operation, the CPU transmits the memory address via MB <1-15 >, and specifies the type of memory operation by transmitting MB0 in either the high or low state. When MB0 is at a high logic level, a refresh operation is selected; when it is at a low logic level, a read or write operation is selected.

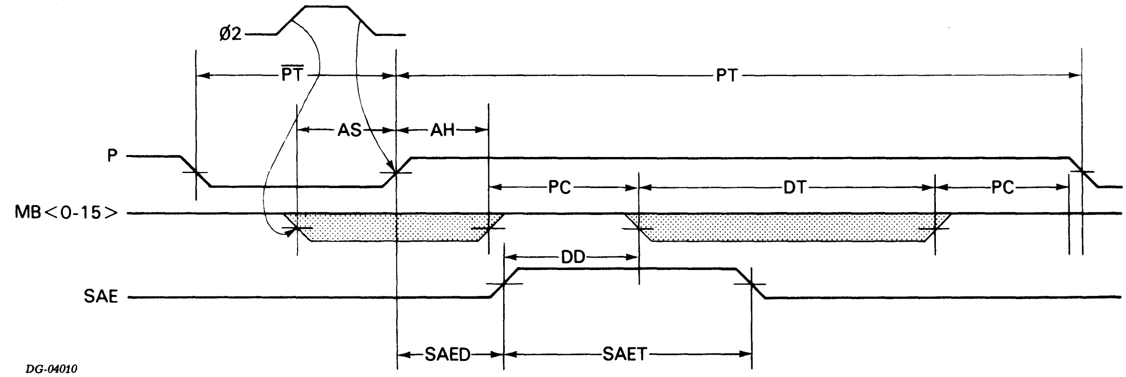
A memory operation is initiated on the rising edge of P. At this time, the memory samples the address on the address/data bus and tests the state of MB0.

> **NOTE:** The memory operations timing diagrams depicted on the following pages show the $\phi 2$ clock as a reference. Each action occurring on the mN601 output pins is synchronized to the rising or falling edge of a clock.

READ OPERATION

After the memory operation is initiated, the CPU transmits the SAE (SENSE AMP ENABLE) signal. After this signal goes to the high state, information is read from memory and placed on the address/data bus, and the CPU retrieves the information via the MB <0-15> pins.

READ TIMING DIAGRAM *



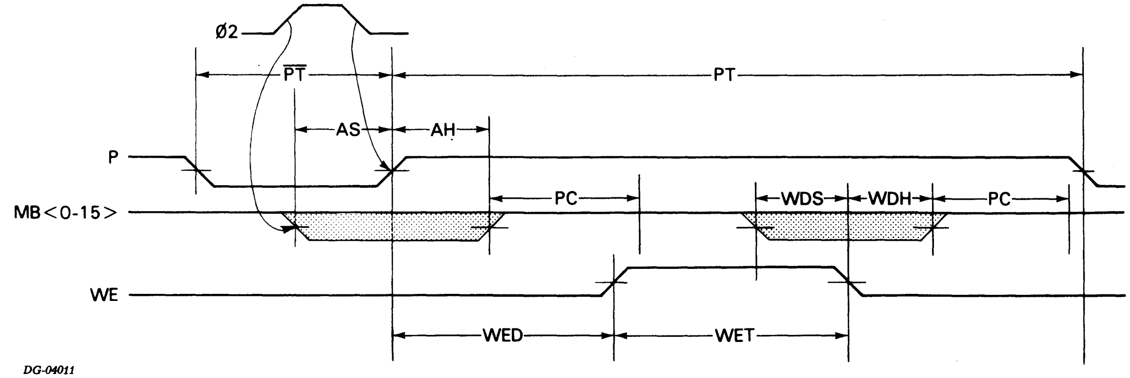
* UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.

FOR TIMING INTERVALS, SEE TIMING TABLE ON PAGE 49.

WRITE OPERATION

After the memory operation is initiated, the CPU transmits both the data to be written into memory via MB <0-15> and the WE (WRITE ENABLE) signal. When WE goes from the high to the low logic level, the information on the address/data bus is valid and is written into memory.

WRITE TIMING DIAGRAM *



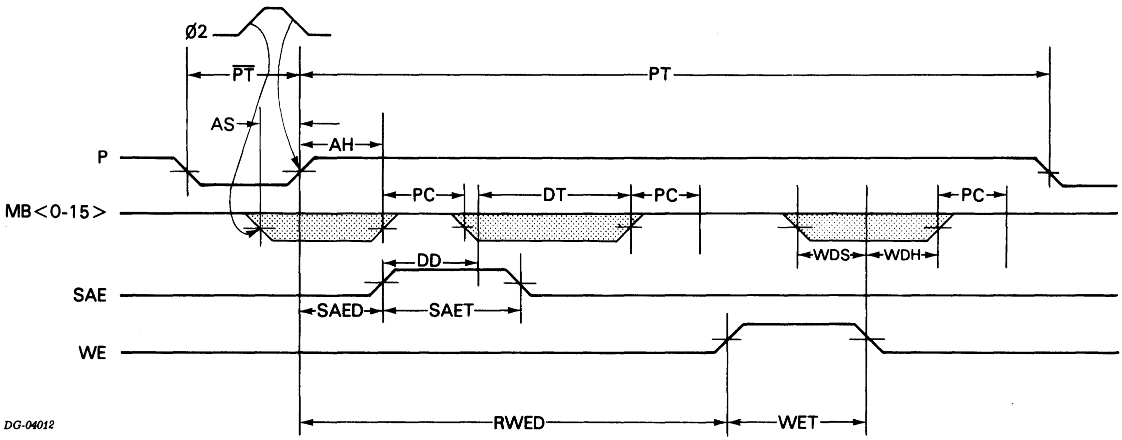
* UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.

FOR TIMING INTERVALS, SEE TIMING TABLE ON PAGE 49.

READ-MODIFY-WRITE OPERATION

After the memory operation is initiated, a read operation is performed. Then, the address/data bus is precharged and a write operation follows. P remains in the high state until after the memory operation is completed.

READ-MODIFY-WRITE TIMING DIAGRAM*



* UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.

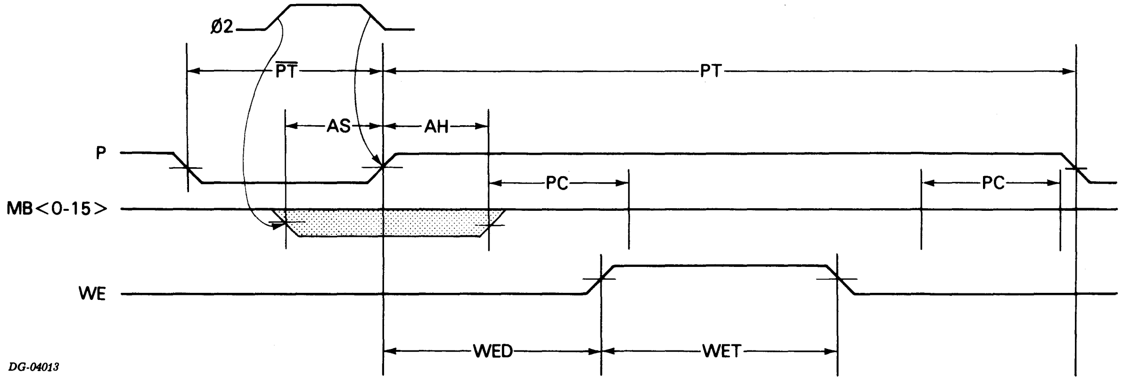
FOR TIMING INTERVALS, SEE TIMING TABLE ON PAGE 49.

REFRESH OPERATION

After the refresh operation is initiated, the CPU transmits the WE signal, but does not transmit data, and the information selected by the address is refreshed.

The mN601 supports dynamic RAM elements by refreshing all memory locations at least once every 1.8432ms.

REFRESH TIMING DIAGRAM*



* UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.

FOR TIMING INTERVALS, SEE TIMING TABLE ON PAGE 49.

NOTE: At times, a memory operation is initiated and then aborted by the CPU. This condition arises when the CPU makes an"intelligent decision"while performing a task;for example, during indirect addressing chains. When this occurs, the memory control signal P is asserted and the address appears on the memory bus; but, no memory operation is performed.

I/O OPERATIONS

Data is passed between the mN601 and the mN603 I/O controllers connected to the microNOVA I/O bus under two operating modes: programmed I/O and data channel break. The CPU is switched to one of these operating modes at the request of the I/O controllers, through the use of the program interrupt and data channel request facilities.

PROGRAMMED I/O

In programmed I/O, the CPU moves information, consisting of a command followed by data, between its accumulators and the I/O bus. There is one exception to this rule; that is, when an I/O Skip instruction is executed. The data received in response to this command contains the state, 0 or 1, of the addressed device's Busy and Done flags. This information is not loaded into an accumulator. It is acted upon directly by the CPU. Using this information together with the command (e.g., SKPDN - Skip If Done Is Non-Zero), the CPU determines whether the test condition specified in the command is true or false. If it is true, the program counter is incremented an additional time.

DATA CHANNEL BREAK

In a data channel break, the CPU moves information between the I/O bus and memory without altering the program flow. This information transfer is transparent to the executing program. When the CPU honors a data channel request, it sends an acknowledgement to the I/O bus. This is responded to by the I/O controller with the highest priority requesting a data channel break. The I/O controller sends to the CPU the appropriate 15-bit memory address and a direction of transfer mode bit (0 equals data out; 1 equals data in). All references to direction of transfer are always in relation to the CPU; that is, data in to the CPU, data out from the CPU. When the CPU receives the address, it begins a read-modify-write operation using the address received from the I/O controller. Then, it examines the mode bit.

If the mode bit is set for a data out, the information received by the CPU during the read portion of the memory operation is sent to the I/O bus. The write portion of the memory operation is not executed.

If the mode bit is set for a data in, the CPU switches to I/O input mode and then waits a certain period of time to receive the data. The 16 data bits received from the I/O controller during this time period are sent to the memory bus and are written into memory during the write portion of the memory operation being performed. The information received by the CPU during the read portion of the memory operation is ignored.

I/O BUS TRANSFERS

Information is passed in serial form between the CPU and the I/O bus via the CPU I/O bus pins. The pins and their functions are listed below.

CPU I/O BUS PINS

PIN MNEMONIC	FUNCTION
I/O DATA 1	Bi-directional, data bus. This line carries the 8 most significant bits (0-7) of a 16-bit word in serial form, plus one control code prefix bit. (Code bits are explained below.)
I/O DATA 2	Bi-directional data bus. This line carries the 8 least significant bits (8-15) of a 16-bit word in serial form, plus one control code prefix bit.
I/O CLOCK	Bi-directional, synchronizing clock line. The I/O CLOCK strobes data into the receiving device at a rate of 8.33Mbits per line.
$\overline{\text{DCHINT}}$	This is a wired-OR of all I/O controllers' data channel request lines. When asserted low by any of the I/O controllers, it tells the CPU that a device, or devices, is requesting a data channel break.
$\overline{\text{EXTINT}}$	This is a wired-OR of all I/O controllers' program interrupt lines. When asserted low by any of the I/O controllers, it tells the CPU that a device, or devices, is requesting a program interrupt.
I/O INPUT	This line indicates the operating mode of the mN601 to the CPU I/O transceiver connected to the I/O bus. When it is at the low logic level, it tells the transceiver that the CPU is sending information, and the transceiver should receive the information and pass it to the I/O bus. When the line is at a high logic level, it tells the transceiver that the CPU is ready to receive data, and the information on the I/O bus should be retrieved and sent to the CPU. Except for the intervals during which the CPU is transmitting information, this line remains in the high logic state.

mN601
I/O OPERATIONS

The I/O data lines (I/O DATA<1-2>) are used to transfer four types of information: Request Enable, Data Channel Address Request, programmed I/O commands, and data. Each type consists of either 2 bits or 18 bits (1 or 9 bits on each I/O data line) and is identified by the first bit transmitted on each line. The encoding is shown below.

TRANSFERS VIA I/O DATA LINES

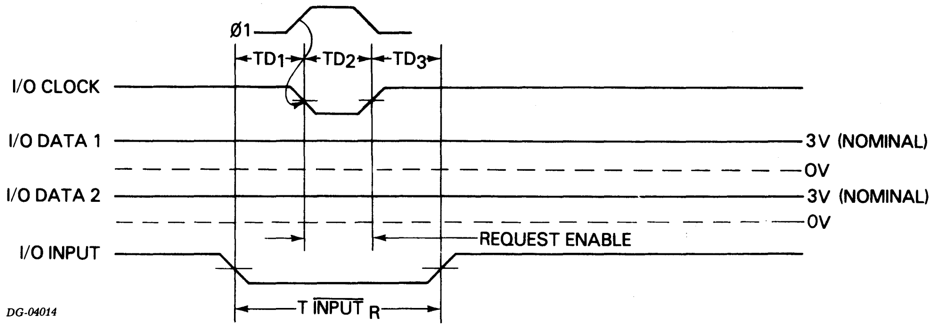
FIRST BIT CODE I/O DATA 1	FIRST BIT CODE I/O DATA 2	I/O TRANSFER TYPE	TOTAL NO. OF BITS TRANSFERRED
1	1	Request Enable	2 code bits only
1	0	Data Channel Address Request	2 code bits only
0	1	Data	18 bits (2 code bits plus 16 data bits)
0	0	I/O Command	18 bits (2 code bits plus 16 command bits)

NOTE: The I/O operations timing diagrams depicted below show the $\phi 1$ clock as a reference. Each action occurring on the mN601 output pins is synchronized to the rising or falling edge of a clock.

REQUEST ENABLE

This is a signal sent by the CPU, in code form, to all devices connected to the I/O bus. It is sent at irregular intervals and is used by the individual I/O controllers to synchronize the assertion of program interrupt and data channel request lines. Request Enable ensures that the $\overline{\text{EXTINT}}$, $\overline{\text{DCHINT}}$, and priority lines are at a steady state when they are examined by the CPU.

REQUEST ENABLE TIMING DIAGRAM *



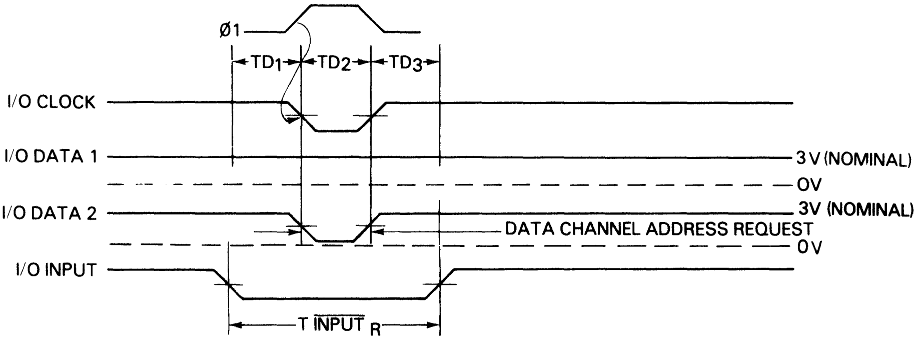
* UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.

FOR TIMING INTERVALS, SEE TIMING TABLE ON PAGE 49.

DATA CHANNEL ADDRESS REQUEST

This is a signal sent by the CPU, in code form, to the I/O bus in response to a data channel request. It tells the I/O controllers that the CPU is ready to receive the data channel address and mode bit. It is responded to by the highest priority device requesting a data channel break. Data Channel Address Request also acts as a Request Enable.

DATA CHANNEL ADDRESS REQUEST TIMING DIAGRAM *



DG-04015

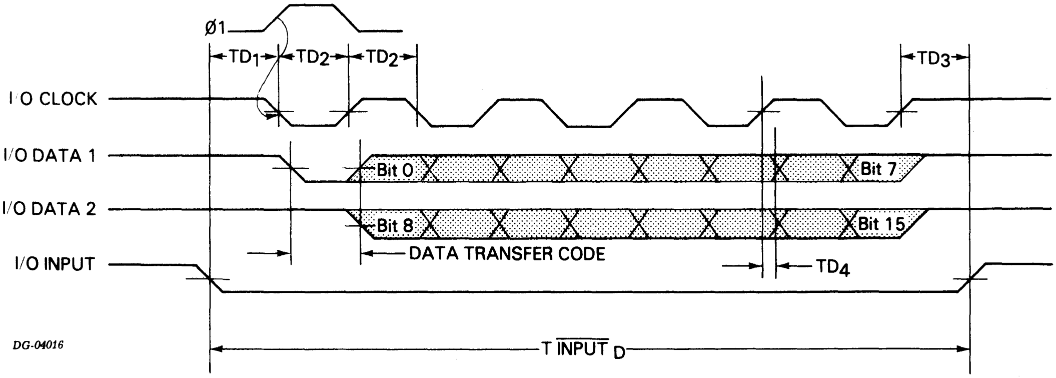
* UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.

FOR TIMING INTERVALS, SEE TIMING TABLE ON PAGE 49.

DATA

The format shown in the Data Timing Diagram is used to transfer 16 data bits on the I/O bus. In programmed I/O, data is any 16-bit transfer which is not an I/O command. In a data channel break, this format is used to transfer the 15-bit memory address plus mode bit and the 16-bit word.

DATA TIMING DIAGRAM *



DG-04016

* UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.

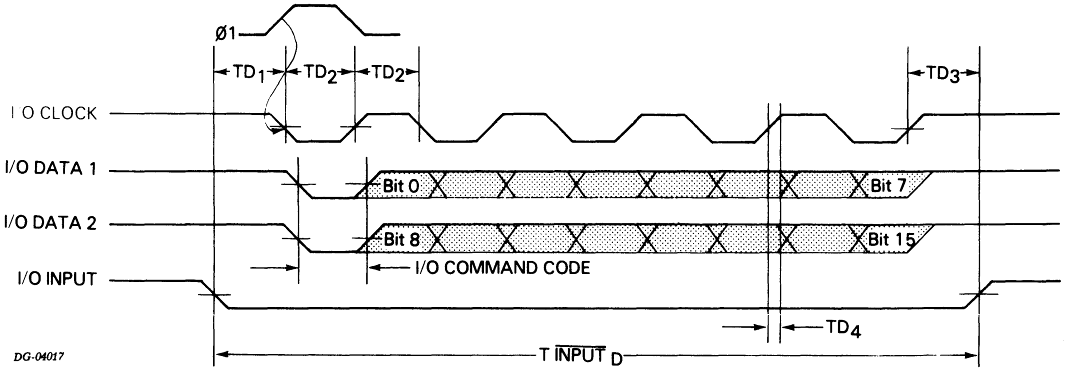
FOR TIMING INTERVALS, SEE TIMING TABLE ON PAGE 49.

NOTE: The timing for incoming data to the CPU is the same as that for outgoing data, except that the I/O INPUT signal remains at a high logic level.

I/O COMMAND

The format shown in the I/O Command Timing Diagram is used by the CPU to transfer all instructions to the I/O bus in which bits 0-2 contain 011 (I/O instruction format).

I/O COMMAND TIMING DIAGRAM *



* UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.

FOR TIMING INTERVALS, SEE TIMING TABLE ON PAGE 49.

RULE: 00, 01_8, 02_8, 03_8, and 77_8 are ILLEGAL DEVICE CODES for any I/O device connected to the microNOVA I/O bus.

The mN601 Instruction Set section describes several groups of instructions which incorporate the I/O format: I/O, CPU I/O, Multiply/Divide, and Stack. Each of these groups is discussed below.

Each I/O instruction should contain a device code in bits 10-15 which specifies one of the existing devices connected to the I/O bus. When one of these instructions is sent to the I/O bus by the CPU, each mN603 I/O controller examines the device code and determines what action, if any, should be taken. Most instructions in this group are followed by a data transfer, either from the CPU or the I/O controller. See mN601 Instruction Set.

Each CPU I/O instruction contains device code 77_8 (illegal device code). Some of these instructions are global in nature and are directed to all I/O controllers; others are acted upon internally by the CPU. The latter should be ignored by all I/O controllers. The effect of each is described in the CPU I/O Instructions section of the mN601 Instruction Set.

Both the Multiply and Divide instructions address device code 01_8 (illegal device code). Although these instructions are internal to the CPU, the instruction is sent to the I/O bus when executed; but, no data is transferred. I/O controllers should ignore these instructions.

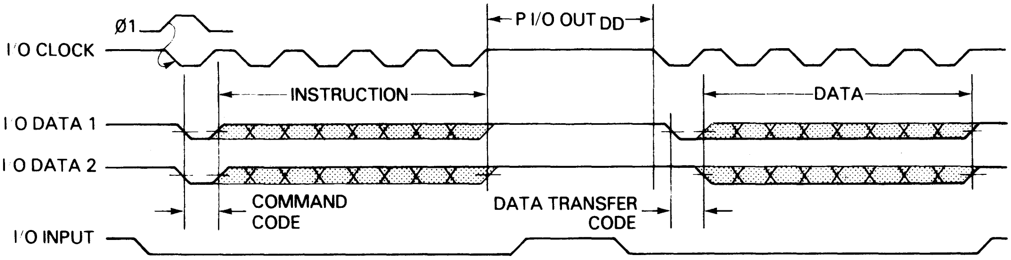
Stack instructions also address device code 01_8 (illegal device code). Their effect is the same as that for Multiply/Divide instructions. I/O controllers should ignore these instructions.

I/O TRANSFER PROTOCOL

Programmed I/O Data Out

The programmed data out timing diagram depicts a transfer of an I/O instruction followed by a data transfer from the CPU; e.g., DOA, DOB, DOC, MSKO. After the instruction is sent to the I/O bus, the CPU waits for a fixed interval of time before sending the data.

PROGRAMMED DATA OUT TIMING DIAGRAM *



DG-04018

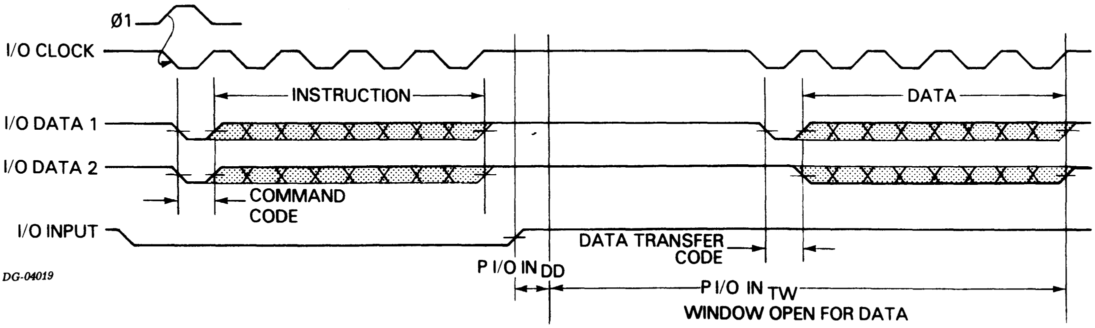
* UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.

FOR TIMING INTERVALS, SEE TIMING TABLE ON PAGE 49.

Programmed I/O Data In

The programmed data in timing diagram depicts a transfer of an I/O instruction followed by a data transfer from an mN603 I/O controller; e,g., DIA, DIB, DIC, INTA, I/O SKIP. After the CPU sends the instruction to the I/O bus, it switches to I/O input mode and opens a window for the reception of a data word for a fixed interval of time. It is expected that the I/O controller addressed by the instruction will send 16 data bits plus the control bits during this time frame. When the window closes, the CPU performs the command specified in the instruction using the data bits received.

PROGRAMMED DATA IN TIMING DIAGRAM *



DG-04019

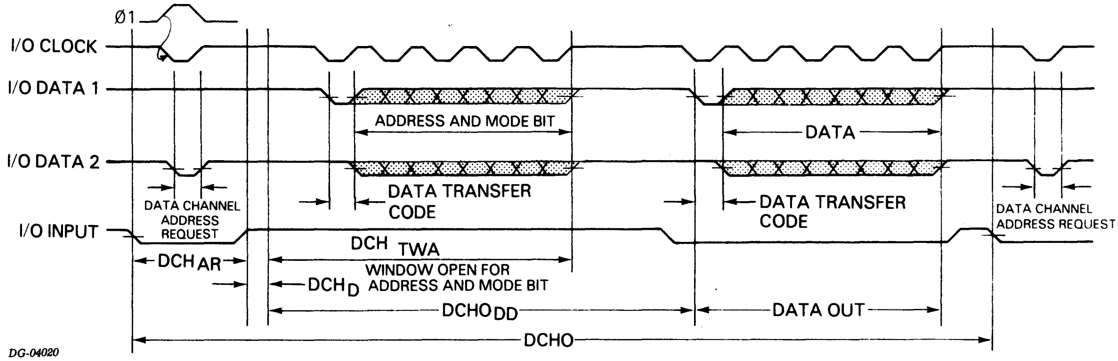
* UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.

FOR TIMING INTERVALS, SEE TIMING TABLE ON PAGE 49.

Data Channel Data Out

The sequence of events from the time the CPU acknowledges a data channel request through the time the data is placed on the I/O bus is depicted in the timing diagram below. After the CPU sends a Data Channel Address Request to the bus, it switches to data channel input mode. At this time the CPU opens a data-in window for a fixed interval of time. It is expected that the I/O controller with the highest priority requesting a data channel break will send the memory address and the direction mode bit during this time frame. When the window closes, the memory address contained in the data transfer is used by the CPU in performing a memory read operation. When the data is retrieved from memory, the CPU transfers it to the I/O bus by performing a data out operation.

DATA CHANNEL MEMORY TO DEVICE TIMING DIAGRAM *



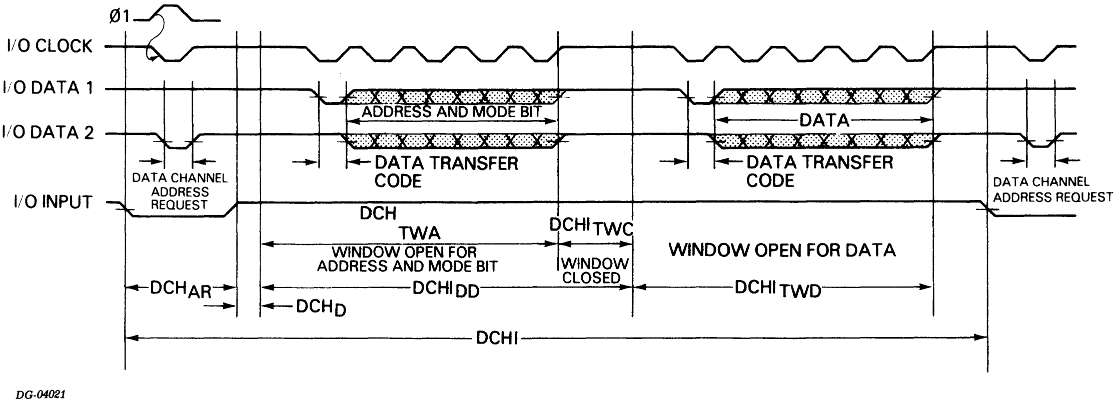
* UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.

FOR TIMING INTERVALS, SEE TIMING TABLE ON PAGE 49.

Data Channel Data In

The sequence of events from the time the CPU acknowledges a data channel request through the time the data is received by the CPU from the I/O bus is depicted in the timing diagram below. After the CPU sends a Data Channel Address Request, it switches to data channel input mode. Then, it opens the data-in window twice for fixed intervals of time. The first time is to retrieve the memory address and mode bit. The second time it is expected that the I/O controller will send one 16-bit word plus the control bits. When the window closes, the data bits received are sent to the memory bus and a memory write operation is performed.

DATA CHANNEL DEVICE TO MEMORY TIMING DIAGRAM *



* UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.

FOR TIMING INTERVALS, SEE TIMING TABLE ON PAGE 49.

TIMING

On systems which have a number of I/O controllers and depend heavily on input/output, both programmed I/O and data channel facilities can be overloaded. This overloading causes certain peripherals to lose data due to the inability of the system to respond to them in time.

Programmed I/O Latency

Most peripherals operating under program control request processor time by setting their Done flags to 1. Whether the CPU determines that a device is requesting service by repeatedly checking the state of the flags using I/O Skip instructions or by means of program interrupts, there is a delay between the time the peripheral requests service (sets its Done flag to 1) and the time the CPU fetches the first instruction of the peripheral service routine. This delay is called programmed I/O latency.

When the program interrupt facility is not used, this time is dependent upon the following: frequency of the I/O Skip instructions and efficiency of the software servicing the peripherals; i.e., the time required by the peripheral service routine to transfer data to or from the peripheral and set the Done flag to 0.

When the program interrupt facility is used, I/O latency is defined as the sum of the following:

1. The time from the setting of the Done flag to 1 to the end of the instruction currently being executed by the CPU.
2. The time when CPU operation is suspended while data channel transfers are in progress. (See the following section.)
3. The time program interrupts are turned off, e.g., during the servicing of an interuput from another peripheral.
4. The time required to execute two of the longest consecutive instructions present in the program. See the table of Instruction Execution Times at the end of this section.
5. Interrupt cycle time, which is comprised of the following: 6 CPU cycles plus 2 cycles per indirect level, where 1 CPU cycle equals 0.480 μsec.

   Interrupt cycle time is the time required for the interrupt facility hardware to store the contents of the program counter plus 1 in location 0, and to simulate a Jump indirect instruction to location 1 (or to location 2 or 3 if the interrupt is generated by the CPU's real-time clock or stack overflow, respectively).
6. The time required by the interrupt handler to identify the peripheral and transfer control to that peripheral's service routine.
7. The time required by the service routine to transfer data to or from the peripheral and set Done to 0.

Items 1, 4, and 5 are machine dependent. Item 2 depends upon the number of data channel devices and their activity rate. Items 3, 6 and 7 are determined by the software and represent the bulk of the programmed I/O interrupt latency.

Additionally, if any higher priority program interrupt, i.e., stack overflow or real-time clock, has a pending interrupt, the time calculation is the same. The CPU always honors the interrupt with the highest priority.

Programmed I/O interrupt latency is important because a peripheral device that must wait too long to be serviced may suffer from loss of data. For this reason, the mN601 incorporates the priority interrupt facility. To minimize the loss of important data, the software priority levels should be assigned on the basis of the following considerations:

1. The maximum allowable programmed I/O latency for each peripheral device.
2. The result of exceeding the maximum allowable programmed I/O latency for each peripheral device; i.e.,slowdown or data loss.

Data Channel Latency

Latency is also a factor when data is transferred between peripheral devices and memory via data channel breaks. The CPU allows data channel devices to access memory only during predefined times. In addition, more than one peripheral may be waiting for service at any given time.

Data channel latency is defined as the time between the I/O device's request for service (the $\overline{\text{DCH SYNC}}$ line is asserted low by the device - see mN603) and the time when data is strobed, either during data reception or transmission. Data channel latency consists of the sum of either of the following values:

Data Channel Data In (Device to Memory) — The time required to execute the two longest consecutive instructions, excluding Multiply and Divide, (see the table of Instruction Execution Times) plus 6 CPU cycles, where 1 CPU cycle equals 0.480 . sec.

Data Channel Data Out (Memory to Device) — The time required to execute the two longest consecutive instructions, excluding Multiply and Divide, plus 11 CPU cycles.

For example, if the two longest consecutive instructions in the program are 15 CPU cycles each, the maximum latency for a data channel data out transfer is:

$$15 + 15 + 11 = 41 \text{ CPU cycles;}$$
$$41 \times .480\mu\text{sec} = 19.68\mu\text{sec}$$

Additionally, there is a minimum data channel latency. The minimum latency for a data channel data in is 11 CPU cycles; the minimum for a data channel data out is 16 CPU cycles.

Minimum latency can be used in computing data channel latency when it is known that the device will request a data transfer. Thus, a data channel request can be initiated 11 CPU cycles (for a data channel data in) or 16 CPU cycles (for a data channel data out) before the device is ready to transfer data; thereby reducing the latency factor from data ready to data transfer.

The above computations assume the request is by the highest priority device requesting service.

When there are other, higher priority devices requesting data channel transfers, the time required for these transfers must be added to the latency. For each data channel out transfer the time is 7.2μsec and for each data channel in transfer the time is 5.8μsec.

STATUS

INITIALIZATION

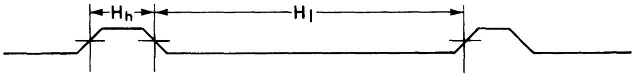
On power up, V_{BB} must be at -3V before V_{GG}, V_{DD} and V_{CC} start going positive.

After the mN601 is initially brought up to the recommended operating specifications (see DC Characteristics), the $\overline{\text{CLAMP}}$ pin must be held at a low logic level for a minimum of 100μsec. Then, when $\overline{\text{CLAMP}}$ goes from the low to high state, the CPU is properly initialized and enters the HALT state (see below). After initialization, pulling $\overline{\text{CLAMP}}$ low for a period of 100μsec causes the mN601 to reset and enter the HALT state when $\overline{\text{CLAMP}}$ goes from the low to high logic state. The contents of internal registers and RAM memory are indeterminate whenever $\overline{\text{CLAMP}}$ is low for a period greater than 100μsec.

HALT

When the mN601 is in the HALT state, it generates a sequence of 0.417MHz pulses from the HALT pin as shown in the timing diagram below; otherwise, the HALT pin remains in the low logic state.

HALT TIMING DIAGRAM *



* UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.

SEE TIMING TABLE ON PAGE 49.

DG-04329

In the HALT state, the CPU performs the following functions: it generates Request Enable signals, it responds to both program interrupts and data channel requests, and it generates dynamic RAM refresh signals. The normal way to start the mN601 when it is in the HALT state is to generate an interrupt.

RESET

The mN601 is reset by pulling the I/O CLOCK pin low for a period of 10μsec. When I/O CLOCK rises to the high logic state after the CPU is reset, the mN601 enters the HALT state and the contents of internal registers are indeterminate.

NOTE: Do NOT pull I/O CLOCK low when I/O INPUT is low.

INSTRUCTION EXECUTION TIMES

The table on the following page gives instruction execution times in CPU cycles by instruction type and addressing mode.

INSTRUCTION EXECUTION TIMES IN CPU CYCLES
(ONE CPU CYCLE=0.480 USEC)

INSTRUCTION GROUP TYPE	ADDRESSING MODES		
MEMORY REFERENCE GROUP	**DIRECT**	**INDIRECT ***	**AUTO INCREMENT/ DECREMENT**
DSZ	8**	10**	13**
ISZ	8**	10**	13**
JMP	6	7	10
JSR	7	9	12
LDA	6	8	11
STA	6	8	11
ARITHMETIC AND LOGICAL GROUP	**REGISTER ADDRESSING**		
ADC	5**		
ADD	5**		
AND	5**		
COM	5**		
INC	5**		
MOV	5**		
NEG	5**		
SUB	5**		
MULTIPLY/DIVIDE GROUP	**REGISTER ADDRESSING**		
DIV	122		
MUL	87		
INPUT/OUTPUT GROUP	**REGISTER ADDRESSING**		
DIA	15		
DIB	15		
DIC	15		
DOA	10		
DOB	10		
DOC	10		
NIO	10		
SKP	15**		
STACK CONTROL GROUP	**REGISTER ADDRESSING**		
MFFP	8		
MFSP	7		
MTFP	6		
MTSP	6		
POPA	7		
PSHA	7		
RET	15		
SAV	16		
CPU CONTROL GROUP	**REGISTER ADDRESSING**		
HALT	10		
INTA	15		
INTDS	10		
INTEN	10		
IORST	10		
MSKO	10		
RTCEN	10		
RTCDS	10		
TRAP	9*		

* FOR EACH ADDITIONAL INDIRECT LEVEL, ADD 2 CYCLES;
FOR EACH AUTO-INCREMENT/DECREMENT, ADD 5 CYCLES
** IF SKIP OCCURS, ADD 2 CYCLES

ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS

Supply voltage, V_{BB}	-7V
Supply voltage, V_{CC}	7V
Supply voltage, V_{DD}	13V
Supply voltage, V_{GG}	17V
Input voltage, V_I	7V
Operating free air temperature range, T_A	0 deg. to 70 deg.C
Storage temperature range, T_{STG}	-55 deg. to 125 deg.C

NOTE: All voltages are referenced to ground. Subjecting a circuit to conditions either outside or at these limits for an extended period of time may cause irreparable damage to the circuit. As such,these ratings are not intended to be used during the operation of the circuit.

RECOMMENDED OPERATING CONDITIONS

Supply voltage, V_{BB}	-4.25V ± 0.5V
Supply voltage, V_{CC}	5V ± 0.25V
Supply voltage, V_{DD}	10V ± 1V
Supply voltage, V_{GG}	14V ± 1V
Operating free air temperature range, T_A	0 deg. to 70 deg.C

DC CHARACTERISTICS

SYMBOL	CHARACTERISTICS	CONDITIONS	MIN	TYP	MAX	UNITS
V_{IL}	φ1, φ2 (clock input) MB <0-15> , $\overline{\text{CLAMP}}$, $\overline{\text{EXTINT}}$, $\overline{\text{DCHINT}}$ I/O CLOCK, I/O DATA 1, I/O DATA 2		-2.0 -1.0 -0.5		0.8 1.0 0.5	V V V
V_{IH}	φ1, φ2 MB <0-15>, $\overline{\text{CLAMP}}$, $\overline{\text{EXTINT}}$, $\overline{\text{DCHINT}}$ I/O CLOCK, I/O DATA 1, I/O DATA 2		13 3.75 2.70		15 5.8 5.8	V V V
I_{IL}	φ1, φ2 MB <0-15> $\overline{\text{EXTINT}}$, $\overline{\text{DCHINT}}$ $\overline{\text{CLAMP}}$ I/O CLOCK, I/O DATA 1, I/O DATA 2	V_{IL} = 0.8V V_{IL} = 1V V_{IL} = 1V V_{IL} = 1V V_{IL} = 0.5V			10 -2 -4 10 -4	μA mA mA μA mA

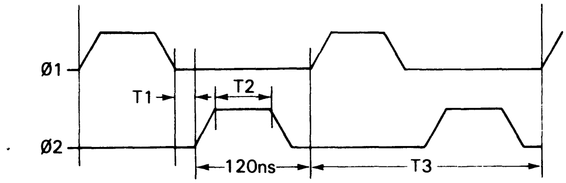
DC CHARACTERISTICS (CONT)

SYMBOL	CHARACTERISTICS	CONDITIONS	MIN	TYP	MAX	UNITS
I_{IH}	ϕ1, ϕ2	V_{IH} = 15V			10	μA
	MB <0-15>	V_{IH} = 5.8V			60	μA
	$\overline{\text{EXTINT}}$, $\overline{\text{DCHINT}}$	V_{IH} = 5.8V			100	μA
	$\overline{\text{CLAMP}}$	V_{IH} = 5.8V			10	μA
	I/O CLOCK, I/O DATA 1, I/O DATA 2	V_{IH} = 5.8V			100	μA
VPCHS	MB <0-15> (VOLTAGE AFTER PRECHARGE)	IPCHS = -60μ A	4.25	4.75	5.25	V
IPCHS	MB <0-15> (CURRENT AFTER PRECHARGE)	VPCHS = 4.25V			-60	μA
V_{OL}	MB <0-15> ,SAE,WE,$\overline{\text{PAUSE}}$,P, I/O INPUT, I/O CLOCK, I/O DATA 1 I/O DATA 2, HALT				0.5	V
V_{OH}	MB <0-15> ,SAE,WE,PAUSE,P, I/O INPUT, I/O CLOCK, I/O DATA 1 I/O DATA 2, HALT		2.75			V
I_{OL}	MB <0-15> ,SAE,WE,$\overline{\text{PAUSE}}$,HALT	V_{OL} = 0.5V			2.0	mA
	I/O CLOCK,I/O DATA 1,I/O DATA 2				2.0	mA
	I/O INPUT, P				4.0	mA
I_{OH}	MB <0-15> ,SAE,WE,$\overline{\text{PAUSE}}$,HALT	V_{OH} = 2.75V			-100	μA
	I/O CLOCK, I/O DATA 1, I/O DATA 2				-100	μA
	P, I/O INPUT				-200	μA
I_{BB}	Average				-250	μA
I_{CC}	Average				20	mA
I_{DD}	Average				60	mA
I_{GG}	Average				30	mA
CAPACITANCE						
Clock line (ϕ1, ϕ2) capacitance					100	pF
Capacitance of all other signal pins					10	pF
External load capacitance on MB <0-15>. I/O CLOCK, I/O DATA 1, I/O DATA 2					20	pF
External load capacitance between any two MB lines.					10	pF
External load capacitance on P. I/O INPUT, SAE, WE, $\overline{\text{PAUSE}}$, HALT					30	pF

NOTE: Positive current is into the pin.

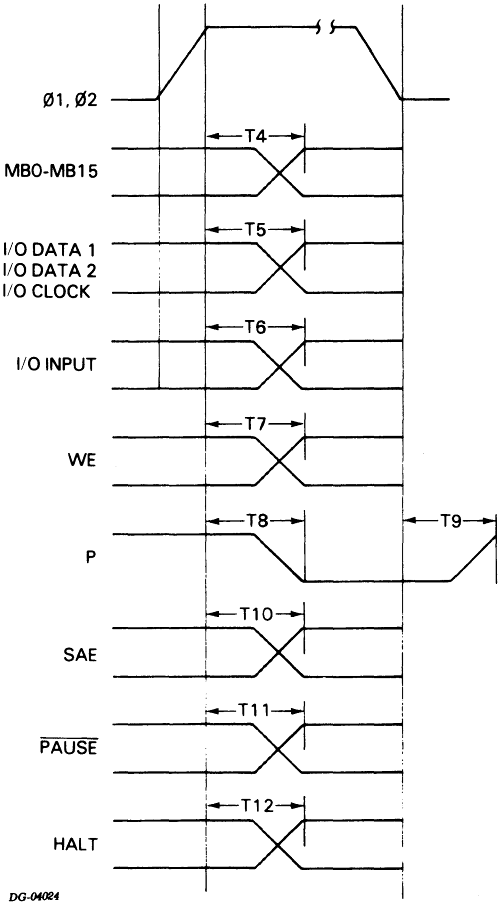
AC CHARACTERISTICS SWITCHING DIAGRAMS

CLOCKS



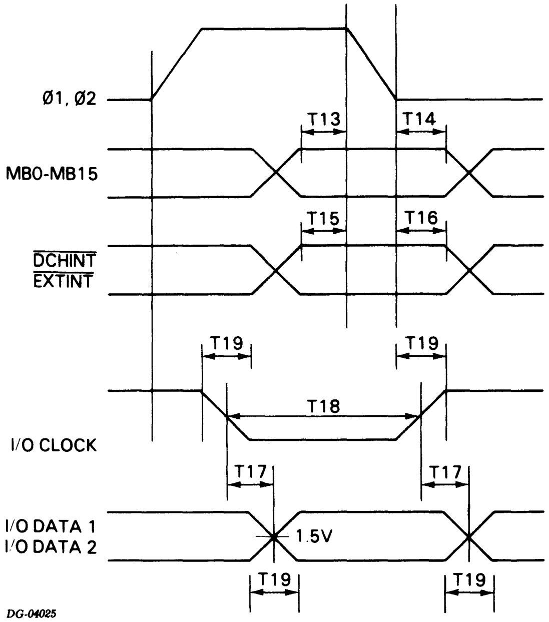
DG-04023

OUTPUT MODE



DG-04024

INPUT MODE



DG-04025

NOTE: Time intervals are all measured between 10% and/or 90% points, unless otherwise specified.

TRANSITION TIMING

FUNCTION	SYMBOL	PIN	MIN.	MAX.	UNITS
MASTER CLOCK FREQUENCY		ACCURACY .1%		8.333	MHz
CLOCKS	T1	ϕ1,ϕ2 SEPARATION	5	25	ns
	T2	ϕ1,ϕ2 WIDTH	75	95	ns
	T3	ϕ1,ϕ2 CYCLE	235	245	ns
OUTPUTS	T4	MB 0 - MB 15		80	ns
	T5	I/O DATA 1, I/O DATA 2, I/O CLOCK		25	ns
	T6	I/O INPUT		25	ns
	T7	WE		25	ns
	T8	P FALL		25	ns
	T9	P RISE		25	ns
	T10	SAE		40	ns
	T11	$\overline{\text{PAUSE}}$		25	ns
	T12	HALT		40	ns
INPUTS	T13	MB 0 - MB 15 SET-UP TIME	40		ns
	T14	MB 0 - MB 15 HOLD TIME	20		ns
	T15	$\overline{\text{DCHINT}}$, $\overline{\text{EXTINT}}$ SET-UP TIME	100		ns
	T16	$\overline{\text{DCHINT}}$, $\overline{\text{EXTINT}}$ HOLD TIME	20		ns
	T17	I/O<CLOCK/DATA 1/DATA 2> SKEW	-5	+15	ns
	T18	I/O<CLOCK/DATA 1/DATA 2> WIDTH	115	125	ns
	T19	I/O<CLOCK/DATA 1/DATA 2> RISE/FALL		10	ns

TIMING TABLES

MEMORY READ/WRITE TIMING

OPERATION	SYMBOL	DESCRIPTION	MIN.	MAX.	UNITS
COMMON TO ALL MEM CYCLES	$\overline{PT}$	P LOW DURATION	0.230	5.570	µs
	PT	P HIGH DURATION	0.710	3.180	µs
	AS	ADDRESS SETUP TIME	0.040	0.130	µs
	AH	ADDRESS HOLD TIME	0.110	0.190	µs
	PC	PRECHARGE		0.120	µs
READ	SAED	P TO SAE DELAY	0.110	1.150	µs
	DD	SAE TO DATA DELAY	0.060	0.130	µs
	SAET	SAE DURATION	0.200	0.260	µs
	DT	DATA DURATION	0.080	0.360	µs
WRITE/ REFRESH	WED	P TO WE DELAY	0.230	0.290	µs
	WET	WE TIME DURATION	0.220	0.260	µs
	WDS *	WRITE DATA SETUP TIME	0.060	0.140	µs
	WDH *	WRITE DATA HOLD TIME	0.100	0.160	µs
READ MODIFY WRITE	RWED	READ MODIFY WRITE WE DELAY	1.190	1.730	µs

DG-04418

* Applies to write operation only.

NOTE: Rise and fall times for input signals are assumed ≤10ns.

I/O BUS TIMING

OPERATION	SYMBOL	DESCRIPTION	MIN.	MAX.	UNITS
COMMON TO ALL OUTPUT OPERATIONS	TD$_1$	I/O INPUT TO I/O CLOCK TIME (START OF TRANSACTION)	0.090	0.150	µs
	TD$_2$	I/O CLOCK 1/2 PERIOD (BIT TIME)	0.110	0.130	µs
	TD$_3$	I/O CLOCK TO I/O INPUT TIME (END OF TRANSACTION)	0.090	0.150	µs
	TD$_4$	I/O CLOCK TO I/O DATA SKEW	-5	+15	ns
I/O COMMAND OR DATA OUTPUT	T $\overline{INPUT}_D$	COMMAND OR DATA TRANSFER TIME	1.310	1.330	µs
I/O REQUEST ENABLE OR DATA CHANNEL ADDRESS REQUEST	T $\overline{INPUT}_R$	REQUEST ENABLE OR DATA CHANNEL ADDRESS REQUEST TRANSFER TIME	0.350	0.370	µs
DATA TRANSFER TIMING					
PROGRAMMED I/O DATA OUT	PI/0 OUT$_{DD}$	PROGRAMMED I/O DATA OUT - DATA DELAY	0.83	0.85	µs
PROGRAMMED I/O DATA IN	PI/O IN$_{TW}$	PROGRAMMED I/O DATA IN - DATA RECEPTION WINDOW	--	3.12	µs
	PI/O IN$_{DD}$	PROGRAMMED I/O DATA IN - DATA DELAY	0.11	0.13	µs
COMMON TO ALL DATA CHANNEL OPERATIONS	DCH$_{AR}$	DATA CHANNEL ADDRESS REQUEST	0.350	0.370	µs
	DCH$_D$	DATA CHANNEL DELAY	0.11	0.13	µs
	DCH$_{TWA}$	DATA CHANNEL ADDRESS AND MODE BIT RECEPTION WINDOW	--	2.6	µs
DATA CHANNEL OUT - MEMORY TO DEVICE	DCHO	CYCLE TIME FOR 16-BIT WORD TRANSFER	--	5.8	µs
	DCHO$_{DD}$	DATA CHANNEL DATA OUT DELAY	3.11	3.13	µs
DATA CHANNEL IN - DEVICE TO MEMORY	DCHI	CYCLE TIME FOR 16-BIT WORD TRANSFER	--	7.2	µs
	DCHI$_{DD}$	DATA CHANNEL DATA IN DELAY	3.54	3.56	µs
	DCHI$_{TWC}$	DATA CHANNEL WINDOW CLOSED	--	0.96	µs
	DCHI$_{TWD}$	DATA CHANNEL DATA RECEPTION WINDOW	--	1.9	µs

HALT TIMING

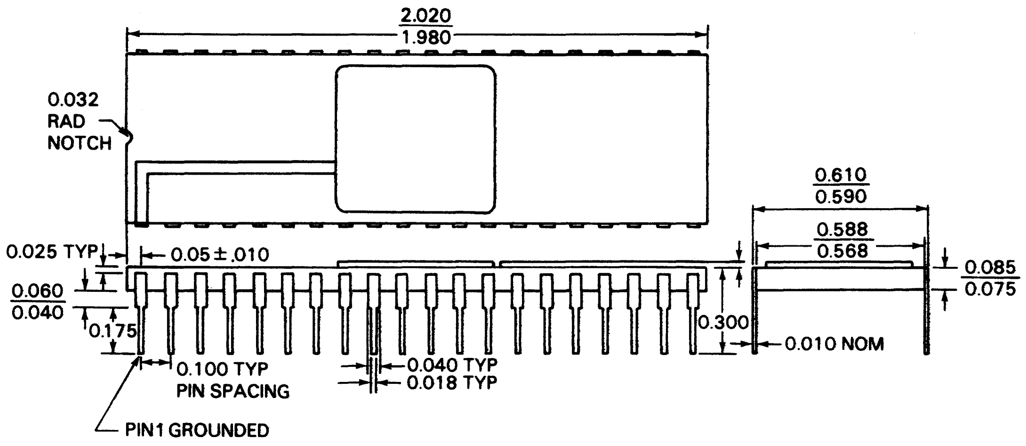
OPERATION	SYMBOL	DESCRIPTION	TYP.	UNITS
HALT	H_H	DENOTES HALT SIGNAL IN HIGH STATE.	0.24	µs
	H_I	DENOTES HALT SIGNAL IN LOW STATE.	2.16	µs

DG-04419

PACKAGE SPECIFICATIONS

The physical dimensions (in inches) of the mN601 chip are given in the diagram below.

DIMENSIONS OF CHIP



DG-04001

2

mN602
16-BIT MICROPROCESSOR

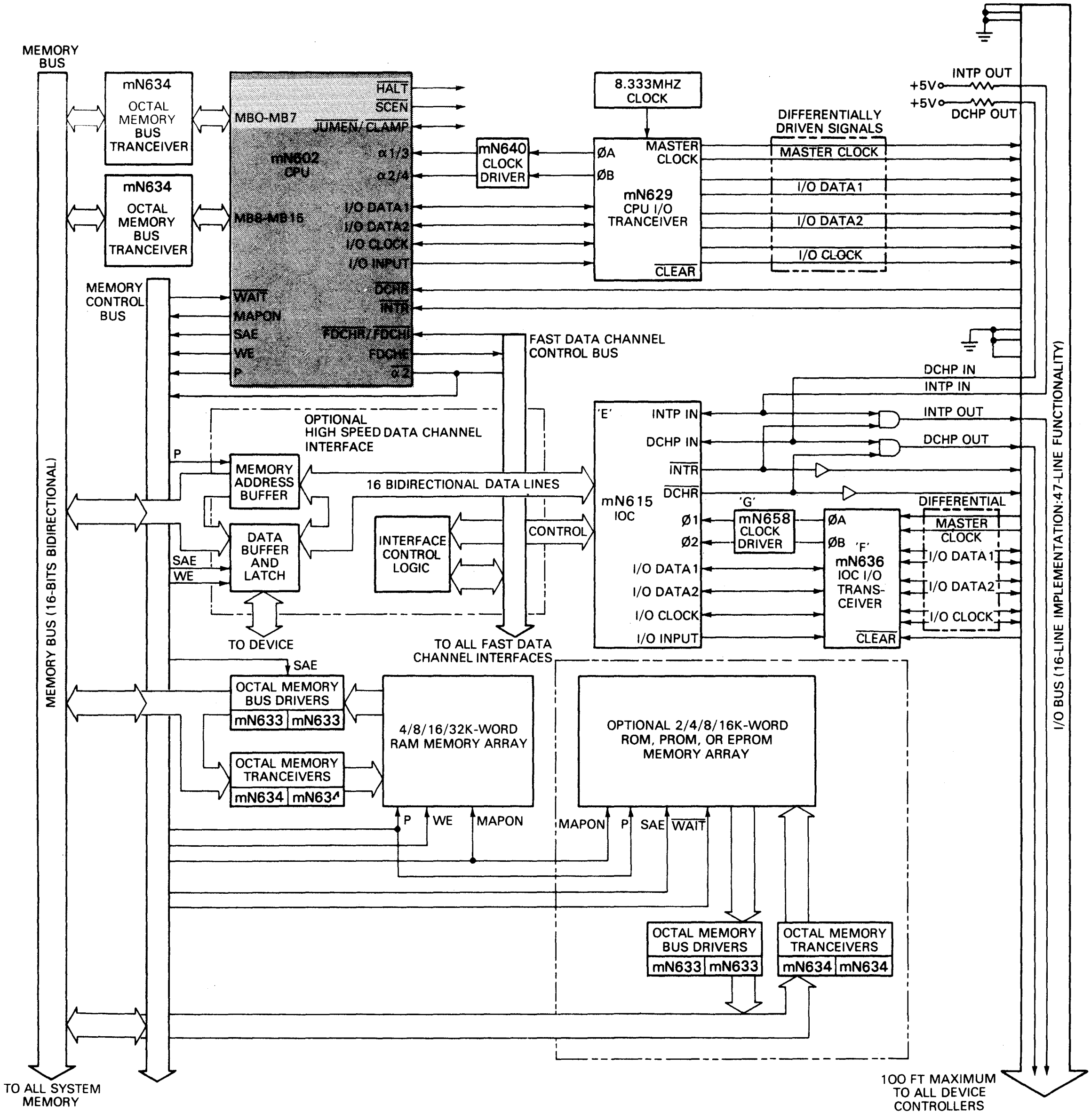
3

Data General
Data General Corporation, Westboro, Massachusetts 01581

CONTENTS

3

mN602
16-BIT MICROPROCESSOR



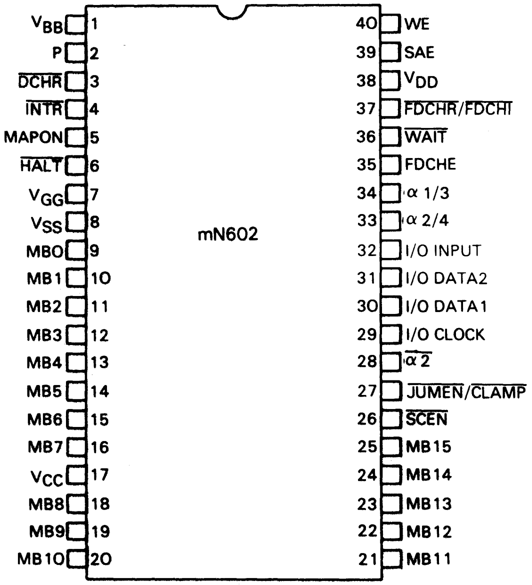
DG-06030

mN602
16-BIT MICROPROCESSOR

FEATURES

- FULL STANDARD mNOVA ARCHITECTURE AND INSTRUCTION SET ON A SINGLE 40-PIN NMOS SILICON-GATE CHIP
- INTEGRAL DATA CHANNEL, REAL-TIME CLOCK, AND MULTIPLY/DIVIDE
- INPUT FOR EXTERNAL REAL-TIME CLOCK
- ABILITY TO RUN SLOW MEMORY
- INTEGRAL TRANSPARENT REFRESH LOGIC FOR 16K-BIT AND 4K-BIT DYNAMIC RAMs
- 64K-WORD PHYSICAL MEMORY MAPPED INTO TWO 32K-WORD LOGICAL ADDRESS SPACES
- FULL ADDRESSING MODES TO 16-BIT MEMORY WORDS
- BUILT-IN POWER-UP, AUTO RESTART, POWER-FAIL, AND TURNKEY CONSOLE FUNCTIONS
- POWERED FROM STANDARD TTL AND MOS VOLTAGE SUPPLIES ±5V, +12V

PACKAGE



DG-06013

GENERAL DESCRIPTION

The mN602 is a 16-bit NMOS microprocessor that provides a central processing function for Data General's microNOVA family.

mN602 architecture features mapped memory, slow memory access capability, and a high speed data channel in addition to hardware multiply/divide, four accumulators, stack registers, and multiple addressing modes. Available addressing modes are absolute, relative, and indexed. Indirect addressing and auto increment/decrement are also provided.

mN602 refresh logic can handle 16K dynamic RAMs as well as 4K dynamic RAMs. Any portion of memory can be made up of slow memories. A slow memory is any memory that has a slower access time than RAM or ROM. EPROM is an example of slow memory.

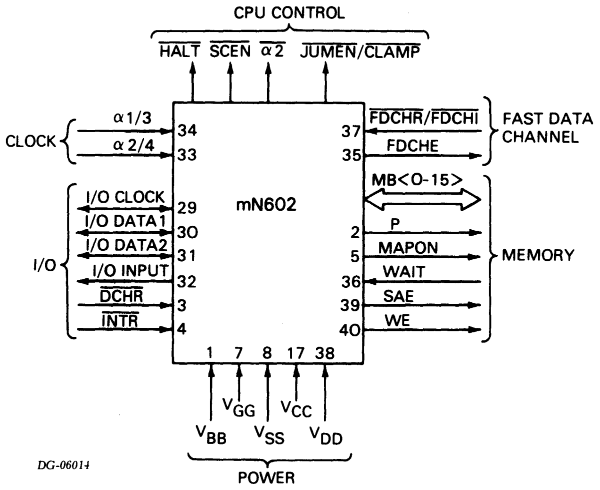
The map feature of the mN602 provides the user with two address spaces of 32K words each. The unmapped (lower) 32K word space is accessible with any memory reference instruction. The mapped (upper) space is accessible under certain well-defined conditions.

The mN602 uses the mN603, the mN613, or the mN615 I/O controller (IOC). The IOC is the I/O interface between the CPU and an external device. It provides access to the 47-line I/O bus of the NOVA line.

PIN DESCRIPTIONS

The diagram below shows the pin connections and the table describes the function of each pin shown on the diagram.

FUNCTIONAL PIN CONNECTION DIAGRAM



PIN FUNCTIONS

MNEMONIC	PIN NO.	I/O	FUNCTION
			CLOCK
α1/3 α2/4	34 33	I I	Two-phase system clock operating between 0 and 12v. Generates the 4-phase internal clock for internal control timing.
			MEMORY BUS
MB0-MB15	9-16 18-25	I/O	**Bidirectional data, status and address bus. When the bus carries data, MB0 is the Most Significant Bit (MSB) and MB 15 the Least Significant Bit (LSB). When the bus carries an address, MB<1-15> contains the address, and MB0 defines the memory operation as either refresh or read/write. When the bus carries status, it sends the CPU a system status word during α3 of the memory refresh cycle.** The CPU precharges the memory bus on the **α2** phase of the clock. Pull-up resistors on each line hold the bus high. On output, the CPU discharges lines transmitting 0, while lines transmitting 1 are not affected.

PIN FUNCTIONS (CONT.)

MNEMONIC	PIN NO.	I/O	FUNCTION
			MEMORY CONTROL
P	2	O	Active high. Initiates all memory cycles. On positive going edge of P, the memory bus has a valid address.
SAE	39	O	Active high. Signals read operation.
WE	40	O	Active high. Enables write operation. On falling edge of WE, the memory bus writes into memory.
$\overline{\text{WAIT}}$	36	I	Active low. Driven low by slow memories while busy. Creates additional WE or SAE signals.
$\overline{\alpha 2}$	28	O	Active low. Slow memory interfaces can use $\overline{\alpha 2}$ as a clock. Signal used by fast data channel as a Request Enable.
MAPON	5	O	Active high. Determines which 32K block of memory the CPU accesses. CPU references mapped memory when MAPON is high. MAPON is driven high on system power-up, and valid only on the rising edge of P.
			I/O BUS
I/O DATA1 I/O DATA2	31 30	I/O I/O	Two-line, bidirectional bus. All data and I/O command information is transmitted serially between the mN602 and an I/O device through an IOC at the system clock rate. During transfers, I/O DATA1 carries the most significant bits (0-7), and I/O DATA2 carries the least significant bits (8-15).
I/O INPUT	32	I/O	Active high indicates a transfer into the mN602. When low, indicates a transfer from the mN602.
I/O CLOCK	29	I/O	Synchronizes all I/O transfers to and from the CPU. Holding the I/O CLOCK low for 10 usec resets the processor, but the I/O CLOCK should not be pulled low when I/O INPUT is low.
$\overline{\text{INTR}}$	4	I	Active low. When driven low by a device, while CPU interrupts are enabled, the device interrupts the CPU.
$\overline{\text{DCHR}}$	3	I	Active low. When driven low by a device, the CPU executes a data channel break. The mN602 responds to a data channel break request even while in the HALT state.

PIN FUNCTIONS (CONT.)

MNEMONIC	PIN NO.	I/O	FUNCTION
			HIGH SPEED DATA CHANNEL CONTROL
$\overline{\text{FDCHR}}$/ $\overline{\text{FDCHI}}$	37	I	Active low. $\overline{\text{FDCHR}}$ is driven low by a device requesting a direct memory access (DMA). Input after CPU sends FDCHE to device gives the direction of transfer. $\overline{\text{FDCHI}}$ low is a memory write, high a memory read to the device using the Memory Bus.
FDCHE	35	O	Active high. Sent to device to enable fast data channel transfer. Transfer starts in direction $\overline{\text{FDCHI}}$.
			CPU CONTROL
$\overline{\text{SCEN}}$	26	O	Active low. Strobes in the system status word on the memory bus during a memory refresh cycle.
$\overline{\text{JUMEN}}$/ $\overline{\text{CLAMP}}$	27	O/I	Bidirectional active low for memory reference and power up. $\overline{\text{JUMEN}}$ is an output strobe for reading the Jumper Word Register during an $\alpha 4$ phase. $\overline{\text{CLAMP}}$ is an input held low during power up for a minimum of 100μsec.
$\overline{\text{HALT}}$	6	O	Active low level. Functions when CPU is halted (idling), as the result of hardware conditions or execution of a HALT instruction.
			POWER
V_{BB}	1		-5v
V_{CC}	17		+5v
V_{DD}	38		+12v
V_{GG}	7		+12v
V_{SS}	8		Ground

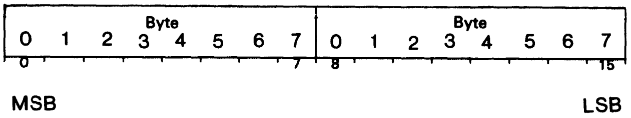
ARCHITECTURE

INTRODUCTION

The mN602 is a 16-bit parallel processor. It can execute the full NOVA instruction set and an extended instruction set for stack and trap routines. It also has control lines that allow memory mapping and that support an asynchronous memory system (one that may contain slow memories, such as EPROMs).

The instruction set can address 64K words of physical memory. This memory is divided into two spaces, each of 32K words. An address is 15 bits long. The word stored at that address is 16 bits long, consisting of two 8-bit bytes.

Word

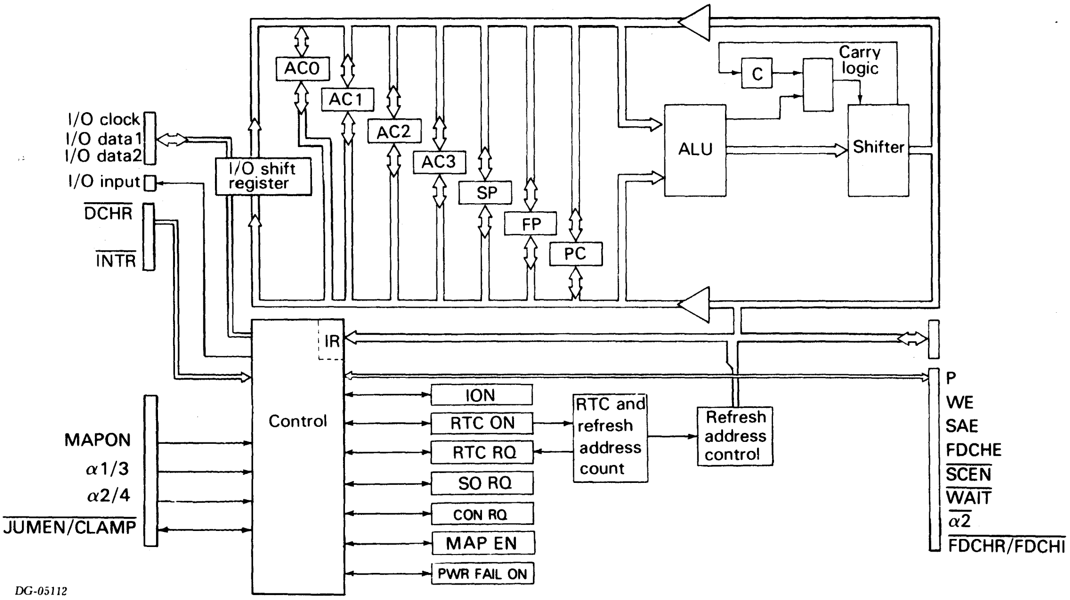


The mN602 communicates with memory via a bidirectional, 16-bit parallel address and data bus (MB<0-15>). Three control lines direct the memory operations: read, write, and refresh. During refresh another control line strobes the system status word into the instruction register from MB<0-15>. The System Status Word carries status and control information to the CPU. Certain bits of the System Status Word can cause interrupts.

Other control lines idle the CPU during direct memory access by an I/O device, clock data during a slow memory transfer, and determine which of the two memory spaces to access.

The block diagram shows the basic internal architecture of the mN602. It illustrates the memory bus, the I/O bus, internal registers, and control lines.

CPU INTERNAL BLOCK DIAGRAM



CPU INTERNAL REGISTERS

REGISTER	SIZE (BITS)	FUNCTION
Accumulator 0 (AC0)* Accumulator 1 (AC1)* Accumulator 2 (AC2)* Accumulator 3 (AC3)*	16 16 16 16	Provide working space during program execution.
Stack Pointer (SP)*	15	Contains address of top of stack. Increments as words are pushed on stack; decrements as words are popped off.
Frame Pointer (FP)*	15	Contains address of top of current return block in stack. On RETurn instruction, stack is popped from this point.
Program Counter (PC)*	15	Contains memory address in current 32K-word address space. Used to fetch next instruction. Is either modified by the instruction or incremented when instruction is completed.
Carry bit (C)*	1	Contains result of addition overflow, subtraction underflow, and initialization.
Interrupt Enable (ION)*	1	Enabled by program. When set to 1, the processor responds to interrupt requests; cleared by interrupts.
Real-time Clock Enable (RTC ON)*	1	This RTC can be the internal real-time clock or the external real-time clock. The internal real-time clock generates an interrupt every 1.9651 ms (every 4094 CPU cycles). The external real-time clock can cause an interrupt at any frequency. (See the description of the System Status Word for information on the external RTC.)
Real-time Clock Request (RTC RQ)	1	Generates a program interrupt every 1.9651 ms. when RTC ON and ION are set.
Stack Overflow Request (SO RQ)	1	Generates an interrupt when the stack pointer crosses a 256-word boundary (e.g., 377_8 to 400_8).
Refresh Address Counter	7	Generates a 7-bit address used to refresh 1/128 of memory every 16 μsec. (All 65,536 RAM locations are refreshed at least once every 2.0 msec.)
I/O Shift Register	16	Performs serial/parallel conversions between the I/O data lines and the rest of the CPU.
Instruction Register (IR)	16	Loaded from the memory bus with the contents of the address in the program counter. A new instruction is loaded on each fetch.

* User accessible

REGISTER FILE

The register file contains four 16-bit accumulators available to users for general programming. Three other addressing registers of 15 bits each control program flow and stack addressing. The PC (program counter) contains the address of the next instruction to be executed. The SP (stack pointer) points to the top of the current LIFO (last-in-first-out) stack in memory. The FP (frame pointer) usually points to the top of the last return block pushed on the stack. A return block is a group of five memory locations that are used to store program reentry information. The memory referred to by the three address registers is confined to the current 32K-word working space.

ALU SHIFT LOGIC

The register file is dual ported to the ALU (arithmetic logic unit). The ALU performs all the arithmetic functions (add, subtract, increment etc.) for the CPU. The shift logic, acting as one unit with the ALU, can manipulate the result of an operation before it is transferred to its destination. The carry bit is an integral part of the shifter. The carry is set by addition overflow or subtraction underflow. It may be initialized by the instruction before the ALU has a result. All transfers from a register to memory must pass through the ALU/shifter path to get to the memory bus. The shifter rotates the result and carry one bit right or left, or swaps the upper and lower bytes of the result without affecting the carry. The carry and result can both be tested to skip the next memory location (see ALU instructions).

INTERNAL FLAGS AND REFRESH CONTROL

The CPU contains four single-bit registers (flags) and one 7-bit register (counter). One flag enables and disables interrupts, and another controls the internal real-time clock. The two request flags, Real Time Clock and Stack Overflow, interrupt the CPU. The 7-bit Refresh Address Counter can address 16K-bit dynamic memory chips. (A memory refresh occurs at least once every 16 usec.). Only the flags controlling interrupts and the internal real-time clock are accessible to the user.

STATE CONTROL LOGIC

The state control logic, includes the basic mNOVA logic, plus three extended features: the high speed data channel controller, the slow memory enable logic, and the memory map. The high speed data channel logic can suspend the CPU's use of memory to allow an I/O device to do a direct memory access (DMA) within time frames bounded by the refresh logic. The slow memory logic will cause the CPU to wait for data, using $\overline{\alpha 2}$ as a clock. The memory map defines two 32K-word memory blocks addressable by the user.

One bus connects the controller, the ALU and shifter, the register file, and the refresh logic to the external memory bus. (The controller distinguishes between addresses, data, and instructions.) This bus is also used during refresh periods to strobe system status information to the controller. The controller is driven by the present instruction held in the instruction register (IR), the current state of the entire processor, and the state of the system it is part of.

INTRODUCTION TO INSTRUCTION SET

A microNOVA instruction is contained in one 16-bit word and has the same format as a standard NOVA instruction. Programs consist of sequences of instruction stored in external memory. The 15-bit program counter (PC) always contains the address of the instruction currently being executed. The PC is incremented by one after each instruction or modified by the current instruction. The PC can address the complete logical address space.

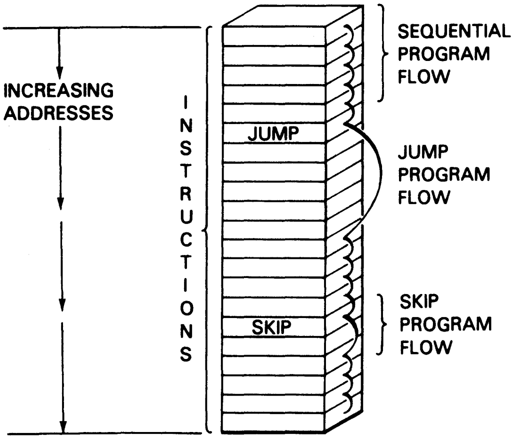
For the mN602, a logical address space refers to one of two 32K-word memory spaces. These two logical address spaces comprise the physical address space of 64K words. One is the unmapped (lower) logical address space that the user normally runs programs in; the other is the mapped (upper) logical address space. Within a logical address space, location 0 is the next sequential memory location after 77777_8.

The signal $\overline{\text{MAPON}}$ determines which logical address the CPU will access. It confines the CPU to one address space at a time. When MAPON is 1, the CPU accesses the mapped (upper) address space. When MAPON is 0, the CPU accesses the unmapped (lower) address space. The 15 logical address translates to a 16-bit physical address. MAPON is valid only during the rising edge of P.

Program Flow Alteration

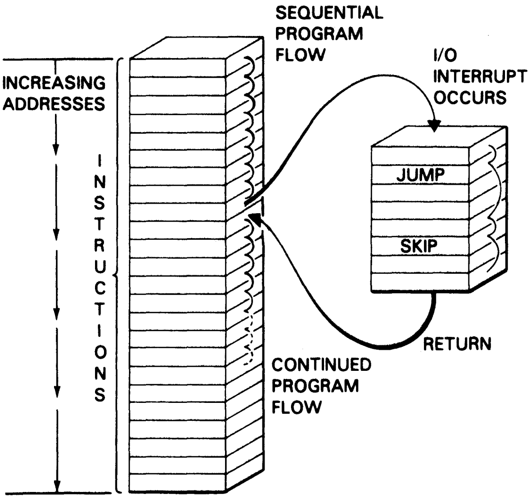
The sequence of program execution can be altered in two ways: by inserting a new value into the PC with a JUMP or SKIP instruction, or by servicing a program interrupt. When a program interrupt is serviced, the present state of the processor is saved and the starting address of the interrupt handler routine is placed in the PC. At the conclusion of this routine, the previous state of the CPU is restored and execution continues with the next instruction after the point of interruption.

NON-INTERRUPTED PROGRAM FLOW



DG-00543

INTERRUPTED PROGRAM FLOW



DG-00544

INSTRUCTION EXECUTION TIMES
(ONE CPU CYCLE=0.480 USEC)

INSTRUCTION GROUP TYPE	CPU CYCLES		
MEMORY REFERENCE GROUP	**DIRECT**	**INDIRECT ***	**AUTO INCREMENT/ DECREMENT**
DSZ	9 **	11 **	14 **
ISZ	9**	11 **	14 **
JMP	6	7	10
JSR	7	9	12
LDA	6	8	11
STA	6	8	11
ARITHMETIC AND LOGICAL GROUP			
ADC	5**		
ADD	5**		
AND	5**		
COM	5**		
INC	5**		
MOV	5**		
NEG	5**		
SUB	5**		
MULTIPLY/DIVIDE GROUP			
DIV	122		
MUL	87		
INPUT/OUTPUT GROUP			
DIA	15		
DIB	15		
DIC	15		
DOA	11		
DOB	11		
DOC	11		
NIO	11		
SKP	15**		
STACK CONTROL GROUP			
MFFP	6		
MFSP	6		
MTFP	6		
MTSP	6		
POPA	7		
PSHA	7		
RET	15		
SAV	16		
CPU CONTROL GROUP			
HALT	7		
INTA	15		
INTDS	11		
INTEN	11		
IORST	11		
MSKO	11		
RTCEN	11		
RTCDS	11		
TRAP	10*		
NIOP 0, CPU **(BRKPT)**	4		
NIOP 1, CPU **(MEP)**	4		
NIOP 2, CPU **(INTRE)**	4		
SKPBZ 1 **(SKPM)**	15**		

* FOR EACH ADDITIONAL INDIRECT LEVEL, ADD 2 CYCLES;
FOR EACH AUTO-INCREMENT/DECREMENT, ADD 5 CYCLES
** IF SKIP OCCURS, ADD 2 CYCLES

MEMORY REFERENCE INSTRUCTIONS

This group of instructions reads and writes to/from memory and loads the program counter with a new address.

INSTRUCTION FORMAT

OP CODE	*	@ INDEX	DISPLACEMENT
0 1 2	3 4	5 6 7	8 9 10 11 12 13 14 15

* AC or OP CODE

MEMORY REFERENCE INSTRUCTIONS
JMP
JSR
ISZ
DSZ
LDA
STA

TERMS USED IN MEMORY ADDRESSING

ADDRESSING MODES refer to the four reference pointers used with a DISPLACEMENT to access the desired address. Different modes use different pointers.

INDIRECT ADDRESSING uses the first address as a pointer to another address. If bit 0 of the succeeding address is 1, the address is taken as a pointer to a third address. If bit 0 of the third address is 1, yet another address is pointed to, etc. Up to 16 levels of indirection are possible. A series of indirect addresses is called an INDIRECTION CHAIN.

The INDIRECT BIT (bit 5 of the instruction and bit 1 of succeeding addresses in an indirection chain) controls each step of the addressing process.

The INDEX BITS (bits 6 and 7 of the instruction) determine which of the four reference pointers will reference memory. When the index bits equal 00, the displacement is an unsigned integer. For the values 01, 10, and 11, the displacement is a signed integer.

ADDRESSING MODE INDEX BITS

INDEX BITS (BITS 6-7)	MODE	RANGE OF DISPLACEMENT	MEMORY ADDRESSABLE
00	Absolute	0 to 377 (0 to 256)	0 to 377
01	PC Relative	-200 to +177 (-128 to +127)	0 to 077777
10	AC2 Relative		
11	AC3 Relative		

The DISPLACEMENT (bits 8-15 of the instruction) gives the distance, in number of memory locations, between the reference point (determined by the addressing mode) and the desired address.

EFFECTIVE ADDRESS CALCULATIONS use the index, indirect and displacement bits, along with the MAPON flag, to convert a logical address in the program into a physical address that can be referenced.

INTERMEDIATE ADDRESSES are effective addresses before they have been tested for indirection.

PAGE ZERO refers to the first 400_8 locations ($0\text{-}377_8$) of each 32K-word address space.

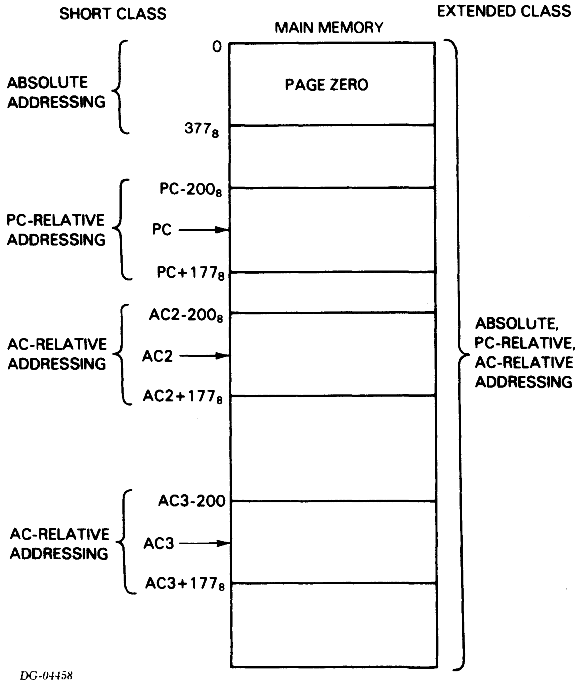
ADDRESSING MODES

Word addressing in the microNOVA can be done in the following modes:

- Absolute addressing (displacement from location 0).
- PC relative addressing (displacement from program counter).
- AC2 relative addressing (displacement from accumulator 2).
- AC3 relative addressing (displacement from accumulator 3).

Direct or indirect addressing can be used in any mode. By addressing indirectly from page zero (absolute addressing), you can access any word in your logical address space. You can also do this by addressing directly from a relative address.

ADDRESSING RANGES



DG-04458

This discussion of addressing modes assumes that MAPON is 0 and that no MAPON change is pending. For information about the memory map, see the section, "Mapped Memory References."

ABSOLUTE ADDRESSING

In absolute addressing mode, the intermediate address is the unmodified displacement. As a result, the instruction specifies locations in the range 0-377 $_8$. The displacement is restricted to one byte.

Lower page zero is important because any memory reference instruction can address this area. With this feature, common data or addresses can link separate sections of the user space.

RELATIVE ADDRESSING

In PC relative mode, the intermediate address is the sum of the program counter (the address of the present instruction) and the signed displacement of the current instruction. Since addresses are 15 bits long, overflows are ignored.

In accumulator relative mode, the intermediate address is the sum of the accumulator specified by the index bits (AC2 or AC3) and the signed displacement. Only the last 15 bits of the accumulator are used. Bit 0 is ignored.

DIRECT AND INDIRECT ADDRESSING

If no indirection is intended, bit 5 equals 0; then the effective address is determined by the intermediate address and the state of the MAPON bit .

If the indirect bit (bit 5) contains a 1, the calculated address is an indirect address. The CPU fetches the contents of the addressed location and tests bit 0.

When a memory location contains a stored address, bit 0 is the indirect bit. If it is 0, then bits 1 to 15 are the effective address. Control is transferred to the location addressed by bits 1 to 15. If the indirect bit is 1, then bits 1-15 denote another indirect address. In this case the chain continues until bit 0 of the fetched word equals 0 (in which case the effective address is found) or 16 levels of the chain have been executed.

An internal counter is used to keep track of the number of levels addressed. When the CPU decodes the indirect bit in the instruction, it initializes the counter to 0. When the CPU tests bit 0 of the fetched word, it increments the counter by 1 if bit 0 is a 1. If the accessed location is an auto-increment or auto-decrement location, the counter is incremented again. If the counter is greater than 17_{10} before reaching an effective address, the CPU halts.

AUTO-INCREMENTING AND AUTO-DECREMENTING

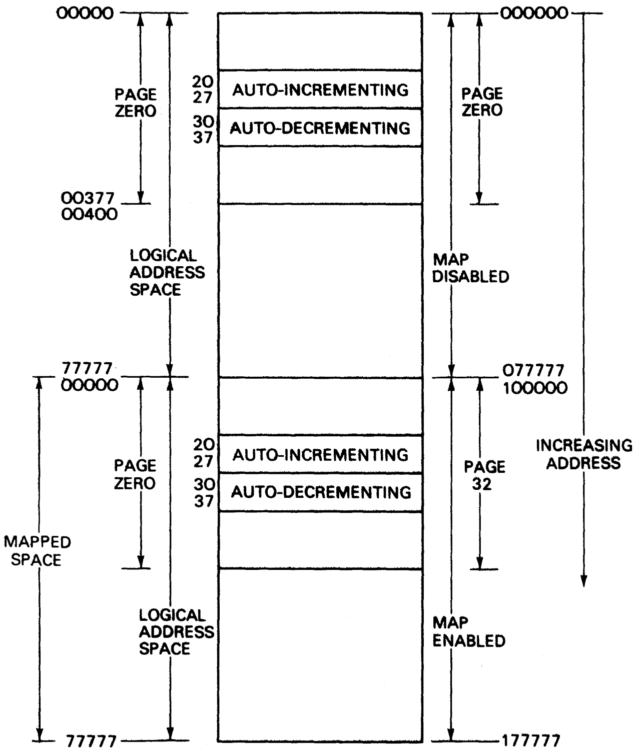
If the intermediate address is in the range of 20 - 27_8, in page zero, and the indirect bit is 1, then the contents of the addressed location are incremented by 1. The indirection chain continues with the new value of that location. When these locations are addressed directly, the incrementing feature is not invoked.

If the indirect address is in the range 30 - 37_8, and the indirect bit is 1, then the contents of the location addressed are decremented by 1. The indirection chain continues with the new value of that location. When these locations are addressed directly, the decrementing feature is not invoked.

NOTE: The state of bit 0 before the increment or decrement determines if the indirection chain continues, whereas the state of bit 0 after the increment or decrement determines the address.

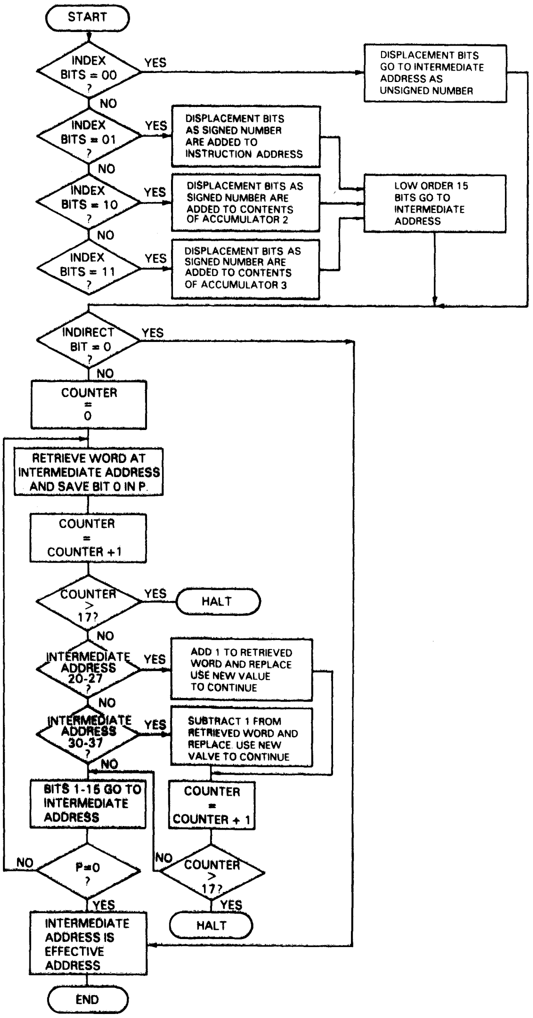
When non-existent memory locations are accessed by a memory reference instruction, the contents appear as all ones. This feature is not recommended for use in new designs.

PHYSICAL ADDRESS SPACE MAP



DG-06015

EFFECTIVE ADDRESS CALCULATION


START
INDEX BITS = 00 ?
YES
NO
DISPLACEMENT BITS GO TO INTERMEDIATE ADDRESS AS UNSIGNED NUMBER
INDEX BITS = 01 ?
YES
NO
DISPLACEMENT BITS AS SIGNED NUMBER ARE ADDED TO INSTRUCTION ADDRESS
INDEX BITS = 10 ?
YES
NO
DISPLACEMENT BITS AS SIGNED NUMBER ARE ADDED TO CONTENTS OF ACCUMULATOR 2
INDEX BITS = 11 ?
YES
DISPLACEMENT BITS AS SIGNED NUMBER ARE ADDED TO CONTENTS OF ACCUMULATOR 3
LOW ORDER 15 BITS GO TO INTERMEDIATE ADDRESS
INDIRECT BIT = 0 ?
YES
NO
COUNTER = 0
RETRIEVE WORD AT INTERMEDIATE ADDRESS AND SAVE BIT 0 IN P.
COUNTER = COUNTER +1
COUNTER > 17?
YES
HALT
NO
INTERMEDIATE ADDRESS 20-27 ?
YES
ADD 1 TO RETRIEVED WORD AND REPLACE USE NEW VALUE TO CONTINUE
NO
INTERMEDIATE ADDRESS 30-37 ?
YES
SUBTRACT 1 FROM RETRIEVED WORD AND REPLACE. USE NEW VALVE TO CONTINUE
NO
BITS 1-15 GO TO INTERMEDIATE ADDRESS
COUNTER = COUNTER + 1
NO
P=0 ?
YES
NO
COUNTER > 17?
YES
HALT
INTERMEDIATE ADDRESS IS EFFECTIVE ADDRESS
END

DG-02403

EFFECTIVE ADDRESS INSTRUCTIONS (NO ACCUMULATOR)

The general format and assembler version of this class of instruction are given below. The table shows the specific format and function of each instruction.

INSTRUCTION FORMAT

0	0	0	OP CODE		@	INDEX		DISPLACEMENT							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

MNEMONIC [@] displacement,[index]

INSTRUCTION FORMATS AND FUNCTIONS

MNEMONIC	MEANING	FUNCTION *	INSTRUCTION FORMAT
JMP	Jump	Compute (E) and load E into PC.	0 0 0 0 0 | @ | INDEX | DISPLACEMENT 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
JSR	Jump To Subroutine	Compute E. Store contents of PC+1 in bits 1-15 of AC3 and set bit 0 to 0; then, load E into PC.	0 0 0 0 1 | @ | INDEX | DISPLACEMENT 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
ISZ	Increment And Skip If Zero	Compute E. Increment contents of location E and write result back into location E. If result equals 000000, increment PC by two; otherwise increment PC by one.	0 0 0 1 0 | @ | INDEX | DISPLACEMENT 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
DSZ	Decrement And Skip If Zero	Compute E. Decrement contents of location E and write result back into location E. If result equals 000000, increment PC by two; otherwise increment PC by one.	0 0 0 1 1 | @ | INDEX | DISPLACEMENT 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

*E = effective address; AC = accumulator

EFFECTIVE ADDRESS INSTRUCTIONS (ONE ACCUMULATOR)

The general format, assembler version, and accumulator codes of this class of instruction are given below. The table shows the specific format and function of each instruction.

INSTRUCTION FORMAT

0	OP CODE		AC		@	INDEX		DISPLACEMENT							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

MNEMONIC ac,[@] displacement [,index]

ACCUMULATOR CODE

AC	Bits 3	4
AC0	0	0
AC1	0	1
AC2	1	0
AC3	1	1

INSTRUCTION FORMATS AND FUNCTIONS

MNEMONIC	MEANING	FUNCTION *	INSTRUCTION FORMAT
LDA	Load Accumulator	Compute E. Load specified AC with contents of location E.	0 0 1 AC @ INDEX DISPLACEMENT 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
STA	Store Accumulator	Compute E. Store contents of specified AC in location E.	0 1 0 AC @ INDEX DISPLACEMENT 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

* E = effective address; AC = accumulator

MAPPED MEMORY REFERENCES

The 64K words of physical memory supported by the mN602 consist of two 32K-word logical address spaces, the unmapped (lower) and mapped (upper) address spaces. The 15 bit logical address translates to a 16 bit physical address.

The state of the MAPON signal determines what space the CPU accesses. If MAPON is 0, the CPU accesses the unmapped address space. If MAPON is 1, the CPU accesses the mapped address space. MAPON is valid only on the rising edge of the P memory control signal, that is, only during the address phase of the memory reference. (For information on P, see "Memory Operations.")

The MEP instruction controls the state of the MAPON signal. This instruction does not immediately change the state of the MAPON signal. When the MEP instruction is executed, the MAPON change becomes pending. When the MAPON change does occur, the MAPON signal changes state. If it was 1, it becomes 0. If it was 0, it becomes 1.

If you have executed an MEP instruction and a MAPON change is pending, that change occurs when the CPU executes any of the following instructions: LDA, STA, JSR @, JMP @. The MAPON change will remain pending during any number of instructions other than these four. Once the CPU executes one of those four instructions, a pending change occurs.

The LDA and STA instructions have a different effect on a pending MAPON change than the JSR @ and JMP @ instructions.

When the CPU executes an LDA or STA instruction and a MAPON change is pending, the MAPON change occurs. The signal MAPON is complemented and remains complemented only for the duration of one memory reference. After the memory reference is complete, MAPON returns to its previous value, and a MAPON change is no longer pending.

When the CPU executes a JSR @ or JMP @ instruction and a MAPON change is pending, the MAPON change occurs after one memory reference. The signal MAPON is complemented and remains complemented. A MAPON change is no longer pending. To return the MAPON signal to its previous value, you must issue another MEP instruction followed by one of the four memory reference instructions listed above. As before, that change occurs when the memory reference instruction is executed and is permanent only if the instruction was a JSR @ or JMP @.

NOTES:

If a MAPON change is pending, the memory reference instructions (ISZ, DSZ) and the stack instructions (PSHA, POPA, SAV, RET) are executed but do not cause a pending MAPON change to occur.

The use of a direct JSR or JMP instruction should not be attempted when a MAPON change is pendng.

If you are in one logical address space and want to transfer control to a routine stored in the other logical space, issue a MEP instruction followed by a JSR @ or JMP @. Other ways are by using the TRAP instruction and by using the BRKPT instruction. (See the descriptions of these instructions for further information.)

EFFECTS OF MEP INSTRUCTION

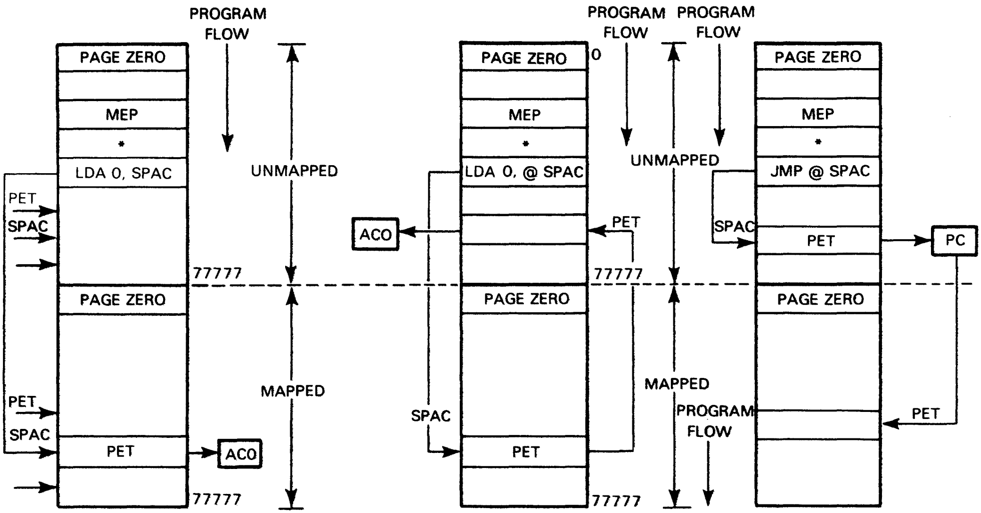


FIGURE A

FIGURE B

FIGURE C

*ANY NUMBER OF NON-MEMORY REFERENCE INSTRUCTIONS

NOTES: MEP INSTRUCTION IS EQUAL TO NIOP 1, CPU (064377_8). ONLY LDA, STA, JMP AND JSR INSTRUCTIONS CAN BE USED TO ACCESS ANOTHER 32K-WORD ADDRESS SPACE.

The figure shows the differences in program flow for the two types of memory reference instructions (LDA, STA and JSR @, JMP @) when a MAPON change is pending.

In Figure A, an LDA 0 SPAC instruction follows an MEP instruction. MAPON changes state only for the duration of one memory reference. The instruction says to put the contents of the location whose address is SPAC into ACO. The memory reference resolves to a location in mapped space, but the program continues to execute in unmapped space.

In Figure B, an LDA 0 @SPAC follows an MEP instruction. The instruction says to put the contents of the location whose address (called PET) is in the location called SPAC into ACO. MAPON is complemented only for the first memory reference. This means that the CPU uses the contents of location SPAC in mapped space as an address of a location in unmapped space. The memory reference resolves to a location in unmapped space, and the program continues to execute in unmapped space.

In Figure C, a JMP @SPAC instruction follows an MEP instruction. The instruction says to transfer control to the address represented by the contents of SPAC. The first memory reference is to SPAC. MAPON is not yet complemented. The CPU references SPAC in unmapped space. Then MAPON is complemented, and the contents of SPAC are used as an address (called PET), referencing a location in mapped space. The memory reference resolves to a location in mapped space, and the program continues to execute in mapped space. If the indirection chain continued, only the first indirect reference would be in unmapped space. Subsequent indirect levels would reference mapped space.

Some programming cautions to keep in mind when a MAPON change is pending are

- Do not reference auto-increment and auto-decrement locations.
- Do not issue a HALT instruction.
- Do not issue a JMP or JSR direct.
- Remember that all interrupts (normal and non-maskable) may be inhibited when a MAPON change is pending. See the section, "Program Interrupts," for details.

TRAP INSTRUCTION

FORMAT OF TRAP INSTRUCTION

<table>
<tr><td>1</td><td colspan="11">TRAP NUMBER</td><td>1</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td></tr>
</table>

The TRAP instruction is a single word call to a subroutine. If the $\overline{\text{MAP ON TRAP}}$ bit of the System Status Word is 0, the TRAP instruction sets MAPON to 1, and program execution transfers to mapped memory space. The TRAP instruction "traps to mapped space." If MAPON was already 1 (the CPU was already executing instructions in mapped space), MAPON stays 1. The instruction still traps to mapped space. If $\overline{\text{MAP EXISTS}}$ is low, all interrupts (normal and non-maskable) are inhibited after a mapped trap until the CPU executes an INTRE instruction.

3

If $\overline{\text{MAP ON TRAP}}$ is 1, the TRAP instruction does not change the state of MAPON, and program execution continues from the current memory space. The TRAP instruction "traps to the current map space." No interrupts are inhibited after the trap.

<table>
<tr><th>MNEMONIC</th><th>MEANING</th><th>FUNCTION</th><th>FORMAT</th></tr>
<tr><td>TRAP</td><td>Trap</td><td>The address of this instruction is placed in memory location 46_8 and bit 0 of that location is set to 0. Then the processor performs a jump indirect through memory location 47_8. The contents of ACS and ACD do not affect this instruction.</td><td><table><tr><td>1</td><td colspan="2">ACS</td><td colspan="2">ACD</td><td colspan="7">TRAP NUMBER</td><td colspan="4">1000</td></tr><tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td></td><td></td><td></td><td></td><td></td><td>11</td><td>12</td><td></td><td></td><td>15</td></tr></table></td></tr>
</table>

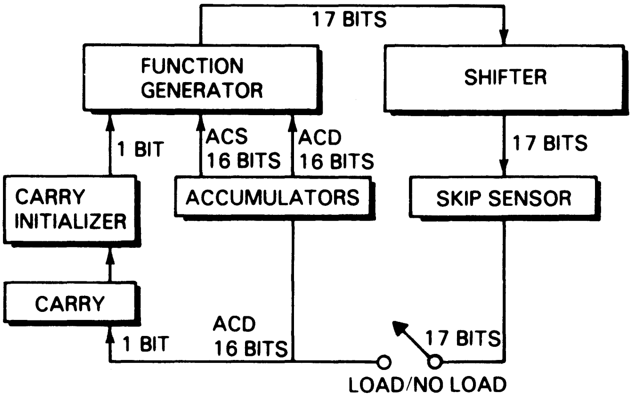
In order to use this instruction the user must write a trap handler routine and place its starting address in location 47_8.

Each TRAP instruction contains a trap number, which you can use to access a subroutine that belongs to the trap handler. You can define the trap number to mean anything you choose. (It is often used to refer to the condition that caused the trap or to an instruction to be emulated.) To read the trap number, the trap handler has to read the last instruction trapped. You can do that with an LDA ac,@46.

ARITHMETIC / LOGICAL INSTRUCTIONS

The instructions in this group perform arithmetic and logical operations, using the four accumulators and the carry bit; they can also alter program flow. Each instruction requires both a source accumulator (ACS) and a destination accumulator (ACD). These can be the same accumulator or two different ones. The carry bit may be either modified explicitly by the instruction, or implicitly by the result of the instruction. Program flow can be altered by testing the result of the instruction. (This test does not affect the result placed in ACD).

ALU INTERNAL STRUCTURE



DG-00927

TWO ACCUMULATOR-MULTIPLE OPERATION

INSTRUCTION FORMAT

1	ACS	ACD	OP CODE	SH	C	#	SKIP
0	1 2	3 4	5 6 7	8 9	10 11	12	13 14 15

MNEMONIC [c] [sh] [#] ACS,ACD [,skip]

ACCUMULATOR FIELDS

	ACS		ACD	
BITS	1	2	3	4
AC0	0	0	0	0
AC1	0	1	0	1
AC2	1	0	1	0
AC3	1	1	1	1

ALU FUNCTION CODES

MNEMONIC	FUNCTION
COM	$ACD \leftarrow \overline{ACS}$
NEG	$ACD \leftarrow \overline{ACS} + 1$
MOV	$ACD \leftarrow ACS$
INC	$ACD \leftarrow ACS + 1$
ADC	$ACD \leftarrow ACD + \overline{ACS}$
SUB	$ACD \leftarrow ACD + \overline{ACS} + 1$
ADD	$ACD \leftarrow ACD + ACS$
AND	$ACD \leftarrow ACD \wedge ACS$
MUL	Multiply contents of AC1 by AC2
DIV	Divide 32 bit contents of AC0 and AC1 by AC2

*$\overline{ACS}$ = one's complement of source accumulator.
$\overline{ACS} + 1$ = two's complement of source accumulator.

INSTRUCTION EXECUTION PROTOCOL

The CPU executes arithmetic instructions in one fixed sequence that gives this small set its great flexibility.

The first step is to initialize the carry bit. The optional extension [c] to the mnemonic sets the carry bit accordingly (see table below). In the absence of this option, the last value of the carry is used.

The second step is execution of the function itself by the ALU. The result of the function on ACS and ACD, along with the carry, are passed along to the shifter. The ALU will pass either the last value of the carry or the initialized value. If an overflow condition is reached in the function, then the carry sent to the shifter is set to 1.

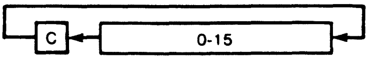
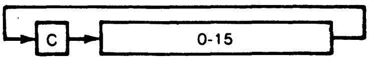
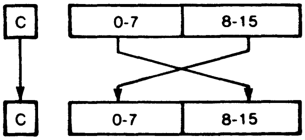
The third step is defined by the shift option [sh]. If no shift operation is specified, the 17-bit result goes to the Skip Sensor. If a shift is specified, the 17-bit result is rotated either left or right one bit, with the carry supplying bit 15 and receiving bit 0, or supplying bit 0 and receiving bit 15, respectively. The shifter is also capable of swapping the upper and lower bytes of the word without altering the carry.

ALU CARRY, SHIFT AND LOAD OPTIONS

CLASS ABBREV.	CODED CHARACTER	RESULT BITS	OPERATION
C	(option omitted)	00	Do not initialize the carry bit.
	Z	01	Initialize the carry bit to 0.
	O	10	Initialize the carry bit to 1.
	C	11	Initialize the carry bit to the complement of its present value.
SH	(option omitted)	00	Leave the result of the arithmetic or logical operation unaffected.
	L	01	Combine the carry and the 16-bit result into a 17-bit number and rotate it one bit left.
	R	10	Combine the carry and the 16-bit result into a 17-bit number and rotate it one bit right.
	S	11	Exchange the two 8-bit halves of the 16-bit result without affecting the carry.
#	(option omitted)	0	Load the result of the shift operation into ACD.
	#	1	Do not load the result of the shift operation into ACD.

DG-04424

ALU SHIFT OPERATIONS

CODED CHARACTER	SHIFTER OPERATION
L	Left rotate one place. Bit 0 is rotated into the carry position, the carry bit into bit 15 
R	Right rotate one place. Bit 15 is rotated into the carry position, the carry bit into bit 0 
S	Swap the halves of the 16-bit result. The carry bit is not affected. 

DG-04423

The fourth step alters program flow. If the [skip] option is specified, the 16-bit word and/or the carry are tested for a zero value; the program counter will be incremented by 1 if the value is zero or 2 if the value is one.

If the no-load option [#] is specified, the above four steps do not affect the value of ACD. In this event, only the carry and the program counter are affected.

ALU SKIP OPTIONS

CLASS ABBREV.	CODED CHARACTER	RESULT BITS	OPERATION
SKIP	(option omitted)	000	Never skip
	SKP	001	Always skip
	SZC	010	Skip if carry = 0
	SNC	011	Skip if carry = 0
	SZR	100	Skip if result = 0
	SNR	101	Skip if result = 0
	SEZ	110	Skip if either carry or result = 0
	SBN	111	Skip if both carry and result = 0

DG-04425

NOTE: Instructions in this group must NOT specify NO LOAD and NO SKIP at the same time. This bit pattern is reserved for the TRAP instruction.

TWO ACCUMULATOR-MULTIPLE OPERATION INSTRUCTIONS

In the function descriptions below, it is assumed that the carry bit is set to the specified value and the shift, conditional skip and load functions are performed as stipulated in the instruction field. When a conditional skip is specified in the instruction, the program counter is incremented an additional time if the condition is true.

INSTRUCTION FORMATS AND FUNCTIONS

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
COM	Complement	The one's complement of the contents of ACS is placed in the shifter. The output of the shifter is placed in ACD and the carry bit.	1 ACS ACD 0 0 0 SH C # SKIP 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
NEG	Negate	The two's complement of the contents of ACS is placed in the shifter. If the operation produces a carry of 1, carry is complemented. The output of the shifter is placed in ACD and the carry bit.	1 ACS ACD 0 0 1 SH C # SKIP 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
MOV	Move	Contents of ACS are placed in the shifter. The output of the shifter is placed in ACD and the carry bit.	1 ACS ACD 0 1 0 SH C # SKIP 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
INC	Increment	Contents of ACS are incremented by one. If the operation produces a carry of 1, carry is complemented. The output of the shifter is placed in ACD and the carry bit.	1 ACS ACD 0 1 1 SH C # SKIP 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
ADC	Add Complement	The one's complement of the contents of ACS is added to the contents of ACD. If the operation produces a carry of 1, carry is complemented. The output of the shifter is placed in ACD and the carry bit.	1 ACS ACD 1 0 0 SH C # SKIP 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

INSTRUCTION FORMATS AND FUNCTIONS (CONT)

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
SUB	Subtract	The two's complement of the contents of ACS is added to the contents of ACD. If the operation produces a carry of 1, carry is complemented. The output of the shifter is placed in ACD and the carry bit.	1 ACS ACD 1 0 1 SH C # SKIP 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
ADD	Add	The contents of ACS are added to the contents of ACD. If the operation produces a carry of 1, the carry is complemented. The output of the shifter is placed in ACD and the carry bit.	1 ACS ACD 1 1 0 SH C # SKIP 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
AND	And	The contents of ACS and ACD are logically AND'd. The output of the shifter is placed in ACD and the carry bit.	1 ACS ACD 1 1 1 SH C # SKIP 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

MULTIPLY/DIVIDE INSTRUCTIONS

The following instructions utilize an I/O instruction format and bits 10 to 15 specify device code 1. For this reason, device code 1 is an illegal device code and should not be assigned to any I/O interface device. No options or AC's specified in instruction format.

INSTRUCTION FORMATS AND FUNCTIONS

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
MUL	Multiply	The contents of AC1 are multiplied by the contents of AC2, yielding a 32-bit intermediate result. The intermediate result is added to the contents of AC0, yielding a final 32-bit result. The high order bits (0-15) of the result are loaded into AC0 and the low order bits are loaded into AC1. Bit 0 of AC0 contains the high-order bit of the result; bit 15 of AC1 contains the low-order bit. The contents of AC2 are unchanged.	0 1 1 | 1 0 | 1 1 0 | 1 1 | 0 0 0 0 0 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
DIV	Divide	The 32-bit contents of AC0 and AC1 are divided by the contents of AC2. Bit 0 of AC0 is the high-order bit of the dividend; bit 15 of AC1 is the low-order bit. The resulting quotient is placed in AC1 and the remainder in AC0. NOTE: Before the divide operation is performed, the contents of AC0 are compared to the contents of AC2. If the contents of AC0 are greater than AC2, an overflow condition is indicated. In this case, the carry bit is set to 1 and the operation is terminated; otherwise, the carry bit is set to 0.	0 1 1 | 1 0 | 1 1 0 | 0 1 | 0 0 0 0 0 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

STACK INSTRUCTIONS

An important programming tool supported by mN602 is a stack facility. The stack is a last-in/first-out buffer that can occupy up to 256 continuous locations in memory. Each entry in the stack is a full 16-bit word. The size of the stack depends on the proximity of the bottom of the stack to the next 256-word boundary, e.g., $xx377_8$ to $xx400_8$.

INSTRUCTION FORMAT

0	1	1	AC		OP CODE					0	0	0	0	0	1
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

MNEMONIC ac (PSHA, POP, MTSP, MTFP, MFSP, MFFP)
or
MNEMONIC (SAV, RET)

Stack instructions use the I/O instruction formats with a device code of 01$_8$ in bits 10 to 15. This code should be used by any I/O device. For SAV and RET, the AC field (bits 3 and 4) must be 00.

STACK FUNCTION CODES

MNEMONIC	FUNCTION
MTFP	FP←AC*
MFFP	AC**←FP**
MTSP	SP←AC*
MFSP	AC**←SP**
PSHA	SP←SP + 1 @SP←AC#

POP	AC←@SP#
	SP←SP - 1

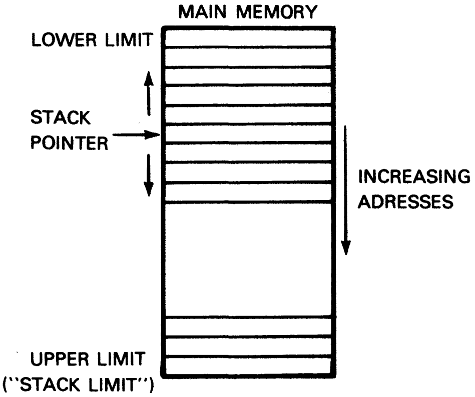
SAV	PUSH RETURN BLOCK ON THE STACK
RET	POP RETURN BLOCK OFF THE STACK

* Least significant 15 bit (1 to 15) of accumulator are used.
**Bit 0 of the accumulator is set to 0.
# @ denotes that the contents of the stack pointer are used as an address to a location in memory.

DEFINING A STACK

Three parameters define a stack: the lower limit, the upper limit, and the present top of the stack (stack pointer or SP). The upper and lower limits define the bounds of the stack in memory. The stack pointer defines the location of the last word moved to the stack and the first word to be removed from the stack. The stack pointer must be initialized to the starting address of the stack minus one.

STACK BOUNDARIES



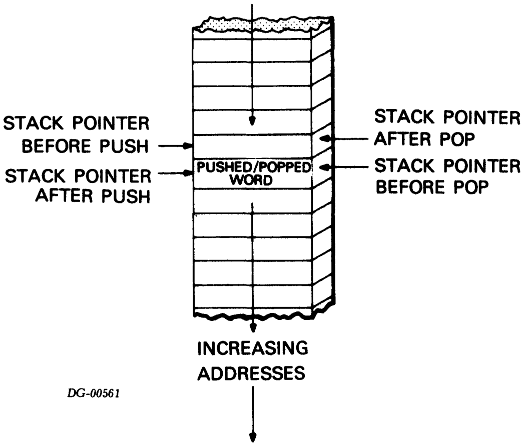
DG-04426

PUSH AND POP

The stack provides quick sequential storage for the contents of any of the accumulators. The 15-bit stack pointer (SP) contains the address of the current top of the stack. If the stack pointer overflows, the carry is not affected. All addressing to the stack is directed to the top of the stack. Thus, the last word moved to the stack by PUSH is the first word removed by POP.

During a PUSH instruction the stack pointer is first incremented, then used as the address to store the contents of the specified accumulator. The POP instruction is the exact reverse of PUSH. When a POP is executed, the contents of the top of the stack are loaded into the specified AC, after which SP is decremented. The word pointed to by the top of the stack still exists in memory, but is no longer part of the stack. If a 400-word boundary is crossed when CPU interrupts are enabled, a stack overflow interrupt is generated after the instruction has been executed. If a stack overflow occurs and ION = 0, a stack overflow interrupt will occur when ION returns to 1.

PUSH/POP OPERATIONS



DG-00561

SAVE AND RETURN

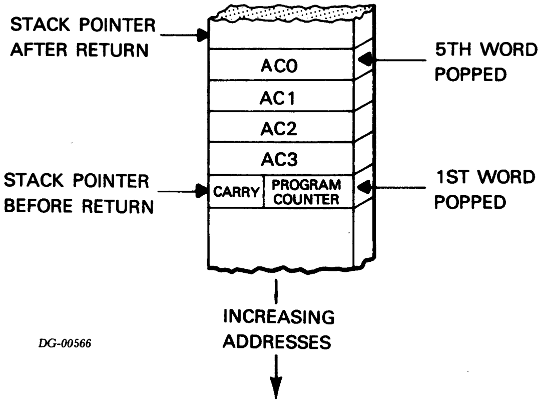
One of the main uses for a stack is to save the state of the CPU during a subroutine call. A subroutine call is made with a JSR instruction. Since a subroutine, like any program, needs the accumulators, and the past values are generally important, they have to be stored until the main program regains control.

The return block, created by the SAVE instruction, saves the state of the CPU at that point. The first three words pushed on the stack are AC0, AC1, and AC2, in that order. The next value pushed on the stack is the current frame pointer (FP).

The frame pointer (FP) holds the address of the top of the present return block in the stack. When the save is completed, FP and SP point to the top of the stack (the starting address of the return block). In this way, the previous state of the CPU and the stack can be restored when returning to the calling program. The frame pointer, like the stack pointer, is 15 bits long.

The last word of the return block is AC3. If the SAVE occurred after a JSR, AC3 holds the return address of the subroutine in bits 1-15 and the carry in bit 0.

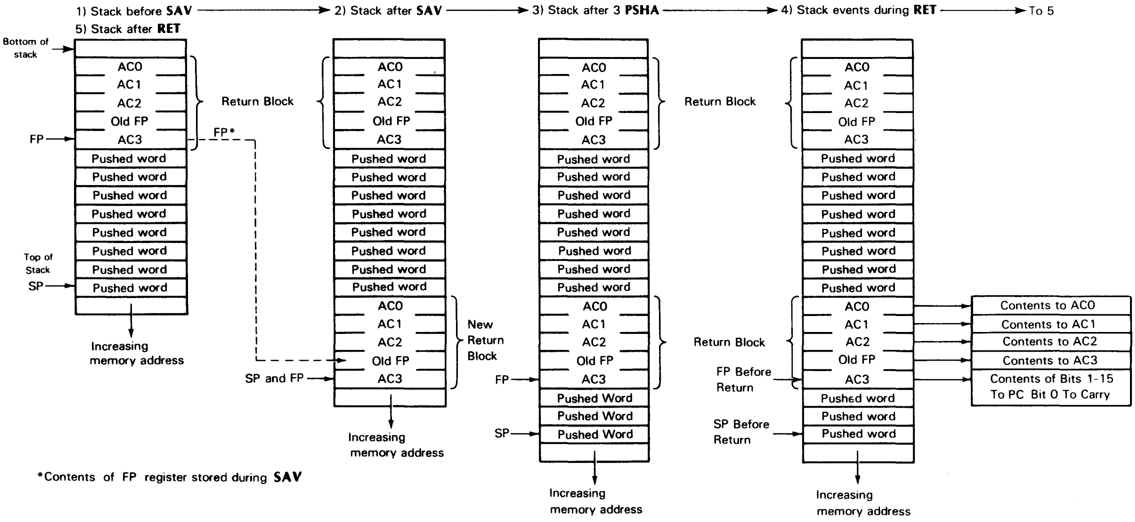
RETURN BLOCK FORMAT



DG-00566

The RETURN instruction removes the return block from the stack. If the last SAVE was done when entering a subroutine, the CPU returns to the calling program at the instruction following the JSR. The instruction starts popping at the frame pointer. The first word reinitializes the carry and the program counter. The second pop moves the frame pointer (now pointing to the next oldest return block) to AC3 and loads the frame pointer register. The next three pops on the stack return the old values of AC2, AC1, and AC0, respectively.

SAVE/RETURN OPERATIONS



*Contents of FP register stored during **SAV**

Note: The SP register contains the address of the top memory location in the stack. Thus, the three single variables pushed on the stack after the last return block (see 4) are effectively pushed off the stack when the return instruction is performed. These variables could be retained elsewhere in memory by executing a series of three pop instructions, each followed by a **STA** (*Store Accumulator*) instruction prior to executing the return instruction. Then, if desired, the variables can be pushed back on the stack after the execution of the return instruction.

After the return instruction was executed, the contents of AC0, AC1, AC2, PC, carry and the SP and FP registers were restored to their original values before the save instruction was executed. The new value of the FP register is left in AC3 after the return instruction (see 4 and 5)

DG-06017

Four instructions affect the accumulators, stack pointer, and frame pointer directly. MOVE TO STACK POINTER (MTSP) moves the contents of the accumulator to the stack pointer. MOVE TO FRAME POINTER (MTFP) moves the contents of the accumulator to the frame pointer. Neither of these instructions affect the contents of the accumulators. MFSP and MFFP move from the stack pointer and the frame pointer, respectively, to the specified accumulator without affecting either pointer.

mN602
STACK INSTRUCTIONS

INSTRUCTION FORMATS AND FUNCTIONS

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
PSHA	Push	The SP is incremented by 1; then, the contents of the specified AC are loaded into the memory location addressed by the contents of SP. The contents of the specified AC are unchanged. If the contents of bits 8-15 of SP are 000 (a 256-word boundary has been crossed), a stack overflow interrupt request is generated.	0 1 1 AC 0 1 1 0 0 0 0 0 0 0 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
POPA	Pop	The specified AC is loaded with the contents of the memory location addressed by SP; then, SP is decremented by 1.	0 1 1 AC 0 1 1 1 0 0 0 0 0 0 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
MTSP	Move To Stack Pointer	The contents of bits 1-15 of the specified AC are placed in SP. The contents of the specified AC are unchanged.	0 1 1 AC 0 1 0 0 0 0 0 0 0 0 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
MTFP	Move To Frame Pointer	The contents of bits 1-15 of the specified AC are placed in FP. The contents of the specified AC are unchanged.	0 1 1 AC 0 0 0 0 0 0 0 0 0 0 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
MFSP	Move From Stack Pointer	The contents of SP are placed in bits 1-15 of the specified AC and bit 0 of the specified AC is set to 0. The contents of the SP are unchanged.	0 1 1 AC 0 1 0 1 0 0 0 0 0 0 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
MFFP	Move From Frame Pointer	The contents of FP are placed in bits 1-15 of the specified AC and bit 0 of the specified AC is set to 0. The contents of FP are unchanged.	0 1 1 AC 0 0 0 1 0 0 0 0 0 0 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

INSTRUCTION FORMATS AND FUNCTIONS (CONT)

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
SAV	Save	Information is pushed on stack as follows: 1) the SP is incremented by 1; then, the contents of AC0 are stored in the memory location addressed by the contents of SP; 2) the SP is incremented by 1; then, the contents of AC1 are stored in the memory location addressed by the contents of SP; 3) the SP is incremented by 1; then, the contents of AC2 are stored in the memory location addressed by the contents of SP; 4) the SP is incremented by 1; then, the contents of FP (before the SAVE instruction) are stored in bits 1-15 of the memory location addressed by the contents of SP and 0 is written into bit 0. 5) the SP is incremented by 1; then, the content of the carry bit is stored in bit 0 and the contents of bits 1-15 of AC3 are stored in bits 1-15 of the memory location addressed by the contents of SP. If the contents of bits 8-15 of SP are 000 (a 256-word boundary was crossed) at any time during execution of this instruction, a stack overflow interrupt request is generated. At the end of the instruction, the new contents of SP are loaded in both the FP and bits 1-15 of AC3. Bit 0 of AC3 is set to 0.	0 1 1 0 0 1 0 1 0 0 0 0 0 0 0 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

INSTRUCTION FORMATS AND FUNCTIONS (CONT)

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
RET	Return	Information is popped off the stack as follows: 1) the contents of the FP are loaded into the SP; 2) bit 0 of the memory location addressed by the contents of SP is loaded into the carry bit and bits 1-15 are loaded into the PC; 3) the SP is decremented by 1; then, the contents of the location addressed by the contents of SP are loaded into AC3; 4) the SP is decremented by 1; then, the contents of the location addressed by the contents of SP are loaded into AC2; 5) the SP is decremented by 1; then, the contents of the location addressed by the contents of SP are loaded into AC1; 6) the SP is decremented by 1; then, the contents of the location addressed by the contents of SP are loaded into AC0. 7) the SP is decremented by 1. At the end of the instruction, the new contents of AC3 (bits 1-15) are loaded into FP.	0 1 1 0 0 1 0 1 1 0 0 0 0 0 0 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

INPUT / OUTPUT INSTRUCTIONS

Instructions in this group communicate directly with an I/O device, one word at a time. This kind of transfer is called programmed I/O. All transfers take place on the I/O bus, which connects the CPU and the peripheral. The instruction specifies which of the device's three input or output buffers is to be addressed. One of the fields of the instruction specifies the CPU accumulator to receive or transmit information.

Each instruction has a 6-bit device code (bits 10 to 15) that can specify one of 64 possible device codes. An I/O controller connects the device and the I/O bus. The controller can set up a unique device code for a device or the protocol for devices using the same device code.

NOTE: Device codes 01_8, 02_8, 03_8, and 77_8 should not be used for any I/O device. The CPU uses code 01_8 for MUL, DIV, and stack instructions and it uses code 77_8 for internal control.

INSTRUCTION FORMAT

0	1	1	AC*		OP CODE			CTRL**		DEVICE CODE					
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

* The AC field is set to 00 for NIO and SKP instructions.

** The control field sets or tests the Busy/Done flags, depending on the instruction. In the instruction bit patterns these are represented by t or f.

BUSY AND DONE FLAGS

A set of Busy/done flags, part of the I/O device controller (IOC), tells the CPU the state of the device. When both Busy and Done are zero, the device is idle, waiting for the CPU to start it. To start the device, an I/O instruction sets Busy to 1, and resets Done to 0, regardless of its past state. When the device has finished its operation, it sets its Done flag to 1 and resets Busy to 0. At this point the device can issue a program interrupt request to the CPU.

NOTE: The mN602 requires either an mN603, mN613, or mN615 device controller.

STATE OF BUSY/DONE FLAGS

BUSY	DONE	FUNCTION
0	0	Device idle, waiting for a command from CPU to start.
0	1	Device finished operation. Can start interrupt to CPU.
1	0	Device busy with operation.
1	1	Not used.

NON-EXISTENT DEVICE CODES

If an instruction is issued to a non-existent device, the CPU acts as if the device exists and makes the transfer on the I/O bus. If the Busy/Done flags of a non-existent device are read, they appear as 00. If the CPU tries to read one of the buffers of a non-existent deivce, the result in the specified accumulator is 177777_8 (all bits set to 1).

BUSY/DONE SET OPTIONS

CLASS ABBREV.	CODED CHARACTER	RESULT BITS	OPERATION
f	(option omitted)	00	Does not affect the Busy and Done flags.
	S	01	Start the device by setting Busy to 1 and Done to 0.
	C	10	Idle the device by setting both Busy and Done to 0.
	P	11	The effect, if any, depends on the device.

DG-04421

BUSY/DONE TEST OPTIONS

CLASS ABBREV.	CODED CHARACTER	RESULT BITS	OPERATION
t	BN	00	Tests for Busy = 1
	BZ	01	Tests for Busy = 0
	DN	10	Tests for Done = 1
	DZ	11	Tests for Done = 0

DG-04422

When the instructions described below are executed, the CPU sends the I/O instruction, exactly as it is stored in memory, to the I/O bus. Following the instruction, data is transferred either from the CPU or the I/O controller. See I/O Operations section.

INSTRUCTION FORMATS AND FUNCTIONS

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
NIO	No I/O Transfer	The Busy and Done flags of the addressed device are set according to the convention specified in bits 8-9 of the instruction. The data transferred by the CPU is meaningless and is ignored by the I/O controller.	0 1 1 0 0 0 0 0 F DEVICE CODE 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
DIA	Data In A	Following the receipt of this instruction, the addressed I/O controller is expected to send a 16-bit word to the I/O bus from its A buffer. This word is retrieved from the I/O bus by the CPU and placed in the accumulator specified in bits 3-4 of the instruction. After the data transfer, the Busy and Done flags are set according to the convention specified in bits 8-9 of the instruction.	0 1 1 AC 0 0 1 F DEVICE CODE 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
DIB	Data In B	Same as DIA, except B buffer.	0 1 1 AC 0 1 1 F DEVICE CODE 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
DIC	Data In C	Same as DIA, except C buffer.	0 1 1 AC 1 0 1 F DEVICE CODE 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
DOA	Data Out A	Following the receipt of this instruction, the addressed I/O controller should retrieve the next 16 bits of data appearing on the I/O bus and load them into its A buffer. The data bits are the contents of the accumulator specified in bits 3-4 of the instruction. After the data transfer, the Busy and Done flags are set according to the convention specified in bits 8-9 of the instruction.	0 1 1 AC 0 1 0 F DEVICE CODE 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
DOB	Data Out B	Same as DOA, except B buffer.	0 1 1 AC 1 0 0 F DEVICE CODE 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
DOC	Data Out C	Same as DOA, except C buffer.	0 1 1 AC 1 1 0 F DEVICE CODE 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

INSTRUCTION FORMATS AND FUNCTIONS (CONT)

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
SKP	I/O Skip	Following the receipt of this instruction, the addressed I/O controller sends the state, 0 or 1, of its Busy and Done flags to the I/O bus. This information is contained in a 16-bit data word. Data bit 0 contains the complement of the state of the Done flag. Data bit 1 contains the complement of the state of the Busy flag. The CPU retrieves this information from the I/O bus and ignores the contents of bits 2-15. It takes the appropriate action depending upon the content of the Busy and Done bits and the specified test condition contained in bits 8-9 of the instruction. If the test condition is true, the program counter is incremented an additional time.	0 1 1 / 0 0 / 1 1 1 / T / DEVICE CODE 0 1 2 / 3 4 / 5 6 7 / 8 9 / 10 11 12 13 14 15

PROGRAM INTERRUPTS

There are two classes of interrupts: normal interrupts and non-maskable interrupts. Normal interrupts are disabled when the CPU ION flag is 0. Non-maskable interrupts are not.

NORMAL INTERRUPTS

When a normal interrupt occurs, the CPU sets ION to 0, disabling any further normal interrupts. Then it increments the PC and stores its value in location 0 of unmapped space, setting bit 0 of location 0 to 0. Location 0 now contains the program's return address.

If the program was executing in unmapped space when the CPU received the normal interrupt, the Busy flag of device 01 is set to 0. If the program was executing in mapped space when the CPU got the normal interrupt, the Busy flag of device 01 is set to 1.

Then the CPU jumps indirectly through a location in unmapped space to an interrupt handler stored somewhere in unmapped space. Which location the CPU jumps indirectly through depends on the source of the interrupt.

NON-MASKABLE INTERRUPTS

When a non-maskable interrupt occurs, the CPU increments the PC and stores its value in location 0 of mapped space, setting bit 0 of location 0 to 0. Location 0 now contains the program's return address. Then the CPU jumps indirectly through location 1 of mapped space to an interrupt handler stored somewhere in mapped space.

INTERRUPT HANDLER

The interrupt handler should save state before transferring to the service routine. If the service routine uses the accumulators, stack pointer and frame pointer, the interrupt handler should save their previous values. It should also save the state of MAPON. It's good programming practice to keep track of the state of MAPON and record its current value in an accessible location. The interrupt handler may then establish a new priority mask with the MSKO instruction and re-enable interrupts. At the conclusion of the service routine, the interrupt handler disables interrupts, restores the CPU to its prior state, re-enables interrupts, and then jumps indirectly through location 0. The interrupted program continues.

DISABLING INTERRUPTS

It is important to understand the different ways that the CPU disables interrupts.

1) When a normal interrupt occurs, the CPU sets ION to 0, disabling all normal interrupts. The CPU will still honor a non-maskable interrupt, however. You can re-enable interrupts by issuing an INTEN instruction. This instruction returns ION to 1.

2) When the CPU executes an INTDS instruction, ION becomes 0 and all normal interrupts are disabled. The CPU will still honor a non-maskable interrupt, however. You can re-enable interrupts by issuing an INTEN instruction. This instruction returns ION to 1.

3) When the CPU executes a MEP instruction, a MAPON change becomes pending. If $\overline{\text{MAP EXISTS}}$ (bit 1 of the System Status Word) is low, all interrupts (normal and non-maskable) are disabled and any breakpoint instructions will become no-ops. If the CPU then executes an LDA or STA instruction, interrupts are re-enabled. The ION flag does not change.

4) When the CPU executes a MEP instruction, a MAPON change becomes pending. If $\overline{\text{MAP EXISTS}}$ is low, all interrupts (normal and non-maskable) are disabled and any breakpoint instructions will become no-ops. If the CPU executes a JMP @ or JSR @ instruction, control transfers to the other memory space and all interrupts are re-enabled. The ION flag does not change.

5) When the CPU takes a non-maskable interrupt, control transfers to the mapped memory space. If $\overline{\text{MAP EXISTS}}$ is low, all interrupts (normal and non-maskable) are disabled and any breakpoint instructions will become no-ops. They remain disabled until the CPU executes an INTRE instruction. The ION flag does not change.

6) When the CPU executes a TRAP instruction and the $\overline{\text{MAP ON TRAP}}$ bit of the System Status Wor is low, the CPU traps to mapped space. If $\overline{\text{MAP EXISTS}}$ is low, all interrupts (normal and non_maskable) are disabled and any breakpoint instructions will become no-ops. They remain disabled until the CPU executes an INTRE instruction. The ION flag does not change.

7) When the CPU executes a BRKPT instruction, it stores the PC in location 0 of mapped space and jumps indirectly through location 1 of mapped memory. If $\overline{\text{MAP EXISTS}}$ is low, all interrupts (normal and non-maskable) are disabled and any breakpoint instructions will become no-ops until the CPU executes an INTRE instruction. The ION flag does not change.

INTERRUPT SOURCES

An interrupt can come from three sources. These sources are:

Two internal registers internal to the mN602
Three bits of the System Status Word
Any external I/O device

CPU INTERRUPTS

INTERRUPT SOURCE	INTERRUPT FLAG	INDIRECT JUMP THROUGH THIS LOCATION	MAPON	PRIORITY
INTERNAL REGISTERS	SO RQ	000003	Low	Second
	RTC RQ	000002	Low	Third
SYSTEM STATUS WORD	$\overline{\text{EXT RTC}}$	000002	Low	Third
	$\overline{\text{POWERFAIL}}$	000001	Low	Fourth
	$\overline{\text{NMR}}$	000001	High	First
EXTERNAL I/O DEVICE	$\overline{\text{INTR}}$	000001	Low	Fourth

Normal interrupts are all interrupts except those due to $\overline{\text{NMR}}$. Non-maskable interrupts are those due to $\overline{\text{NMR}}$.

Internal Registers

The two internal registers that can cause interrupts are Stack Overflow Request (SO RQ) and Real Time Clock Request (RTC RQ). Both single bit registers are automatically cleared after they cause the interrupt.

SO RQ interrupts when the stack pointer increments over a 256 word boundary (xxx377_8 to xxx400_8). No interrupt results when the stack pointer decrements over the same boundary.

RTC RQ interrupts every 1.9651 ms. For this interrupt to occur both the Real Time Clock Enable bit (RTC ON) and ION must be 1. If RTC ON is 0, RTC RQ cannot cause an interrupt even if ION is 1.

System Status Word

The CPU reads the System Status Word (SSW) off the memory bus on the data phase of every memory refresh. Certain bits of this word can cause interrupts at this time. These bits are $\overline{\text{EXT RTC}}$, $\overline{\text{POWERFAIL}}$, and $\overline{\text{NMR}}$.

The CPU samples these bits after every refresh. The interrupt is edge sensitive. That means an interrupt occurs if the bit has changed state in the appropriate direction since the last sampling.

$\overline{\text{EXT RTC}}$ (External Real Time Clock) is bit 12 of the SSW. If $\overline{\text{EXT RTC}}$ has experienced a rising edge since the last refresh, the CPU disables the internal real time clock and responds as if the internal RTC generated the interrupt. It saves the PC in location 0 and jumps indirectly through location 2.

$\overline{\text{POWERFAIL}}$ is bit 6 of the SSW. This bit should be driven low by the power supply if power to the mN602 goes below minimum levels. If $\overline{\text{POWERFAIL}}$ has experienced a falling edge since the last refresh, an interrupt occurs.

$\overline{\text{NMR}}$ (Non-maskable Request) is bit 0 of the SSW. If $\overline{\text{NMR}}$ has experienced a falling edge since the last refresh, a non-maskable interrupt occurs. When the mN602 gets a non-maskable interrupt, it stores the PC in location 0 of mapped memory and jumps indirectly through location 1 of mapped memory.

External I/O Device Interrupts

An interrupt occurs when ION is 1 and $\overline{\text{INTR}}$ goes low. The mN602 stores the PC in location 0 of unmapped space and jumps indirectly through location 1 of unmapped space.

PRIORITY INTERRUPTS

The mN602 provides 16 levels of priority for standard I/O devices. The Maskout instruction (MSKO) controls the priority of the interrupt, using a 16-bit mask word. Each bit of the mask denotes a priority level. One mask bit controls one device, or a number of devices with the same transfer rate. If a mask bit is 1, devices assigned to that priority level are disabled from generating interrupts. If the bit is 0, interrupts from that level are enabled. Each I/O controller contains an Interrupt Disable flag that is set by the Maskout instruction to enable or disable interrupts.

CPU I/O INSTRUCTIONS

I/O instructions addressed to device code 77_8 are CPU I/O instructions. This instruction group performs special functions rather than controlling specific devices. Some CPU I/O instructions are global in nature. Global instructions are directed to all I/O controllers; others are acted upon internally by the CPU.

In all but the I/O Skip instruction, I/O instructions addressed to device code 77_8 use bits 8-9 to control the condition of the Interrupt On flag (also referred to in the text as the interrupt enable flag). The table below shows the results of these bits when used with device code 77_8.

CPU ION FLAG SET AND TEST OPTIONS

CLASS ABBREV.	CODED CHARACTER	RESULT BITS 8 - 9	OPERATION
Set (f)	(option omitted)	00	Does not affect the state of the Interrupt On flag.
	S	01	Set the Interrupt On flag to 1.
	C	10	Set the Interrupt On flag to 0.
	P	11	Do not use.
Test (t)	BN	00	Tests for Interrupt On = 1.
	BZ	01	Tests for Interrupt On = 0.
	DN	10	Skip if there is a power fail (the CPU Done flag is 1 if the System Status Word bit $\overline{\text{POWERFAIL}}$ is low)
	DZ	11	Skip if no power fail.

CPU I/O INSTRUCTIONS

INTEN
INTDS
INTA
MSKO
IORST
HALT
SKP

RTCEN
RTCDS
NIOP 0,CPU (BRKPT)
NIOP 1,CPU (MEP)
NIOP 2,CPU (INTRE)
SKPDN 1 (SKPM)

When the instructions described below are executed, the CPU sends the instruction, exactly as it is stored in memory, to the I/O bus. In most cases, after the instruction is transmitted, data is transferred either from the CPU or the I/O controller. See I/O Operations section.

INSTRUCTION FORMATS AND FUNCTIONS

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
INTEN	Interrupt Enable	This instruction is internal to the CPU. When executed, the Interrupt On flag is set to 1 after one more instruction is executed; then interrupt requests are honored by the CPU. The data transmitted by the CPU following this instruction should be ignored by the I/O controllers.	0 1 1 | 0 0 | 0 0 0 | 0 1 | 1 1 1 1 1 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
INTDS	Interrupt Disable	This instruction is internal to the CPU. When executed, the Interrupt On flag is set to 0 and interrupt requests are not honored by the CPU. The data transmitted by the CPU following this instruction should be ignored by the I/O controllers.	0 1 1 | 0 0 | 0 0 0 | 1 0 | 1 1 1 1 1 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
INTA	Interrupt Acknowledge	This is a global instruction directed to all I/O controllers. It is sent to the I/O bus in response to a program interrupt request. The highest priority device requesting an interrupt should respond by sending its device code to the I/O bus in a 16-bit data word. The device code is contained in data bits 10-15. When the CPU retrieves this information, the device code is placed in bits 10-15 of the accumulator specified in bits 3-4 of the instruction. If the device code returned is 77_8, the interrupt was due to a power fail. Data bits 0-9 are transmitted and loaded in the specified AC as 0's. If no device is requesting an interrupt, bits 0-15 of the specified accumulator are set to ones.	0 1 1 | AC | 0 1 1 | 00 | 1 1 1 1 1 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

INSTRUCTION FORMATS AND FUNCTIONS (CONT)

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
MSKO	Mask Out	This is a global instruction directed to all I/O controllers. After the CPU sends the instruction, it sends the contents of the accumulator specified in bits 3-4 of the instruction. The contents of the accumulator are interpreted by the I/O controllers as a 16-bit software priority mask. **Each I/O controller responds (after one more instruction is executed) by turning its Interrupt Disable flag on or off according to the device's selected priority level. A 1 in a device's mask bit turns its Interrupt Disable flag on; a 0 turns it off.**	0 1 1 AC 1 0 0 0 0 1 1 1 1 1 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
IORST	Reset	This is a global instruction directed to all I/O controllers and the CPU. The I/O controllers respond by setting their Busy and Done flags to 0 and by turning their Interrupt Disable flags off. **The CPU responds by disabling its real-time clock from generating interrupts.** The data transferred by the CPU following this instruction should be ignored by the I/O controllers.	0 1 1 0 0 0 1 0 1 0 1 1 1 1 1 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

3

INSTRUCTION FORMATS AND FUNCTIONS (CONT)

MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
HALT	Halt	This instruction is internal to the CPU. It is used to stop the execution of instructions. The Interrupt On flag is set to 1. The CPU honors both program interrupt and data channel requests while it is in the halted state. When this instruction is executed, no data is transferred as part of the I/O transaction. See I/O Operations.	0 1 1 / 0 0 / 1 1 0 / 0 0 / 1 1 1 1 1 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
SKP	CPU Skip	This instruction is internal to the CPU. When executed, the CPU switches to I/O input mode (see I/O Operations section); but, no device should respond by placing data on the I/O bus. **CPU skip tests the state of the Interrupt On flag or the power fail flag. If the test condition specified in bits 8-9 of the instruction is true, the program counter is incremented an additional time.**	0 1 1 / 0 0 / 1 1 1 / T / 1 1 1 1 1 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
RTCEN	Real-time Clock Enable	This instruction is internal to the CPU. It enables the CPU's real-time clock to generate program interrupt requests. The data transmitted by the CPU following this instruction should be ignored by I/O controllers.	0 1 1 1 0 0 1 0 / 0 0 / 1 1 1 1 1 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

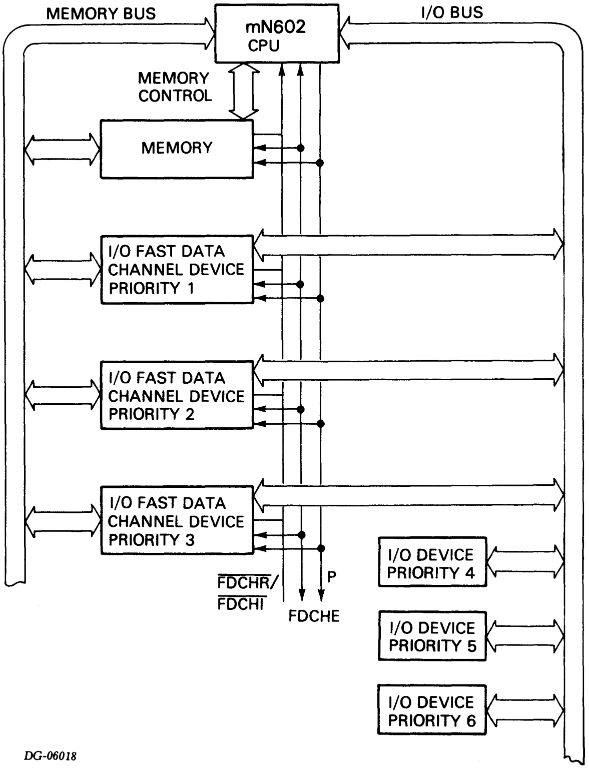
MNEMONIC	MEANING	FUNCTION	INSTRUCTION FORMAT
RTCDS	Real-time Clock Disable	This instruction is internal to the CPU. It disables program interrupt requests by the CPU'S real-time clock. The data transmitted by the CPU following this instruction should be ignored by I/O controllers.	0 1 1 0 1 0 1 0 0 0 1 1 1 1 1 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
BRKPT*	Breakpoint	Turns on map. Stores PC in Location 0 of upper memory. Jumps indirectly through Location 1 of upper memory. Executes upper memory locations. Breakpoints are possible in a mapped program, but only after the INTRE instruction, and if no map change is pending.	0 1 1 0 0 0 0 0 1 1 1 1 1 1 1 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 Equivalent mnemonic: NIOP 0,CPU
MEP*	Map Enable Pending	The next memory reference instruction complements MAPON and accesses the other logical address space. (The MAPON change remains pending during non-memory reference instructions.) If next instruction is LDA or STA, data is transferred between CPU and the other memory. MAPON remains complemented only for one memory reference. If next instruction is JSR@ or JMP@, MAPON is complemented after the first memory reference and remains complemented. The CPU resolves the first level of indirection, complements MAPON, and transfers control to a location in the other memory space.	0 1 1 0 1 0 0 0 1 1 1 1 1 1 1 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 Equivalent mnemonic: NIOP 1,CPU
INTRE*	Interrupt Reenable	Enables all interrupts (normal and non-maskable)	0 1 1 1 0 0 0 0 1 1 1 1 1 1 1 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 Equivalent mnemonic: NIOP 2,CPU
SKPM*	Skip on Previous Map	Used in interrupt handlers. Tests state of MAPON before the last normal interrupt occured. Skip next instruction if MAPON was 1.	0 1 1 0 0 1 1 1 0 0 0 0 0 0 0 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

*At present, the assembler does not recognize a mnemonic for this instruction. Use equivalent mnemonic: SKPBN 1.

OPERATIONS PROTOCOL

The mN602 communicates with memory and I/O interfaces via the memory bus and I/O bus, respectively. Four types of transfers are possible: memory reference (read/write), programmed I/O, standard data channel, and high speed data channel. Read/write and high speed data channel transfers are made via the memory bus. Programmed I/O and standard data channel transfers are made via the I/O bus. Since the fast data channel transfers data only on the memory bus, devices using it require the I/O bus to take commands from the CPU.

mN602 SYSTEM WITH FAST DATA CHANNEL

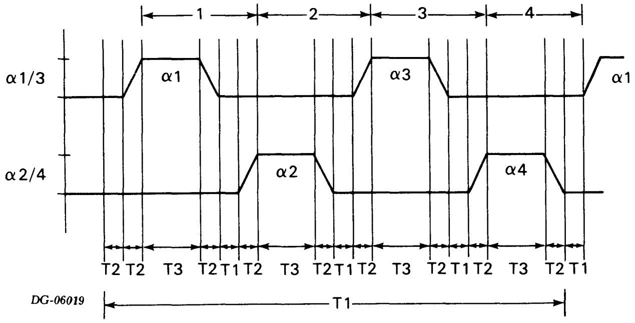


DG-06018

FOUR PHASE CLOCK

The CPU clock is composed of two lines, $\alpha 1/3$ and $\alpha 2/4$, which synchronize all transfers external to the mN602. These two lines give a 4-phase, non-overlapped clock that defines one CPU cycle. One CPU cycle consists of the sequence $\alpha 1$, $\alpha 2$, $\alpha 3$, $\alpha 4$.

FOUR-PHASE CLOCK



DG-06019

NOTE: See AC Transition Timing Table (p.67) for values.

MEMORY OPERATIONS

Memory references involve not only the memory reference instructions but also CPU memory transfers. These include any form of read (LDA, POPA, or RET), write (STA, PSHA, or SAV), and the instruction fetch cycle. (Memory sees an instruction fetch as any other read. The only difference is that the contents of the memory location referenced go to the instruction register instead of an accumulator.

Signals P, SAE, and WE are used for the normal memory references. These signals strobe data in or out of the CPU along the memory bus. The $\overline{\text{WAIT}}$ signal, when driven low by slow memories, forces the CPU to initiate another SAE or WE. Slow memories use $\overline{\alpha 2}$ as a clock to strobe onto the memory bus. When the mapped memory space is being accessed, MAPON goes high during the setup time for the address.

Both addresses and data are transferred on the memory bus. During the $\alpha 2$ phase of the clock, the memory bus lines are precharged. The CPU places information on the bus by discharging lines for logic zero, and leaving lines for logic one charged. The last part of the transfer comes during the $\alpha 1$ clock phase. During the $\alpha 2$ phase, the memory bus lines are precharged again in preparation for the next transfer.

Bit 0 of the memory bus (MB0) determines if the next event on the bus is a memory reference or a refresh. If MB0 is low, then the next event is a memory reference. The remaining 15 bits give the memory address. If MB0 is high, the next event is a memory refresh.

The mN602 contains a refresh address and a refresh counter. When a refresh occurs, the refresh counter is reset and the refresh address is incremented. During a refresh operation, the CPU transfers a 7-bit address specifying 1/128 of total memory. Memory bus bits 9-15 hold the address while bits 0-8 contain all ones.

There are two types of refreshes: transparent refreshes and demand refreshes.

A transparent refresh occurs at irregular intervals during the execution of instructions. Transparent refreshes cannot occur during data channel activity (high speed and standard).

A demand refresh is a refresh operation initiated by the refresh logic. It occurs when the refresh counter counts to 16 μs. When a demand refresh occurs, any high speed and standard data channel activity is suspended. Demand refreshes insure that memory is refreshed at least once every 16 μs.

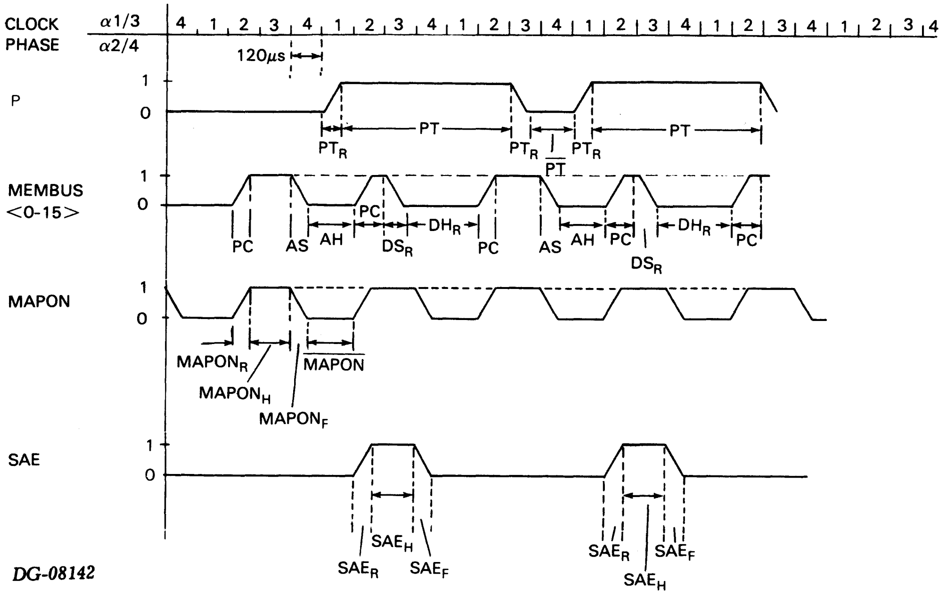
Both memory references and refresh operations are started on the rising edge of P. When P rises, the memory tests MB0 and proceeds with the indicated operation.

Read

The mN602 precharges the memory bus during every $\alpha 2$ phase. If the mN602 wants to read memory, it sets up an address on the memory bus. Then during the next $\alpha 1$ phase, the mN602 drives P high. When P goes high, memory assumes that the memory bus contains a valid address.

Then the CPU transmits SAE to the memory bus transceivers. Depending on the instruction, there may be a delay of a few CPU cycles from the time when P goes high to the time when SAE is transmitted. The falling edge of SAE signifies that valid read data are on the memory bus. The valid data remain on the bus until the bus is precharged during the next $\alpha 2$ phase. P drops on the falling edge of this $\alpha 2$.

MEMORY READ TIMING DIAGRAM



DG-08142

NOTE: See Timing Tables near the end of this chapter.

Read (Slow Memories)

Unmapped mN602 memory can contain slow memories (e.g., EPROMs). If the mN602 reads a slow memory location, the slow memory drives $\overline{\text{WAIT}}$ low. The CPU samples $\overline{\text{WAIT}}$ on the falling edge of $\alpha 2$. If $\overline{\text{WAIT}}$ was low the last time the CPU sampled it, the CPU does not recognize the data on the memory bus as valid. The CPU transmits another SAE to the memory bus transceivers.

During the next $\alpha 2$ phase the CPU again precharges the memory bus. On the falling edge of this $\alpha 2$ the CPU samples $\overline{\text{WAIT}}$. If $\overline{\text{WAIT}}$ is now high (slow memory required only one wait cycle), the falling edge of SAE signifies that valid data are now on the bus. The valid data remain on the bus until the bus is precharged during the next $\alpha 2$ phase. P drops on the falling edge of this $\alpha 2$.

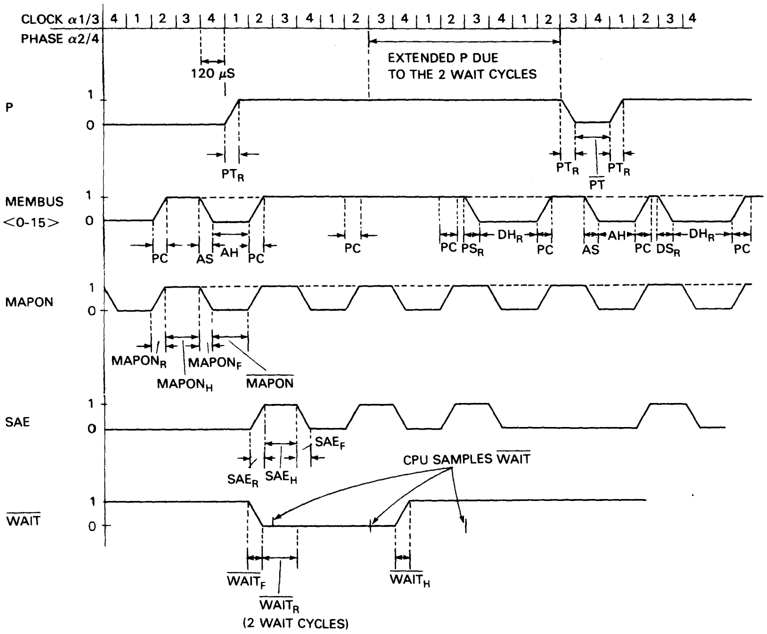
The first figure on the next page illustrates the use of wait cycles. It shows memory keeping $\overline{\text{WAIT}}$ low for two wait cycles.

The CPU can tolerate up to 50 wait cycles. More than 50 wait cycles on one memory reference halts the CPU.

The figure, $\overline{\text{WAIT}}$ goes low during $\alpha 2$. The CPU samples it on the falling edge of $\alpha 2$ and consequently finds it low. $\overline{\text{WAIT}}$ stays low during the next $\alpha 3$, $\alpha 4$, $\alpha 1$, $\alpha 2$ phases. The CPU samples $\overline{\text{WAIT}}$ again on the falling edge of this last $\alpha 2$ phase and again finds it low. Because $\overline{\text{WAIT}}$ was low both times the CPU sampled it, the CPU sent out two extra SAE signals. On the falling edge of the next $\alpha 2$, the CPU again samples $\overline{\text{WAIT}}$ and this time finds it high.

The falling edge of the third SAE signal signifies that valid data are on the bus. The third SAE signal is the one that signifies valid data because $\overline{\text{WAIT}}$ was high the last time the CPU sampled it. P drops on the falling edge of the next $\alpha 2$.

MEMORY READ TIMING DIAGRAM WITH WAIT CYCLES



DG-08143

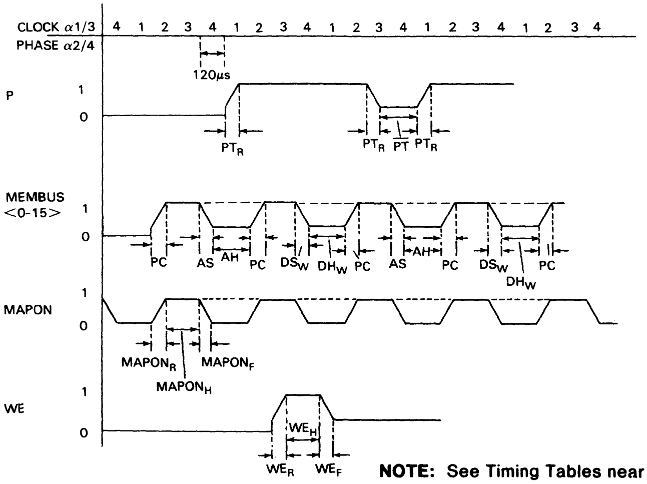
NOTE: See Timing Tables near the end of this chapter.

Write

The mN602 precharges the memory bus during every α2 phase. If the mN602 wants to write to memory, it sets up an address on the memory bus. Then during the following α1, the mN602 drives P high. When P goes high, memory assumes that the memory bus contains a valid address.

The CPU precharges the memory bus during the next α2 phase. Then, the CPU transmits WE to the memory bus transceivers. Depending on the instruction there may be a delay of a few CPU cycles from the time when P goes high to the time when WE is transmitted. The falling edge of WE signifies that valid write data are on the memory bus. The valid data remain on the bus until the bus is precharged during the next α2 phase. P drops on the falling edge of this α2.

MEMORY WRITE TIMING DIAGRAM



DG-08144

NOTE: See Timing Tables near the end of this chapter.

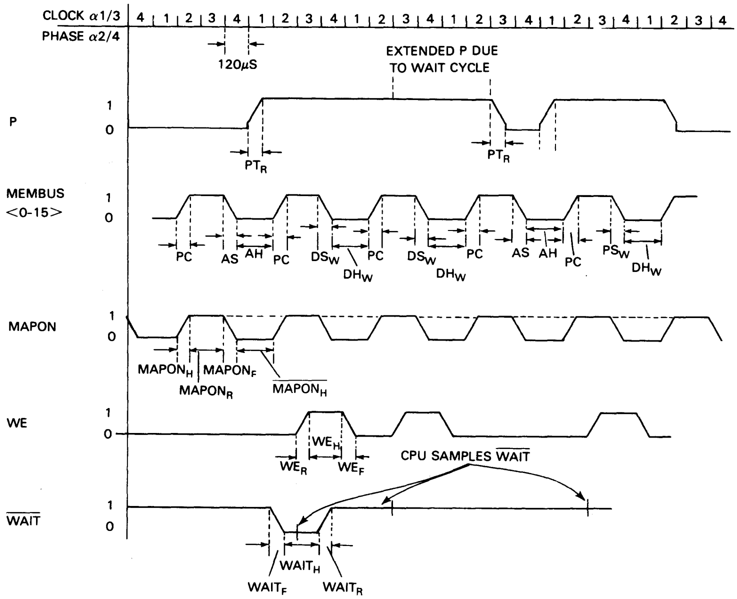
Write (Slow Memories)

Unmapped mN602 memory can contain slow memories (e.g. EPROMs). If the mN602 writes to a slow memory location, the slow memory drives $\overline{\text{WAIT}}$ low. The CPU samples $\overline{\text{WAIT}}$ on the falling edge of $\alpha 2$. If $\overline{\text{WAIT}}$ is low, the CPU knows that memory did not accept the data on the memory bus. The CPU transmits another WE to the memory bus transceivers.

The memory bus is precharged during the following $\alpha 2$ phase. On the falling edge of this $\alpha 2$ the CPU again samples $\overline{\text{WAIT}}$.

If $\overline{\text{WAIT}}$ was high (slow memory required only one wait cycle), the falling edge of WE signifies that valid write data are on the bus. Since $\overline{\text{WAIT}}$ was high the last time the CPU sampled it, the CPU knows that memory has accepted the data. The valid data remain on the bus until the bus is precharged during the next $\alpha 2$ phase. P drops on the falling edge of this $\alpha 2$.

MEMORY WRITE TIMING DIAGRAM WITH WAIT CYCLE



DG-08145

NOTE: See Timing Tables near the end of this chapter.

The figure illustrating the use of wait cycles shows memory keeping $\overline{\text{WAIT}}$ low for one wait cycle.

The CPU can tolerate up to 50 wait cycles. More than 50 wait cycles on one memory reference halts the CPU.

In the figure, $\overline{\text{WAIT}}$ goes low during $\alpha 2$. The CPU samples it on the falling edge of $\alpha 2$ and consequently finds it low. Because $\overline{\text{WAIT}}$ was low when the CPU sampled it, the CPU sent out an extra WE signal. On the falling edge of the next $\alpha 2$, the CPU again samples $\overline{\text{WAIT}}$ and this time finds it high.

The falling edge of the second WE signal signifies that valid write data are on the bus. The second WE signal is the one that signifies valid data because $\overline{\text{WAIT}}$ was high the last time the CPU sampled it. P drops on the falling edge of the next $\alpha 2$.

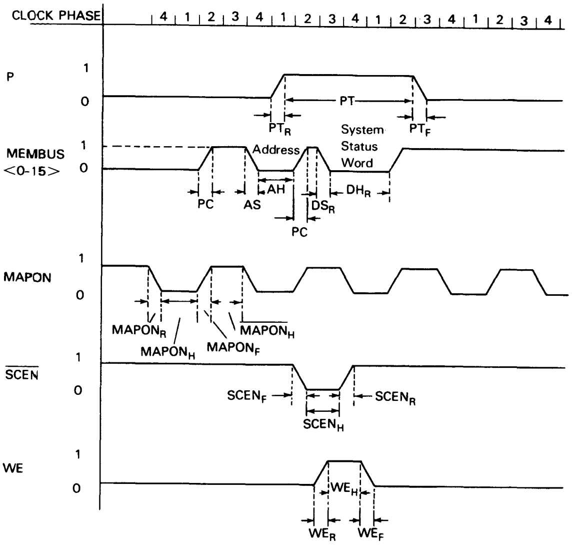
Refresh

On the rising edge of P, memory tests MB0. If MB0 is a 1, memory knows that a refresh is occurring.

P rises during $\alpha 1$, telling memory that a valid address is on the memory bus. During the following $\alpha 2$ phase, the CPU drives $\overline{\text{SCEN}}$ low. $\overline{\text{SCEN}}$ strobes the System Status Word onto the memory bus.

During the following $\alpha 3$ phase, the CPU drives WE high. The falling edge of $\alpha 3$ strobes the System Status Word into an internal CPU register. The System Status Word is not program accessible.

MEMORY REFRESH TIMING DIAGRAM



DG-08146 **NOTE:** See Timing Tables near the end of this chapter.

HIGH SPEED DATA CHANNEL

The mN602 contains logic that synchronizes high speed data channel requests and provides the memory control signals that enable data transfers between a high speed data channel device and memory via the memory bus. When the CPU is not referencing memory, it allows a high speed data channel device to use the memory bus.

The mN602 puts out $\overline{\alpha 2}$ as a request enable for high speed data channel requests. When a device requests the high speed data channel, it sets its own high speed data channel request flag. Then the rising edge of the next $\overline{\alpha 2}$ causes $\overline{\text{FDCHR}}$ to go down. The CPU samples $\overline{\text{FDCHR}}$ during an $\alpha 4$ phase. If the CPU is using the memory bus, the high speed data channel request remains pending. When the CPU finishes with memory, it asserts FDCHE at the rising edge of the next $\alpha 2$, thus enabling high speed data channel transfers.

While high speed data channel transfers are going on, the CPU does not perform memory references. It stops the process of the current instruction and ignores its memory bus inputs.

By driving FDCHE high the CPU allows the highest priority high speed data channel device to gain control of the memory bus. The device closest to the CPU along the priority chain has the highest priority. The priority chain is created by gating the $\overline{\text{DCHR}}$ line through each I/O interface. When the device drives $\overline{\text{DCHR}}$ low, it locks out any other devices below it on the priority chain. A high speed data channel device can lock out not only the CPU and lower priority fast data channel devices, but also all normal I/O devices.

During the time that FDCHE is high the high speed data channel device that has priority indicates the direction of the upcoming transfer to the CPU. The CPU samples $\overline{\text{FDCHR}}$ (now considered $\overline{\text{FDCHI}}$) during the $\alpha 4$ phase following the assertion of FDCHE. When $\overline{\text{FDCHI}}$ is high, it indicates a data out transfer (the device reads from memory). When $\overline{\text{FDCHI}}$ is low, it indicates a data in transfer (the device writes to memory).

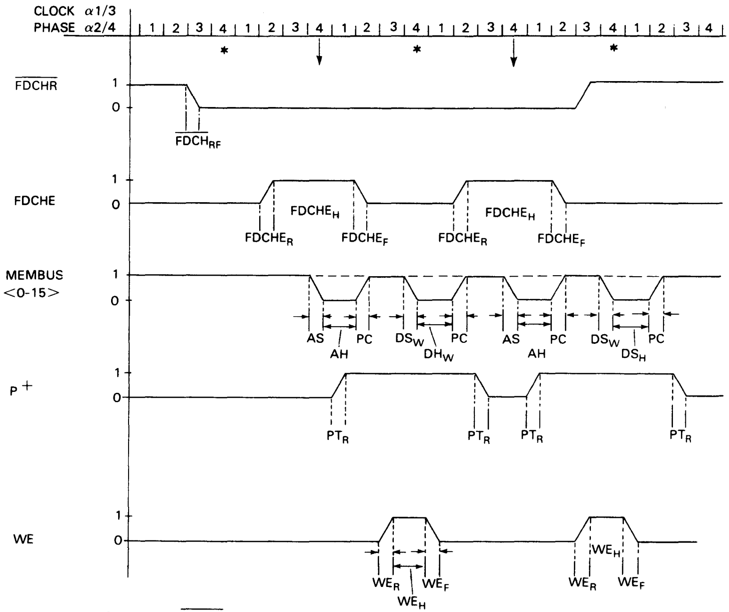
While the memory bus is being precharged in preparation for the address of the upcoming transfer, FDCHE goes low. Now the CPU issues the proper control signals for a memory read or write. Control line P goes high to signify a valid memory address and falls when the memory transfer is completed. The CPU issues the appropriate WE or SAE signals.

High Speed Data Channel Memory Access

The high speed data channel cannot access slow memories. If $\overline{WAIT}$ goes low during a transfer, the CPU will halt. $\overline{WAIT}$ being low before a transfer occurs, does not cause the CPU to halt.

The high speed data channel is intended to access only unmapped memory. The CPU drives MAPON low during high speed data channel activity.

HIGH SPEED DATA CHANNEL DEVICE TO MEMORY TIMING DIAGRAM



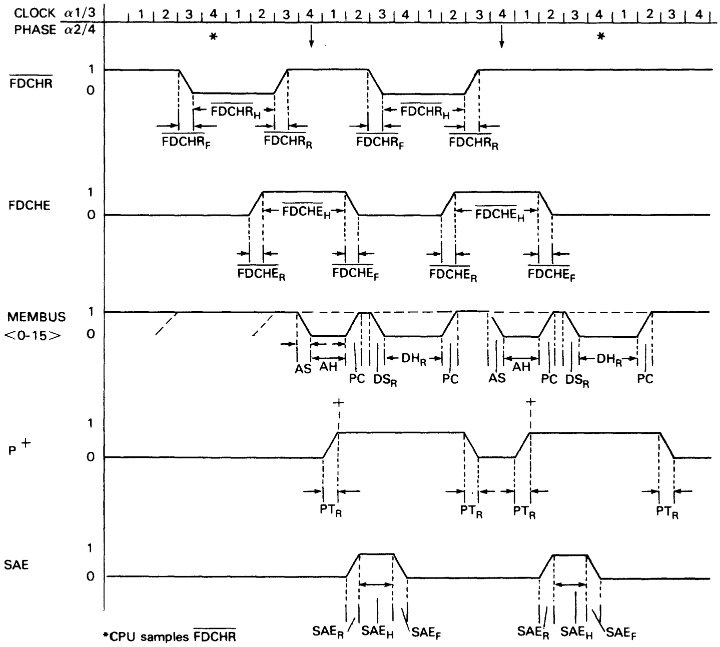
*CPU samples $\overline{FDCHR}$
↓CPU samples direction
+P rising edge denotes a valid address or the memory bus

NOTE: See Timing Tables near the end of this chapter.

Refresh during High Speed Data Channel Activity

When 16 μs have elapsed since the last refresh, the mN602 can take the memory bus away from the high speed data channel to perform a demand refresh. Transparent refreshes are not performed during high speed data channel activity. After the CPU performs the demand refresh, it returns to program execution. If the device still requires the high speed data channel, the device must request it again.

MEMORY TO HIGH SPEED DATA CHANNEL DEVICE TIMING DIAGRAM



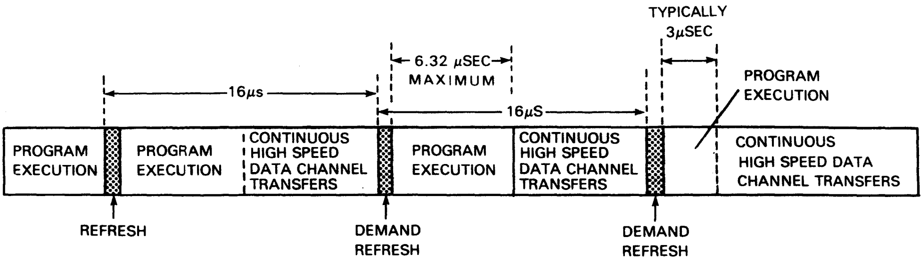
*CPU samples $\overline{\text{FDCHR}}$

| CPU samples direction

+P rising edge denotes a valid address on the memory bus

DG-08148 **NOTE:** See Timing Tables near the end of this chapter.

HIGH SPEED DATA CHANNEL SEQUENCE OF EVENTS



DG-08149

I/O OPERATIONS

Data are passed between the mN602 and the I/O controllers connected to the microNOVA I/O bus via programmed I/O or the standard data channel. The CPU is switched to one of these modes of transfer at the request of the I/O controllers, through the use of the program interrupt and data channel request facilities.

PROGRAMMED I/O

In programmed I/O, the CPU moves information, consisting of a command followed by data, between its accumulators and the I/O bus. There is one exception to this rule; that is when an I/O Skip instruction is executed. The data received in response to this command contains the state, 0 or 1, of the addressed device's Busy and Done flags. This information is not loaded into an accumulator. It is acted upon directly by the CPU. Using this information together with a skip command (e.g., Skip If Done is Non-Zero), the CPU determines whether the test condition specified in the command is true or false. If it is true, the program counter is incremented an additional time.

STANDARD DATA CHANNEL

Using the standard data channel, the CPU moves information between the I/O bus and memory without altering the program flow. This information transfer is transparent to the executing program. When the CPU honors a data channel request, it sends an acknowledgement to the I/O bus. This is responded to by the I/O controller with the highest priority requesting a data channel break. The I/O controller sends to the CPU the appropriate 15-bit memory address and a direction of transfer mode bit (0 equals data out; 1 equals data in). All references to direction of transfer are always in relation to the CPU; that is, data in to the CPU, data out from the CPU. When the CPU receives the address, it begins a read operation using the address received from the I/O controller. Then, it examines the mode bit.

If the mode bit is set for a data out, the information received by the CPU during the read portion of the memory operation is sent to the I/O bus. The write portion of the memory operation is not executed.

If the mode bit is set for a data in, the CPU starts a write operation. The CPU switches to I/O input mode and then waits a certain period of time to receive the data. The 16 data bits received from the I/O controller during this time period are sent to the memory bis and are written into memory during the write portion of the memory operation being performed. The information received by the CPU during the read portion of the memory operation is ignored.

I/O BUS TRANSFERS

Information is passed in serial form between the CPU and the I/O bus via the CPU I/O bus pins. The pins and their functions are listed below.

I/O SIGNALS

PIN MNEMONIC	FUNCTION
I/O DATA 1	Bidirectional, data bus. This line carries the 8 most significant bits (0-7) of a 16-bit word in serial form, plus one control code prefix bit. (Code bits are explained below.)
I/O DATA 2	Bidirectional data bus. This line carries the 8 least significant bits (8-15) of a 16-bit word in serial form, plus one control code prefix bit.
I/O CLOCK	Bidirectional, synchronizing clock line. The I/O CLOCK strobes data into the receiving device at a rate of 8.33Mbits per line.
$\overline{\text{DCHR}}$	This is a wired-OR of all I/O controllers' data channel request lines. When asserted low by any of the I/O controllers, it tells the CPU that a device, or devices, is requesting a data channel break.
$\overline{\text{INTR}}$	This is a wired-OR of all I/O controllers' program interrupt lines. When asserted low by any of the I/O controllers, it tells the CPU that a device, or devices, is requesting a program interrupt.
I/O INPUT	This line indicates the operating mode of the mN602 to the CPU I/O transceiver connected to the I/O bus. When it is at the low logic level, it tells the transceiver that the CPU is sending information, and the transceiver should receive the information and pass it to the I/O bus. When the line is at a high logic level, it tells the transceiver that the CPU is ready to receive data, and the information on the I/O bus should be retrieved and sent to the CPU. Except for the intervals during which the CPU is transmitting information, this line remains in the high logic state.

The I/O data lines (I/O DATA<1-2>) are used to transfer four types of information: Request Enable, Data Channel Address Request, programmed I/O commands, and data. Each type consists of either 2 bits or 18 bits (1 or 9 bits on each I/O data line) and is identified by the first bit transmitted on each line. The encoding is shown below.

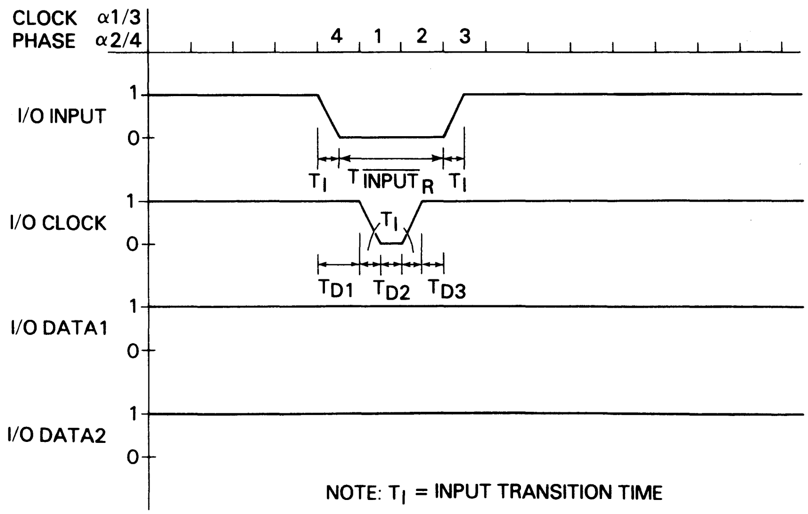
TRANSFERS VIA I/O DATA LINES

FIRST BIT CODE I/O DATA 1	FIRST BIT CODE I/O DATA 2	I/O TRANSFER TYPE	TOTAL NO. OF BITS TRANSFERRED
1	1	Request Enable	2 code bits only
1	0	Data Channel Address Request	2 code bits only
0	1	Data	18 bits (2 code bits plus 16 data bits)
0	0	I/O Command	18 bits (2 code bits plus 16 command bits)

REQUEST ENABLE

This is a signal sent by the CPU, in form, to all devices connected to the I/O bus. It is sent between instructions and at other irregular intervals. It is used by the individual I/O controllers to synchronize the assertion of program interrupt and data channel request lines. Request Enable ensures that the $\overline{\text{INTR}}$, $\overline{\text{DCHR}}$, and priority lines are at a steady state when they are examined by the CPU.

REQUEST ENABLE TIMING DIAGRAM

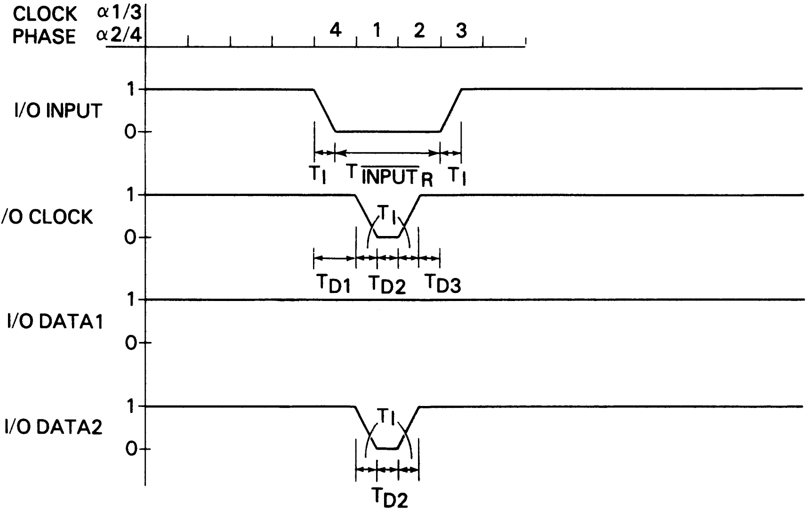


DG-06025

DATA CHANNEL ADDRESS REQUEST

This is a signal sent by the CPU, in code form, to the I/O bus in response to a data channel request. It tells the I/O controllers that the CPU is ready to receive the data channel address and mode bit. It is responded to by the highest priority device requesting a data channel break. Data Channel Address Request also acts as a Request Enable.

DATA CHANNEL ADDRESS REQUEST TIMING DIAGRAM



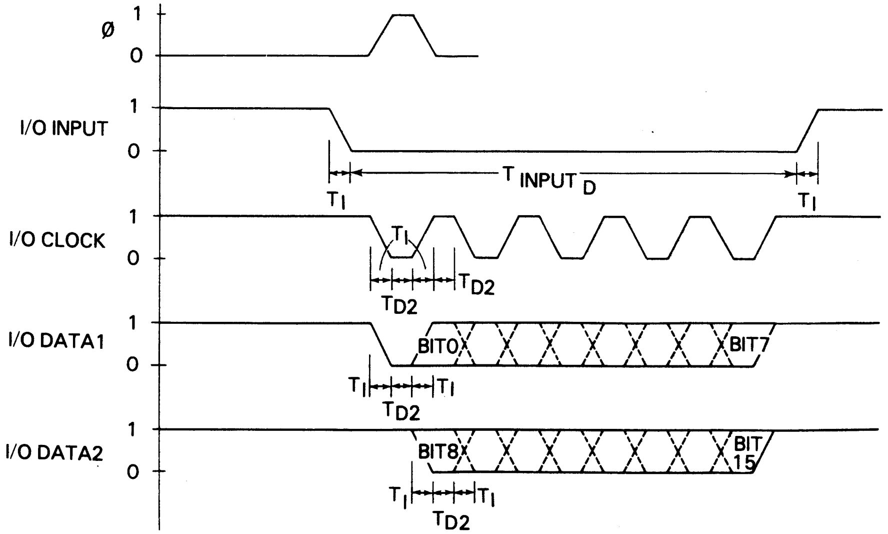
DG-06026

NOTE: See Timing Tables near the end of this chapter.

DATA

The format shown in the Data Timing Diagram is used to transfer 16 data bits on the I/O bus. In programmed I/O, data is any 16-bit transfer which is not an I/O command. In a data channel break, this format is used to transfer the 15-bit memory address plus mode bit and the 16-bit word.

DATA TIMING DIAGRAM



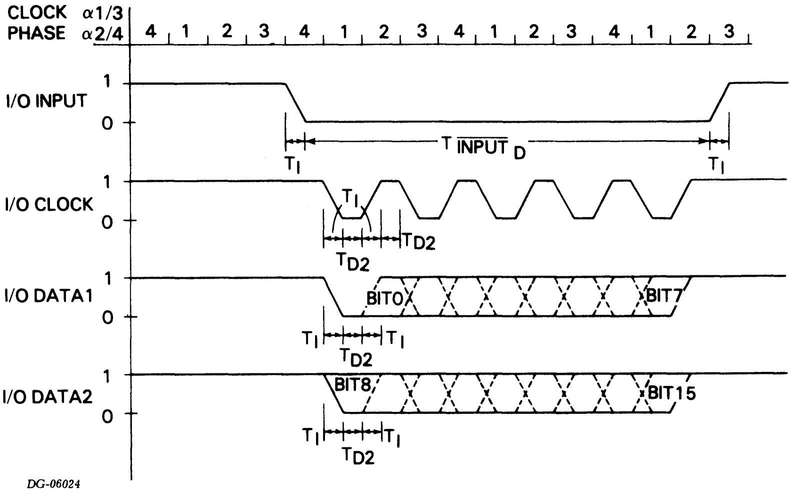
DG-06023

NOTE: See Timing Table near the end of this chapter.

I/O COMMAND

The format shown in the I/O Command Timing Diagram is used by the CPU to transfer all instructions to the I/O bus in which bits 0-2 contain 011 (I/O instruction format).

I/O COMMAND TIMING DIAGRAM *



DG-06024

* UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.

FOR TIMING INTERVALS, SEE TIMING TABLE ON PAGE 68.

RULE: 00, 01_8, 02_8, 03_8, and 77_8 are ILLEGAL DEVICE CODES for any I/O device connected to the microNOVA I/O bus.

The mN602 Instruction Set section describes several groups of instructions which incorporate the I/O format: I/O, CPU I/O, Multiply/Divide, and Stack. Each of these groups is discussed below.

Each I/O instruction should contain a device code in bits 10-15 which specifies one of the existing devices connected to the I/O bus. When one of these instructions is sent to the I/O bus by the CPU, each mN603 I/O controller examines the device code and determines what action, if any, should be taken. Most instructions in this group are followed by a data transfer, either from the CPU or the I/O controller. See mN601 Instruction Set.

Each CPU I/O instruction contains device code 77_8 (illegal device code). Some of these instructions are global in nature and are directed to all I/O controllers; others are acted upon internally by the CPU. The latter should be ignored by all I/O controllers. The effect of each is described in the CPU I/O instructions section of the mN602 Instruction Set.

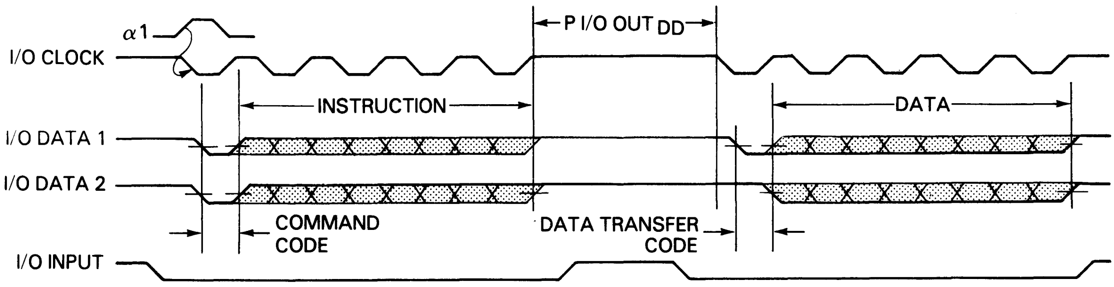
The Multiply and Divide instructions and the Stack instructions address device code 01_8 (illegal device code). Because these instructions are internal to the CPU, the instruction is not sent to the I/O bus when executed; and no data is transferred. I/O controllers should ignore these instructions.

I/O TRANSFER PROTOCOL

Programmed I/O Data Out

The programmed data out timing diagram depicts a transfer of an I/O instruction followed by a data transfer from the CPU; e.g., DOA, DOB, DOC, MSKO. After the instruction is sent to the I/O bus, the CPU waits for a fixed interval of time before sending the data.

PROGRAMMED DATA OUT TIMING DIAGRAM *



DG-04018

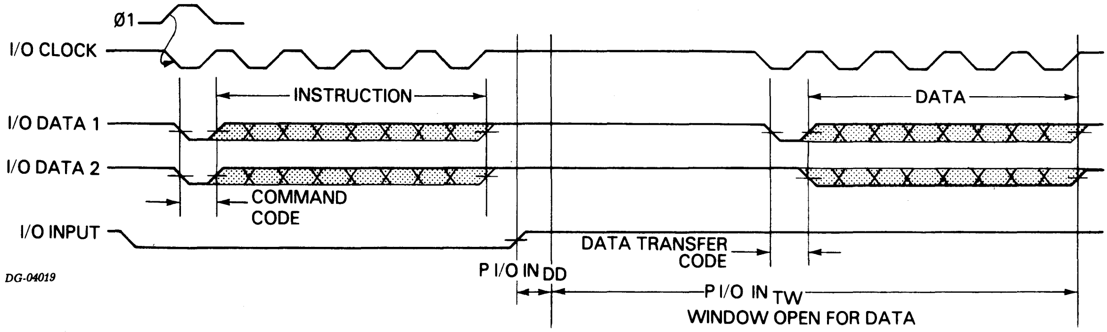
* UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.

NOTE: See Timing Tables near the end of this chapter.

Programmed I/O Data In

The programmed data in the timing diagram depicts a transfer of an I/O instruction followed by a data transfer from an IOC; e.g., DIA, DIB, DIC, INTA, I/O SKIP. After the CPU sends the instruction to the I/O bus, it switches to I/O input mode and opens a window for the reception of a data word for a fixed interval of time. It is expected that the I/O controller addressed by the instruction will send 16 data bits plus the control bits during this time frame. When the window closes, the CPU performs the command specified in the instruction using the data bits received.

PROGRAMMED DATA IN TIMING DIAGRAM *



DG-04019

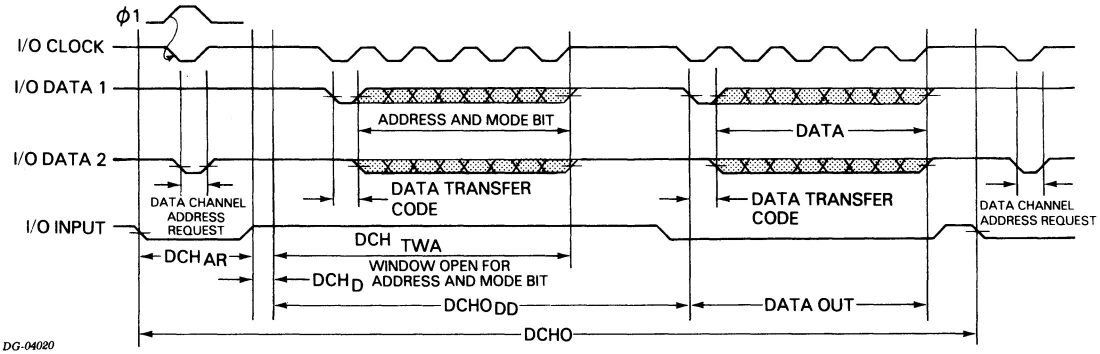
* UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.

NOTE: See Timing Table near the end of this chapter.

Data Channel Data Out

The sequence of events from the time the CPU acknowledges a data channel request through the time the data is placed on the I/O bus is depicted in the timing diagram below. After the CPU sends a Data Channel Address Request to the bus, it switches to data channel input mode. At this time the CPU opens a data-in window for a fixed interval of time. It is expected that the I/O controller with the highest priority requesting a data channel break will send the memory address and the direction mode bit during this time frame. When the window closes, the memory address contained in the data transfer is used by the CPU in performing a memory read operation. When the data is retrieved from memory, the CPU transfers it to the I/O bus by performing a data out operation.

DATA CHANNEL MEMORY TO DEVICE TIMING DIAGRAM *



DG-04020

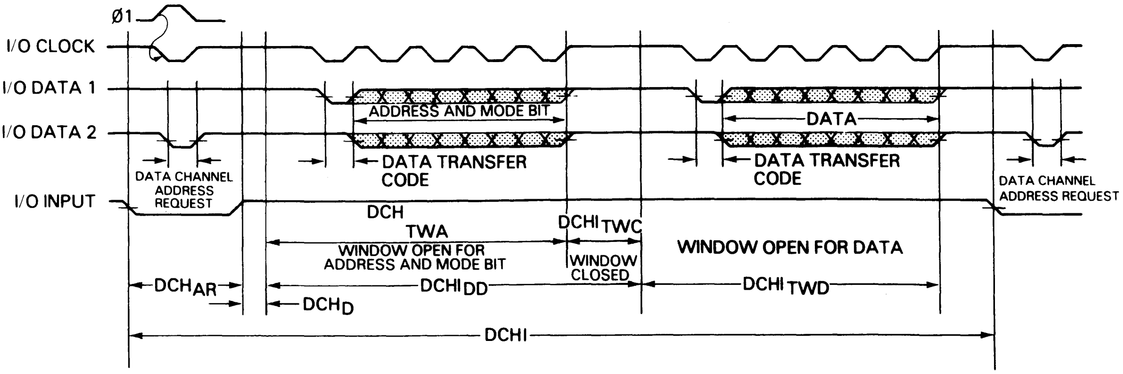
* UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.

NOTE: See Timing Tables near the end of this chapter.

Data Channel Data In

The sequence of events from the time the CPU acknowledges a data channel request through the time the data is received by the CPU from the I/O bus is depicted in the timing diagram below. After the CPU sends a Data Channel Address Request, it switches to data channel input mode. Then, it opens the data-in window twice for fixed intervals of time. The first time is to retrieve the memory address and mode bit. The second time it is expected that the I/O controller will send one 16-bit word plus the control bits. When the window closes, the data bits received are sent to the memory bus and a memory write operation is performed.

DATA CHANNEL DEVICE TO MEMORY TIMING DIAGRAM *



DG-04021

* UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.

NOTE: See Timing Table near the end of this chapter.

LATENCY

A system that depends heavily on slow memories and a number of I/O controllers can be overloaded. This means that peripherals and memories may lose data, since the system cannot respond in time to requests for service. Hence, it is important to understand the timing constraints of the system you are using.

MEMORY LATENCY

Memory latency is the time delay between the start of a memory reference (the CPU sets P) and the placement of valid data on the memory bus. This interval normally consists of one full CPU cycle (480 ns.)

Another delay is introduced when a slow memory is accessed. Each $\overline{\text{WAIT}}$ signal, generated while the memory bus is being precharged and set up, forces the CPU to generate another WE or SAE signal one CPU cycle later. The CPU tolerates up to 50 $\overline{\text{WAIT}}$ requests from memory for one memory reference. This creates an absolute maximum delay of 24 μs. However, most accesses are made with one $\overline{\text{WAIT}}$.

NOTE: More than 50 $\overline{\text{WAIT}}$ signals to the CPU on one memory reference will halt the CPU and require a processor reset.

Slow memories can be used in mapped and unmapped space. Slow memories cannot be addressed by the high speed data channel.

Memory latency = memory reference time + the number of wait cycles.

Memory latency = CPU cycles, typically.

HIGH SPEED DATA CHANNEL LATENCY

High speed data channel latency is the delay between the request for service by the device and the placement of valid data on the memory bus. It is composed of

1. The time from when the high speed data channel device sets its own request flag to when it gets a Request Enable from the CPU. Request Enables for high speed data channel requests are $\overline{\alpha 2}$ signals. When the device gets an α2, it asserts $\overline{\text{FDCHR}}$.
2. The time from when $\overline{\text{FDCHR}}$ is asserted to when the CPU asserts FDCHE. If the CPU is making a memory reference when $\overline{\text{FDCHR}}$ goes down, it waits until the completion of the memory reference before asserting FDCHE.
3. The time from when the CPU asserts FDCHE to when valid data appears on the memory bus. This is typically 1.5 CPU cycles for a read from memory and 1.75 CPU cycles for a write to memory.

Programmed I/O Latency

Most peripherals operating under program control request processor time by setting their Done flags to 1. Whether the CPU determines that a device is requesting service by repeatedly checking the state of the flags using I/O Skip instructions or by means of program interrupts, there is a delay between the time the peripheral requests service (sets its Done flag to 1) and the time the CPU fetches the first instruction of the peripheral service routine. This delay is called programmed I/O latency.

When the program interrupt facility is not used, this time is dependent upon the following: frequency of the I/O Skip instructions and efficiency of the software servicing the peripherals; i.e., the time required by the peripheral service routine to transfer data to or from the peripheral and set the Done flag to 0.

When the program interrupt facility is used, I/O latency is defined as the sum of the following:

1. The time from the setting of the Done flag to 1 to the end of the instruction currently being executed by the CPU.

2. The time when CPU operation is suspended while data channel transfers are in progress. (See the following section.)

3. The time program interrupts are turned off, e.g., during the servicing of an interruput from another peripheral.

4. **The time from the setting of the Done flag to the time when the CPU issues a Request Enable.**

   The CPU may issue this Request Enable while completing the execution of the current instruction. If not, the CPU begins and completes the next instruction.

5. Interrupt cycle time, which is comprised of the following: 6 CPU cycles plus 2 cycles per indirect level, where 1 CPU cycle equals 0.480 μsec.

   Interrupt cycle time is the time required for the interrupt facility hardware to store the contents of the program counter plus 1 in location 0, and to simulate a Jump indirect instruction to location 1 (or to location 2 or 3 if the interrupt is generated by the CPU's real-time clock or stack overflow, respectively).

6. The time required by the interrupt handler to identify the peripheral and transfer control to that peripheral's service routine.

7. The time required by the service routine to transfer data to or from the peripheral and set Done to 0.

Items 1, 4, and 5 are machine dependent. Item 2 depends upon the number of data channel devices and their activity rate. Items 3, 6 and 7 are determined by the software and represent the bulk of the programmed I/O interrupt latency.

Additionally, if any higher priority program interrupt, i.e., stack overflow or real-time clock, has a pending interrupt, the time calculation is the same. The CPU always honors the interrupt with the highest priority.

Programmed I/O interrupt latency is important because a peripheral device that must wait too long to be serviced may suffer from loss of data.For this reason,the mN602 incorporates the priority interrupt facility.To minimize the loss of important data, the software priority levels should be assigned on the basis of the following considerations:

1. **The maximum allowable programmed I/O latency for each peripheral device.**

2. **The result of exceeding the maximum allowable programmed I/O latency for each peripheral device; i.e.,slowdown or data loss.**

Data Channel Latency

Latency is also a factor when data is transferred between peripheral devices and memory via the standard data channel. The CPU allows data channel devices to access memory only during predefined times. In addition, more than one peripheral may be waiting for service at any given time.

Data channel latency is defined as the time between the I/O devices's request for service (the DCH SYNC line is asserted low by the device - see mN613) and the time when data is strobed, either during data reception or transmission. Data channel latency consists of the sum of either of the following values:

Data Channel Data In (Device to Memory)
The time from when the data channel device sets its request flag to when the CPU issues a Request Enable followed by a Data Channel Address Request plus 6 CPU cycles, where 1 CPU cycle equals 0.480 sec.

Data Channel Data Out (Memory to Device)
The time from when the data channel device sets its request flag to when the CPU issues a Request Enable followed by a Data Channel Address Request plus 11 CPU cycles.

For example, if the two longest consecutive instructions in the program are 15 CPU cycles each, the maximum latency for a data channel data out transfer is:

$$15 + 15 + 11 = 41 \text{ CPU cycles;}$$
$$41 \times .480\mu\text{sec} = 19.68\mu\text{sec}$$

Additionally, there is a minimum data channel latency. The minimum latency for a data channel data in is 11 CPU cycles; the minimum for a data channel data out is 16 CPU cycles.

Minimum latency can be used in computing data channel latency when it is known that the device will request a data transfer. Thus, a data channel request can be initiated 11 CPU cycles (for a data channel data in) or 16 CPU cycles (for a data channel data out) before the device is ready to transfer data; this reduces the latency factor from data ready to data transfer.

The above computations assume the request is by the highest priority device requesting service.

When there are other, higher priority devices requesting data channel transfers, the time required for these transfers must be added to the latency. For each data channel out transfer the time is 7.2μsec and for each data channel in transfer the time is 5.8μsec.

STATUS

INITIALIZATION AND RESET

On power up, VBB must be -3.0V before VGG, VDD, and VCC go positve.

After mN602 is within operating specifications (see Electrical Specifications, DC Characteristics), the $\overline{\text{JUMEN}}/\overline{\text{CLAMP}}$ must be held low, i.e., input logic 0, for 100 μsec minimum. When the input is brough high, the CPU enters the HALT state. The contents of internal registers and RAM memory are indeterminate.

While in operation if $\overline{\text{JUMEN}}/\overline{\text{CLAMP}}$ goes low again, the CPU will reset and enter the HALT state on the rising edge of $\overline{\text{JUMEN}}/\overline{\text{CLAMP}}$. If $\overline{\text{JUMEN}}/\overline{\text{CLAMP}}$ is low for more than 100 μsec, the contents of RAM memory becomes indeterminate. (Memory refresh is disabled when $\overline{\text{JUMEN}}/\overline{\text{CLAMP}}$ is low). Another method of resetting the CPU is pulling I/O CLOCK low for 10 μsec. When I/O CLOCK goes high again, the CPU resets and enters the HALT state. This is the only way to reset the I/O device. It does not affect memory or registers.

NOTE: Do NOT pull I/O CLOCK low when I/O INPUT is low.

HALT

With the mN602 in the HALT state, the $\overline{\text{HALT}}$ line remains low until the CPU is restarted. While in the HALT state, the CPU generates Request Enable and refreshes memory. In generating Request Enables, the CPU responds to data channel requests. High speed data channel transfers now do not compete with program execution. They have full control of the memory bus except during refresh cycles. Generating a program interrupt restarts the CPU, ION, of course, must be 1 for this to occur.

SYSTEM STATUS WORD

The System Status Word is read by the CPU from the memory bus lines during the first $\alpha3$ phase of a memory refresh cycle. When the CPU drives $\overline{\text{SCEN}}$ low, the System Status Word should be placed on the memory bus. The falling edge of $\alpha3$ strobes it into the CPU. The System Status Word goes into an internal register and cannot be read by the program.

MEMORY BUS BITS	NAME	FUNCTION
0	$\overline{\text{NMR}}$	Falling edge generates a non-maskable interrupt. Must stay low until mN602 reads the system status word.
1	$\overline{\text{MAP EXISTS}}$	If low, all interrupts (normal and non-maskable) are disabled between the time when MAPON rises and the next INTRE instruction. If high, interrupts are not disabled.
2	$\overline{\text{MAP ON TRAP}}$	If low, a TRAP instruction sets MAPON to 1. The TRAP instruction traps to mapped space.
3	$\overline{\text{REBOOT}}$	If low, the CPU does a regular boot. $\overline{\text{LOCK}}$ must be low, and it must be a power up situation.
4	$\overline{\text{BOOTEN}}$	If low, then a boot occurs when mN602 halts. When a boot occurs, MAPON goes high and the mN602 jumps indirectly through mapped 77777$_8$.
5	$\overline{\text{LOCK}}$	If low, disables $\overline{\text{BOOT}}$ and $\overline{\text{CONT}}$ and enables $\overline{\text{REBOOT}}$. If high, enables $\overline{\text{BOOT}}$ and $\overline{\text{CONT}}$ and disables $\overline{\text{REBOOT}}$.
6	$\overline{\text{PWRFAIL}}$	Powerfail signal from power supply. If low, interrupts the CPU.
7	$\overline{\text{CONT}}$	If low, the mN602 begins execution at the address contained in the PC. The mN602 must be halted. The mN602 provides internal debouncing for this signal.
8	$\overline{\text{BOOT}}$	If low, the mN602 does a boot. The mN602 must be halted. The mN602 provides internal debouncing for this signal.
9	—	Reserved.
10	—	Reserved. Must be high.
11	$\overline{\text{JUMPER EXIST}}$	When low, the mN602 will not issue the signal $\overline{\text{JUMEN}}$. When high, the mN602 will issue $\overline{\text{JUMEN}}$.
12	$\overline{\text{EXTRTC}}$	When pulled low, disables the internal RTC. Must stay low until the mN602 reads the system status word. (The only way the internal RTC can be re-enabled is by powering up with $\overline{\text{EXTRTC high}}$.) When the internal RTC clock is disabled, an RTC interrupt occurs when the CPU detects a low to high transition for $\overline{\text{EXTRTC}}$. Must stay high until the mN602 reads the system word.
13	—	Reserved. Must be high.
14	—	Reserved. Must be high.
15	JUMPER POSITION	When $\overline{\text{JUMPER EXIST}}$ is low and JUMPER POSITION is high, the mN602 issues $\overline{\text{JUMEN}}$ when it addresses mapped location 77777$_8$. When $\overline{\text{JUMPER EXIST}}$ is low and JUMPER POSITION is low, the mN602 issues $\overline{\text{JUMEN}}$ when it addresses mapped location 77776$_8$.

POWER UP

When the mN602 is powered up, MAPON and ION go to 0, and the CPU enters the halt state. It can service the high speed and standard data channels.

The mN602 refreshes memory. When it refreshes memory, it reads the System Status Word.

$\overline{\text{LOCK}}$ High

If $\overline{\text{LOCK}}$ is high, the mN602 checks $\overline{\text{BOOTEN}}$. If $\overline{\text{BOOTEN}}$ is low, the mN602 begins the boot procedure.

It sets MAPON to 1 and stores the PC in mapped location 0. Then it checks $\overline{\text{JUMPER EXISTS}}$. If $\overline{\text{JUMPER EXISTS}}$ is 1, the mN602 jumps indirect through 77777 of mapped memory. If $\overline{\text{JUMPER EXISTS}}$ is 0, the mN602 will issue the signal $\overline{\text{JUMEN}}$.

The mN602 now looks at $\overline{\text{JUMPER POSITION}}$. If $\overline{\text{JUMPER POSITION}}$ is 1, the mN602 will issue $\overline{\text{JUMEN}}$ when addressing location 77777 of mapped memory. If $\overline{\text{JUMPER POSITION}}$ is 0,the mN602 will issue $\overline{\text{JUMEN}}$ when addressing location 77776 of mapped memory.

The mN602 now jumps indirect through 77777. If the mN602 does not issue $\overline{\text{JUMEN}}$, the returned address is the contents of location 77777 in mapped memory. If it issues $\overline{\text{JUMEN}}$, the mN602 forces bits 8-14 of the returned address to 0 and bit 15 to 1.

$\overline{\text{LOCK}}$ Low

If $\overline{\text{LOCK}}$ is low, the mN602 checks to see if it is in a power up situation. It does this by checking an internal flip flop. This internal flip flop always comes up in a known state.

If the mN602 has just received DC power, it checks $\overline{\text{REBOOT}}$. If $\overline{\text{REBOOT}}$ is high, the mN602 returns to a halt. If $\overline{\text{REBOOT}}$ is low, the the mN602 sets MAPON to 1 and continues with the normal boot procedure.

If this is not a power up situation (the mN602 is not just receiving DC power), it sets MAPON to 1 and continues with the normal boot procedure, except that when it fetches the contents of mapped location 77777, it forces bit 15 to 0.

DESIGN SUGGESTIONS

The signal $\overline{\text{LOCK}}$ is intended to be held low when a system is unattended. It is often connected to the LOCK switch on a front panel. When the LOCK switch is on, $\overline{\text{LOCK}}$ is low.

The signal $\overline{\text{BOOTEN}}$ is intended to be determined by a jumper. If it's low, an automatic boot occurs when the lock switch is off and the mN602 halts.

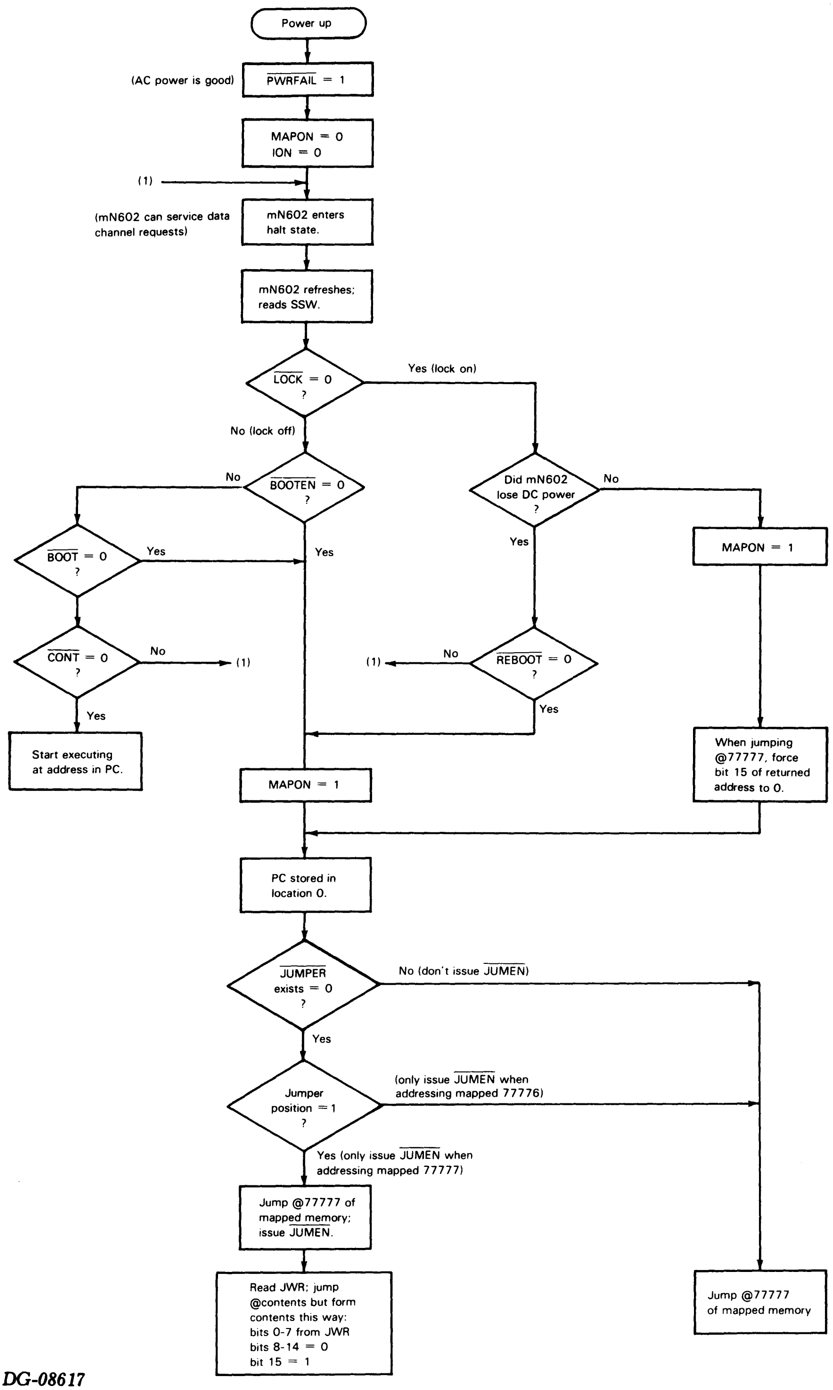
The signal $\overline{\text{BOOT}}$ is intended to come from a front panel switch. It allows you to do manually what $\overline{\text{BOOTEN}}$ does automatically.

The signal $\overline{\text{PWRFAIL}}$ is intended to come from the power supply and, when high, to signify the presence of AC power. If $\overline{\text{PWRFAIL}}$ is high, the mN602 is halted, and $\overline{\text{LOCK}}$ is low, this may represent an unattended system that has lost AC power, but has had it returned. The mN602 contains an internal flip flop that always comes up in a known state when the mN602 is powered up. By checking this flip flop, the mN602 can distinguish between these two cases:

1. AC power returned but the mN602 has lost DC power.
2. AC power returned but the mN602 has not lost DC power.

The signal $\overline{\text{JUMEN}}$ is intended to enable a jumper word register. If the mN602 addresses mapped 77777 or mapped 77776 and also issues $\overline{\text{JUMEN}}$, you can have the address come from a set of jumpers rather than from memory.

POWER-UP FLOWCHART


Power up
(AC power is good)
$\overline{\text{PWRFAIL}}$ = 1
MAPON = 0
ION = 0
(1)
(mN602 can service data channel requests)
mN602 enters halt state.
mN602 refreshes; reads SSW.
$\overline{\text{LOCK}}$ = 0 ?
Yes (lock on)
No (lock off)
No
$\overline{\text{BOOTEN}}$ = 0 ?
Did mN602 lose DC power ?
No
$\overline{\text{BOOT}}$ = 0 ?
Yes
Yes
Yes
MAPON = 1
$\overline{\text{CONT}}$ = 0 ?
No
(1)
(1)
No
$\overline{\text{REBOOT}}$ = 0 ?
Yes
Yes
Start executing at address in PC.
MAPON = 1
When jumping @77777, force bit 15 of returned address to 0.
PC stored in location 0.
$\overline{\text{JUMPER}}$ exists = 0 ?
No (don't issue $\overline{\text{JUMEN}}$)
Yes
Jumper position = 1 ?
(only issue $\overline{\text{JUMEN}}$ when addressing mapped 77776)
Yes (only issue $\overline{\text{JUMEN}}$ when addressing mapped 77777)
Jump @77777 of mapped memory; issue $\overline{\text{JUMEN}}$.
Read JWR; jump @contents but form contents this way:
bits 0-7 from JWR
bits 8-14 = 0
bit 15 = 1
Jump @77777 of mapped memory

DG-08617

ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS

Supply voltage, V_{BB}	-7V
Supply voltage, V_{CC}	7V
Supply voltage, V_{DD}	15V
Supply voltage, V_{GG}	17V
Input voltage, V_I	7V
Operating free-air temperature range, T_A	O deg. to 70 deg.C
Storage temperature range, T_{STG}	-55 deg. to 125 deg.C

NOTE: All voltages are referenced to ground (Vss). Subjecting a circuit to conditions either outside of or at these limits for an extended period of time may cause irreparable damage to the circuits. Hence, these circuits are not intended to be operated at maximum ratings.

RECOMMENDED OPERATING CONDITIONS

Supply voltage, V_{BB}	-5V ± 0.5V
Supply voltage, V_{CC}	5V ± 0.25V
Supply voltage, V_{DD}	12V ± 0.6V
Supply voltage, V_{GG}	12V ± 0.6V
Operating free air temperature range, T_A	0 deg. to 70 deg. C

DC CHARACTERISTICS

SYMBOL	CHARACTERISTICS	CONDITIONS	MIN	TYP	MAX	UNITS
V_{IL}	α1, α2 (clock inputs)		-0.5	0	0.5	V
	I/O CLOCK		-0.5	0.25	0.5	V
	I/O DATA1, I/O DATA2		-0.5	0.25	0.5	V
	$\overline{\text{DCHR, INTR}}$		-0.8	0.25	0.8	V
	$\overline{\text{WAIT, FDCHR}}$		-0.8	0.25	0.8	V
	$\overline{\text{JUMEN/CLAMP}}$		-0.8	0.25	0.8	V
	MB<0-15>		-0.8	0.25	0.8	V
V_{IH}	α1, α2 (clock inputs)		11.4	12.0	12.6	V
	I/O CLOCK		2.7	5.0	5.8	V
	I/O DATA1, I/O DATA2		2.7	5.0	5.8	V
	$\overline{\text{DCHR, INTR}}$		4.25	5.0	5.8	V
	$\overline{\text{WAIT, FDCHR}}$		4.25	5.0	5.8	V
	$\overline{\text{JUMEN/CLAMP}}$		4.25	5.0	5.8	V
	MB<0-15>		4.25	5.0	5.8	V
I_{IL}*	α1, α2 (clock inputs)	@V_{IL} = 0.5V			2.0	mA
	MB<0-15>				-3.0	mA
	$\overline{\text{JUMEN/CLAMP}}$				-3.0	mA
	I/O CLOCK				-8.0	mA
	I/O DATA1, I/O DATA2				-8.0	mA
	$\overline{\text{DCHR, INTR}}$				-8.0	mA
	$\overline{\text{WAIT, FDCHR}}$				-8.0	mA

DC CHARACTERISTICS (CONT.)

SYMBOL	CHARACTERISTICS	CONDITIONS	MIN	TYP	MAX	UNITS
I_{IH}	α1, α2 (clock input)**					
	MB<0-15>	@V_{IH}= 4.5V	-60			uA
	$\overline{\text{INTR}}$		-100			uA
	I/O CLOCK	@V_{IH}= 2.7V	-100			uA
	I/O DATA1, I/O DATA2		-100			uA
	$\overline{\text{DCHR, WAIT}}$	@V_{IH}= 4.5V	-100			uA
	$\overline{\text{FDCHR}}$		-100			uA
	$\overline{\text{JUMEN/CLAMP}}$		-60			uA
VPCHS	MB<0-15> (VOLTAGE AFTER PRECHARGE)	IPCHS= 60ua	4.25	4.75	5.25	V
IPCHS	MB<0-15> (CURRENT AFTER PRECHARGE)***	VPCHS=4.25V			60	uA
V_{OL}	MB<0-15>,P,SAE,WE	@LOAD CURRENTS			0.6	V
	I/O CLOCK $\overline{\alpha 2}$				0.6	V
	I/O DATA1, I/O DATA2				0.6	V
	$\overline{\text{HALT}}$, MAPON, FDCHE				0.6	V
	$\overline{\text{SCEN}}$, I/O INPUT				0.6	V
	$\overline{\text{JUMEN/CLAMP}}$				0.6	V
V_{OH}	MB<0-15>,P,SAE,WE	@LOAD CURRENTS	2.7			V
	I/O CLOCK $\overline{\alpha 2}$		2.7			V
	I/O DATA1, I/O DATA2		2.7			V
	$\overline{\text{HALT}}$, MAPON, FDCHE		2.7			V
	$\overline{\text{SCEN}}$, I/O INPUT		2.7			V
	$\overline{\text{JUMEN/CLAMP}}$		2.7			V
I_{OL}	MB<0-15>,P,SAE,WE	@VOL= 0.6V	2.0			mA
	I/O CLOCK $\overline{\text{SCEN}}$		2.0			mA
	I/O DATA1, I/O DATA2		2.0			mA
	$\overline{\text{HALT}}$, MAPON, FDCHE		2.0			mA
	$\overline{\text{JUMEN/CLAMP}}$		2.0			mA
	I/O INPUT		4.0			mA
	$\overline{\alpha 2}$		4.5			mA
I_{OH}	MB<0-15>,P,SAE,WE	@VOH= 2.7V			-100	uA
	I/O CLOCK $\overline{\text{SCEN}}$				-100	uA
	I/O DATA1, I/O DATA2				-100	uA
	$\overline{\text{HALT}}$, MAPON, FDCHE				-100	uA
	$\overline{\text{JUMEN/CLAMP}}$				-100	uA
	I/O INPUT				-200	uA
	$\overline{\alpha 2}$				-300	uA
I_{BB}	Average				-250	uA
I_{CC}	Average				10	mA
I_{DD}	Average				100	mA

* Internal devices will attempt to make $V_{IL}=0$.

** Clocks tested for 10uA static leakage.

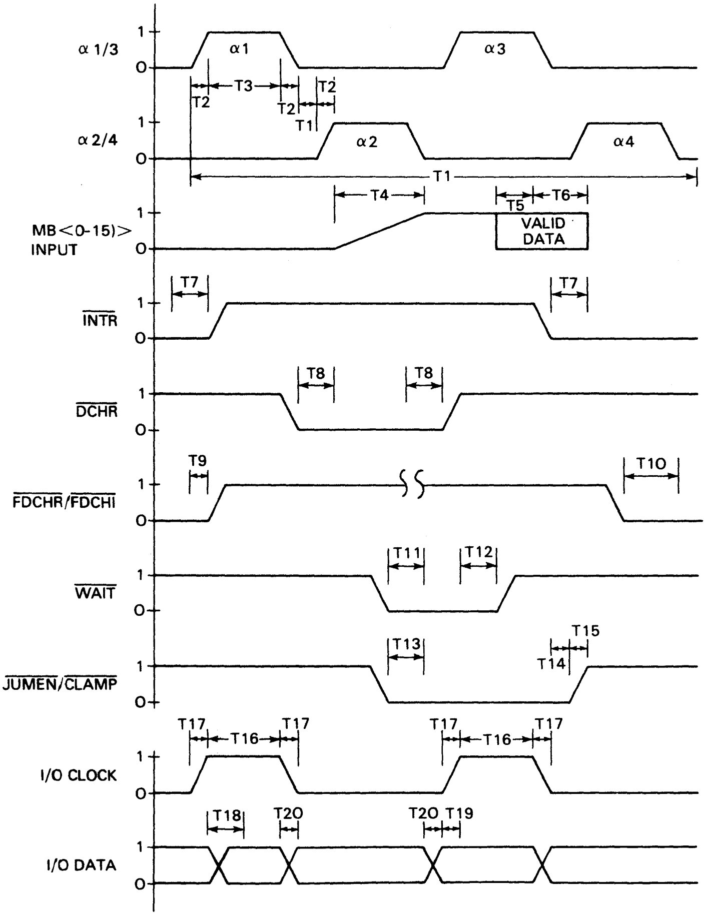
*** Precharge guaranteed by checking VOH on MB lines.

CAPACITANCE

Clock line ($\alpha 1, \alpha 2$) capacitance (input)	100 pf
Capacitance of all other signal pins (inputs)	10 pf
External load capacitance (output drive capability)	
MB0-15; I/O CLOCK; I/O DATA1; I/O DATA2; P; SAE;	20 pf
WE; I/O INPUT; $\overline{\text{HALT}}$; $\overline{\alpha 2}$; MAPON;	20 pf
$\overline{\text{SCEN}}$; FDCHE; $\overline{\text{JUMEN/CLAMP}}$	30 pf
External load capacitance between MB lines	10 pf

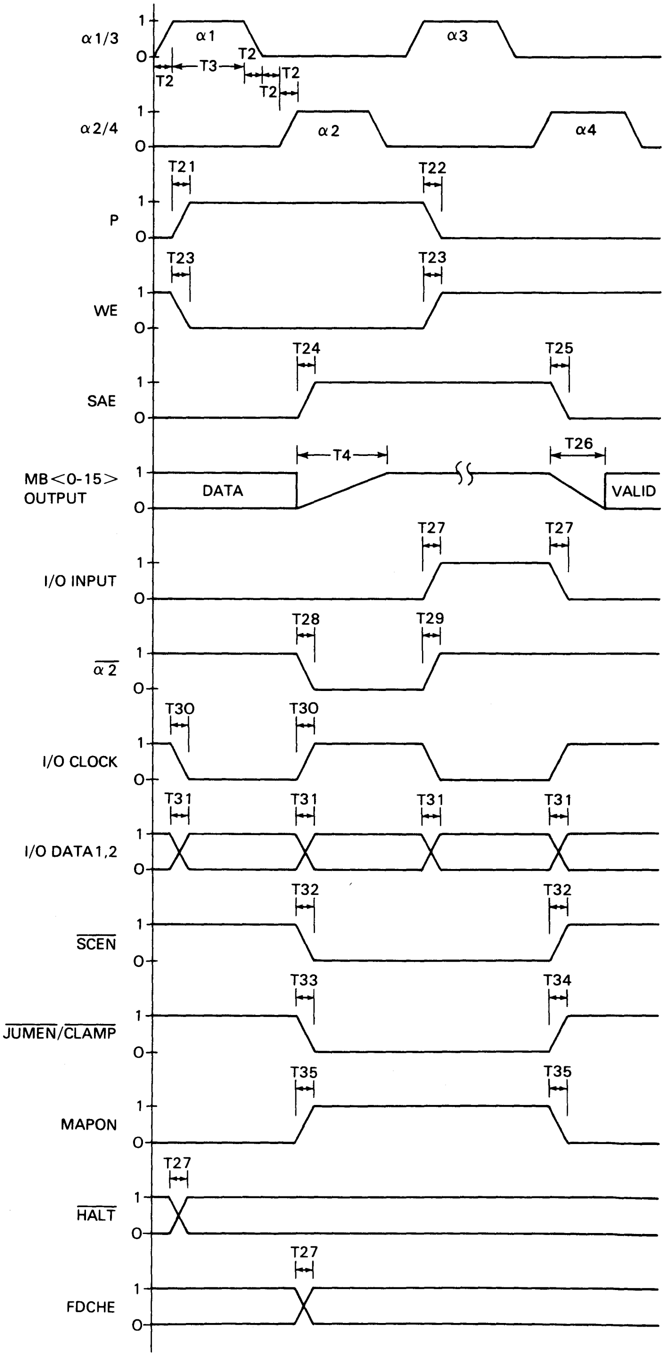
NOTE: Positive currents are sunk by device. Negative currents are sourced by device.

AC INPUT TIMING CHARACTERISTICS



DG-06028

AC OUTPUT TIMING CHARACTERISTICS



DG-06029

AC TRANSITION TIMING

			TIMING (NS)		
DIRECTION	SYMBOL	FUNCTION	MIN	TYP	MAX
INPUT	T1	Clock period, CPU cycle.	470	480	490
	T1'*	Clock separation	5	10	15
	T2*	Clock rise, fall time	5	10	20
	T3	Clock pulse time	75	90	105
	T4	MB<0-15> precharge time		60	80
	T5	MB<0-15> set-up time	40		
	T6	MB<0-15> hold time	10		
	T7	$\overline{\text{INTR}}$ set-up and hold	10		
	T8	$\overline{\text{DCHR}}$ set-up and hold	10		
	T9	$\overline{\text{FDCHR/FDCHI}}$ hold	10		
	T10	$\overline{\text{FDCHR/FDCHI}}$ set-up	30		
	T11	$\overline{\text{WAIT}}$ set-up time	30		
	T12	$\overline{\text{WAIT}}$ hold time	10		
	T13	$\overline{\text{JUMEN/CLAMP}}$ set-up	30		
	T14	$\overline{\text{JUMEN/CLAMP}}$ hold	0		
	T15	$\overline{\text{JUMEN/CLAMP}}$ precharge time	0	80	
	T16	I/O CLOCK pulse width	100		
	T17	I/O CLOCK rise, fall time	0	10	
	T18	I/O DATA lags I/O CLOCK	0	5	
	T19	I/O DATA leads I/O CLOCK	0	15	
	T20	I/O DATA rise, fall time	0	10	
OUTPUT	T21	P true state delay	0	60	
	T22	P false state delay	0	30	
	T23	WE true, false delay	0	30	
	T24	SAE true state delay	0	60	
	T25	SAE false state delay	0	30	
	T26	MB<0-15> data delay	0		80
	T27	I/O INPUT true, false state delay $\overline{\text{HALT}}$ delay, FDCHE delay	0		30
	T28	$\alpha 2$ false state delay	0		20
	T29**	$\overline{\alpha 2}$ true state delay	0		30
	T30	I/O CLOCK true, false state delay	0		30
	T31	I/O DATA true, false state delay	0		30
	T32	$\overline{\text{SCEN}}$ true, false state delay	0		30
	T33	$\overline{\text{JUMEN/CLAMP}}$ false state delay	0		30
	T34	$\overline{\text{JUMEN/CLAMP}}$ true state delay	0		60
	T35	MAPON true, false state delay	0		30

*Assumes cycle time of 480 ns. Both clocks may be turned off for 20 ns.

**$\overline{\alpha 2}$ may become true after the end of $\alpha 2$ due to clock separation.

NOTE: All timing points are measured at 10% and 90%.

TIMING TABLES

MEMORY REFERENCE TIMING

SYMBOL	DESCRIPTION	TIMING (NS) TYPICAL
PT_R	Rising edge of P delay.	60
PT	Memory reference period	660
PT_F	Falling edge of P delay.	30
$\overline{\text{PT}}$	Minimum P low period between memory references.	210
PC	Memory bus precharge time.	80
A_S	Address set up time.	80
A_H	Address hold time.	160
DS_R	Data setup for a read.	40
DH_R	Data hold for a read.	200
DS_W	Data setup for a write	80
DH_W	Data hold for a write	160
MAPON_R	32K-word bank select bit rise time delay.	30
MAPON_H	Bank select hold time.	210
$\overline{\text{MAPON}}_H$		210
MAPON_F	Bank select fall time delay.	30
SAE_R	CPU read pulse rise time delay.	60
SAE_H	CPU read hold time.	180
SAE_F	CPU read pulse fall time delay.	30
$\overline{\text{WAIT}}_F$	Memory delay request fall time delay.	30
$\overline{\text{WAIT}}_H$	Memory delay request hold time.	210
$\overline{\text{WAIT}}_R$	End of delay request, rise time delay.	30
WE_R	Write enable rise time delay.	30
WE_H	Write enable hold time.	210
WE_F	Write enable fall time, valid data on memory bus, delay.	30
$\overline{\text{SCEN}}_F$	System Status Word strobe fall time delay.	30
$\overline{\text{SCEN}}_H$	System Status Word hold time.	210
$\overline{\text{SCEN}}_R$	Rising edge strobes status word delay.	30
$\overline{\text{FDCHR}}_F$	Fast data channel request fall time.	30
$\overline{\text{FDCHR}}_H$	Fast data channel request hold time.	450
$\overline{\text{FDCHR}}_R$	Fast data channel request rise time.	30
$\overline{\text{FDCHI}}_R$	FDCH direction bit rise time delay.	30
$\overline{\text{FDCHI}}_H$	FDCH direction bit hold time.	90
$\overline{\text{FDCHE}}_F$	FDCH enable fall time delay.	30
$\overline{\text{FDCHE}}_H$	FDCH enable hold time, for one transfer.	450
$\overline{\text{FDCHE}}_R$	FDCH enable rise time delay.	30

I/O BUS TIMING (See I/O Operations)

OPERATION	SYMBOL	DESCRIPTION	TIMING (μS)	
			MIN.	MAX.
COMMON TO ALL OUTPUT OPERATIONS	TD_1 TD_2 TD_3 TD_4	I/O INPUT TO I/O CLOCK TIME (START OF TRANSACTION) I/O CLOCK 1/2 PERIOD (BIT TIME) I/O CLOCK TO I/O INPUT TIME (END OF TRANSACTION) I/O CLOCK TO I/O DATA SKEW	.090 .110 .090 -5	.150 .130 .150 +15
I/O COMMAND OR DATA OUTPUT	$T\overline{INPUT}_D$	COMMAND OR DATA TRANSFER TIME	1.310	1.330
I/O REQUEST ENABLE OR DATA CHANNEL ADDRESS REQUEST	$T\overline{INPUT}_R$	REQUEST ENABLE OR DATA CHANNEL ADDRESS REQUEST TRANSFER TIME	0.350	0.370

DG-04419

DATA TRANSFER TIMING

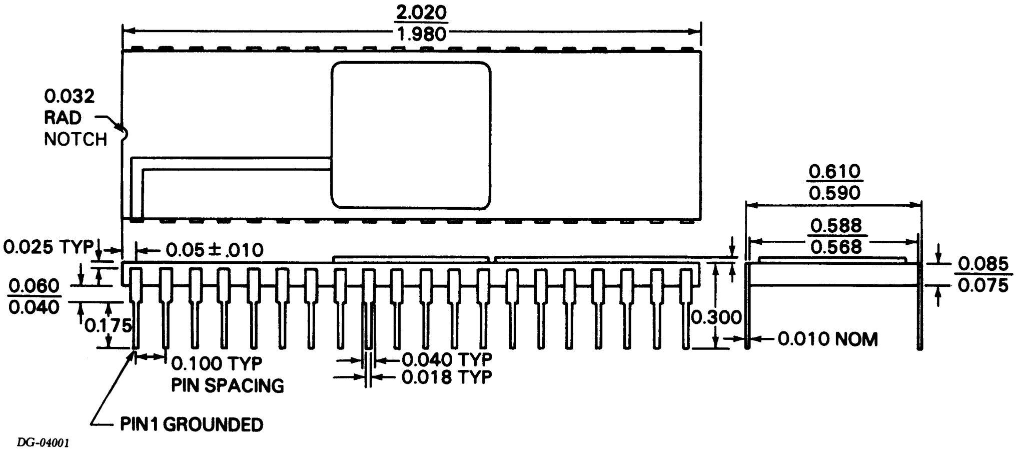
OPERATION	SYMBOL	DESCRIPTION	TIMING (μS)	
			MIN.	MAX.
PROGRAMMED I/O DATA OUT	PI/0 OUT_{DD}	PROGRAMMED I/O DATA OUT - DATA DELAY	0.83	0.85
PROGRAMMED I/O DATA IN	PI/O IN_{TW} PI/O IN_{DD}	PROGRAMMED I/O DATA IN - DATA RECEPTION WINDOW PROGRAMMED I/O DATA IN - DATA DELAY	-- 0.11	3.12 0.13
COMMON TO ALL DATA CHANNEL OPERATIONS	DCH_{AR} DCH_D DCH_{TWA}	DATA CHANNEL ADDRESS REQUEST DATA CHANNEL DELAY DATA CHANNEL ADDRESS AND MODE BIT RECEPTION WINDOW	0.350 0.11 --	0.370 0.13 2.6
DATA CHANNEL OUT - MEMORY TO DEVICE	DCHO $DCHO_{DD}$	CYCLE TIME FOR 16-BIT WORD TRANSFER DATA CHANNEL DATA OUT DELAY	-- 3.11	5.8 3.13
DATA CHANNEL IN - DEVICE TO MEMORY	DCHI $DCHI_{DD}$ $DCHI_{TWC}$ $DCHI_{TWD}$	CYCLE TIME FOR 16-BIT WORD TRANSFER DATA CHANNEL DATA IN DELAY DATA CHANNEL WINDOW CLOSED DATA CHANNEL DATA RECEPTION WINDOW	-- 3.54 -- --	7.2 3.56 0.96 1.9

DG-04419

PACKAGE SPECIFICATIONS

The physical dimensions (in inches) of the mN602 chip are given in the diagram below.

DIMENSIONS OF CHIP



DG-04001

3

mN603/mN613/mN615
I/O CONTROLLER

4

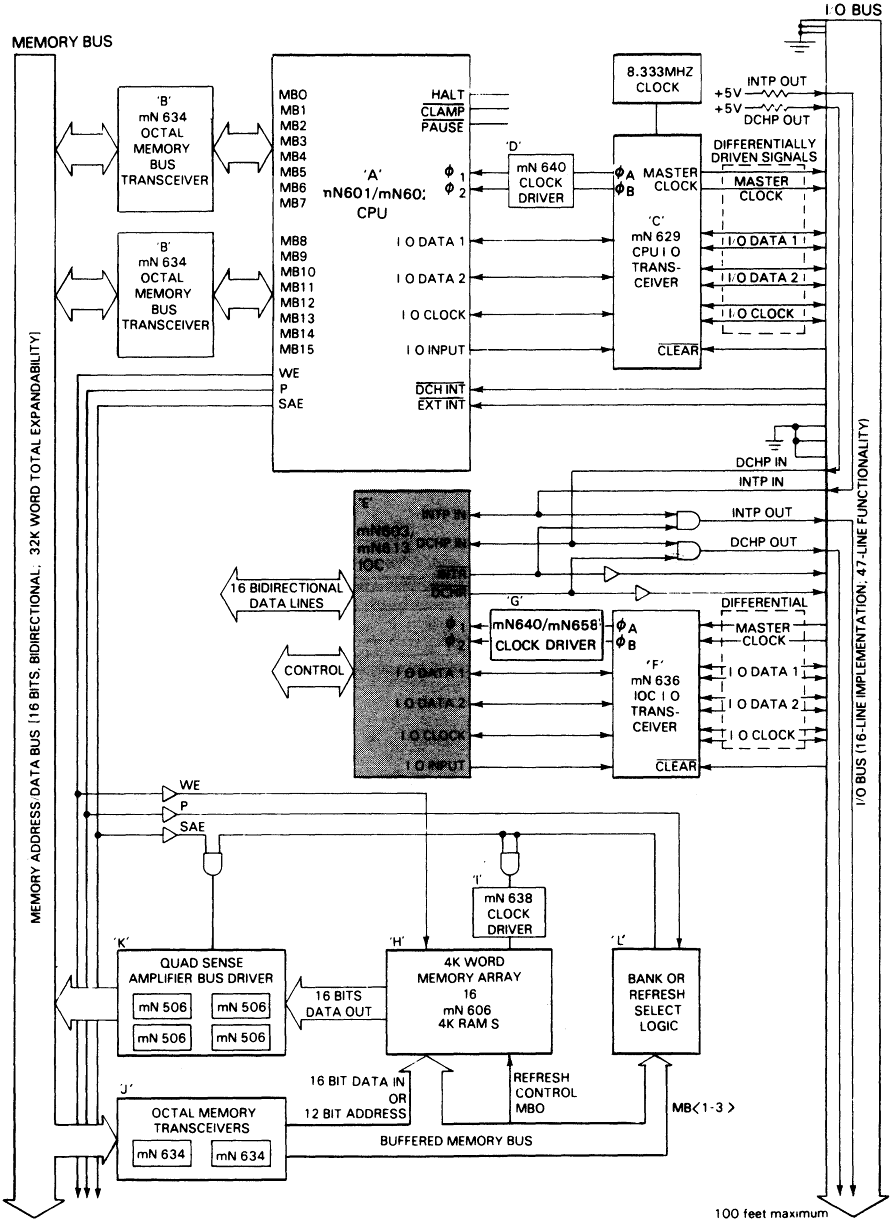
DataGeneral
Data General Corporation, Westboro, Massachusetts 01581

CONTENTS

4

4

mN603/mN613/mN615 I/O CONTROLLER



DG-04326

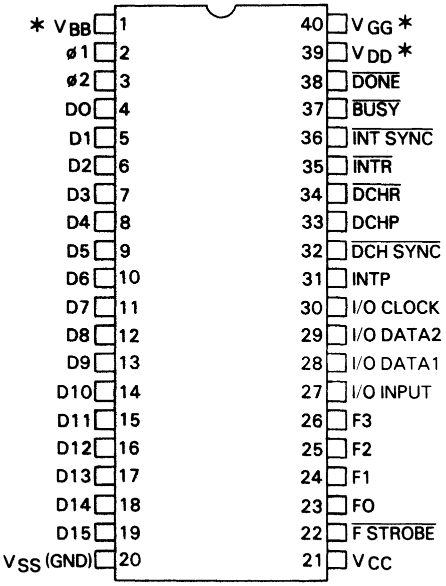
mN603/mN613/mN615

I/O CONTROLLER

FEATURES

- I/O DEVICE CONTROLLER ON A SINGLE 40-PIN SILICON-GATE NMOS CHIP
- INTERPRETS AND EXECUTES DATA GENERAL'S STANDARD NOVA COMPUTER I/O INSTRUCTION SET
- PROVIDES SIMPLE 16-BIT PARALLEL USER INTERFACE
- INTEGRAL BUSY AND DONE CONTROL
- INTEGRAL DEVICE IDENTIFICATION AND INTERRUPT CONTROL
- CONTAINS DATA CHANNEL CONTROL LOGIC
- FULL ADDRESS AND WORD COUNT REGISTERS FOR DATA CHANNEL OPERATION
- USER SELECTABLE DATA BUS SIGNAL POLARITY
- mN613 REQUIRES ONLY +/-5V AND +12V POWER SUPPLY
- mN615 REQUIRES ONLY +5V

PACKAGE



*For mN615

- V_{BB} is generated internally and brought out through pin 1.
- Pins 39 and 40 have no internal connection.

GENERAL DESCRIPTION

The mN603, mN613, and mN615 are I/O controllers (IOCs) that provide the full functional capability of Data General's 47-line NOVA I/O bus. The IOC decodes an encoded data stream from the CPU and presents a parallel 16-bit bidirectional interface, four encoded function bits, and a function strobe, to the peripheral for simple interfacing. The mN603, mN613, and mN615 provide the same functionality, but they have different electrical characteristics. The mN603 requires ±5V, +10V, and +14V; the mN613 requires ±5V and +12V; the mN615 requires +5V.

The IOC includes a number of bus adapter functions found in the most powerful minicomputer systems. It includes integral device identification, BUSY/DONE interrupt logic, and a per-device interrupt masking capability. For block-oriented controllers, the IOC includes data channel bus hand shaking and full address and word counters.

The IOC allows the construction of a complete I/O controller module using a minimum of additional components. These components are:

1) an mN640 Clock Driver (for the mN603 and mN613)

2) an mN658 Clock Driver (for the mN615)

3) an mN636 IOC I/O Tranceiver

4) interrupt and data channel priority circuitry

5) decode logic for the IOC's four encoded function bits

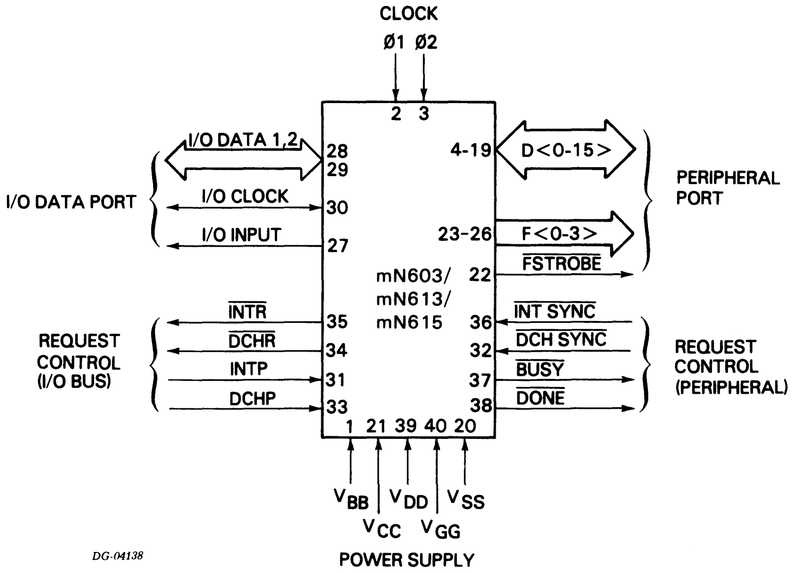
6) additional gating to interface a peripheral and buffer the IOC

Recommendation: Use the mN658/mN615 combination in preference to the mN640/mN603 or the mN640/mN613 combinations.

PIN DESCRIPTIONS

The diagram below shows the pin connections, and the table describes the function of each pin shown in the diagram.

FUNCTIONAL PIN CONNECTION DIAGRAM



DG-04138

PIN FUNCTIONS

MNEMONIC	PIN NO.	IN/ OUT	FUNCTION
			CLOCKS
Ø1 Ø2	2 3	IN IN	Two-phase non-overlapping clock. mN603/mN613 operate between 0 and 14V amplitude. mN615 operates between 0 and 5V. Generates 4-phase internal clock, providing internal timing.
			I/O DATA PORT
I/O DATA1 I/O DATA2	28 29	I/O I/O	Two-line, bi-directional bus. Receives all data and I/O command information transmitted serially from the CPU at the Master Clock rate. Transmits data to the CPU at this rate. During a 16 bit transfer, I/O DATA1 carries bits 0-7 while I/O DATA2 carries bits 8-15.
I/O CLOCK	30	I/O	Synchronizes all I/O transfers on I/O DATA <1,2>. If receiving, I/O CLOCK strobes information received on I/O DATA <1,2> into the IOC. If transmitting, I/O CLOCK is generated by the IOC. Note: The IOC will be reset if I/O CLOCK is held low for more than 8 cycles of MASTER CLOCK*
I/O INPUT	27	OUT	Indicates whether the IOC is transmitting or receiving on I/O DATA <1,2>. High = IOC receiving. Low = IOC transmitting.

PIN FUNCTIONS (CONT.)

MNEMONIC	PIN NO.	IN/ OUT	FUNCTION
			PERIPHERAL PORT
D<0-15>	4-19	I/O	16-line, bi-directional data bus. Transfers all data between an IOC and a peripheral device. Data is interpreted using either positive or negative logic as selected during initilization.
F<0-3>	23-26	OUT	Four line encoded control bus. Four bit Function codes may be decoded into one of 16 separate functions which control the gating of data between a peripheral and D<0-15>. Code is valid while $\overline{\text{FSTROBE}}$ is low.
$\overline{\text{FSTROBE}}$	22	OUT	Used to synchronize function code. Code is valid when $\overline{\text{FSTROBE}}$ is low. (One $\overline{\text{FSTROBE}}$ clock period is equal to four MASTER CLOCK* periods.)
			REQUEST CONTROL
$\overline{\text{INTR}}$	35	OUT	Asserted by an IOC to request program interrupts from the CPU.
$\overline{\text{DCHR}}$	34	OUT	Asserted by an IOC to request a Data Channel Break from the CPU.
INTP	31	IN	Indicates whether an IOC requesting a program interrupt has priority to respond to an Interrupt Acknowledge instruction. INTP high indicates priority.
DCHP	33	IN	Indicates whether an IOC requesting data channel service has priority to respond to a Data Channel Address Request. DCHP high indicates priority.
$\overline{\text{INT SYNC}}$	36	IN	Asserted by peripheral; prompts the IOC to request a program interrupt from the CPU (assert its $\overline{\text{INTR}}$ line).
$\overline{\text{DCH SYNC}}$	32	IN	Asserted by peripheral; prompts the IOC to request data channel service from the CPU (assert its $\overline{\text{DCHR}}$ line).
$\overline{\text{BUSY}}$	37	I/O	Indicates the state of the internal BUSY flag; also used by the device to set the BUSY flag. When $\overline{\text{BUSY}}$ is low, the peripheral device is performing an operation. BUSY set to 1 by: ● device asserting $\overline{\text{BUSY}}$ ● an I/O instruction with an S in the F field (see I/O Data Port, Programmed I/O Transactions) BUSY set to 0 by: ● device asserting $\overline{\text{DONE}}$ ● an I/O instruction with a C in the F field (see I/O Data Port, Programmed I/O Transactions) ● I/O Reset Instruction
$\overline{\text{DONE}}$	38	I/O	Indicates the state of the internal DONE flag; also used by the device to set the DONE flag. When $\overline{\text{DONE}}$ is low, the peripheral device has finished an operation. A Program interrupt will be requested when $\overline{\text{DONE}}$ is low if the Interrupt Disable Flag is not set (see Internal Structure, Interrupt Request Logic). DONE set to 1 by: ● device asserting $\overline{\text{DONE}}$ DONE set to 0 by: ● an I/O instruction with an S, C, or P in the F field (see I/O Data Port, Programmed I/O Transactions) ● an I/O Reset Instruction

PIN FUNCTIONS (CONT.)

POWER				
V_{BB}	1	I	-4.25 ± 0.5V	mN603
V_{CC}	21	I	+5 ± 0.25V	
V_{DD}	39	I	+10 ± 1.0V	
V_{GG}	40	I	+14 ± 1.0V	
V_{SS}	20		GROUND	
V_{BB}	1	I	-4.25 ± 0.5V	mN613
V_{CC}	21	I	+5 ± 0.25V	
V_{DD}	39	I	+12 ± 1.0V	
V_{GG}	40	I	+12 ± 1.0V	
V_{SS}	20		GROUND	
V_{BB}*	1	O	-2.5 ± 0.5V	mN615
V_{CC}	21	I	+5 ± 0.25V	
V_{DD}	39		N/C	
V_{GG}	40	I	N/C	
V_{SS}	20		GROUND	

*On the mN615, V_{BB} should be connected to ground through a 0.1μf capacitor.

OVERVIEW

The IOC, together with its supporting elements, provides a parallel user interface from the serial microNOVA I/O bus. This resulting interface is functionally equivalent to but not identical to the 47-line bus used on Data General's NOVA and ECLIPSE line computer systems.

Each time the CPU fetches an I/O instruction from memory, it transmits the complete instruction over the I/O bus. Every IOC connected to the bus receives a copy of this instruction and internally decodes it. If the IOC determines that the instruction is directed to it, the IOC executes the instruction. Since it has an exact copy of the I/O instruction, the IOC can emulate all the programmed I/O control signals found in the NOVA/ECLIPSE I/O bus.

In addition to decoding programmed I/O instructions, the IOC contains all the logic necessary to implement a complete program interrupt system. This system is compatible with the mN601 or an mN602 and includes Busy and Done flags, an interrupt request line, interrupt masking logic, and interrupt source identification.

The IOC also provides the logic necessary to perform standard (not high speed) data channel transfers. This includes the request logic and two internal registers, a memory address register and a word count register, which control the data transfer.

The IOC automatically manages all the protocols found on the microNOVA I/O bus. These protocols include the transfer of data, of programmed I/O instructions, and of the system synchronization signals needed by the interrupt and data channel facilities.

Four major areas of the IOC perform the above tasks.

- The Peripheral Data Port provides the device with a 16-bit wide, bidirectional data bus and four function code control lines.
- The I/O Data Port performs the serial/parallel conversions of information transferred to and from the I/O bus.
- The internal registers, data paths, and logic arrays decode instructions and connect the two data ports.
- The Request Control Facility integrates the service requests into the rest of the microNOVA system.

These four areas along with the System Requirements are discussed in the following sections as described below:
PERIPHERAL DATA PORT - This section discusses how data is transferred between a peripheral device and an IOC. The Peripheral Data Port includes the 16 data lines (D<0-15>), the four function code lines (F<0-3>), and the clock ($\overline{\text{FSTROBE}}$). The 16 data lines transfer data to and from the peripheral device. The function codes, with their clock, control the gating of data on these data lines.

I/O DATA PORT - This section explains how data is transferred between the serial I/O bus and an IOC. The I/O data port includes two serial data lines (I/O DATA1 and I/O DATA2), their strobe (I/O CLOCK), and a transmit/receive control line (I/O INPUT). The I/O DATA <1,2> lines receive commands and data from the I/O bus and transmit data to the I/O bus. I/O CLOCK strobes the information transmitted on these data lines. I/O INPUT indicates whether the IOC is transmitting or receiving on the data lines.

INTERNAL STRUCTURE - This section describes the internal data flow between the two data ports. A set of internal registers, together with their connecting data paths, join the I/O data port to the peripheral data port. All control of information transfers and service requests comes from the internal logic array and state change logic. Two of the internal registers used during data channel breaks may be replaced with external registers. Interrupt requests and data channel break requests are controlled by the Request Logic.

REQUEST CONTROL - This section describes the program interrupt and data channel break facilities. Each of these facilities has separate request and priority networks. Most of this logic is internal to the IOC. However, the designer may choose to use either internal or external BUSY and DONE logic.

SYSTEM REQUIREMENTS - This section covers the IOC's power-up and initialization requirements. The Power-Up portion discusses the sequence required when first applying power and the clock to the chip. The Initialization portion discusses how synchronization with the CPU is achieved and how an IOC's internal registers are initialized.

PERIPHERAL PORT

The Peripheral Port transfers data between an I/O Controller and the associated peripheral device. The port consists of 16 bi-directional data lines (D<0-15>), four function code control lines (F<0-3>), and a function code clock (FSTROBE).

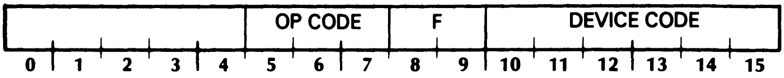
DATA LINES

The data lines transfer data to or from an IOC in a 16 bit parallel format. Data may be interpreted using either positive or negative logic; selectable during initialization. Some of the data lines are used to load initialization information (see System Requirements, Initialization). Each data line can drive one Schottky TTL load.

CONTROL LINES

The function code lines, with their clock, either indicate how the data lines are to be used or provide control pulses. These lines react in particular ways according to the information received from the I/O bus by the I/O Data Port. In particular, there are 9 I/O Instructions and two types of Data Channel Transactions which invoke specific responses from the Peripheral Port.

The four function code lines specify one of 16 possible codes at any one time. Most of these function codes are generated as a result of I/O instructions received from the CPU via the I/O Data Port. The I/O instruction format, the Op Code and F field mnemonics, and their corresponding bit patterns are shown below:



I/O INSTRUCTION	OP CODE (BITS 5,6,7)
NO I/O Transfer	NIO - 0 0 0
DATA IN A	DIA - 0 0 1
DATA OUT A	DOA - 0 1 0
DATA IN B	DIB - 0 1 1
DATA OUT B	DOB - 1 0 0
DATA IN C	DIC - 1 0 1
DATA OUT C	DOC - 1 1 0
I/O SKIP	SKP - 1 1 1

F FIELD (BITS 8,9)
NONE - 0 0
S(STRT) - 0 1
C(CLR) - 1 0
P(IOPLS) - 1 1

NOTE: The I/O Skip instruction is listed above for completeness; however, no function codes are generated as a result of this instruction.

There are three I/O instructions not listed above which follow a slightly different format (see I/O Data Port, Programmed I/O Instructions). Of these three, the Interrupt Acknowledge Instruction generates no function code. However, the Mask-out and I/O Reset Instructions each produce a specific function code. Data Channel Transactions account for the four remaining function codes.

The various function codes, their labels, and their purpose are listed below:

FUNCTION CODE TABLE

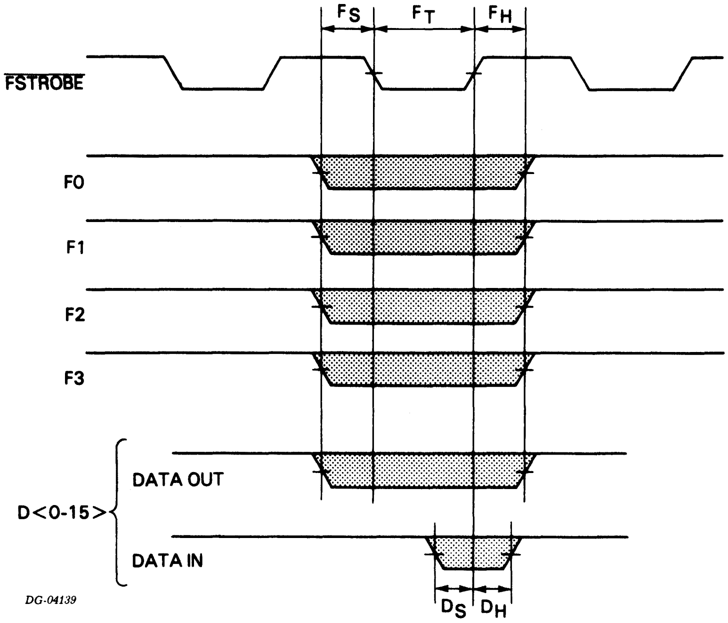
FUNCTION CODE F<0-3>	LABEL	FUNCTION
0000	DOA	Load the A register with data from IOC data lines D <0-15>.
0001	DIA	Gate data into IOC from the A register via data lines D <0-15>.
0010	DOB	Load the B register with data from IOC data lines D <0-15> (see Internal Structure, Data Channel Registers).
0011	DIB	Gate data into IOC from the B register via data lines D <0-15>. If internal data channel registers are being used, D <0-15> are ignored (see Internal Structure, Data Channel Registers).
0100	DOC	Load the C register with data from IOC data lines D <0-15> (see Internal Structure, Data Channel Registers).
0101	DIC	Gate data into IOC from the C register via data lines D <0-15>.
0110	STRT	Start I/O device.
0111	CLR	Clear I/O device.
1000	IOPLS	I/O pulse.
1001	IORST*	Gate device code, Polarity, External Register Enable, and External BUSY/DONE Enable bits into IOC via the following data lines: D<11-15> = DEV CODE D9 = POLARITY D8 = EXT REG ENB D7 = EXT BUSY/DONE ENB NOTES: This data is always interpreted using negative logic (high = 0). See System Requirements, Initialization.
1010	MSKO	Gate device maskout priority bit into IOC via data lines D <0-15>.
1011	DCHA	Gate data channel direction bit into IOC via data line DO. If using external registers,gate the external address register into the IOC via data lines D <0-15>.
1100	DCHI	Gate data from peripheral data register onto IOC's data lines (D <0-15>) during a Data Channel In Transaction.
1101	DCHO	Load the peripheral data register with data from the IOC's data lines (D <0-15>) during a Data Channel Out Transaction.
1110	WCEZ	Indicates that the internal data channel Word Count Register has overflowed (equals zero). Denotes the end of a block data channel transfer.
1111	NOP	No function.

* Should be used to clear all external registers and flags (Busy/Done flags, word count register and address register).

PERIPHERAL PORT TIMING

Function codes are valid only when $\overline{\text{FSTROBE}}$ is low. Each function code lasts for one cycle of $\overline{\text{FSTROBE}}$. If either the data lines are not being used by the IOC or control pulses are not occurring, the NOP code is transmitted. If the data lines are being used to transfer data to the IOC, they are sampled on the rising edge of $\overline{\text{FSTROBE}}$. Timing constraints are shown below:

PERIPHERAL PORT TIMING



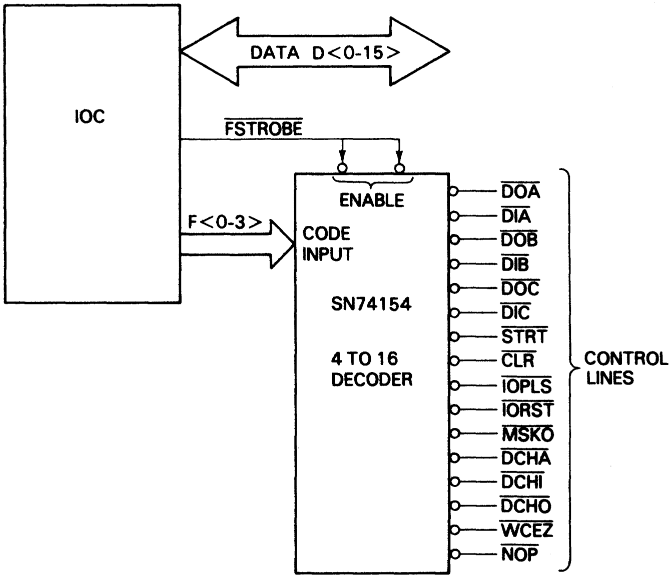
NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE ON PAGE 43.
- D<0-15> MUST NOT BE PULLED LOW AT ANY TIME OTHER THAN THAT INDICATED BY D_S AND D_H IN THE DIAGRAM ABOVE.

DECODING THE CONTROL LINES

There are many ways to decode the four bit code produced by the four function code lines into separate control lines. Described below are two such methods:

The first approach incorporates a 4 to 16 decoder (TI SN74154 or equivalent) which decodes the 4 bit function codes into individual control lines. $\overline{\text{FSTROBE}}$ should enable the decoder when the function codes are valid. In the arrangement shown below, only one control line is asserted during each cycle of $\overline{\text{FSTROBE}}$:

PERIPHERAL PORT WITH DECODER

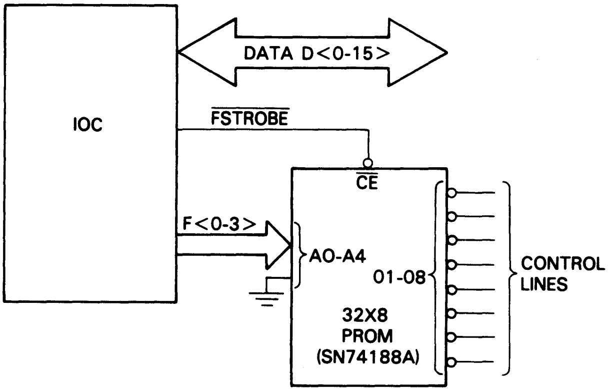


DG-04140

A second approach is to use one or two PROM(s) (TI SN74188A or equivalent) to decode the function code lines. The advantage of using PROMs is that they allow two or more function codes to assert the same control line without requiring additional logic. For example, when using external BUSY and DONE logic, it is frequently necessary to clear both flags when either a CLR function code or an IORST code is issued. PROMs allow full flexibility in decoding both these function codes.

In the application shown below, $\overline{\text{FSTROBE}}$ enables the outputs of the PROM by pulling the chip enable ($\overline{\text{CE}}$) input low. Only four of the five address lines are used.

PERIPHERAL PORT WITH PROM DECODER



DG-04141

Most of the control lines resulting from the decoding of F <0-3> are used to control the flow of the data to and from the Peripheral Port Data lines. These data lines are used when an IOC executes certain programmed I/O instructions or when it performs Data Channel Breaks. The following sections explain how the Peripheral Port reacts during the execution of these instructions and commands.

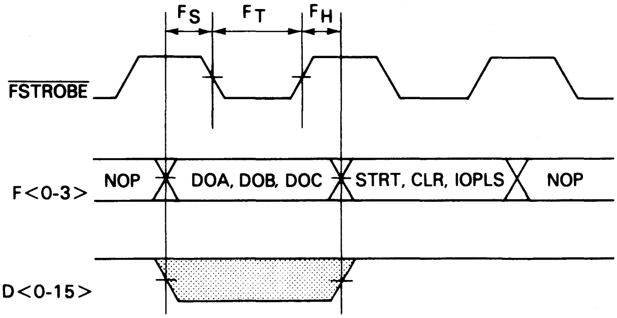
PROGRAMMED I/O INSTRUCTIONS

Data Out Instructions

Data Out Instructions (DOA, DOB, and DOC) perform a transfer of up to 16 bits of data from an IOC to a peripheral register. The IOC places the data on D<0-15>when the DOA, DOB, or DOC code is valid. These codes may be used to strobe infomation on D <0-15> into the appropriate register (A, B, or C).

In addition, one of three function codes (STRT, CLR, or IOPLS), if coded in the instruction, is issued during the next $\overline{\text{FSTROBE}}$ cycle. These codes may be used to control the state of the peripheral device (see Request Control, External Busy/Done Logic).

IOC TO DEVICE DATA TIMING



NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE NEAR THE END OF THIS CHAPTER

DG-04142

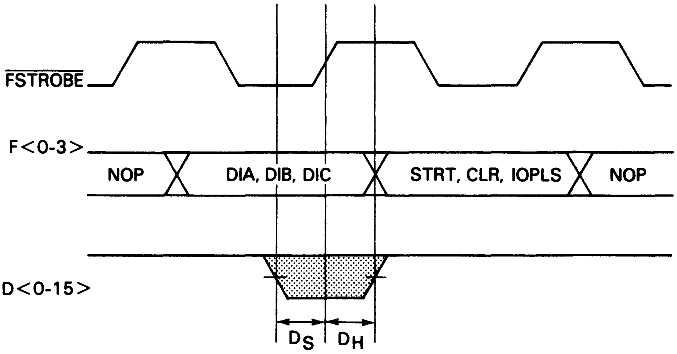
NOTE: The No I/O Transfer Instruction is similar to a Data Out Instruction in that a STRT, CLR, or IOPLS function code is issued if coded in the original instruction. However, the NOP function code is always issued in place of the DOA, DOB, or DOC codes.

Data In Instructions

Data In Instructions (DIA, DIB, and DIC) perform a transfer of up to 16 bits of data from a peripheral register to an IOC. The function codes may be used to strobe information from the respective peripheral register (A,B, or C) to D <0-15>. The IOC samples D <0-15> on the rising edge of $\overline{\text{FSTROBE}}$ when these codes are valid.

In addition, one of three function codes (STRT, CLR, and IOPLS), if coded in the instruction, is issued during the next $\overline{\text{FSTROBE}}$ cycle. These codes may be used to control the state of the peripheral device (see Request Control, External Busy/Done Logic).

DEVICE TO IOC DATA TIMING



NOTES:

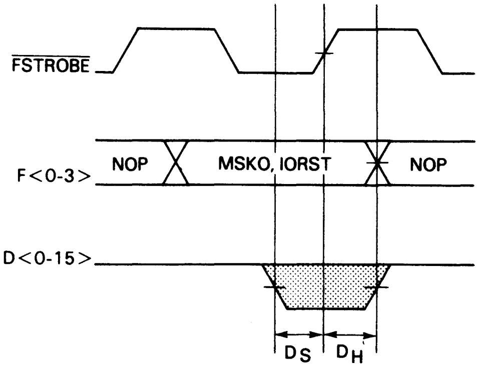
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE NEAR THE END OF THIS CHAPTER

DG-04143

Maskout and I/O Reset Instructions

The Maskout and I/O Reset Instructions both require the transfer of data to an IOC. The IOC samples D <0-15> on the rising edge of $\overline{\text{FSTROBE}}$ when either the MSKO or the IORST code is valid. Therefore, these codes should be used to gate data onto these lines (for MSKO see Internal Structure, Interrupt Request Logic; for IORST see System Requirements, Initialization).

MSKO/IORST DATA TIMING



NOTES:

- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE NEAR THE END OF THIS CHAPTER

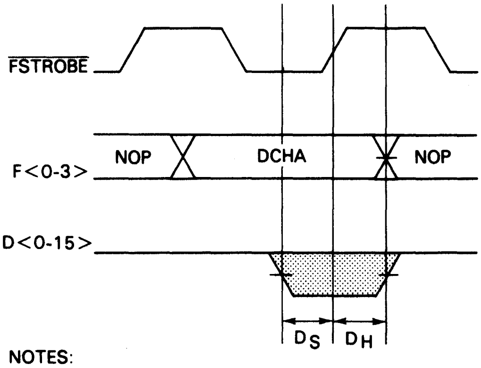
DG-04144

DATA CHANNEL TRANSACTIONS

The Data Channel transfers blocks of data between memory and a peripheral device via the I/O bus and CPU. Each word of a block requires a complete Data Channel Transaction. A Data Channel Transaction requires two information transfers between a peripheral device and an IOC. The first transfer of the pair is to the IOC. This first transfer specifies the direction of the second transfer and may provide the address of the memory location to be accessed. The second transfer of the pair is the data.

During the first transfer of the pair, the IOC samples D <0-15> on the rising edge of $\overline{\text{FSTROBE}}$ when the DCHA function code is valid. DCHA may be used to gate the direction bit to D0 and the Data Channel Address to D <1-15>. The data on D <1-15> is ignored if the IOC has been initialized to use the internal Address Register (see Internal Structure, Data Channel Registers).

DATA CHANNEL ADDRESS TRANSFER



NOTES:

- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE NEAR THE END OF THIS CHAPTER

DG-04145

If the direction bit is a 0, a Data Channel Out Transaction is performed. This transfers a 16 bit word "out" from the memory location specified by the Data Channel Address to the peripheral device. If the direction bit is a 1, a Data Channel In Transaction is performed. In this case, a 16 bit word is transferred from the peripheral device "in" to the memory location specified by the Data Channel Address.

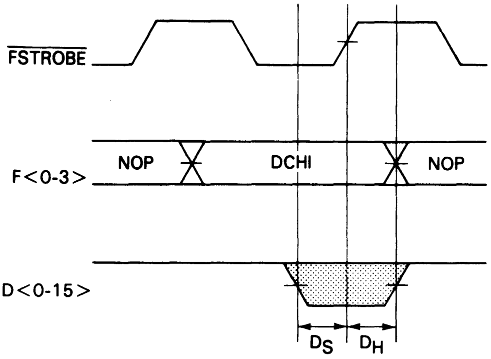
DATA CHANNEL CONTROL WORD

D	DATA CHANNEL ADDRESS														
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

D = Direction Bit
1 = Data Channel In Transaction
0 = Data Channel Out Transaction

If the first transfer specifies a Data Channel In Transaction, the DCHI code is issued during the second transfer. This code may be used to strobe the data word onto D < 0-15 >. The IOC samples these lines on the rising edge of $\overline{\text{FSTROBE}}$ when the DCHI code is valid.

DATA CHANNEL IN TRANSFER

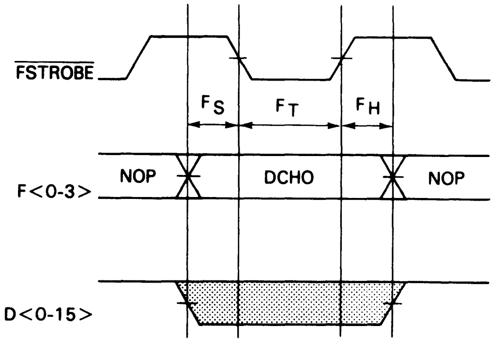


NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE NEAR THE END OF THIS CHAPTER

DG-04146

If the first transfer specifies a Data Channel Out Transaction, the DCHO code is issued during the second transfer. The IOC places the contents of the second data transfer on D < 0-15 > when the DCHO code is valid. This code may be used to strobe the data word into the appropriate peripheral register.

DATA CHANNEL OUT TRANSFER

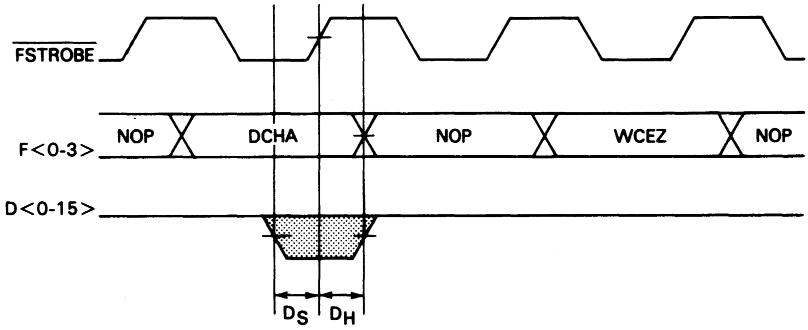


NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE NEAR THE END OF THIS CHAPTER

DG-04147

When the last transaction of a block transfer begins, the internal word count register (if in use) overflows (see Internal Structure, Data Channel Registers). The WCEZ (word count equals zero) function code is issued as shown below. This code may be used to set the DONE flag signaling the completion of a block transfer.

DATA CHANNEL END OF TRANSFER



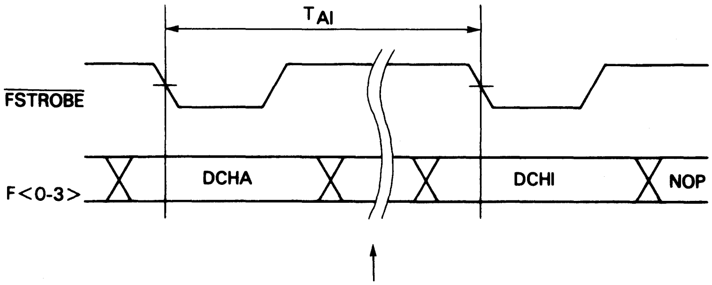
NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE NEAR THE END OF THIS CHAPTER

DG-04148

Data Channel Timing

The minimum interval between the time a DCHA function code is issued and the time a DCHI or DCHO function code is issued is given below. This assumes that the IOC requesting Data Channel service is the highest priority device (see Request Control, Priority Networks).

DATA CHANNEL IN TRANSACTION

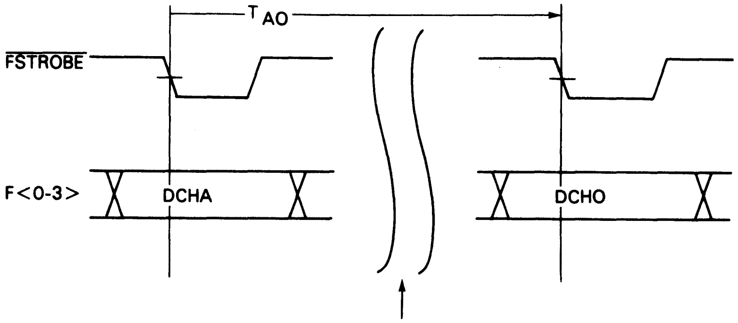


NOTE:
ANOTHER DCHA OR OTHER FUNCTION CODE MAY OCCUR HERE, SEE INTERNAL STRUCTURE, DATA CHANNEL LOGIC

NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE NEAR THE END OF THIS CHAPTER

DG-04149

DATA CHANNEL OUT TRANSACTION



NOTE:
ANOTHER DCHA OR OTHER FUNCTION CODE
MAY OCCUR HERE, SEE INTERNAL
STRUCTURE, DATA CHANNEL LOGIC

NOTES:

- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE NEAR THE END OF THIS CHAPTER

DG-04150

For the mN601 the maximum interval of time is calculated by adding the execution time of the longest instruction (excluding Multiply and Divide) to the minimum interval shown above. A table listing mN601 instruction execution times is given in the section describing the mN602.

The mN602 issues Request Enables and enters an interrogation state between instructions and at irregular intervals during the execution of instructions. While in the interrogation state, the mN602 checks whether it is time to do a demand refresh, service the high speed data channel, or service the standard data channel.

If the peripheral device has asserted $\overline{\text{DCH SYNC}}$, a Request Enable allows $\overline{\text{DCHR}}$ to fall and the IOC to issue a DCHA function code. If $\overline{\text{DCHR}}$ is down at the next interrogation state, the mN602 issues a Data Channel Address Request which gives the IOC authority to issue a DCHO or DCHI function code.

For the mN602 the maximum interval between a DCHA and a DCHO or DCHI is the maximum time listed above plus the maximum time between a Request Enable and an interrogation state. This time is lengthened further if the mN602 is required to perform a demand refresh or service the high speed data channel.

I/O DATA PORT

The following section describes the operation of the I/O Data Port and the protocols it uses. Although most designers will concern themselves with only the Peripheral Port, the following description of the I/O Data Port provides a better understanding of the IOC's interaction with the system.

The I/O Data Port includes two serial data lines (I/O DATA <1,2>), their strobe (I/O CLOCK), and the transmit/receive control line (I/O INPUT). The data lines transfer information between an IOC and the CPU via the I/O bus. IOC and CPU I/O Transceivers (mN636 and mN629) are used to drive these data lines and their clock on the differential I/O bus. I/O CLOCK, generated by the transmitting device, strobes information into the receiving device. I/O INPUT, generated by the IOC, selects either the transmit mode (when low) or the receive mode (when high) of the associated transceiver. The IOC monitors the I/O bus by remaining in receive mode unless required to respond to the CPU.

INFORMATION TYPES

Four types of information are received by the I/O Data Port; Request Enable, Data Channel Address Request, I/O Instructions, and Data. However, only data is transmitted by this port.

These information types are transferred in one of two formats; short and long. Both Request Enable and Data Channel Address Request use the short format which requires only one I/O CLOCK pulse. I/O Instructions and Data use the long format which requires five I/O CLOCK pulses. In all cases, the first bits of a transfer on the I/O DATA <1,2> lines make up a two bit code indicating the information type. The encoding of these first bits is shown below:

I/O TRANSFER CODES

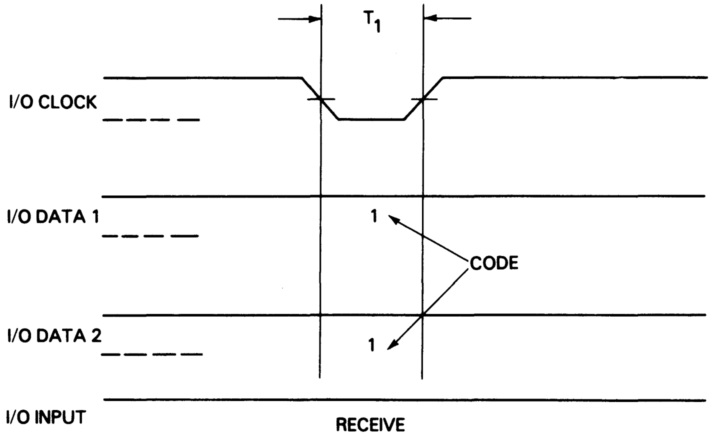
FIRST BIT I/O DATA1	FIRST BIT I/O DATA2	INFORMATION TYPE
1	1	REQUEST ENABLE
1	0	DATA CHANNEL ADDRESS REQUEST
0	1	DATA
0	0	I/O INSTRUCTION

DG-04416

Request Enable

Request Enables are issued at intervals by the CPU. Only the two code bits are transferred. They synchronize program interrupt requests and data channel requests with the CPU. This is necessary to ensure that the request lines to the CPU and priority lines are stable when they are sampled. Request Enables are also used for IOC initialization (see the System Requirements, Initialization).

REQUEST ENABLE TIMING



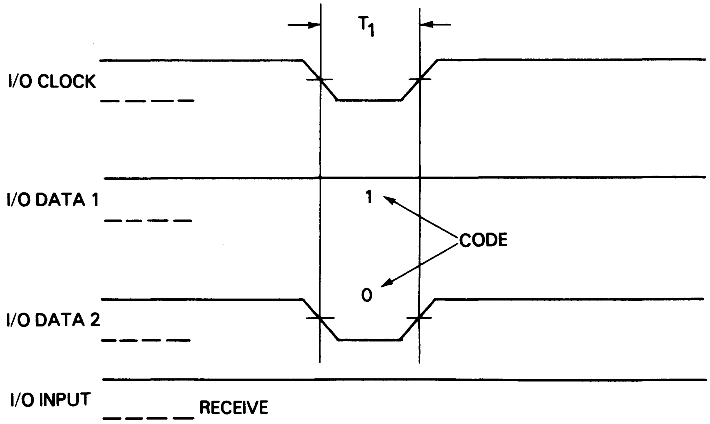
NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE NEAR THE END OF THIS CHAPTER

DG-04151

Data Channel Address Request

Data Channel Address Requests are issued by the CPU when a data channel break is executed. Only the two code bits are transferred. They allow the highest priority IOC requesting data channel service to perform a data channel transaction. In addition, like Request Enable, they synchronize program interrupt requests and additional data channel requests for all IOCs (see Request Enable and Data Channel Transaction sections).

DATA CHANNEL REQUEST TIMING



NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE NEAR THE END OF THIS CHAPTER

DG-04152

I/O Instructions

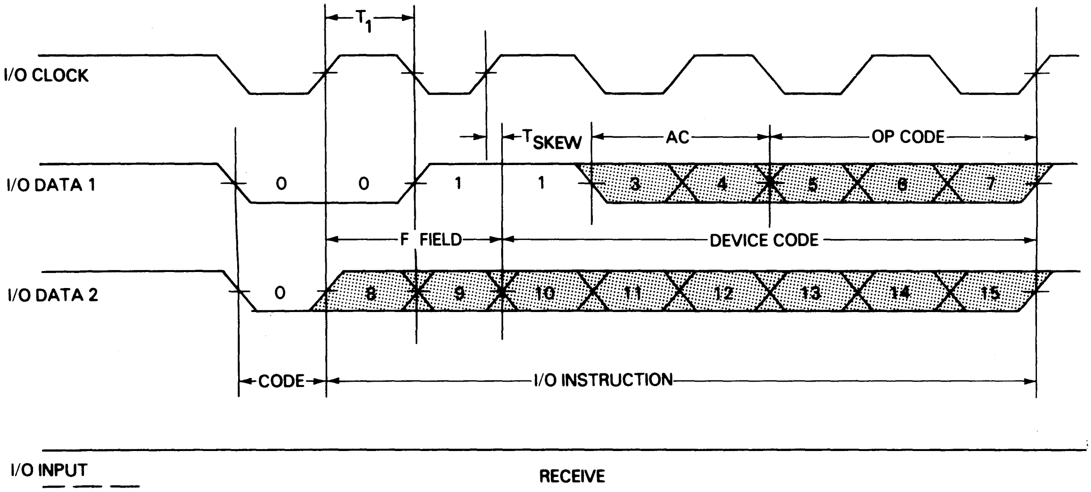
I/O Instructions are issued by the CPU. Eighteen bits are transferred; two code bits and a 16-bit instruction. This is the instruction exactly as fetched from memory by the CPU. The eight high order bits of the instruction (0-7) are transferred via the I/O DATA1 line while the eight low order bits (8-15) are transferred via the I/O DATA2 line. The instruction format is shown in the bit pattern below:

0	1	1	AC		OP CODE			F		DEVICE CODE					
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

AC = accumulator, see mN601
F = F field

The device code contained in the last six bits of the instruction is checked to see if the IOC should execute the instruction. An IOC executes instructions containing its own device code as well as some of those with device code 77_8 (which select all IOCs). Device codes 00_8 through 03_8 ,and 77_8 are ILLEGAL DEVICE CODES for any device connected to the microNOVA I/O bus.

I/O INSTRUCTION TIMING



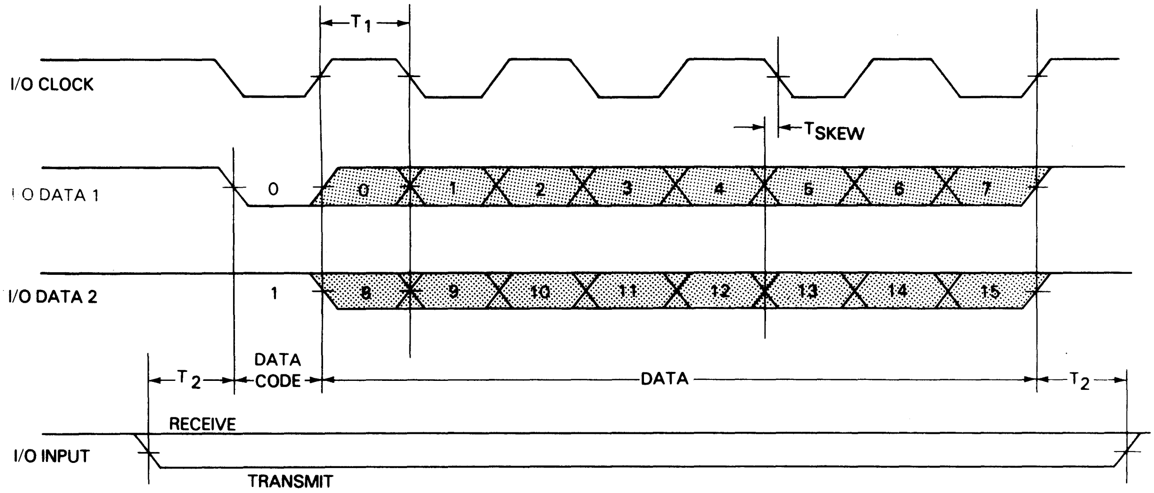
NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE NEAR THE END OF THIS CHAPTER

DG-04153

Data

Data is transmitted by either the CPU or an IOC. Eighteen bits are transferred; two code bits and a 16 bit data word. The I/O DATA1 line transfers the eight high order bits of the data word while the I/O DATA2 line transfers the eight low order bits. Data received by an IOC is ignored unless required by a Programmed I/O Transaction or a Data Channel Transaction. Likewise, data is transmitted only if required by one of these transactions.

DATA TRANSFER TIMING



NOTES:

- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE NEAR THE END OF THIS CHAPTER

DG-04154

TRANSACTION PROTOCOLS

The four types of information transfers discussed above are combined to form three types of transactions; Request Enable Transactions, Programmed I/O Transactions, and Data Channel Transactions. Each type of transaction has its own protocol. Request Enables occur as a single information transfer. Programmed I/O Transactions combine an I/O Instruction transfer with a Data transfer. Data Channel Transactions consist of a Data Channel Address Request followed by two Data transfers.

> **NOTE:** There is a minimum time between two consecutive transactions. This is indicated in the timing diagrams by T_N.

Request Enable Transactions

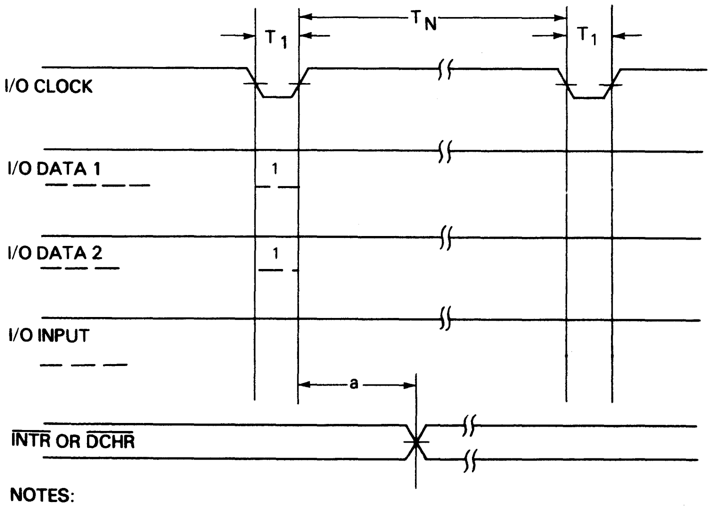
Request Enables are used by an IOC to synchronize the assertion and release of program interrupt and data channel requests. If a peripheral device requires service, a Request Enable allows the associated IOC's interrupt request line ($\overline{\text{INTR}}$) and/or data channel request line ($\overline{\text{DCHR}}$) to be asserted, signaling the CPU for service. When service is received, a Request Enable allows these lines to be released. It is necessary that the transition of these lines occur at specific times to ensure their stability when sampled by the CPU. These lines are also cleared asynchronously if the IOC is reset.

One of two conditions must be met for $\overline{\text{INTR}}$ to be asserted during a Request Enable. Either the peripheral device pulls the IOC's $\overline{\text{INT SYNC}}$ line low or it sets the DONE flag. However, an internal Interrupt Disable Flag can block a program interrupt request; see the section on Request Control.

In order for $\overline{\text{DCHR}}$ to be asserted during a Request Enable, the peripheral device must pull the IOC's $\overline{\text{DCH SYNC}}$ line low.

> **NOTE:** The first two Request Enables received after an IOC is powered up or reset initialize its internal four phase clock and internal registers. As a result, the IOC is synchronized with the CPU (see System Requirements, Initialization).

IOC INITIALIZATION



NOTES:

- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE NEAR THE END OF THIS CHAPTER

DG-04155

Programmed I/O Transactions

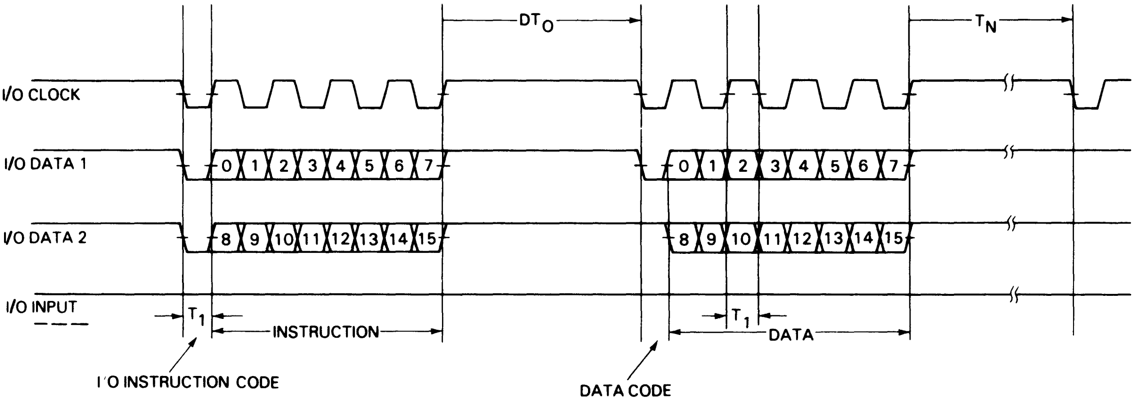
There are two types of Programmed I/O Transactions:
1) CPU to IOC Data Out Transaction
2) IOC to CPU Data In Transaction
Each type of transaction transfers a word of data between an IOC and the CPU.

1) CPU to IOC Data Out Transaction

Data Out Transactions transfer a 16 bit word of data from the CPU to one or all IOCs. The transactions begin with an I/O Instruction transfer, issued by the CPU and are followed by a Data transfer, also issued by the CPU. The instruction transmitted is the I/O Instruction executed by the CPU, exactly as fetched from memory.

An IOC executing a Data Out Instruction must receive the Data transfer during a fixed interval of time after it has received the I/O Instruction transfer. This time interval is illustrated in the following timing diagram by DT_O.

DATA OUT TIMING



NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE NEAR THE END OF THIS CHAPTER

DG-04156

There are two groups of instructions which use the Data Out format. The first group consists of four I/O Instructions which are executed by only one IOC per transaction. The second group consists of two I/O Instructions which are executed by all IOCs simultaneously.

The first group selects individual IOCs by specifying a particular device code. The bit patterns for these I/O Instructions are given below. The instruction is the same as fetched from memory by the mN601 or mN602 CPU.

0	1	1	AC		OP CODE			F		DEVICE CODE					
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

AC	BIT 3	BIT 4
AC0	0	0
AC1	0	1
AC2	1	0
AC3	1	1

DOA (Data Out A) ....Op Code = 010
DOB (Data Out B) ....Op Code = 100
DOC (Data Out C) ....Op Code = 110

DOA, DOB, and DOC transfer data from the CPU to a specific peripheral device via an IOC. The data received by the selected IOC is placed on its Peripheral Data Port to be gated into one of three peripheral registers, A, B, or C. For more information on how these registers are loaded, see Peripheral Port.

In addition, the state of two internal flags (BUSY and DONE) are controlled by these instructions (see Internal Structure, Internal BUSY/DONE Logic).

> **NOTE:** Two internal registers, used during Data Channel Breaks, are loaded using the DOB and DOC instructions (see Request Control, Data Channel Registers).

NIO (No I/O) ....Op Code = 000

The No I/O Transfer instruction is a Data Out instruction in which data may be transferred from the CPU to the IOC. If data is received by the selected IOC, it may be placed on the Peripheral Port data lines. However, this data is unpredictable and no function code is generated.

Like DOA, DOB, and DOC, the NIO instruction can be used to control the state of the BUSY and DONE flags (see Internal Structure, Internal BUSY/DONE Logic).

The second group of data out transactions consists of two I/O Instructions which select all IOCs by specifying device code 77_8 . The bit patterns and timing for these two instructions are given below:

MSKO (Mask Out)
DOB [f] ac, CPU

0	1	1	AC		1	0	0	F		1	1	1	1	1	1
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

AC, F = see mN601, CPU

The Mask Out Instruction transfers a 16-bit mask (data) from the CPU to all IOCs connected to the I/O bus. This data affects the state of each IOC's Interrupt Disable flag, after executing one more instruction. For more information on this flag, see Interrupt Request Logic under Internal Structure.

IORST (I/O Reset)
DOA [f] 0, CPU

0	1	1	0	0	0	1	0	F		1	1	1	1	1	1
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

The I/O Reset Instruction is used to initialize all an IOC's internal registers and flags. The instruction portion of this transaction is executed by all IOCs, however, the data portion of the transaction is ignored. For futher information see the section titled System Requirements, Initialization.

FLAG SETTING

CLASS ABBREV.	CODED CHARACTER	RESULT BITS	OPERATION
f	(option omitted)	00	Does not affect the Busy and Done flags.
	S	01	Start the device by setting Busy to 1 and Done to 0.
	C	10	Idle the device by setting both Busy and Done to 0.
	P	11	The effect, if any, depends on the device.

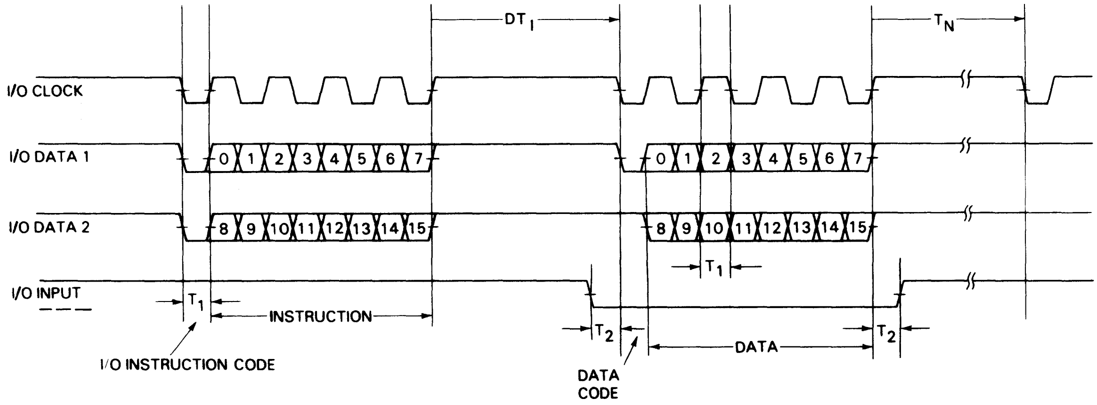
NOTE: IORST sets bits 8 and 9 to 10, while DOA [f] 0, CPU clears the device.

2) IOC to CPU Data In Transactions

Data In Transactions transfer a 16 bit word of data from an IOC to the CPU. These transactions begin with an I/O Instruction transfer issued by the CPU and are followed by a data transfer issued by the responding IOC.

An IOC executing a Data In Instruction begins transmitting the data within a fixed interval of time after it has received the I/O Instruction transfer. This time interval is illustrated in the following timing diagram by DT_I.

DATA IN TIMING



NOTES:

- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE NEAR THE END OF THIS CHAPTER

DG-04157

There are two groups of instructions which use the Data In format. The first group consists of four I/O Instructions which select only one IOC per transaction. The second group consists of one I/O Instruction which selects all IOCs simultaneously.

The first group selects individual IOCs by specifying a particular device code. The bit pattern for these instructions is given below:

0	1	1	AC		OP CODE			F		DEVICE CODE					
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

DIA (Data In A)..............Op Code = 001
DIB (Data In B)..............Op Code = 011
DIC (Data In C)..............Op Code = 101

DIA, DIB, and DIC transfer data from a specific peripheral device to the CPU via an IOC. The selected IOC gates data into its Peripheral Data Port from one of three peripheral registers (A, B, or C) and transmits this data to the CPU.

In addition, the state of two internal flags (BUSY and DONE) may be controlled by these instructions (see Internal Structure, Internal Busy/Done Logic).

> **NOTE:** There is one internal register used during data channel breaks whose contents may be transferred from an IOC to the CPU using the DIB instruction (see Request Control, Data Channel Registers).

SKP (I/O Skip)...............Op Code = 111

The I/O Skip Instruction transfers a word of data containing the state of the BUSY and DONE flags from the selected IOC to the CPU. Bit 0 of this word contains the complement of the state of the DONE flag while bit 1 contains the complement of the state of the BUSY flag. Bits 2-15 contain zeros.

The second group of Data In Instructions consists of the Interrupt Acknowledge Instruction. This instruction incorporates device code 77_8 which selects all IOCs.

INTA (Interrupt Acknowledge)
DIB [f] ac, CPU

0	1	1	AC		0	1	1	F		1	1	1	1	1	1
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

The Interrupt Acknowledge Instruction causes the responding IOC to transmit its device code to the CPU. This instruction is issued by the CPU in response to one or more IOCs requesting program interrupts (asserting their $\overline{\text{INTR}}$ line). Only the highest priority IOC responds (see Request Control, Priority Networks).

The responding IOC transmits the contents of an internal register containing its device code to the CPU. Bits 0-9 of the data transfer are zero while bits 10-15 contain the device code.

Data Channel Transactions

There are two types of Data Channel Transactions:

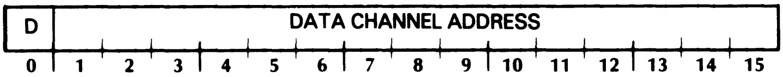
1) CPU to IOC Data Channel Out Transactions
2) IOC to CPU Data Channel In Transactions

Each type of transaction transfers a 16 bit word of data between a peripheral device and memory via the CPU. The direction of the transfer indicated by the name refers to the CPU.

Both types of Data Channel Transactions begin with a Data Channel Address Request, issued by the CPU and are followed by a Data transfer issued by the responding IOC. The second Data transfer is issued by the CPU or IOC depending on which type of transaction is taking place.

NOTE: Though all IOCs receive the Data Channel Address Request, only the highest priority IOC requesting a data channel break completes the transaction (see Request Control section).

The first data transfer, from IOC to CPU, provides the CPU with two kinds of information. Bit 0 of this 16 bit word indicates the direction of the second transfer (data). Bits 1-15 contain the address of the memory location to be accessed.



D = direction bit:
0 = Data Channel Out Transaction
1 = Data Channel In Transaction

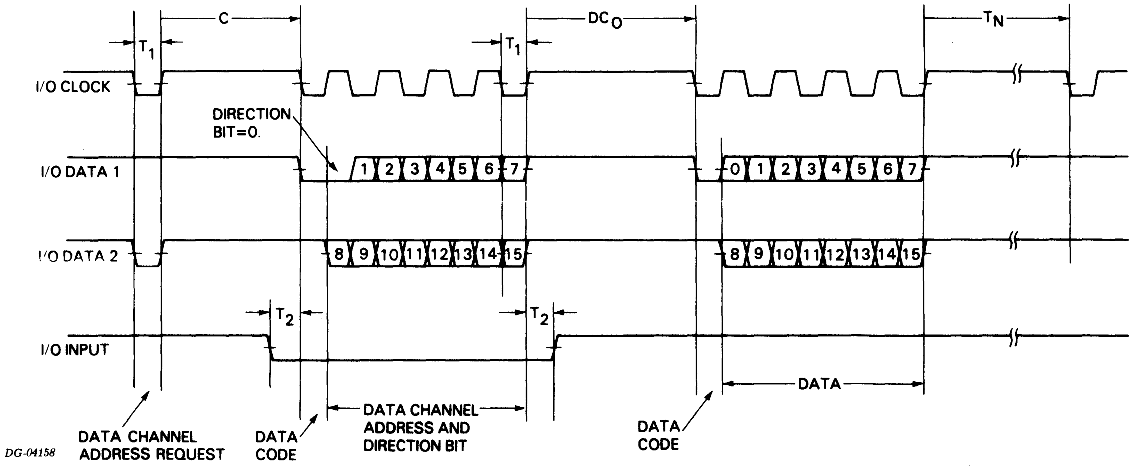
NOTE: See Internal Structure, Data Channel Logic.

DATA CHANNEL TRANSACTIONS

1) CPU to IOC Data Channel Out Transactions

A Data Channel Out Transaction transfers a word of data from memory "out" to an IOC. The IOC begins transmitting the address and direction bit to the CPU during a set interval of time after the reception of the Data Channel Address Request. This is illustrated by C in the timing diagram. Likewise, the IOC must begin reception of the Data transfer a set interval of time after it transmits the Control Word. This interval is illustrated by DC_O.

DATA CHANNEL OUT TIMING

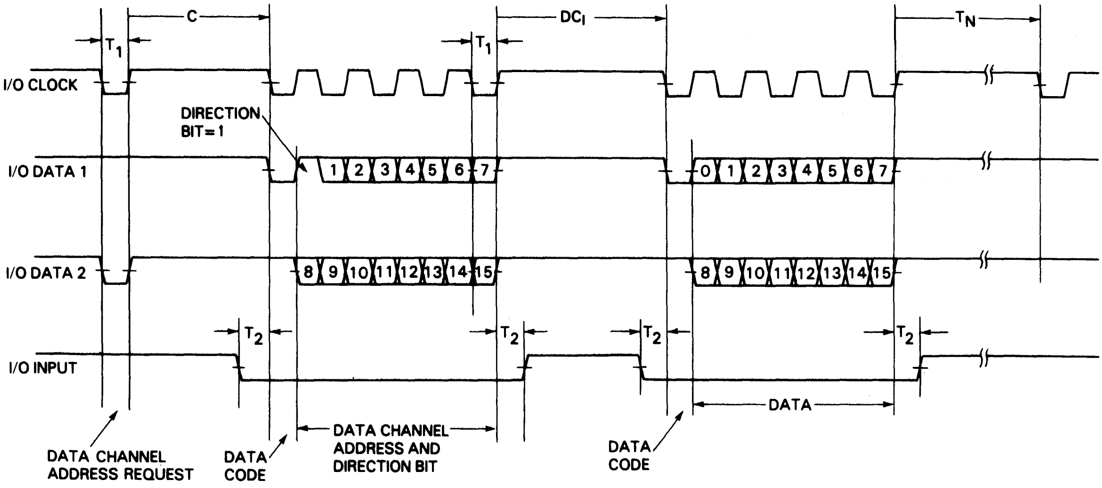


NOTE: UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
SEE TIMING TABLE NEAR THE END OF THIS CHAPTER

2) IOC to CPU Data Channel In Transactions

A Data Channel In Transaction transfers a word of data from an IOC "in" to memory. The IOC begins transmitting the address and direction bit to the CPU a set interval of time after the reception of the Data Channel Address Request. This is illustrated in the timing diagram by C. Likewise, the IOC will begin transmitting the data to the CPU (memory) during a set interval of time after it transmits the Control Word. This interval is illustrated by DC_I.

DATA CHANNEL IN TIMING

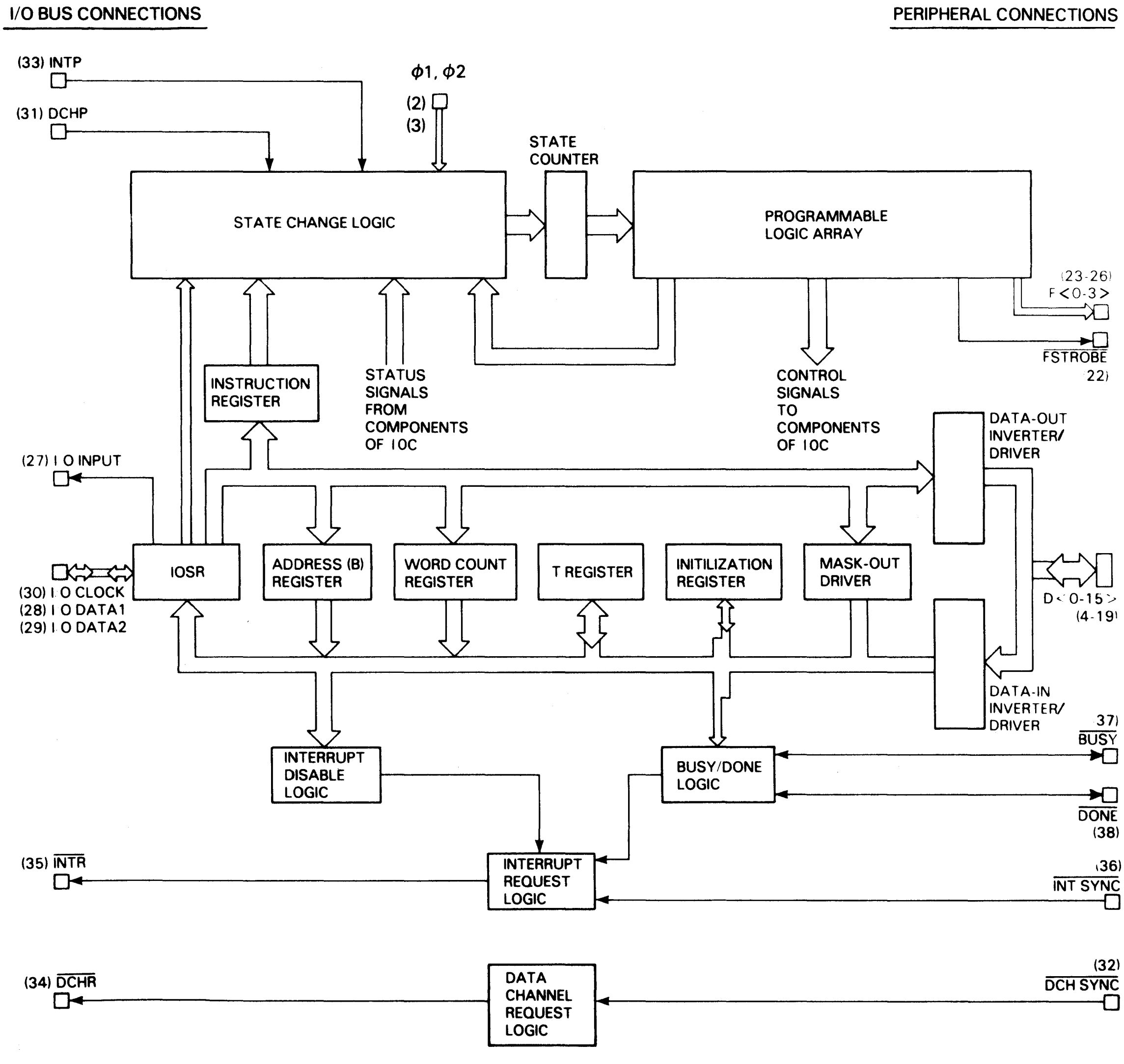


NOTE: UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
SEE TIMING TABLE NEAR THE END OF THIS CHAPTER

DG-04159

IOC INTERNAL STRUCTURE

In order to ınake full use of an IOCs capabilities, an understanding of some of its internal elements is necessary. The block diagram below is a simplified representation of the IOC's internal structure.



DG-04160

Six internal registers are used during transactions as shown in the table below:

REGISTERS	FUNCTION
IOSR (I/O Shift Register)	Performs the serial/parallel conversion of information transferred on I/O DATA <1,2>.
Instruction Register	Holds all I/O instructions received by the IOC.
Initialization Register	Holds the device code, external register enable bit, external BUSY/DONE enable bit, and the polarity bit (see System Requirements, Initialization).
Address Register	Holds the 15-bit memory address used during data channel transactions when internal registers are used.
Word Count Register	Holds the two's complement of the number of words remaining in a data channel block transfer when internal registers are used.
T Register	Buffers the 15-bit memory address and the direction bit used during data channel transfers.

NOTE: Each IOC must have a unique device code. An IOC compares the device code contained in its Initialization Register with that received in all I/O instructions. If they match, the instruction is executed. In addition, some instructions specifying device code 77_8 are executed.

INTERNAL BUSY/DONE LOGIC

The BUSY and DONE flags indicate the state of the peripheral device. In normal use, the BUSY flag being set to 1 indicates the peripheral device is performing an operation; the DONE flag being set to 1 indicates the peripheral device has completed an operation.

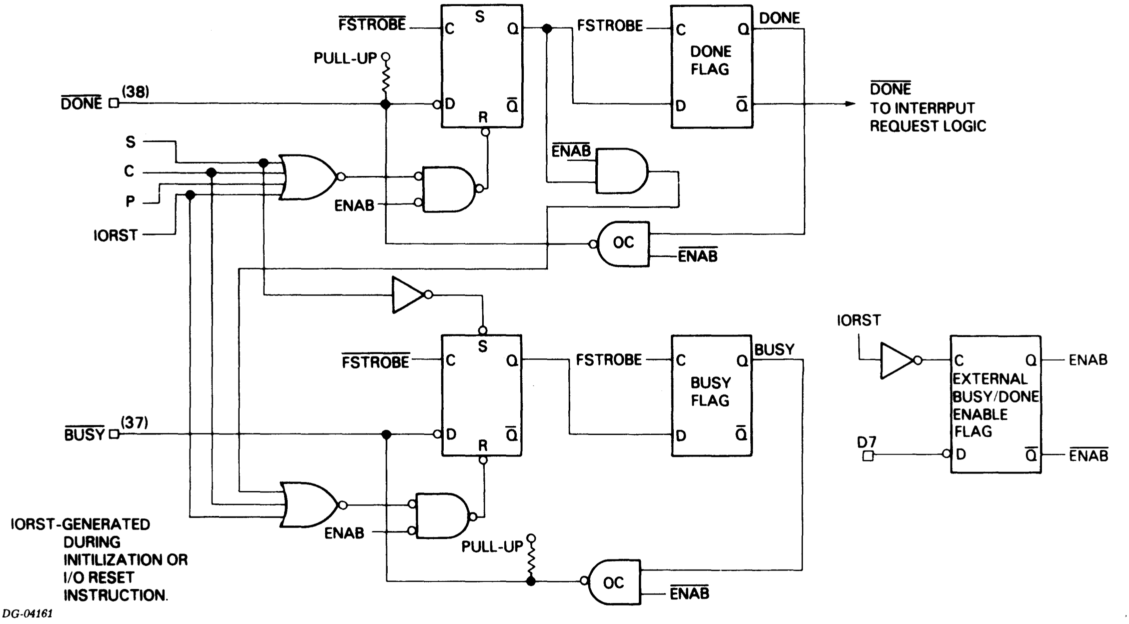
If the internal BUSY and DONE flags are active (External BUSY/DONE Enable bit is 0) they are controlled by both programmed I/O instructions and the peripheral device. For those programmed I/O instructions which contain an IOC's selected device code, bits 8 and 9 (flag control or F field) from the Instruction Register affect the BUSY and DONE flags as follows:

BITS		PROGRAM MNEMONIC	EFFECT
8	9		
0	0	-	No effect
0	1	S	set BUSY to 1, DONE to 0
1	0	C	set both BUSY and DONE to 0
1	1	P	set DONE to 0, no effect on BUSY

DG-04417

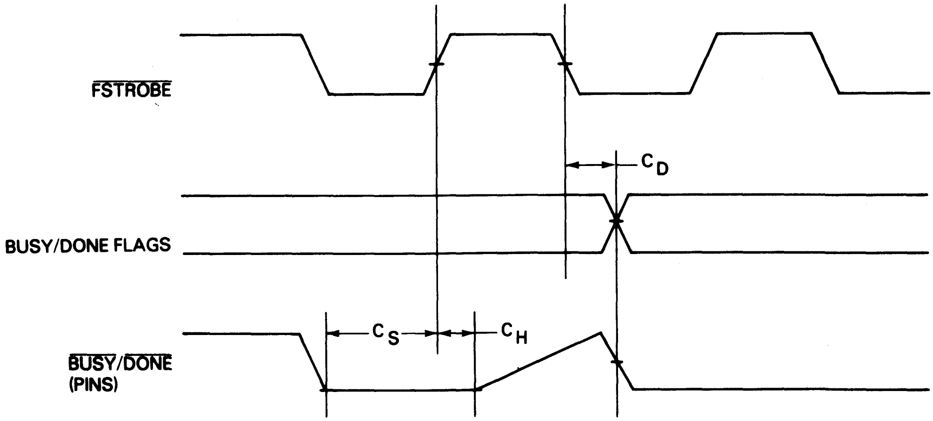
The peripheral device can set either the BUSY or DONE flag to 1. When the DONE flag gets set to 1, the BUSY flag is set to 0. The peripheral device controls these flags by pulling either the $\overline{\text{BUSY}}$ or the $\overline{\text{DONE}}$ lines low. Shown below is a representation of the internal BUSY/DONE logic:

INTERNAL BUSY/DONE LOGIC



There is a set relationship between the time the $\overline{\text{BUSY}}$/$\overline{\text{DONE}}$ lines are asserted by the peripheral device and the time that the internal BUSY/DONE flags are set. The state of the $\overline{\text{BUSY}}$/$\overline{\text{DONE}}$ lines are sampled on the rising edge of $\overline{\text{FSTROBE}}$. However, the internal flags are not set until the next falling edge of $\overline{\text{FSTROBE}}$. If the peripheral device does not hold the $\overline{\text{BUSY}}$/$\overline{\text{DONE}}$ lines low, they may float high until the internal flags are set.

BUSY/DONE TIMING



NOTES:

- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE NEAR THE END OF THIS CHAPTER

DG-04162

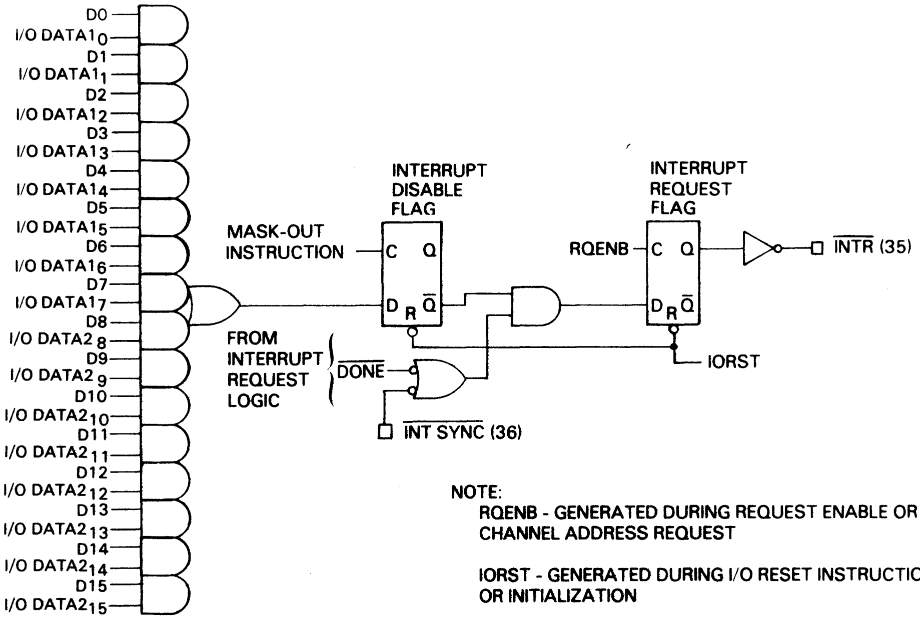
INTERRUPT REQUEST LOGIC

The Interrupt Request Logic is responsible for requesting program interrupts from the CPU. An IOC requests interrupts by asserting its $\overline{\text{INTR}}$ line. The state of this line is controlled by a flag called the Interrupt Request Flag. This flag is set or cleared during either a Request Enable or a Data Channel Address Request Transaction. This ensures that the request line ($\overline{\text{INTR}}$) is stable when it is sampled by the CPU. The section titled "Request Control, Interrupt Requests" discusses how this flag may be set.

Each IOC contains an Interrupt Disable Flag which allows the program to disable interrupts from that IOC. When this flag is set to one, the IOC is prevented from asserting the $\overline{\text{INTR}}$ line.

The Interrupt Disable Flags for all IOCs are manipulated by the Mask-Out Instruction. When this instruction is executed, the 16-bit mask received from the I/O Data Port is logically ANDed with the data received from the Peripheral Port. If any bit of the result is a one, the interrupt Disable Flag is set to one; otherwise, it is set to 0. It is unconditionally cleared by an I/O Reset Instruction.

INTERRUPT REQUEST LOGIC



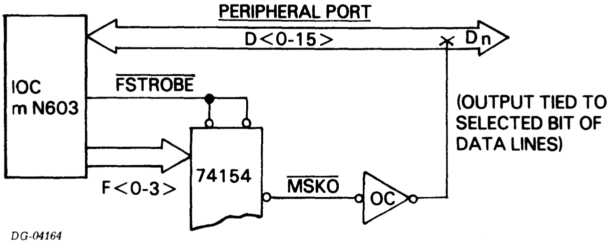
NOTE:
RQENB - GENERATED DURING REQUEST ENABLE OR DATA CHANNEL ADDRESS REQUEST

IORST - GENERATED DURING I/O RESET INSTRUCTION OR INITIALIZATION

DG-04163

A mask-out bit is selected by pulling one of the data lines low during the execution of the Mask-Out Instruction. The MSKO function code generated during this instruction should be used for this purpose. During the time that the MSKO code is valid, the Data Port is forced to interpret all data using negative logic (low = 1). As a result, the buffered output of the function code decoder may be used to tie the selected data line low for the duration of the code as shown below. (see mN601 or mN602 Program Interrupts)

MASKOUT BIT SELECT



DG-04164

DATA CHANNEL LOGIC

The Data Channel Logic is responsible for transferring device service requests to the CPU and performing data channel transfers. A peripheral device requests data channel service by asserting an IOC's $\overline{\text{DCH SYNC}}$ line. On the Reqest Enable or Data Channel Address Request following the assertion of $\overline{\text{DCH SYNC}}$, the IOC's Data Channel Request Flag is set to 1. This flag controls the state of the data channel request line ($\overline{\text{DCHR}}$). When the flag is set to one, the $\overline{\text{DCHR}}$ line is asserted requesting service from the CPU. Synchronizing the request in this way ensures the stability of the $\overline{\text{DCHR}}$ line when it is sampled by the CPU.

Upon receiving data channel service, an IOC must transmit control information about the transfer to the CPU (see I/O Data Port, Data Channel Transactions). If the control information is ready for transmission when service is received, the entire transaction can be accomplished more quickly. Therefore, the internal T Register is loaded with this control information when the service request is made. This shortens the total transacton time.

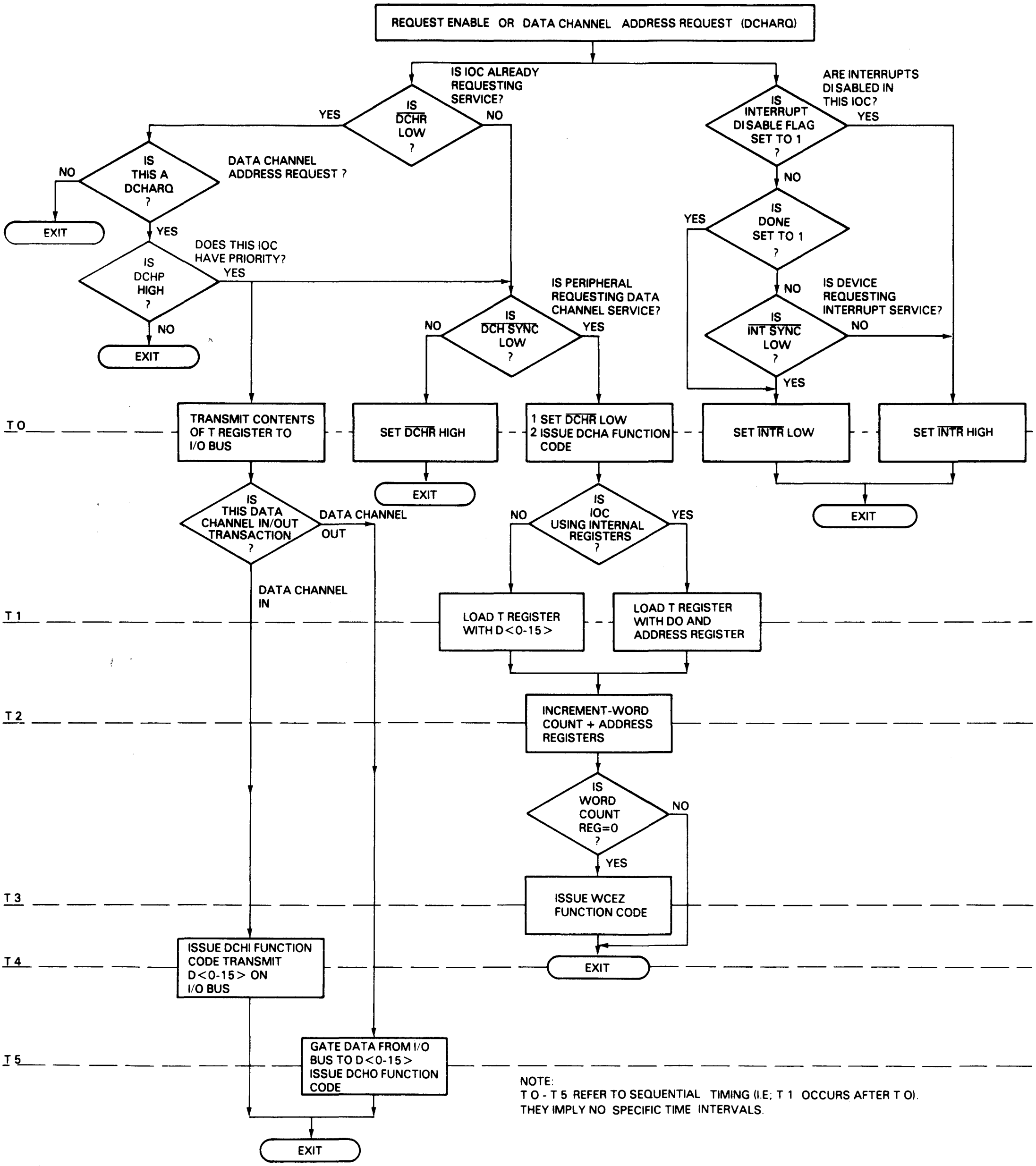
The first thing an IOC does after receiving data channel service from the CPU (a Data Channel Address Request when the IOC has priority), is to transmit the contents of the T Register to the CPU. At the same time, the IOC checks to see if the peripheral is requesting another transfer ($\overline{\text{DCH SYNC}}$ asserted). If the peripheral is ready for a second transfer, the T Register is loaded again. However, if no more transfers are being requested, the Data Channel Request Flag is cleared releasing the $\overline{\text{DCHR}}$ line. In either case, the data word is then transferred completing the first transaction.

Each time the T Register is to be loaded, a DCHA function code is issued to the peripheral. This function code indicates that the IOC expects the appropriate control information on its data lines as described in the Peripheral Port section. Since the T Register may be loaded twice before the first data word is actually transferred, two DCHA function codes may be issued by the IOC before the DCHI/DCHO code which gates the first data word to/from the IOC. In addition, it is possible for one I/O Instruction to be executed between the time data channel service is requested and the time the first transaction occurs. However, once the first transaction commences, the CPU continues to transmit Data Channel Address Requests until all requests are satisfied ($\overline{\text{DCHR}}$ is released).

NOTE: For correct operation at least 4 CPU cycles (1.92 μSec) must exist between a Request Enable and a Data Channel Address Request. the mN601 and the mN602 meet this criterion.

INTERRUPT AND DATA CHANNEL SEQUENCING DIAGRAM

The sequence of events which occur as a result of the IOC receiving a Request Enable or Data Channel Address Request are illustrated in the following flow chart. T0 - T5 refer to the sequential timing of specific events. They in no way imply specific time intervals.


REQUEST ENABLE OR DATA CHANNEL ADDRESS REQUEST (DCHARQ)
IS IOC ALREADY REQUESTING SERVICE?
IS DCHR LOW ?
YES
NO
ARE INTERRUPTS DISABLED IN THIS IOC?
IS INTERRUPT DISABLE FLAG SET TO 1 ?
YES
NO
IS THIS A DCHARQ ?
DATA CHANNEL ADDRESS REQUEST ?
NO
EXIT
YES
DOES THIS IOC HAVE PRIORITY?
IS DCHP HIGH ?
YES
NO
EXIT
IS DONE SET TO 1 ?
YES
NO
IS PERIPHERAL REQUESTING DATA CHANNEL SERVICE?
IS DCH SYNC LOW ?
NO
YES
IS DEVICE REQUESTING INTERRUPT SERVICE?
IS INT SYNC LOW ?
NO
YES
T O
TRANSMIT CONTENTS OF T REGISTER TO I/O BUS
SET DCHR HIGH
1 SET DCHR LOW
2 ISSUE DCHA FUNCTION CODE
SET INTR LOW
SET INTR HIGH
EXIT
EXIT
IS THIS DATA CHANNEL IN/OUT TRANSACTION ?
DATA CHANNEL OUT
DATA CHANNEL IN
IS IOC USING INTERNAL REGISTERS ?
NO
YES
T 1
LOAD T REGISTER WITH D<0-15>
LOAD T REGISTER WITH DO AND ADDRESS REGISTER
T 2
INCREMENT-WORD COUNT + ADDRESS REGISTERS
IS WORD COUNT REG=0 ?
NO
YES
T 3
ISSUE WCEZ FUNCTION CODE
T 4
ISSUE DCHI FUNCTION CODE TRANSMIT D<0-15> ON I/O BUS
EXIT
T 5
GATE DATA FROM I/O BUS TO D<0-15> ISSUE DCHO FUNCTION CODE
EXIT

NOTE:
T 0 - T 5 REFER TO SEQUENTIAL TIMING (I.E; T 1 OCCURS AFTER T 0).
THEY IMPLY NO SPECIFIC TIME INTERVALS.

DG-04165

REQUEST CONTROL

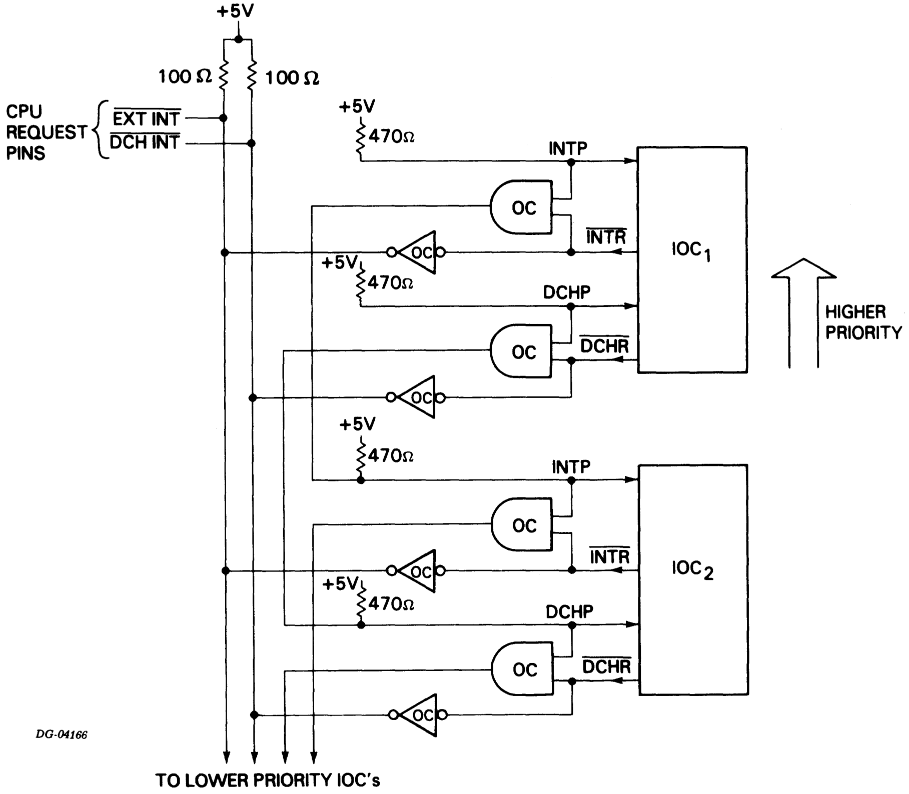
The request control facility uses 6 lines of the IOC; three for program interrupts, INTP, $\overline{\text{INTR}}$, and $\overline{\text{INT SYNC}}$ and three for data channel requests, DCHP, $\overline{\text{DCHR}}$, and $\overline{\text{DCH SYNC}}$.

As mentioned previously, the IOC may request both interrupt and data channel service from the CPU. In both cases, the peripheral device prompts the IOC to request these services. When two or more IOCs are requesting the same type of service, a mechanism for determining which IOC receives service first is necessary. This mechanism is called the priority network.

PRIORITY NETWORKS

Since there are two types of service requests, interrupt and data channel, there are two priority networks. Each network contains a priority line (INTP or DCHP) which is daisy chained from controller to controller. The controller closest to the CPU requiring service can remove priority from all the other controllers further down the priority chain. An IOC has priority for a particular type of service if the appropriate priority pin is high. (See figure below)

PRIORITY CONTROL LOGIC



INTERRUPT REQUESTS

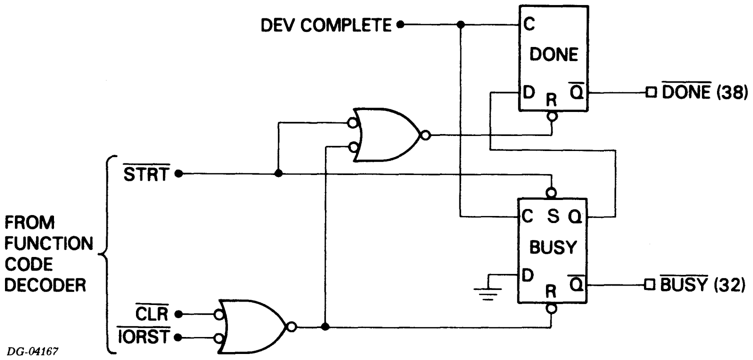
An IOC requests a program interrupt if either its DONE flag is set to one or the peripheral device asserts the IOC's INT SYNC line (assuming that a Mask-out Instruction has not set the Interrupt Disable Bit). On the Request Enable or Data Channel Address Request following the setting of DONE or the assertion of INT SYNC, the IOC requests service (asserts INTR). The service request should remain in effect (DONE set or INT SYNC asserted) until the CPU honors the interrupt request.

The CPU honors a request by issuing the appropriate I/O instruction. If the request was generated as a result of DONE being set to one, the instruction should set DONE to 0 (see Internal Structure, Internal BUSY/DONE Logic). If the request occurred as a result of INT SYNC being asserted, the instruction should specify a function code which the peripheral should use to release INT SYNC. In both cases, the IOC stops requesting interrupt service (asserting INTR) on the Request Enable or Data Channel Address following reception of the I/O Instruction honoring the request.

EXTERNAL BUSY/DONE LOGIC

The following circuit is an example of how external BUSY/DONE flags may be connected to an IOC. In this case, the External BUSY/DONE Enable flag must be set to 1 (see Internal Structure and System Requirements, Initialization).

EXTERNAL BUSY/DONE LOGIC

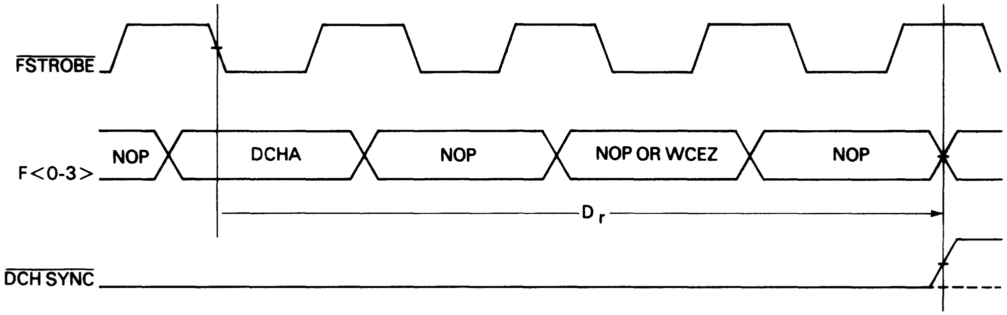


NOTE: This circuit is similar in operation to the internal BUSY/DONE logic. However, the designer may alter this circuit to fit specific applications.

DATA CHANNEL REQUESTS

An IOC requests data channel service if a peripheral device asserts its $\overline{\text{DCH SYNC}}$ line. On the first Request Enable or Data Channel Address Request following the assertion of $\overline{\text{DCH SYNC}}$, the $\overline{\text{DCHR}}$ line is asserted by the IOC, requesting data channel service from the CPU. At this time a DCHA function code is issued to the peripheral. This function code should be used to release the $\overline{\text{DCH SYNC}}$ line unless another data channel transfer is desired. The interval Dr in the following diagram indicates the time in which $\overline{\text{DCH SYNC}}$ must be removed to prevent prompting extraneous data channel requests:

DATA CHANNEL REQUEST TIMING



NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED FROM A 1.5 VOLT LEVEL.
- SEE TIMING TABLE NEAR THE END OF THIS CHAPTER.

DG-04168

DATA CHANNEL REGISTERS

Two registers are used during data channel transfers, a 15 bit address register and a 16 bit word count register. The address register contains the address of the memory location to be accessed during a particular data channel transfer. The word count register contains the 2's complement of the number of words to be transferred in a block transfer. Under normal operation, both registers are incremented each time a transfer takes place. This assumes that the data is being transferred in or out of a contiguous block of memory. When the word count register overflows, a block transfer is complete.

If internal registers are selected (see System Requirements, Initialization), the internal Address and Word Count Registers are used during data channel transactions. In this case, a DOB instruction loads the internal Address Register as well as placing the data on the Peripheral Port data lines. A DOC instruction loads the internal Word Count Register as well as placing the data on the data lines. A DIB instruction transmits the contents of the internal Address Register to the CPU instead of the data received on the data lines.

If external registers are used, they may be loaded and read using programmed I/O instructions as selected by the designer. However, a DCHA function code indicates that the IOC expects the contents of the Address Register to be gated on data lines D<1-15>and the direction bit on D0. This code should also be used to increment both registers, readying them for a subsequent transfer.

INTERRUPT AND DATA CHANNEL SERVICE

If a peripheral requires service, either a Request Enable or a Data Channel Address Request allows the assertion or clearing of an IOC's request lines. The priority networks determine which IOC receiving these commands is allowed to respond. How Request Enable and Data Channel Address Request affect an IOC's interrupt logic, data channel logic, I/O data port, and peripheral port is illustrated in the flow chart shown in the Internal Structure section.

SYSTEM REQUIREMENTS

POWER UP

When powering up the mN603 or the mN613, V_{BB} must be brought within its specified operating range before the clocks Ø 1 and Ø 2 are applied to the chip. After all power supply voltages have reached their operating range, the chip is considered to be in a cleared or Reset state. The four phase internal clock generated by Ø 1 and Ø 2 is halted while an IOC is in this state.

Once initialized, an IOC may be placed in the reset state if its I/O CLOCK line is held low for more than 8 cycles of MASTER CLOCK during reception (I/O INPUT high). However, I/O CLOCK should never be held low while the IOC is transmitting to the CPU (I/O INPUT low).

INITIALIZATION

The initialization process performs two functions. First, the four phase internal clock is started, synchronizing the IOC and the CPU. Second, the interrupt Disable flag, Data Channel Request flag, Busy flag, Done flag, Internal Word Count register, and Address register are cleared (set to 0) while the Initialization register is loaded with information received on the Peripheral Port data lines.

The first two Request Enables received from the CPU while an IOC is in the reset state cause it to be initialized. When an IOC is connected to a microNOVA system, this initialization process occurs automatically. This happens because the CPU issues Request Enables even while it is in the Halt state.

If the IOC is already initialized and communicating with the CPU, its internal registers and flags may be reset and the Initialization Register reloaded if the CPU issues an I/O Reset Instruction.

In either case, the following information is loaded into the Initialization Register via data lines D<7-15>:

IOC INITIALIZATION SIGNALS

DATA PIN(S)	LOADS	SIGNAL LEVEL	EFFECT
D 7	EXTERNAL BUSY/DONE ENABLE BIT	LOW	External BUSY/DONE flags are enabled.
		HIGH	Internal BUSY/DONE flags are enabled.
D 8	EXTERNAL REGISTER ENABLE BIT	LOW	External Address and Word Count Registers are enabled.
		HIGH	Internal Address and Word Count Registers are enabled.
D 9	POLARITY BIT	LOW	Data on D<0-15> interpreted using positive logic. (i.e. high = 1)
		HIGH	Data on D<0-15> interpreted using negative logic. (i.e. high = 0)
D<10-15>	DEVICE CODE	LOW/ HIGH	When loaded, the six bit device code on these lines is interpreted using negative logic (i.e. high = 0).

DG-04378

NOTE: The contents of the polarity bit does not take effect until after Initialization is complete.

CLOCKS

The two phase, non-overlapping clock $\phi 1$ and $\phi 2$ generates the IOC's four phase internal clock. $\phi 1$ and $\phi 2$ are generated from MASTER CLOCK as received from the I/O bus by the IOC I/O Transceiver chip (mN636). A Clock Driver chip (mN640) drives the TTL clock outputs of the Transceiver to the MOS levels required by the IOC. All the specifications and timing in this section assume a MASTER CLOCK frequency of 8.333MHz.

ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS mN603

Supply voltage, V_{BB}	-7V
Supply voltage, V_{CC}	7V
Supply voltage, V_{DD}	13V
Supply voltage, V_{GG}	17V
Input voltage, V_I	7V
Operating free-air temperature range, T_A	0 deg. to 70 deg. C
Storage temperature range, T_{STG}	-55 deg. to 125 deg.C

ABSOLUTE MAXIMUM RATINGS mN613

Supply voltage, V_{BB}	-7V
Supply voltage, V_{CC}	7V
Supply voltage, V_{DD}	17V
Supply voltage, V_{GG}	17V
Input voltage, V_I	7V
Operating free-air temperature range, T_A	0 deg. to 70 deg. C
Storage temperature range, T_{STG}	-55 deg. to 125 deg.C

NOTE: All voltages are measured with respect to ground. Subjecting a circuit to conditions either outside these limits or at these limits for an extended period of time may cause irreparable damage to the circuit. Absolute maximum ratings are not intended to be used during the operation of the circuit.

ABSOLUTE MAXIMUM RATINGS mN615

Supply Voltage, V_{CC}	7V
Supply Voltage, V_{BB}	-7V
Input Voltage, V_1	7V
Operating free-air temperature, T_A	0 deg to 70 deg C
Storage temperature range, T_{STG}	-55 deg to 125 deg C

SPECIFIED OPERATING RANGES mN603

Supply voltage, V_{BB}	-4.25V ± 0.5V
Supply voltage, V_{CC}	5V ± 0.25V
Supply voltage, V_{DD}	10V ± 1V
Supply voltage, V_{GG}	14V ± 1V
Operating free air temperature range, T_A	0 deg. to 70 deg. C

SPECIFIED OPERATING RANGES mN613

Supply voltage, V_{BB}	-5V ± 0.5V
Supply voltage, V_{CC}	5V ± 0.25V
Supply voltage, V_{DD}	12V ± 0.6V
Supply voltage, V_{GG}	12V ± 0.6V
Operating free air temperature range, T_A	0 deg. to 70 deg. C

NOTE: All voltages are measured with respect to ground. On power-up, V_{BB} must be within its specified operating range before any other power supply voltages are applied to the circuit.

SPECIFIED OPERATING RANGES mN615

Internal Voltage V_{BB}*	-2.5 ± 0.25V
Supply Voltage, V_{CC}	5 ± 0.25V
Operating free-air temperature, T_A	0 deg to 70 deg C

*V_{BB} is generated internally and brought out through device pin 1.

NOTE: All voltages are measured with respect to ground.

mN603/mN613 ELECTRICAL SPECIFICATIONS

DC CHARACTERISTICS (mN603/mN613)

SYMBOL	CHARACTERISTICS	CONDITIONS	MIN.	MAX.	UNITS
V_{IH}	Ø1,Ø2		13	15	V
	D <0-15> $\overline{\text{BUSY}}$, $\overline{\text{DONE}}$ INTP, DCHP $\overline{\text{INT SYNC}}$, $\overline{\text{DCH SYNC}}$		3.75	6	V
	I/O CLOCK I/O DATA1,I/O DATA2		2.7	6	V
V_{IL}	Ø1,Ø2		-2	+0.8	V
	D<0-15> $\overline{\text{BUSY}}$, $\overline{\text{DONE}}$ INTP, DCHP $\overline{\text{INT SYNC}}$, $\overline{\text{DCH SYNC}}$		-1	+0.8	V
	I/O CLOCK I/O DATA1,I/O DATA2		-0.5	+0.5	V
I_{IH}	Ø1,Ø2	V_I = 15V		+.01	mA
	D<0-15> $\overline{\text{BUSY}}$, $\overline{\text{DONE}}$ INTP, DCHP $\overline{\text{INT SYNC}}$, $\overline{\text{DCH SYNC}}$	V_I = 4V		+70	μA
	I/O CLOCK I/O DATA1,I/O DATA2	V_I = 2.7V		+70	μA
I_{IL}	φ1,φ2	V_I = 0.8V		10	μA
	I/O CLOCK I/O DATA1,I/O DATA2	V_I = OV		-4	mA
	$\overline{\text{BUSY}}$, $\overline{\text{DONE}}$ INTP, DCHP $\overline{\text{INT SYNC}}$, $\overline{\text{DCH SYNC}}$	V_I = OV		-2	mA
	D<0-15>	V_I = OV		-2	mA
V_{OH}	D<0-15> $\overline{\text{BUSY}}$, $\overline{\text{DONE}}$	I_O = -70μA	4	V_{CC}	V
	I/O CLOCK I/O DATA1,I/O DATA2	I_O = -40μA	3	V_{CC}	V
	F<0-3>, FSTROBE $\overline{\text{DCHR}}$, $\overline{\text{INTR}}$ I/O INPUT	I_O = -0.1mA	2.7	V_{CC}	V
V_{OL}	D<0-15> I/O CLOCK I/O DATA1,I/O DATA2 $\overline{\text{BUSY}}$, $\overline{\text{DONE}}$	I_O = 2mA	0	0.5	V
	I/O INPUT F<0-3> $\overline{\text{FSTROBE}}$ $\overline{\text{DCHR}}$, $\overline{\text{INTR}}$	I_O = 4mA	0	0.5	V
I_{BB}	MAX AVERAGE SUPPLY CURRENT	V_{BB} = -4.25 ± 0.5V		-0.5	mA
I_{CC}	MAX AVERAGE SUPPLY CURRENT	V_{CC} = 5.0 ± 0.25V		25	mA
I_{DD}	MAX AVERAGE SUPPLY CURRENT	V_{DD} = 10.0 ± 1.0V		25	mA

DC CHARACTERISTICS (mN603/mN613) (CONT'D)

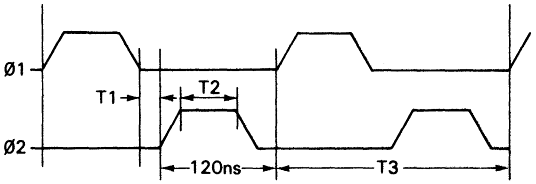
SYMBOL	CHARACTERISTICS	CONDITIONS	MIN.	MAX.	UNITS
I_{GG}	MAX AVERAGE SUPPLY CURRENT	V_{GG} = 14.0 ± 1.0V		20	mA
C_I	Ø1, Ø2 BUSY. DONE			50	pF
	D<0-15> I/O CLOCK I/O DATA1, I/O DATA2 INTP, DCHP INT SYNC, DCH SYNC			30	pF

NOTE: Positive current is into the pin.

AC CHARACTERISTICS

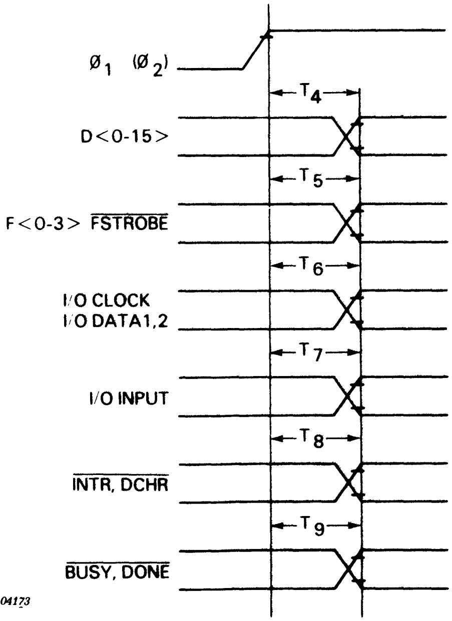
SWITCHING DIAGRAMS (mN603/mN613)

IOC CLOCK TIMING



DG-04023

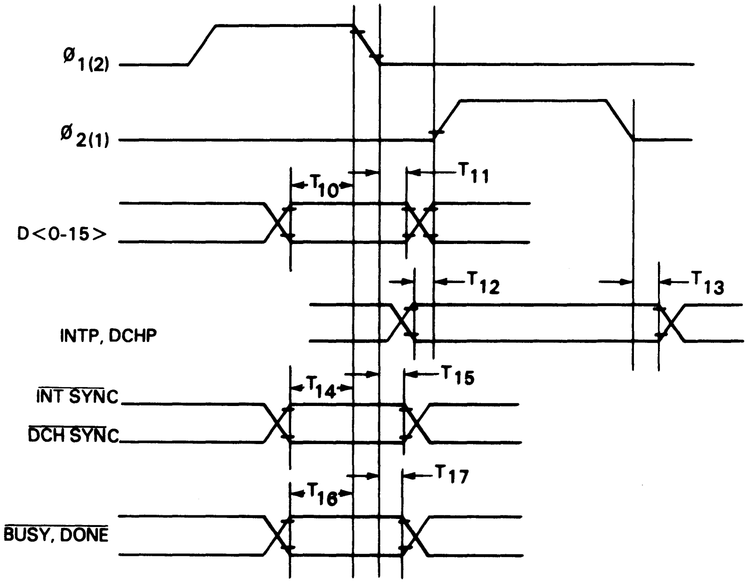
DATA PINS - OUTPUT MODE



DG-04173

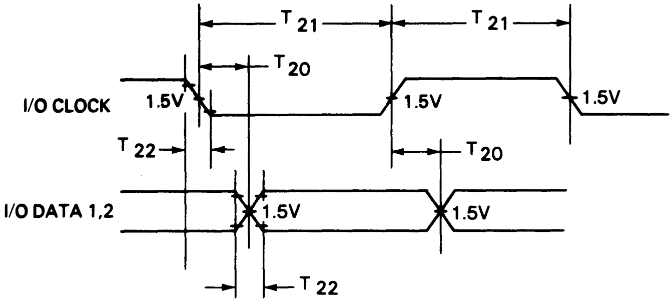
NOTE: TIME INTERVALS ARE MEASURED BETWEEN 10% AND/OR 90% POINTS, UNLESS OTHERWISE SPECIFIED.

DATA PINS - INPUT MODE



DG-04172

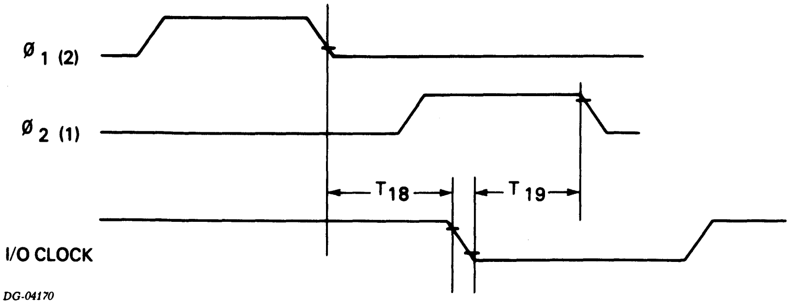
I/O PORT - INPUT MODE



DG-04171

NOTE: TIME INTERVALS ARE MEASURED BETWEEN 10% AND/OR 90% POINTS, UNLESS OTHERWISE SPECIFIED.

I/O CLOCK - OUTPUT MODE



DG-04170

TRANSITION TIMING (mN603/mN613)

CLASS	SYMBOL	PIN	MIN	MAX	UNIT
CLOCKS	T_{1}	Ø1, Ø2 SEPARATION	5	-	ns
	T_{2}	WIDTH	75	-	ns
	T_{3}	CYCLE	235	245	ns
DATA OUTPUTS	T_{4}	D<0-15>	-	60	ns
	T_{5}	F<0-3>, $\overline{\text{FSTROBE}}$	-	60	ns
	T_{6}	I/O CLOCK, I/O DATA1 I/O DATA2	-	30	ns
	T_{7}	I/O INPUT		30	ns
	T_{8}	$\overline{\text{INTR}}$, $\overline{\text{DCHR}}$	-	60	ns
	T_{9}	$\overline{\text{BUSY}}$, $\overline{\text{DONE}}$	-	60	ns
DATA INPUTS	T_{10}	D <0-15> SETUP	50	-	ns
	T_{11}	D <0-15> HOLD	0	-	ns
	T_{12}	INTP, DCHP SETUP	0	-	ns
	T_{13}	INTP, DCHP HOLD	0	-	ns
	T_{14}	$\overline{\text{INT SYNC}}$, $\overline{\text{DCH SYNC}}$ SETUP	50	-	ns
	T_{15}	$\overline{\text{INT SYNC}}$, $\overline{\text{DCH SYNC}}$ HOLD	0	-	ns
	T_{16}	$\overline{\text{BUSY}}$, $\overline{\text{DONE}}$ SETUP	50	-	ns
	T_{17}	$\overline{\text{BUSY}}$, $\overline{\text{DONE}}$ HOLD	0	-	ns
I/O DATA PORT	T_{18}	I/O CLOCK HOLD	0	-	ns
	T_{19}	I/O CLOCK SETUP	70	-	ns
	T_{20}	I/O SKEW	-10	+10	ns
	T_{21}	I/O PULSE WIDTH	115	125	ns
	T_{22}	RISE, FALL TIMES	-	10	ns

NOTE: All the above times assume a MASTER CLOCK frequency of 8.333 Mhz.

DG-04412

TIMING TABLES (mN603/mN613/mN615)

I/O DATA PORT TIMING TABLE

CLASS	MNEMONIC	MIN.	MAX.	UNITS
All	T_{1}	110	130	ns
	T_{2}	110	130	ns
	T_{SKEW}	-5	+15	ns
	T_{N} (next command)	840	--	ns
I/O Instructions	DT_{O} (Data Out Transfer)	470	850	ns
	DT_{I} (Data In Transfer)	1190	1330	ns
Request Enable and Data Channel Address Request	a	360	540	ns
Data Channel Transactions	C (Data Channel Address Request to first transfer)	710	850	ns
	DC_{O} (Data Channel Out)	590	1930	ns
	DC_{I} (Data Channel In)	1190	1210	ns

NOTE: All the above times assume a MASTER CLOCK frequency of 8.333Mhz.

DG-04413

PERIPHERAL PORT TIMING TABLE

OPERATION	MNEMONIC	DESCRIPTION	MIN.	MAX.	UNIT
Common to all operations	F_{S}	Function code setup time prior to $\overline{\text{FSTROBE}}$	60		ns
	F_{T}	$\overline{\text{FSTROBE}}$ duration	180	300	ns
	F_{H}	Function pin hold time	60		ns
I/O data out Instruction and Data Channel Out Transfer		Data output timing same as function code timing above			
I/O data in Instruction and Data Channel In Transfer	D_{S}	Data setup time prior to $\overline{\text{FSTROBE}}$	120	240	ns
	D_{H}	Data hold time after $\overline{\text{FSTROBE}}$	0	120	ns
	T_{AI}	DCHA to DHCI	3.36		µs
	T_{AO}	DCHA to DCHO	6.56		µs

DG-04415

SYNCHRONIZATION TIMING TABLE

OPERATION	MNEMONIC	DESCRIPTION	MIN.	MAX.	UNIT
Request Control	D_{r}	Time in which $\overline{\text{INT SYNC}}$ or $\overline{\text{DCH SYNC}}$ must be removed to prevent further requests		840	ns
BUSY and DONE	C_{D}	Delay between $\overline{\text{FSTROBE}}$ and $\overline{\text{BUSY}}$ or $\overline{\text{DONE}}$ transition	-10	+10	ns
	C_{S}	$\overline{\text{BUSY}}$ or $\overline{\text{DONE}}$ assertion setup time for internal recognition of desired functions	120		ns
	C_{H}	$\overline{\text{BUSY}}$ or $\overline{\text{DONE}}$ assertion hold time for recognition (Note: if C_{H} 240 nS then signal will float toward Vcc until next $\overline{\text{FSTROBE}}$)	0		ns

NOTE: All the above times assume a MASTER CLOCK frequency of 8 333 Mhz.

DG-04414

DC CHARACTERISTICS (mN615)

SYMBOL	CHARACTERISTICS	CONDITIONS	MIN	MAX	UNITS
V_{IH}	Ø1 Ø2		4.5	5.0	V
	D<0-15> $\overline{\text{BUSY}}$ $\overline{\text{DONE}}$ INTP DCHP $\overline{\text{INT SYNC}}$ $\overline{\text{DCH SYNC}}$		2.5	6.0	V
	I/O CLOCK I/O DATA<1-2>		2.7	6.0	V
V_{IL}	Ø1 Ø2		-2	0.3	V
	D<0-15> $\overline{\text{BUSY}}$ $\overline{\text{DONE}}$ INTP DCHP $\overline{\text{INT SYNC}}$ $\overline{\text{DCH SYNC}}$		-1.0	0.8	V
	I/O CLOCK I/O DATA<1-2>		-0.5	0.5	V
I_{IH}	Ø1 Ø2	V_{IH} =5.00V		0.1	mA
	D<0-15> $\overline{\text{BUSY}}$ $\overline{\text{DONE}}$ INTP DCHP $\overline{\text{INT SYNC}}$ $\overline{\text{DCH SYNC}}$	V_{IH} =3.0V	-70		µA
	I/O CLOCK I/O DATA<1-2>	V_{IH} = 2.7V	-40		µA
I_{IL}	Ø1 Ø2	V_{IL} = 0.3V		10	µA
	D<0-15> $\overline{\text{BUSY}}$ $\overline{\text{DONE}}$ INTP DCHP $\overline{\text{INT SYNC}}$ $\overline{\text{DCH SYNC}}$	V_{IL} = 0.0V		-0.5	mA
	I/O CLOCK I/O DATA<1-2>	V_{IL} = 0.0V		-0.2	mA
V_{OH}	D<0-15> $\overline{\text{BUSY}}$ $\overline{\text{DONE}}$	I_O = -70uA	3.0		V
	F<0-3> $\overline{\text{FSTROBE}}$ $\overline{\text{DCHR}}$ $\overline{\text{INTR}}$ I/O INPUT	I_O = -0.1mA	2.7		V
	I/O CLOCK I/O DATA<1-2>	I_O = -40uA	3.0		V

NOTE: The V_{BB} is internally generated and should not be supplied. Positive current is into the device.

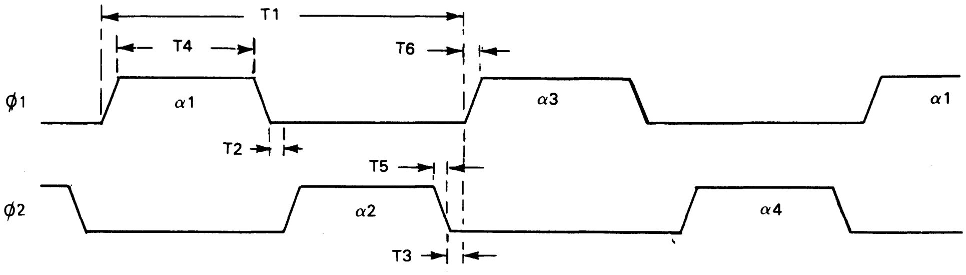
mN615 ELECTRICAL SPECIFICATIONS

SYMBOL	CHARACTERISTICS	CONDITIONS	MIN	MAX	UNITS
V_{OL}	D<0-15> $\overline{\text{BUSY}}$ $\overline{\text{DONE}}$	I_O = 2mA		0.5	V
	F<0-3> $\overline{\text{FSTROBE}}$ $\overline{\text{DCHR}}$ $\overline{\text{INTR}}$ I/O INPUT	I_O = 4mA		0.5	V
	I/O CLOCK I/O DATA<1-2>	I_O = 2mA		0.5	V
I_{OH}	D<0-15> $\overline{\text{BUSY}}$ $\overline{\text{DONE}}$	V_O =3.0V		-70	μA
	F<0-3> $\overline{\text{FSTROBE}}$ $\overline{\text{DCHR}}$ $\overline{\text{INTR}}$ I/O INPUT	V_O =2.7V		-100	μA
	I/O CLOCK I/O DATA<1-2>	V_O =3.0V		-40	μA
I_{OL}	D<0-15> $\overline{\text{BUSY}}$ $\overline{\text{DONE}}$	V_O =0.5V		2	mA
	$\overline{\text{FSTROBE}}$ $\overline{\text{DCHR}}$ $\overline{\text{INTR}}$ I/O INPUT	V_O =0.5V		4	mA
	I/O CLOCK I/O DATA<1-2>	V_O =0.5V		2	mA
I_{BB}		V_{BB} = -2.5 ± 0.25V		500	μA
I_{CC}		V_{CC} = 5 ± 0.25V		100	mA
C_I	Ø1, Ø2			200	pF
	$\overline{\text{BUSY}}$ $\overline{\text{DONE}}$ INTP DCHP $\overline{\text{INTSNYC}}$ $\overline{\text{DCHSYNC}}$ I/O CLOCK I/O DATA<1-2>			7	pF
C_O	D<0-15> $\overline{\text{BUSY}}$ $\overline{\text{DONE}}$ F<0-3> $\overline{\text{FSTROBE}}$ $\overline{\text{DCHR}}$ $\overline{\text{INTR}}$ I/O INPUT I/O CLOCK I/O DATA<1-2>			50	pF

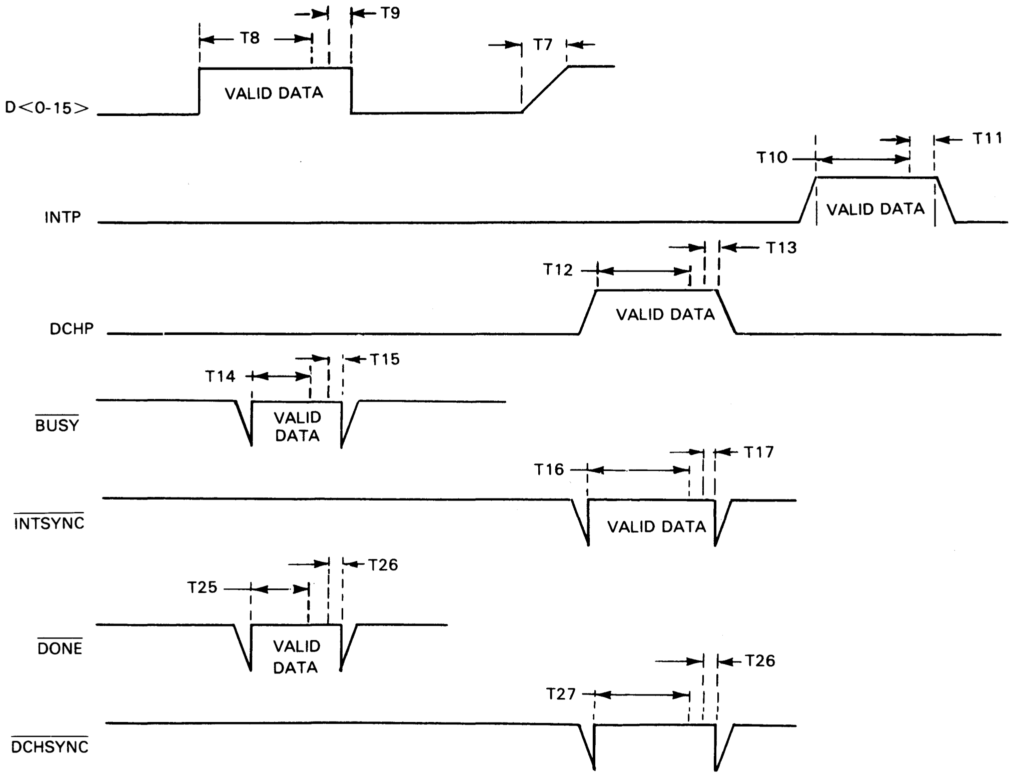
NOTE: The V_{BB} is internally generated and should not be supplied. Positive current is into the device.

AC CHARACTERISTICS SWITCHING DIAGRAM

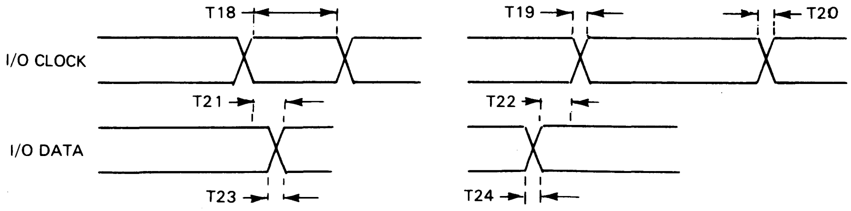
mN615 CLOCK TIMING

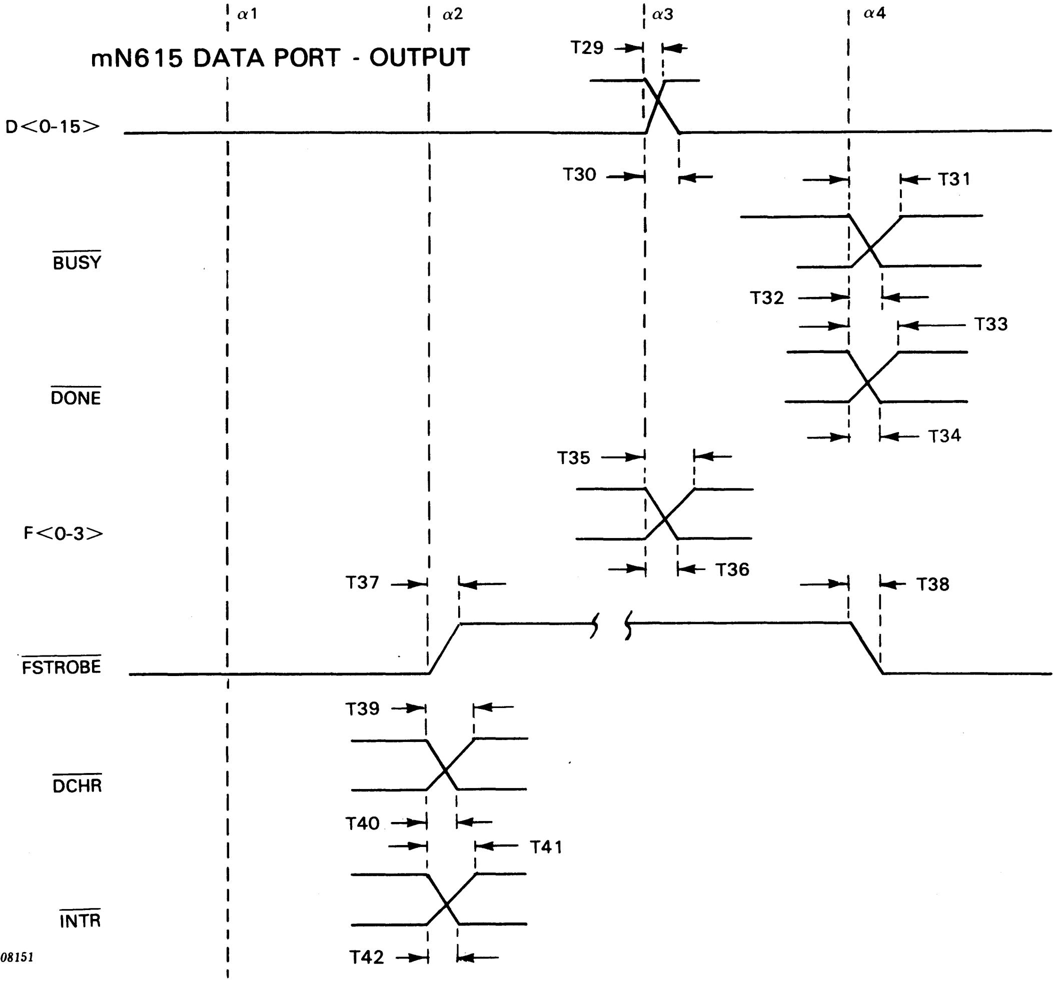


mN615 DATA PORT - INPUT

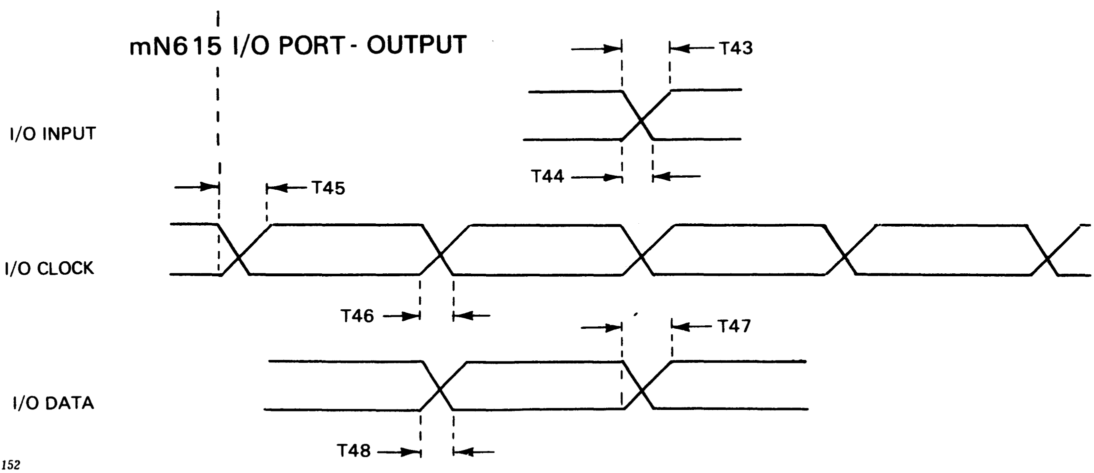


mN615 I/O PORT - INPUT





DG-08151



DG-08152

TRANSITION TIMING (mN615)

CLASS	SYMBOL	PIN	MIN	MAX	UNIT
CLOCKS	T_1	CYCLE PERIOD	235	245	ns
	T_2	CLOCK SEPARATION $\alpha 1$ to $\alpha 2$ or $\alpha 3$ to $\alpha 4$	5		ns
	T_3	CLOCK SEPARATION $\alpha 2$ to $\alpha 3$ or $\alpha 3$ to $\alpha 4$	5		ns
	T_4	CLOCK PULSE WIDTH	75		ns
	T_5	CLOCK FALL TIME	5	25	ns
	T_6	CLOCK RISE TIME	5	25	ns
DATA PORT -- INPUT	T_7	D<0-15> PRECHARGE TIME		60	ns
	T_8	D<0-15> SETUP TIME	50		ns
	T_9	D<0-15> HOLD TIME	0		ns
	T_{10}	INTP SETUP TIME	75		ns
	T_{11}	INTP HOLD TIME	0		ns
	T_{12}	DCHP SETUP TIME	75		ns
	T_{13}	DCHP HOLD TIME	0		ns
	T_{14}	$\overline{\text{BUSY}}$ SETUP TIME	50		ns
	T_{15}	$\overline{\text{BUSY}}$ HOLD TIME	0		ns
	T_{16}	$\overline{\text{INTSYNC}}$ SETUP TIME	50		ns
	T_{17}	$\overline{\text{INTSYNC}}$ HOLD TIME	0		ns
	T_{25}	$\overline{\text{DONE}}$ SETUP TIME	50		ns
	T_{26}	$\overline{\text{DONE}}$ HOLD TIME	0		ns
	T_{27}	$\overline{\text{DCHSYNC}}$ SETUP TIME	50		ns
	T_{28}	$\overline{\text{DCHSYNC}}$ HOLD TIME	0		ns
I/O PORT -- INPUT	T_{18}	I/O CLOCK PULSE WIDTH	115	125	ns
	T_{19}	I/O CLOCK RISE TIME		10	ns
	T_{20}	I/O CLOCK FALL TIME		10	ns
	T_{21}	I/O DATA LAG TIME FROM I/O CLOCK		10	ns
	T_{22}	I/O DATA LEAD TIME OF I/O CLOCK		10	ns
	T_{23}	I/O DATA RISE TIME		10	ns
	T_{24}	I/O DATA FALL TIME		10	ns

NOTE: All timing points measured at 10% and 90% points.

All the above times assume a MASTER CLOCK frequency of 8.333 MHz.

TRANSITION TIMING (mN615)

CLASS	SYMBOL	PIN	MIN	MAX	UNIT
DATA PORT -- OUTPUT	T_{29}	D<0-15> TRUE STATE DELAY		60	ns
	T_{30}	D<0-15> FALSE STATE STATE		60	ns
	T_{31}	$\overline{\text{BUSY}}$ TRUE STATE DELAY		60	ns
	T_{32}	$\overline{\text{BUSY}}$ FALSE STATE DELAY		60	ns
	T_{33}	$\overline{\text{DONE}}$ TRUE STATE DELAY		60	ns
	T_{34}	$\overline{\text{DONE}}$ FALSE STATE DELAY		60	ns
	T_{35}	F<0-3> TRUE STATE DELAY		60	ns
	T_{36}	F<0-3> FALSE STATE DELAY		60	ns
	T_{37}	$\overline{\text{FSTROBE}}$ PRECHARGE TIME		60	ns
	T_{38}	$\overline{\text{FSTROBE}}$ FALSE STATE DELAY		60	ns
	T_{39}	$\overline{\text{DCHR}}$ TRUE STATE DELAY		60	ns
	T_{40}	$\overline{\text{DCHR}}$ FALSE STATE DELAY		60	ns
	T_{41}	$\overline{\text{INTR}}$ TRUE STATE DELAY		60	ns
	T_{42}	$\overline{\text{INTR}}$ FALSE STATE DELAY		60	ns
I/O PORT -- OUTPUT	T_{43}	I/O INPUT TRUE STATE DELAY		45	ns
	T_{44}	I/O INPUT FALSE STATE DELAY		45	ns
	T_{45}	I/O CLOCK TRUE STATE DELAY		45	ns
	T_{46}	I/O CLOCK FALSE STATE DELAY		45	ns
	T_{47}	I/O DATE TRUE STATE DELAY		45	ns
	T_{48}	I/O DATA FALSE STATE DELAY		45	ns

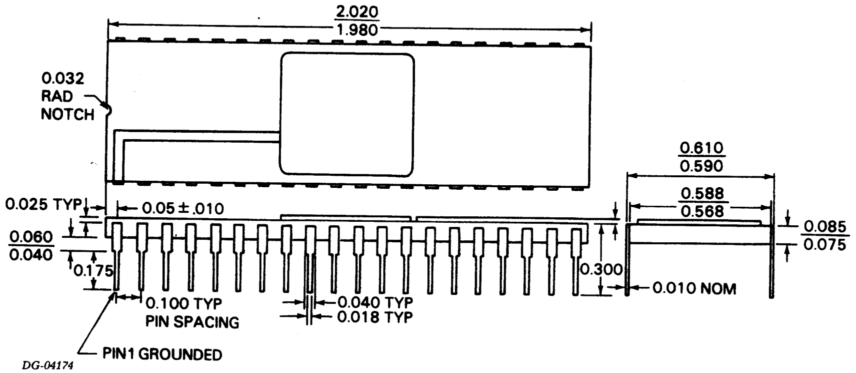
NOTE: All timing points measured at 10% and 90% points.

All the above times assume a MASTER CLOCK frequency of 8.333 MHz.

PACKAGE SPECIFICATION

The physical dimensions (in inches) of the mN603/mN613/mN615 chips are given in the diagram below.

DIMENSIONS OF CHIP



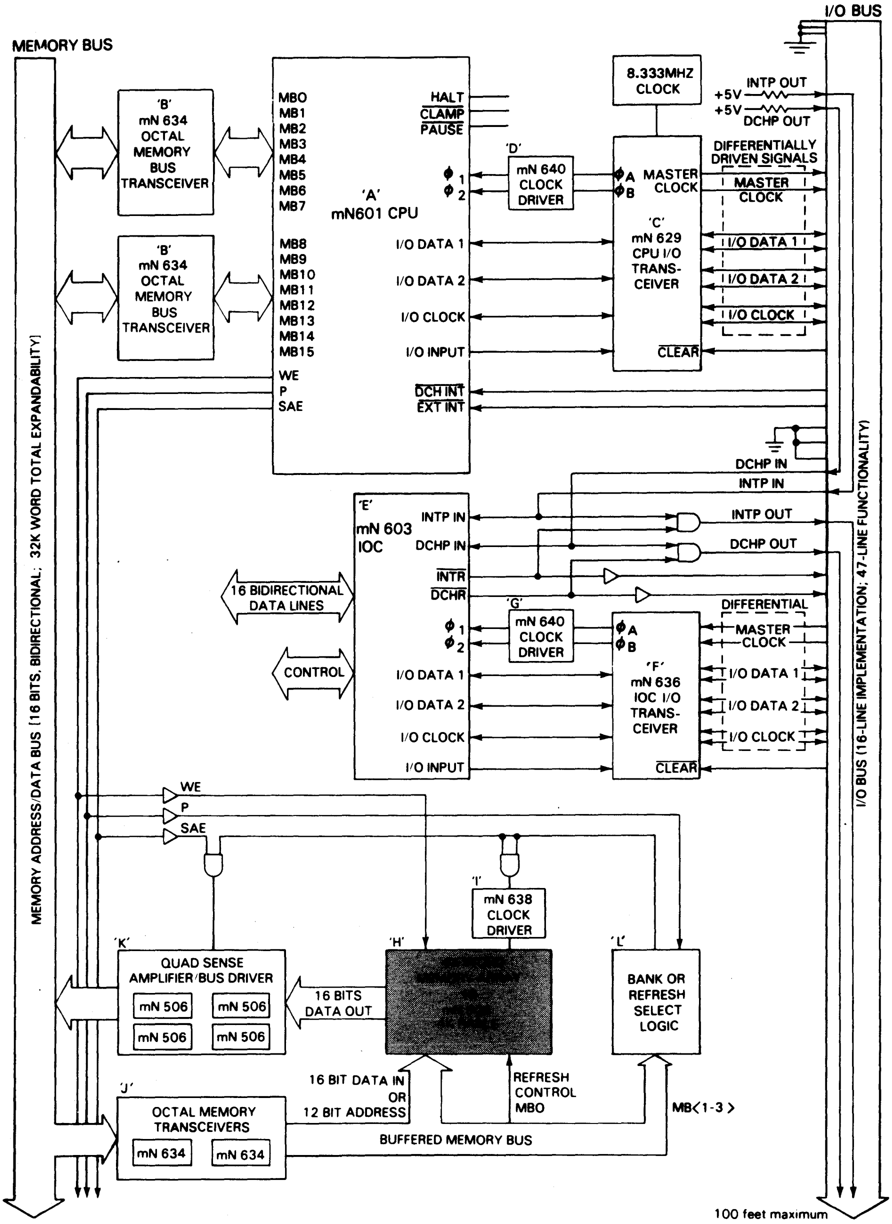
DG-04174

4

mN606
4K DYNAMIC RANDOM ACCESS MEMORY

5

mN606
4K DYNAMIC RANDOM ACCESS MEMORY



DG-04334

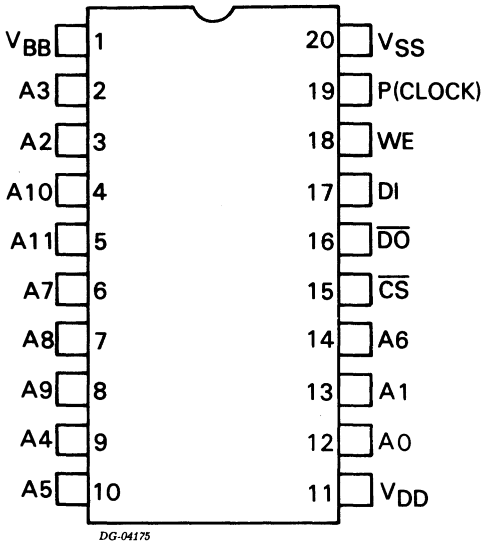
mN606

4K X 1 DYNAMIC RANDOM ACCESS MEMORY

FEATURES

- NMOS DYNAMIC RANDOM ACCESS MEMORY IN A 20 PIN PLASTIC PACKAGE
- ORGANIZED AS 4096 BY 1 BIT
- FAST 180nS ACCESS TIME
- FAST INTERNAL LATCHES FOR ADDRESS AND CHIP SELECT
- REFRESHES 64 BITS AT A TIME
- OPEN DRAIN OUTPUT

PACKAGE



GENERAL DESCRIPTION

The mN606 is a 4096 by 1 bit NMOS memory using cost-efficient dynamic RAM technology to implement the microNOVA family's large memory orientation. Its 20-pin package permits separate pins for each of the 12 address lines as well as for the data in and data out lines. The chip's high speed access time of 180 nanoseconds contributes significantly to the microNOVA family's high performance.

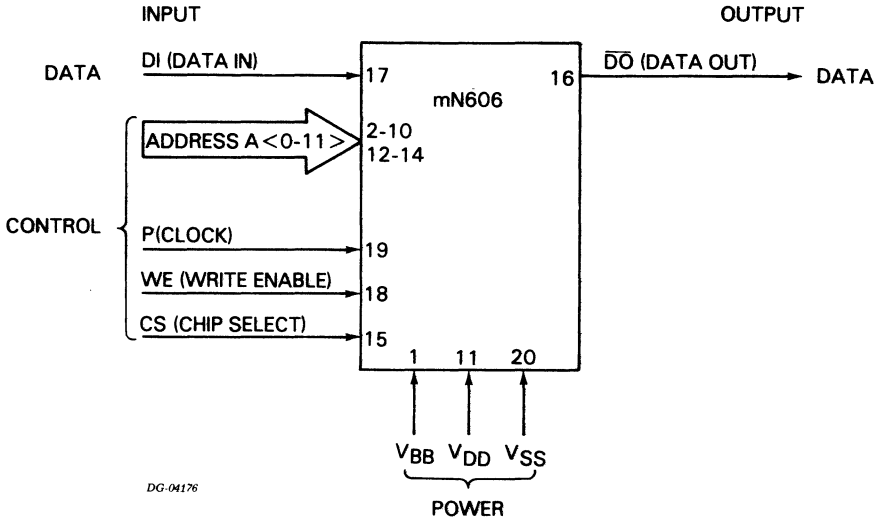
To construct a 16 bit by 4K word memory board, 16 mN606 RAMs are needed as well as two mN634 Memory Transceivers, four mN506 Sense Amp/Bus Drivers, one mN638 Clock Driver, and supporting components. The system block diagram illustrates the basic interconnection of these components.

NOTE: The mN606 has been discontinued. Information has been included to assist those who already have the mN606.

PIN DESCRIPTIONS

The diagram below shows the pin connections and the table describes the function of each pin shown in the diagram.

FUNCTIONAL PIN CONNECTION DIAGRAM



DG-04176

PIN FUNCTIONS

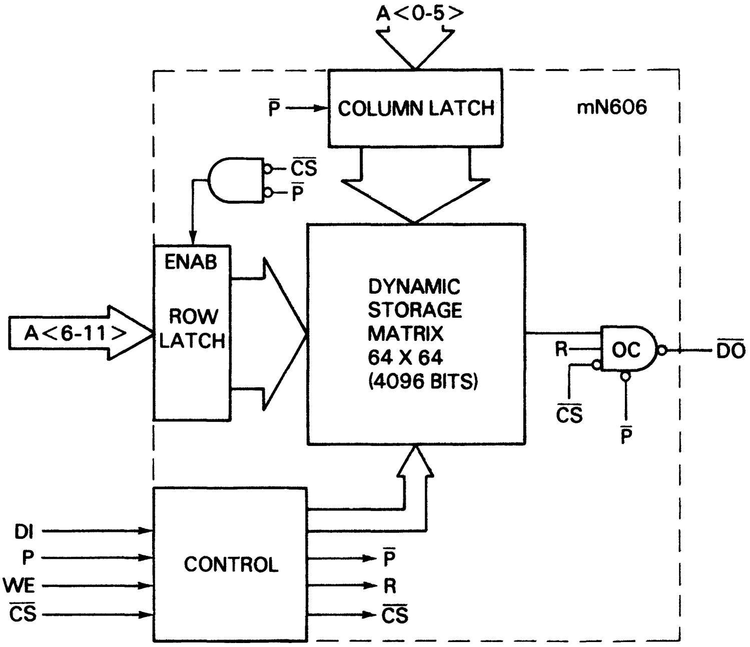
MNEMONIC	PIN NO.	IN/ OUT	FUNCTION
A<0-11> (Address)	2-10 12-14	In	These pins enter an address which selects one of 4096 bits. A<0-5>is used during refresh operations for column selection (1/64th of the RAM). Address is latched on rising edge of P clock.
DI (Data In)	17	In	One bit of data written into location specified on address lines during memory write operation. High level logical one input.
$\overline{\text{DO}}$ (Data Out)	16	Out	One bit of data read from location specified by address pins during memory read operation. High impedence, inverted output (i.e., low = 1, high = 0).
P (Clock)	19	In	All memory operations initiated by latching Address Register and starting internal clock on rising edge of P clock.
WE (Write Enable)	18	In	When asserted high, enables data from DI to be written into specified memory location. If low, enables data at the specified address to be read out to the DO pin.
$\overline{\text{CS}}$ (Chip Select)	15	In	If asserted low when P goes high, a memory read or write operation is initiated. If high when P goes high, a memory refresh operation is initiated.
V_{BB}	1		-4.25 ± 0.5V
V_{DD}	11		+14.0 ± 1.0V
V_{SS}	20		Ground

INTERNAL STRUCTURE

The operation performed by the RAM depends on the level of CS when P clock goes high. If CS is low, a memory Read or Write operation is performed. The address on pins A <0-11> is latched on the rising edge of P clock. If WE remains low, the data appears on DO. A write operation is initiated by pulling WE high. The rising edge of WE forces DO high while the falling edge loads the data from DI. Both operations may be combined to form a Read/Delay/Write (ie Read/Modify/ Write) operation.

Since the memory mechanism is dynamic, the chip must periodically be refreshed. A Refresh operation is initiated during a memory Write operation if CS remains high. In this case, A<0-5> are latched on the rising edge of P clock to select a particular 64 bit column of memory to be refreshed. Since there are 64 columns, 64 refresh operations are necessary to refresh all 4096 memory cells. Each location must be refreshed at least once every 2.0 mS. The Data Out (DO) line remains high throughout a refresh operation.

INTERNAL BLOCK DIAGRAM



DG-04177

ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS

Supply voltage, V_{BB}	-7V
Supply voltage, V_{DD}	17V
Clock input voltage, V_{ICLK}	17V
All other inputs, V_I	7V
Operating temperature range,(air moving at 200 feet/minute),T_A	0 deg. to 70 deg. C
Storage temperature range,T_{STG}	-55 deg. to 85 deg. C

NOTE: Subjecting a circuit to conditions either outside these limits or at these limits for an extended period of time may cause irreperable damage to the circuit. These ratings are not intended to be used during the operation of the circuit.

RECOMMENDED OPERATING CONDITIONS

Supply voltage, V_{BB}	-4.25 ± 0.5V
Supply voltage, V_{DD}	14 ± 1V
Operating temperature range,(air moving at 200 feet/minute)	0 deg. to 70 deg. C

NOTE: On power-up, V_{BB} must be within its specified operating range before V_{DD} is applied to the circuit.

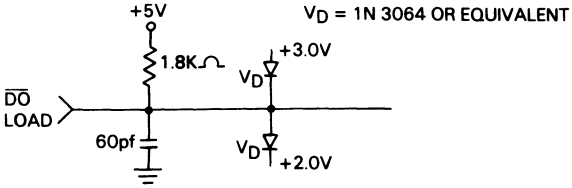
DC CHARACTERISTICS

SYMBOL	CHARACTERISTICS	CONDITIONS	MIN.	MAX.	UNITS
V_{IH}	P(CLOCK)		+13.0	+15.5	V
	A <0-11> $\overline{CS}$, WE, DI		+4.0	+6.0	V
V_{IL}	P(CLOCK), A <0-11> $\overline{CS}$, WE, DI		-1.0	+0.8	V
I_{IH}	P(CLOCK)	V_I = 15.5V	---	+0.8	mA
	$\overline{CS}$, WE, DI A <0-11>	V_I = 6.0V	---	+0.01	mA
I_{IL}	P(CLOCK) $\overline{CS}$, WE, DI	V_I = +0.8V	---	+0.01	mA
I_{OH}	$\overline{DO}$	SEE TEST CIRCUIT, UNDER AC CHARACTERISTICS	0	0.04	mA
I_{OL}	$\overline{DO}$	SEE TEST CIRCUIT, UNDER AC CHARACTERISTICS	1.5	9.0	mA
I_{BB}	SUPPLY CURRENT	V_{BB} = -4.25 ± 0.5V	---	-0.15	mA
I_{DD}	SUPPLY CURRENT	V_{DD} = +14.0 ± 1.0V	---	+20.0	mA
C_I	P(CLOCK)			23	pF
	A <0-11> $\overline{CS}$, WE, DI			7	pF
R_T	MEMORY CELL REFRESH TIME		---	2.0	mS

NOTE: Positive current is into the pin.

AC CHARACTERISTICS

TEST CIRCUIT

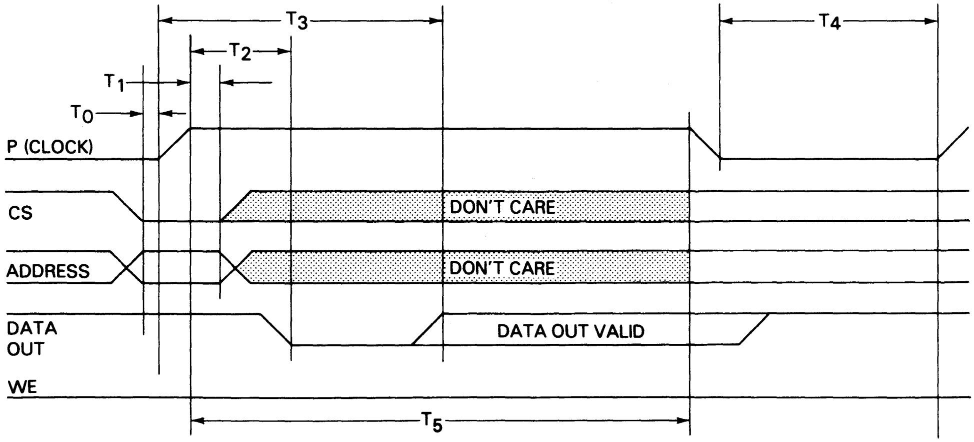


TEST CONDITIONS
P(CLOCK) RISE AND FALL 20NS
ALL OTHERS 10NS
ALL RISE AND FALL TIMES ARE MEASURED
FROM THE 10 TO 90 PERCENT POINTS.

LOAD CAPACITOR INCLUDES
JIG AND WIRING CAPACITANCE.

DG-04178

READ OPERATION

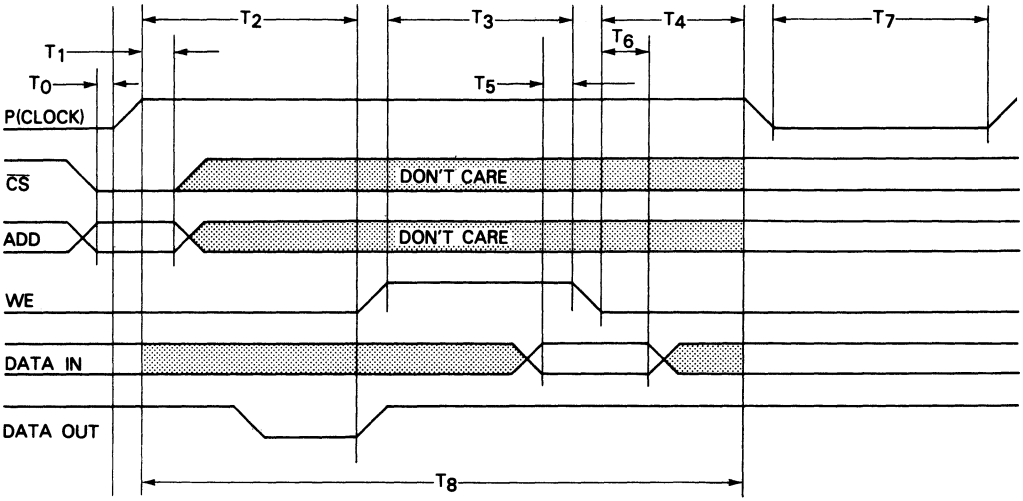


WE MUST REMAIN LOW DURING THE ENTIRE READ CYCLE.

NOTE: SEE AC CHARACTERISTICS TIMING TABLE ON PAGE 7.

DG-04331

WRITE OPERATION

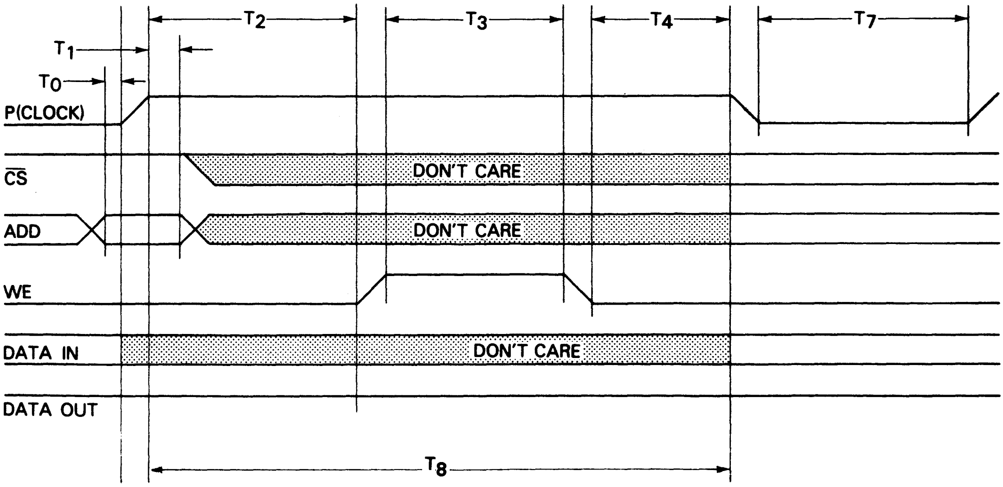


WHEN WE GOES HIGH, A WRITE CYCLE IS INITIATED AND DATA OUT IS DISABLED.

NOTE: SEE AC CHARACTERISTICS TIMING TABLE ON PAGE 7.

DG-04179

REFRESH OPERATION



WHEN REFRESHING, $\overline{CS}$ MUST BE HELD
HIGH FOR THE TIME T1. THE DATA INPUTS
ARE IGNORED.

NOTE: SEE AC CHARACTERISTICS TIMING TABLE ON PAGE 7.

DG-04180

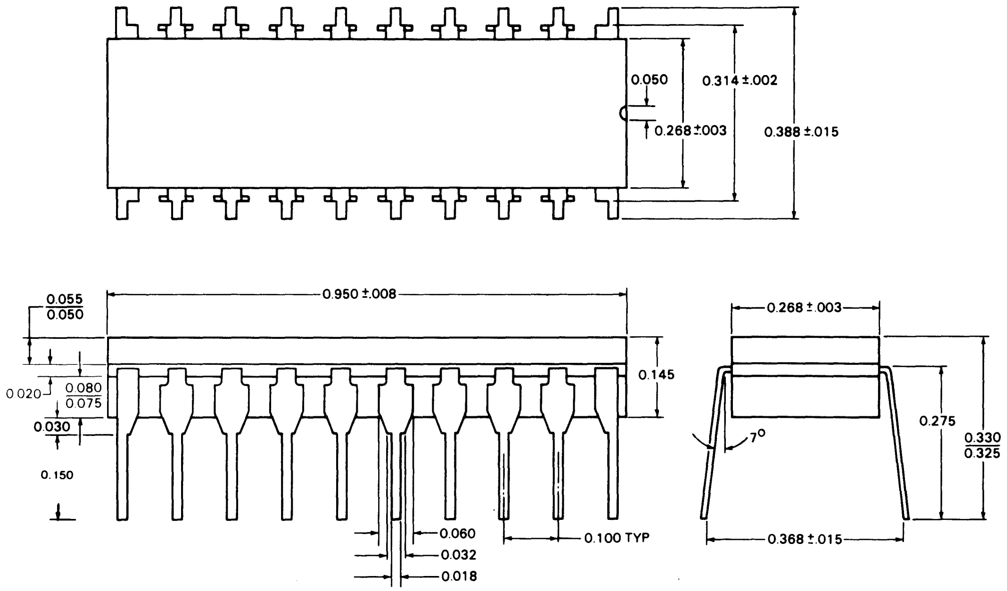
TRANSITION TIMING

OPERATION	SIGNAL	DESCRIPTION	MIN.	MAX.	UNITS
READ	T_0	SETUP TIME ADDRESS AND $\overline{CS}$	0		ns
	T_1	HOLD TIME ADDRESS AND $\overline{CS}$	75		ns
	T_2	DATA LOW TIME		65	ns
	T_3	ACCESS TIME		180	ns
	T_4	P (clock) OFF TIME	160		ns
	T_5	P (clock) HIGH	220	3400	ns
WRITE AND REFRESH	T_0	SETUP TIME ADDRESS AND $\overline{CS}$	0		ns
	T_1	HOLD TIME ADDRESS AND $\overline{CS}$	75		ns
	T_2	DELAY TIME P TO WE	160	2900	ns
	T_3	WE PULSE WIDTH	180	250	ns
	T_4	WRITE TIME	130	270	ns
	T_5	DATA SETUP TIME	50		ns
	T_6	DATA HOLD TIME	85		ns
	T_7	P (clock) OFF TIME	160		ns
	T_8	P (clock) WIDTH	490	3400	ns

PACKAGE SPECIFICATIONS

The physical dimensions (in inches) of the mN606 chip are given below.

DIMENSIONS OF CHIP



DG-04181

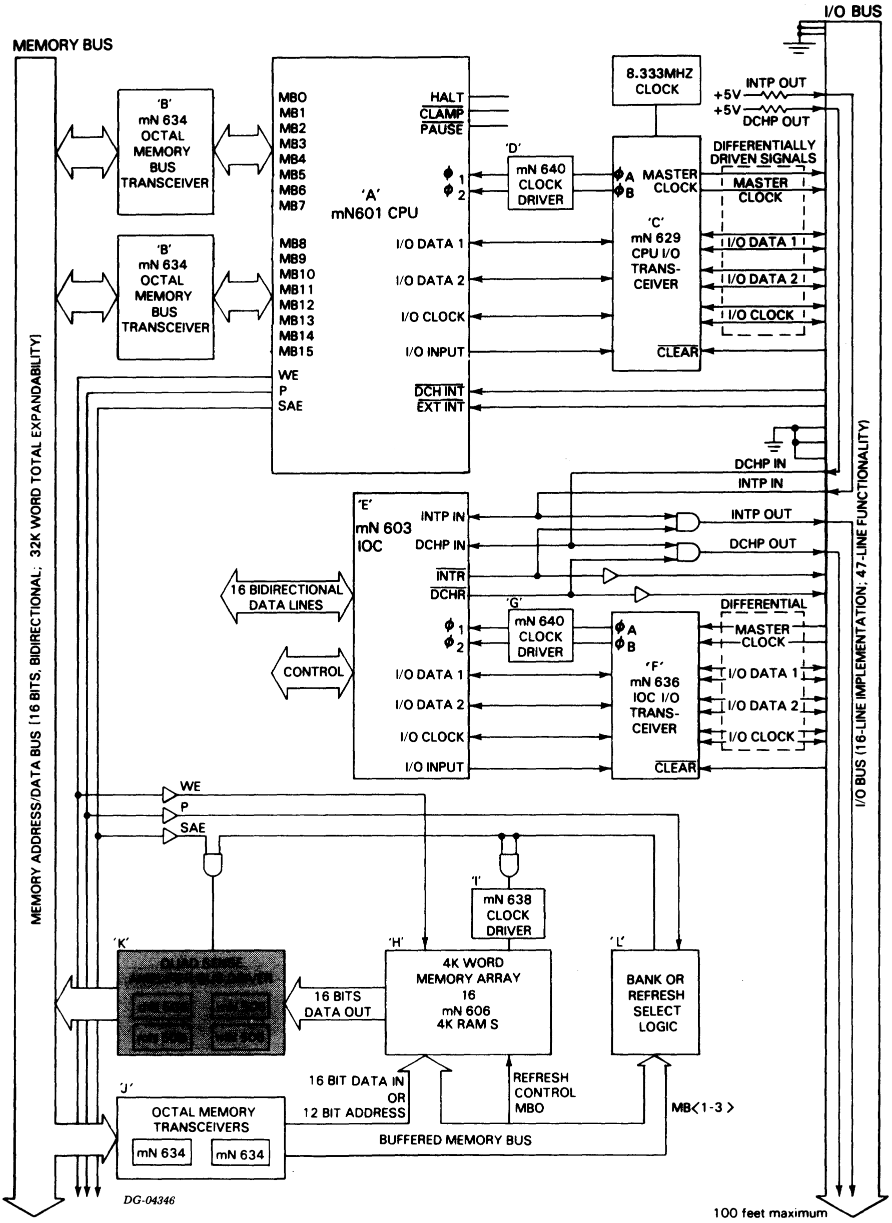
5

mN506
QUAD SENSE AMP/BUS DRIVER

6

CONTENTS

mN506
QUAD SENSE AMP/BUS DRIVER

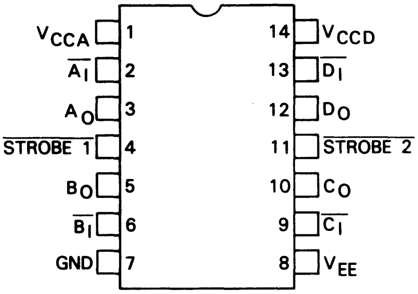


mN506
QUAD SENSE AMP/BUS DRIVER

FEATURES

- FOUR SENSE AMPLIFIERS/BUS DRIVERS IN A 14-PIN CERDIP PACKAGE.
- COMPATIBILE WITH mN606 RANDOM ACCESS MEMORIES.
- TWO INDEPENDENT STROBE LINES.
- INVERTED OPEN COLLECTOR OUTPUTS WITH 35mA CURRENT SINKING CAPABILITIES.
- DRIVERS MAY BE WIRE ORed TO SIMILAR OUTPUT CIRCUITS.

PACKAGE



DG-04132

GENERAL DESCRIPTION

The mN506 Quad Sense Amplifier/Bus Driver is compatible with the mN606 RAM chip. The output drivers may be wire ORed allowing direct connection of multiple memory arrays to the microNOVA memory bus.

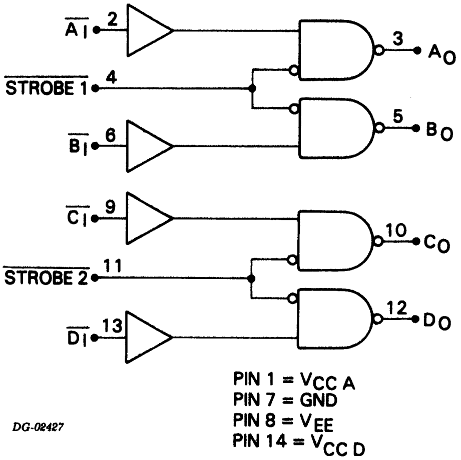
The mN506 contains four analog sense amplifiers and four digital bus drivers. Independent analog and digital supply voltage sources allow power conservation in systems with battery back-up. The analog voltage supply (for the sense amplifiers) may be shut off while the digital voltage supply provides power to maintain the memory bus integrity.

The analog inputs of the mN506 are designed to sense the current drain caused by the outputs of the mN606 Random Access Memory. The bus drivers provide open-collector TTL compatible outputs capable of sinking up to 35mA. This allows the wire ORing of drivers on the Memory Bus. The bus drivers are enabled in pairs with the assertion of the $\overline{\text{STROBE1}}$ and $\overline{\text{STROBE2}}$ signals. In addition, their outputs are inverted. Since the mN606 RAM outputs are also inverted, a low level output from the mN506 indicates that the accessed memory location contains a logical 0.

PIN DESCRIPTIONS

The diagram below shows the pin connections and the table describes the function of each pin shown in the diagram.

FUNCTIONAL PIN CONNECTION DIAGRAM



DG-02427

PIN FUNCTIONS

MNEMONIC	PIN NO.	IN/ OUT	FUNCTION
$\overline{A_I}$ $\overline{B_I}$ $\overline{C_I}$ $\overline{D_I}$	2 6 9 13	IN IN IN IN	Sense amplifier inputs.
A_O B_O C_O D_O	3 5 10 12	OUT OUT OUT OUT	Inverted open collector outputs. A_O and B_O enabled when $\overline{\text{STROBE1}}$ asserted low. C_O and D_O enabled when $\overline{\text{STROBE2}}$ asserted low.
$\overline{\text{STROBE1}}$	4	IN	When asserted low, enables A_O and B_O
$\overline{\text{STROBE2}}$	11	IN	When asserted low, enables $C_O$and D_O
V_{CCA}	1		+5 ± 0.25 volts, Analog circuit power supply.
V_{CCD}	14		+5 ± 0.25 volts, Digital circuit power supply.
V_{EE}	8		-5 ± 0.25 volts, Analog circuit power supply.
GND	7		GROUND

DG-04382

ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS

Supply voltage, V_{CCA}	7V
Supply voltage, V_{CCD}	7V
Supply voltage, V_{EE}	-7V
Input voltage, V_I	+5.5V
Output current, I_O	60mA
Operating free-air temperature range, T_A	0 deg. to 70 deg. C
Storage temperature range, T_{STG}	-25 deg. to 155 deg. C

NOTE: Subjecting a circuit to conditions either outside these limits or at these limits for an extended period of time may cause irreparable damage to the circuit. These ratings are not intended to be used during the operation of the circuit.

RECOMMENDED OPERATING CONDITIONS

Supply voltage, V_{CCA}	5.0 ± 0.25V
Supply voltage, V_{CCD}	5.0 ± 0.25V
Supply voltage, V_{EE}	-5.0 ± 0.25V
Average power dissipation	0 8 W
Operating free-air temperature range, T_A	0 deg to 70 deg C

DG-04383

DC CHARACTERISTICS

SYMBOL	CHARACTERISTICS	CONDITIONS	MIN.	MAX.	UNITS
V_{IH}	$\overline{\text{STROBE1}}$ $\overline{\text{STROBE2}}$		2.0		V
V_{IL}	$\overline{\text{STROBE1}}$ $\overline{\text{STROBE2}}$			0.8	V
V_{IHA}*	$\overline{A}_I, \overline{B}_I, \overline{C}_I, \overline{D}_I$	$V_{CCA} = V_{CCD} = 4.75$ V $V_{EE} = -5.25$ V $I_{IAMP} = -9.0$mA	1.0		V
V_{ILA}*	$\overline{A}_I, \overline{B}_I, \overline{C}_I, \overline{D}_I$	$V_{CCA} = V_{CCD} = 5.25$ V $V_{EE} = -4.75$ V $I_{IAMP} = 0$mA		3.0	V
V_I	$\overline{\text{STROBE1}}$ $\overline{\text{STROBE2}}$ (INPUT CLAMP VOLTAGE)	$V_{CCA} = V_{CCD} = 4.75$V $V_{EE} = -4.75$ V $I_I = -18$mA	-1.5		V
I_{IH}	$\overline{\text{STROBE1}}$ $\overline{\text{STROBE2}}$	$V_{CCA} = V_{CCD} = 5.25$ V $V_{EE} = -5.25$ V $V_I = 2.7$V		200	μA

DG-04384

* **NOTE**: SEE SENSE AMP V-I CHARACTERISTICS ON PAGE 5.

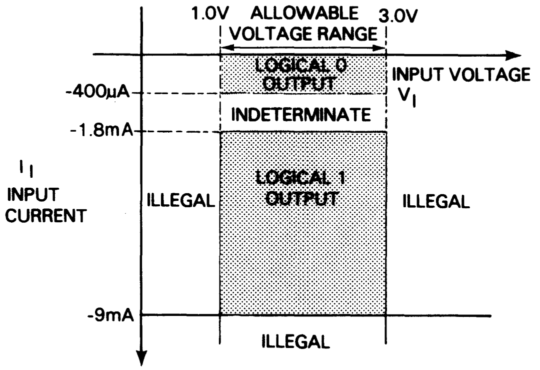
DC CHARACTERISTICS (CONT.)

SYMBOL	CHARACTERISTICS	CONDITIONS	MIN.	MAX.	UNITS
I_{IL}	$\overline{\text{STROBE1}}$ $\overline{\text{STROBE2}}$	V_{CCA} = V_{CCD} = 5.25V V_{EE} = -5.25 V V_I = 0.5V	-4		mA
I_I	$\overline{\text{STROBE1}}$ $\overline{\text{STROBE2}}$	V_{CCA} = V_{CCD} = 5.25V V_{EE} = -5.25V V_I = 5.5V		1	mA
V_{OL}	A_O,B_O,C_O,D_O	V_{CCA} = V_{CCD} = 4.75V V_{EE} = -5.25V $V_{ISTROBE}$ = 0.8V I_{IAMP} = -400uA I_O= 35mA		0.5	V
I_{OH}	A_O,B_O,C_O,D_O	V_{CCA} = V_{CCD} = 4.75V V_{EE} = -4.75V V_O = 5.5V			
		I_{IAMP} = -1.8mA $V_{ISTROBE}$ = 0.8V		50	μA
		I_{IAMP} = -400uA $V_{ISTROBE}$ = 2.0V		50	μA
I_{CCA}	ANALOG POWER SUPPLY CURRENT (V_{CCA})	V_{CCA} = V_{CCD} = 5.25V V_{EE} = -5.25V I_{IAMP} = -1.8mA $V_{ISTROBE}$ = 2.0V		50	mA
		V_{CCA} = V_{CCD} = 5.25V V_{EE} = -5.25V I_{IAMP} = -400uA $V_{ISTROBE}$ = 0.8V		50	mA
I_{CCD}	DIGITAL POWER SUPPLY CURRENT (V_{CCD})	VCC_A = VCC_D = 5.25V V_{EE} = -5.25V I_{IAMP} = -1.8mA $V_{ISTROBE}$ = 2.0V		31	mA
		V_{CCA} = V_{CCD} = 5.25V V_{EE} = -5.25V I_{IAMP} = -400 μA $V_{ISTROBE}$ = 0.8		51	mA
I_{EE}	ANALOG POWER SUPPLY CURRENT (V_{EE})	V_{CCA} = V_{CCD} = 5.25V V_{EE} = -5.25V I_{IAMP} = -1.8mA $V_{ISTROBE}$ = 2.0V		-65	mA
		V_{CCA} = V_{CCD} = 5.25V V_{EE} = -5.25V I_{IAMP} = -400uA $V_{ISTROBE}$ = 0.8V		-65	mA

NOTE: POSITIVE CURRENT IS INTO THE PIN.

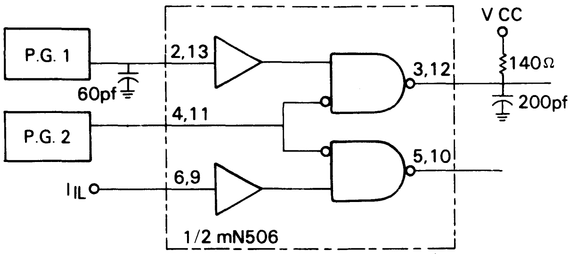
AC CHARACTERISTICS

SENSE-AMP INPUT V-I CHARACTERISTICS



DG-04133

TEST CIRCUIT:

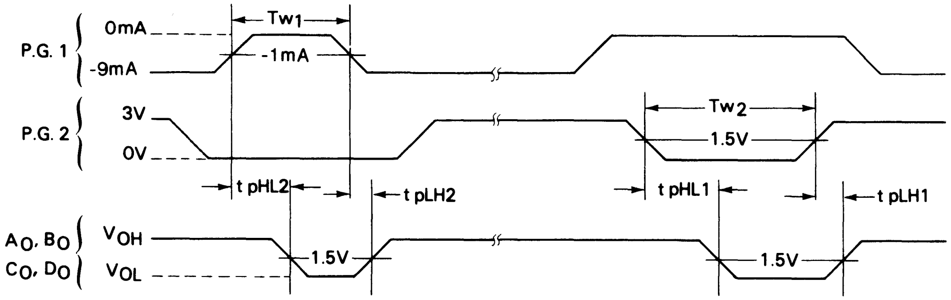


NOTE: PULSE GENERATORS 1 AND 2
NOT INTERNALLY TERMINATED.

$0 < t_r = t_f < 6nS$
PRR = 1MHz
tw_1 = 100nS
tw_2 = 300nS

DG-04135

SWITCHING DIAGRAM



DG-04134

TRANSITION TIMING

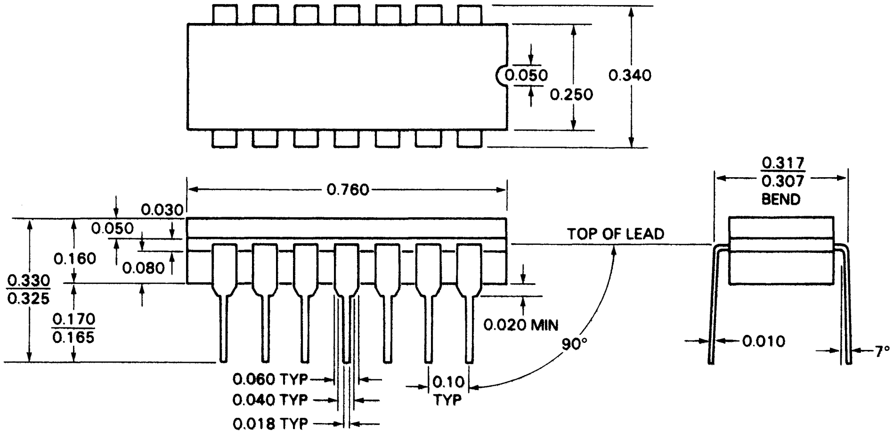
SYMBOL	CHARACTERISTIC	CONDITION*	MIN.	MAX.	UNITS
t pHL1	PROPAGATION DELAY (HIGH TO LOW LEVEL)	$V_{CCA} = V_{CCD} = 5.0V$ $V_{EE} = -5.0V$ $I_{IL} = -1.8mA$	7	18	nS
t pLH1	PROPAGATION DELAY (LOW TO HIGH LEVEL)	$V_{CCA} = V_{CCD} = 5.0V$ $V_{EE} = -5.0V$ $I_{IL} = -1.8mA$	7	18	nS
t pHL2	PROPAGATION DELAY (HIGH TO LOW LEVEL)	$V_{CCA} = V_{CCD} = 5.0V$ $V_{EE} = -5.0V$ $I_{IL} = -1.8mA$	10	25	nS
t pLH2	PROPAGATION DELAY (LOW TO HIGH LEVEL)	$V_{CCA} = V_{CCD} = 5.0V$ $V_{EE} = -5.0V$ $I_{IL} = -1.8mA$	25	38	nS

* See Test Circuit

PACKAGE SPECIFICATIONS

The physical dimensions of the mN506 chip are given below.

DIMENSIONS OF CHIP



DG-04239

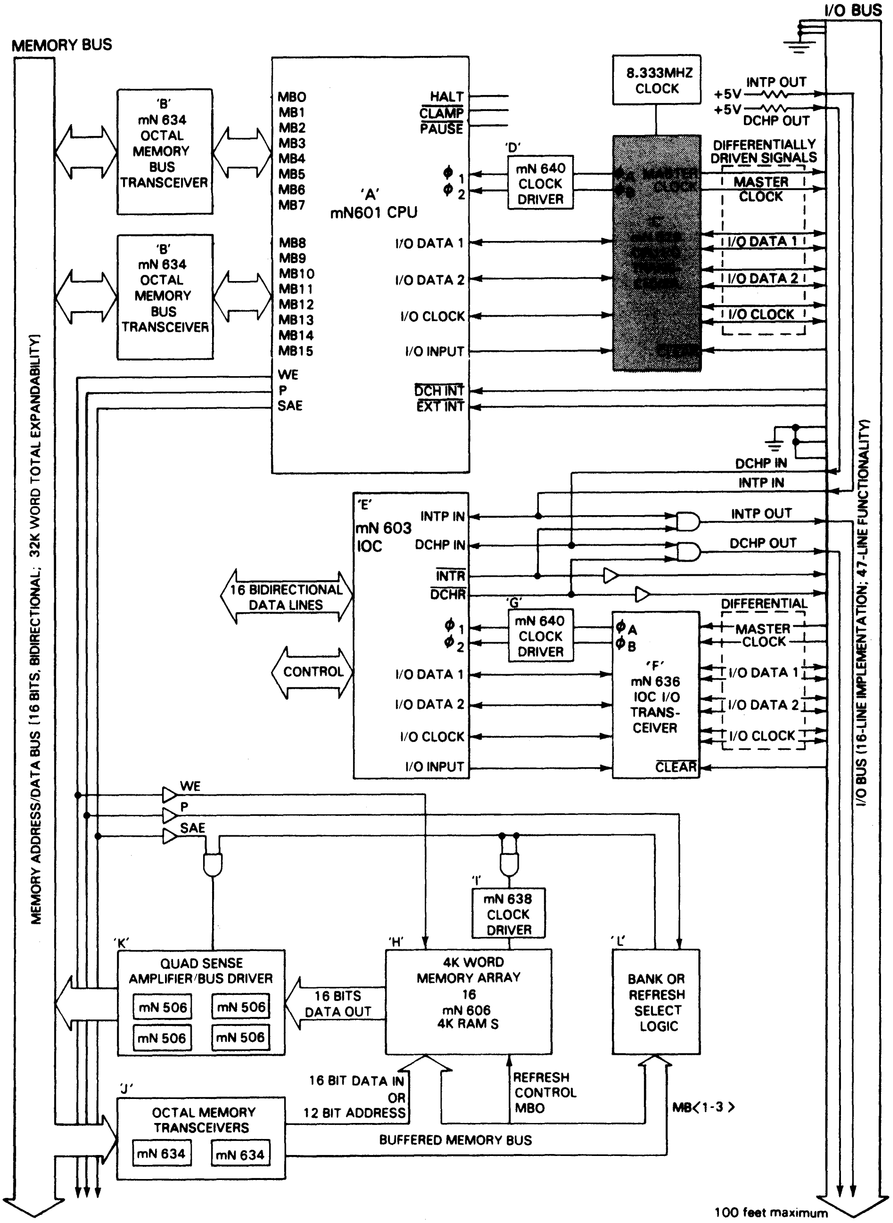
6

mN629
CPU I/O TRANSCEIVER

7

Data General
Data General Corporation, Westboro, Massachusetts 01581

mN629
CPU I/O TRANSCEIVER



DG-04332

mN629
CPU I/O TRANSCEIVER

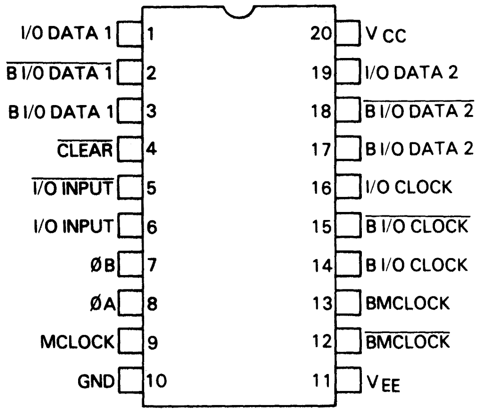
FEATURES

- I/O BUS RECEIVER AND TRANSMITTER IN A SINGLE 20-PIN CERAMIC PACKAGE.
- INTERFACES mN601 CPU to microNOVA DIFFERENTIAL I/O BUS
- DIFFERENTIAL DRIVE CAPABILITIES:

  1. PROVIDE HIGH I/O BUS NOISE IMMUNITY.
  2. ALLOW LONG DISTANCE APPLICATIONS. (UP TO 100 FEET)
  3. ALLOW THE USE OF INEXPENSIVE 16-CONDUCTOR RIBBON CABLE.

- PROVIDES ALL SYSTEM TIMING FROM ONE MASTER CLOCK INPUT.

PACKAGE


I/O DATA 1
1
$\overline{\text{B I/O DATA 1}}$
2
B I/O DATA 1
3
$\overline{\text{CLEAR}}$
4
$\overline{\text{I/O INPUT}}$
5
I/O INPUT
6
ØB
7
ØA
8
MCLOCK
9
GND
10
20
V CC
19
I/O DATA 2
18
$\overline{\text{B I/O DATA 2}}$
17
B I/O DATA 2
16
I/O CLOCK
15
$\overline{\text{B I/O CLOCK}}$
14
B I/O CLOCK
13
BMCLOCK
12
$\overline{\text{BMCLOCK}}$
11
V EE

DG-04182

GENERAL DESCRIPTION

The CPU I/O Transceiver chip interfaces the microNOVA CPU with the bi-directional, differential I/O bus. It communicates with peripheral devices via peripheral I/O Transceivers (mN636s). When the mN629 is in receive mode, it passes information to the CPU. When it is in transmit mode, the CPU can send information to I/O Controllers.

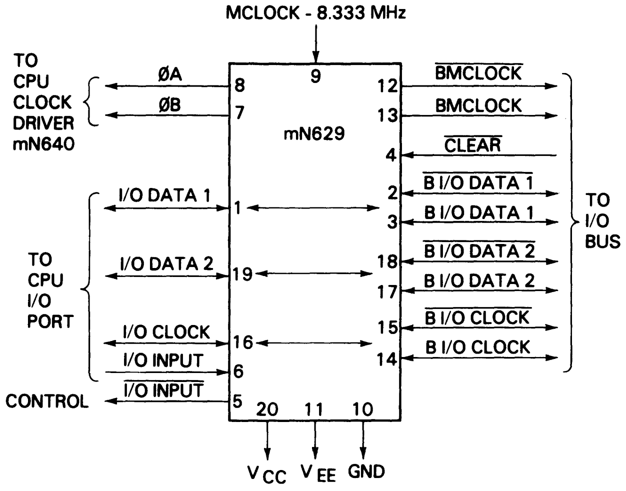
The mN629 uses differential drivers and receivers to communicate with up to 20 IOC I/O Transceivers via the I/O bus. This allows high noise immunity over a long bus. An I/O bus length of up to 100 feet is possible using inexpensive 16-conductor ribbon cable.

In addition to passing data between the CPU and the I/O bus, the CPU I/O Transceiver provides timing for the entire system. The system master clock is an input to the CPU I/O Transceiver. A two phase clock (ØA and ØB) is generated from this input to provide timing for the high voltage clock drivers required by the CPU. In addition, a differential version of master clock is transmitted to all the mN636 I/O Transceivers to provide synchronized timing for the entire system.

PIN DESCRIPTIONS

The diagram below shows the pin connections and the table describes the function of each pin shown in the diagram.

FUNCTIONAL PIN CONNECTION DIAGRAM


MCLOCK - 8.333 MHz
TO CPU CLOCK DRIVER mN640
ØA
ØB
8
7
9
mN629
12
$\overline{\text{BMCLOCK}}$
13
BMCLOCK
4
$\overline{\text{CLEAR}}$
2
$\overline{\text{B I/O DATA 1}}$
3
B I/O DATA 1
18
$\overline{\text{B I/O DATA 2}}$
17
B I/O DATA 2
15
$\overline{\text{B I/O CLOCK}}$
14
B I/O CLOCK
TO I/O BUS
TO CPU I/O PORT
I/O DATA 1
1
I/O DATA 2
19
I/O CLOCK
16
I/O INPUT
6
CONTROL
$\overline{\text{I/O INPUT}}$
5
20
11
10
V_{CC}
V_{EE}
GND

DG-04183

PIN FUNCTIONS

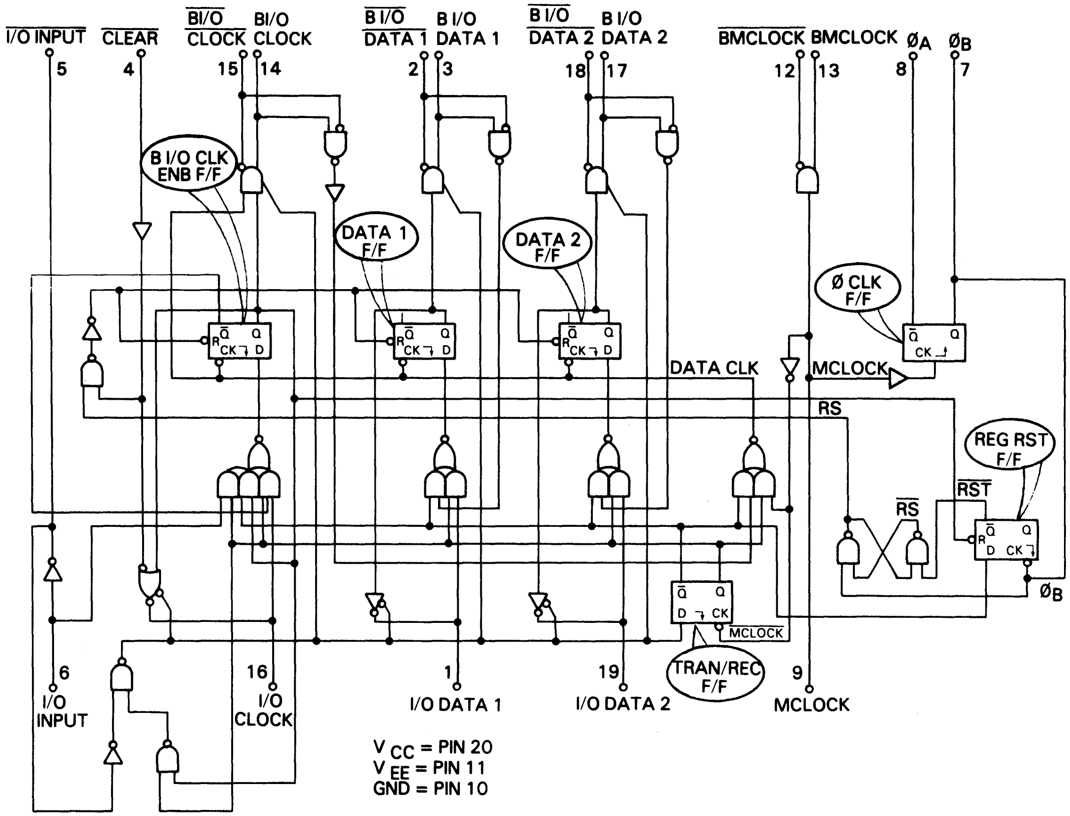
MNEMONIC	PIN NO.	IN/ OUT	FUNCTION
I/O DATA1 I/O DATA2	1 19	I/O I/O	Tri-state, bi-directional data port pins. Transfer serial data between Transceiver and CPU I/O port data pins.
I/O CLOCK	16	I/O	Tri-state, bi-directional clock line used to synchronize transmission and reception of data on I/O DATA1 and I/O DATA2 lines. Note: CLEAR asserted low pulls I/O CLOCK to its low state if Transceiver is in Receive Mode.
I/O INPUT	6	IN	High = mN629 in receive mode. Low = mN629 in transmit mode.
$\overline{\text{I/O INPUT}}$	5	OUT	Inverted output of I/O INPUT.
$\overline{\text{CLEAR}}$	4	IN	When asserted, clears all internal flip/flops and pulls I/O CLOCK (pin 16) to the low state if the Transceiver is in the Receive Mode (used to reset mN601).
MCLOCK	9	IN	MicroNOVA system master clock input.
ØA ØB	8 7	OUT OUT	Two-phase clocks operating at one-half the MCLOCK frequency. TTL level open collector outputs.
$\overline{\text{BMCLOCK}}$ BMCLOCK	12 13	OUT OUT	Open emitter, open collector differential pair. Transmits system clocks to all peripheral Transceivers (mN636).

MNEMONIC	PIN NO.	IN/ OUT	FUNCTION
$\overline{\text{BI/O DATA1}}$ BI/O DATA1	2 3	I/O I/O	Open emitter, open collector differential pair. Transmit and receive serial data on I/O bus. RECEIVE MODE (I/O INPUT high) - receive differential signals from I/O bus. Clocked by BI/O CLOCK. TTL level signal output on I/O DATA1 pin. TRANSMIT MODE (I/O INPUT low) - TTL level input from I/O DATA1 transmitted differentially on I/O bus. Clocked by MCLOCK.
$\overline{\text{BI/O DATA2}}$ BI/O DATA2	18 17	I/O I/O	Open emitter, open collector differential pair. Transmit and receive serial data on I/O bus. RECEIVE MODE (I/O INPUT high) - receives differential signals from I/O bus. Clocked by BI/O CLOCK. TTL level signal output on I/O DATA2 pin. TRANSMIT MODE (I/O INPUT low) - TTL level input from I/O DATA2 transmitted differentilly on I/O bus. Clocked by MCLOCK.
$\overline{\text{BI/O CLOCK}}$ BI/O CLOCK	15 14	I/O I/O	Open emitter, open collector differential pair. Used to synchronize transmission and reception of BI/O DATA <1,2> pairs. RECEIVE MODE (I/O INPUT high) - receive differential synchronizing clock from I/O bus. Clocks data in on BI/O DATA <1,2> differential pairs. TTL level signal output on I/O CLOCK pin. TRANSMIT MODE (I/O INPUT low) - TTL level input from I/O CLOCK pin used to generate a double frequency clock from MCLOCK; is transmitted differentially to the I/O bus.
			POWER
V_{CC}	20		+5.0 ± 0.25 volts
V_{EE}	11		-5.0 ± 0.25 volts
GND	10		Ground

INTERNAL STRUCTURE

The mN629 consists of six flip/flops (F/F), a number of gates, four differential drivers, and three differential receivers. The I/O INPUT pin determines whether the Transceiver is in the Receive mode or the Transmit mode. When I/O INPUT is high, the Transceiver is in the Receive mode. In this case, the I/O DATA1, I/O DATA2, and I/O CLOCK outputs are enabled. When I/O INPUT is asserted low, these output drivers are disabled while the differential drivers are enabled. In this mode, the I/O DATA1, I/O DATA2, and I/O CLOCK inputs provide data to the DATA1 F/F, DATA2 F/F, and BI/O CLK ENB F/F.

INTERNAL BLOCK DIAGRAM



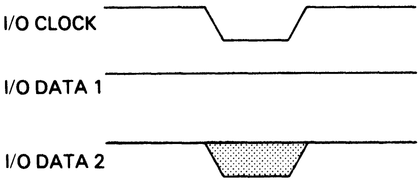
V_{CC} = PIN 20
V_{EE} = PIN 11
GND = PIN 10

DG-02435

DATA TRANSMISSION AND RECEPTION

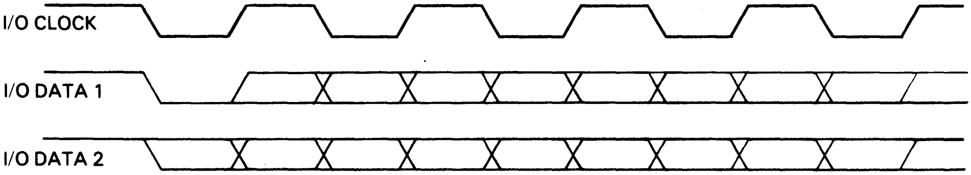
When connected to the microNOVA I/O bus, the mN629 performs two general types of data transfers; short and long as shown below:

SHORT TRANSFER



DG-04184

LONG TRANSFER



DG-04185

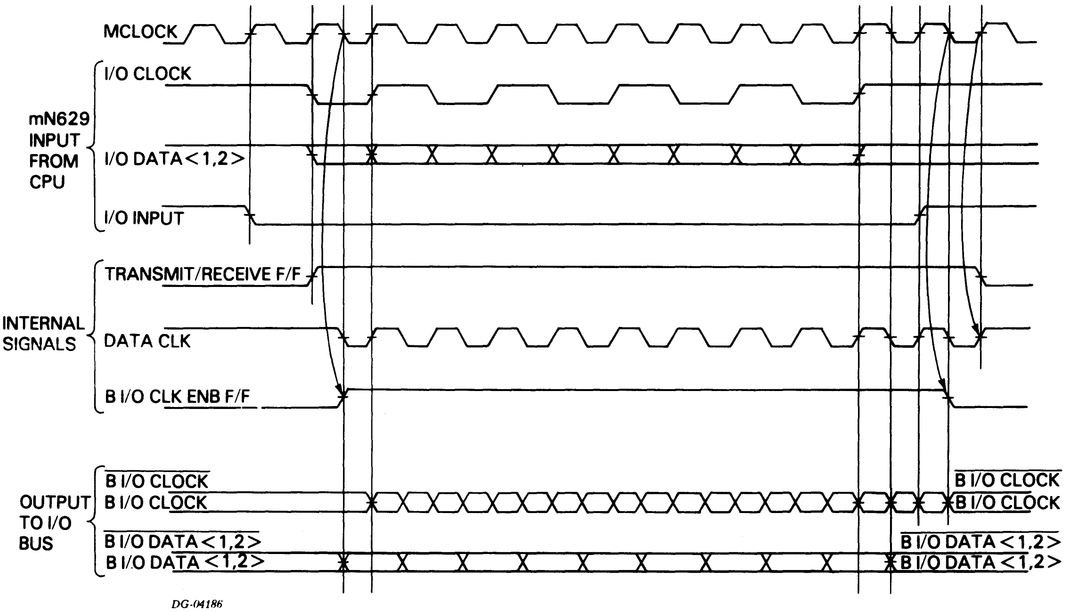
The short type consists of one I/O CLOCK pulse and transfers two bits. The long type consists of five I/O CLOCK pulses and transfers 18 bits. For the purpose of illustrating the operation of the Transceiver, the long transfer is used as an example. The one pulse transfer is simply a shortened version of the long transfer as far as the Transceiver's operation is concerned. See the Introduction and CPU (mN601 or mN602 Operations Protocol, I/O Operations sections for further explanation of the types of transfers.

The transceiver operates under two modes; transmit and receive. The input waveforms, some internal signals, output waveforms, and a brief discussion of each mode are given below.

Transmit Mode

Transmit mode begins when I/O INPUT is asserted low (by the CPU). The next rising edge of MCLOCK sets the Transmit/Receive F/F allowing MCLOCK to clock the DATA CLK line. As shown in the block diagram, DATA CLK clocks data into the DATA1 F/F and DATA2 F/F. These flip/flops supply the data to the differential drivers. In addition, DATA CLK sets the BI/O CLK ENB F/F for the duration of the transmission. The high output of the BI/O CLK ENB F/F allows DATA CLK to be transmitted on the I/O bus via the BI/O CLOCK differential driver. A long transfer from the CPU to the I/O bus is shown in the following timing diagram.

TRANSMIT WAVEFORMS

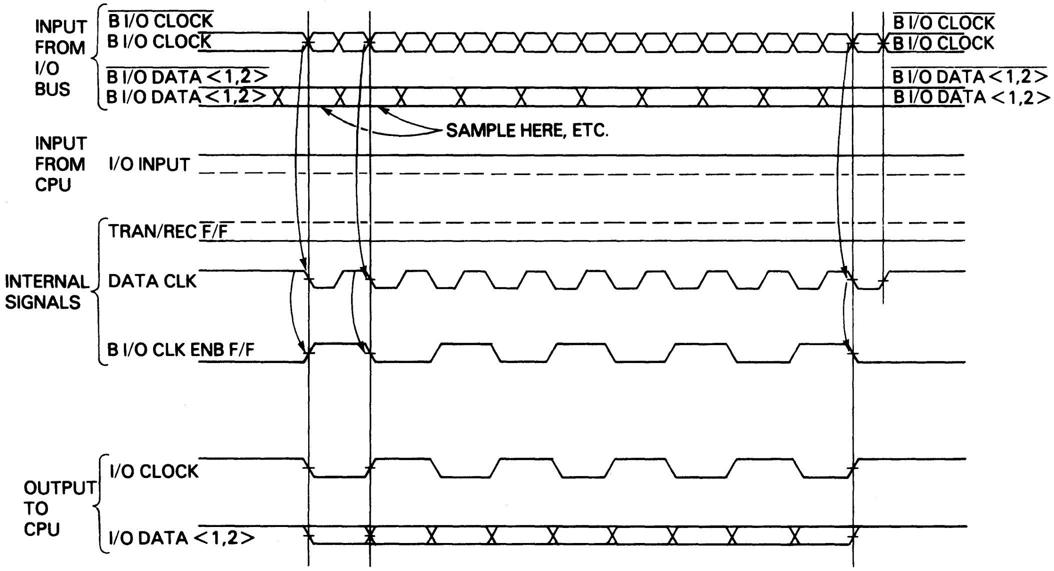


NOTE: MCLOCK and DATA CLK operate at twice the frequency of I/O CLOCK. This provides a transition of BI/O CLOCK near the middle of every BI/O DATA < 1,2 > pulse allowing the receiving transceiver to sample near the middle of each data bit.

Receive Mode

Receive mode begins when I/O INPUT is asserted high. The first BI/O CLOCK transition from the I/O bus initiates data reception. The signals received from the I/O bus are in the same form as those transmitted by the Transceiver. The Transmit/Receive F/F is in the reset state allowing BI/O CLOCK to generate DATA CLK. Data is clocked into the DATA1 F/F and DATA2 F/F on every falling edge of DATA CLK. In addition, the BI/O CLK ENB F/F is toggled on each falling edge of DATA CLK. An inverted output of this flip/flop drives the I/O CLOCK pin. A long transfer from the I/O bus to the CPU is shown in the following timing diagram.

RECEIVE WAVEFORMS



DG-04187

NOTE: The last BI/O CLOCK transition causes the Transceiver to sample the differential data lines after the last bit of data has been received. This ensures that the DATA1 F/F and DATA2 F/F are reset to their normal state at the completion of the transfer.

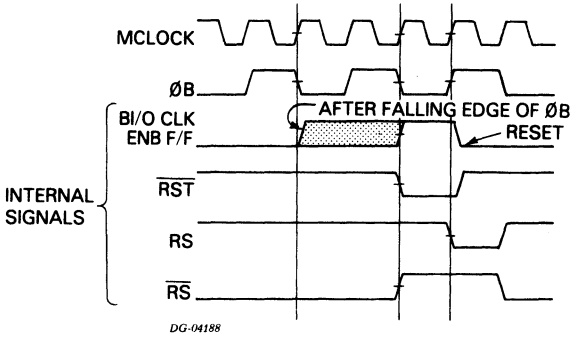
REGISTER RESET CIRCUIT

This circuit consists of the Register Reset Flip/Flop (REG RST F/F), two NAND gates, and the associated control lines (see lower right corner of Internal Block Diagram). While in the receive mode, the Register Reset Circuit ensures that the BI/O CLK ENB F/F will be reset if set by a spurious transition of BI/O CLOCK.

Under normal operation, the BI/O CLK ENB F/F is clocked either twice or ten times depending on whether the Transceiver receives a short or a long type of transfer. Thus, there should always be an even number of BI/O CLOCK transitions. If for some reason a single transition of BI/O CLOCK is received, the BI/O CLK ENB F/F must be reset to avoid holding I/O CLOCK low and resetting the CPU (see mN601 or mN602 Operations Protocol, Status).

If the BI/O CLK ENB F/F is set anywhere during the period shaded in the following timing diagram, it is reset after the rising edge of ØBas described below.

B I/O CLK ENB F/F RESET TIMING



While the BI/O CLK ENB F/F is in the reset state, its Q output is 0. This forces the REG RST F/F to its reset state. If the BI/O CLK ENB F/F is clocked by a fraudulent transition of BI/O CLOCK, its Q output is no longer 0 and the REG RST F/F is no longer forced to the reset state. On the next falling edge of ØB, the REG RST F/F is set placing $\overline{\text{RST}}$ low. Though $\overline{\text{RST}}$ is now low, RS remains high since ØB is low. When ØB goes high, RS goes low clearing the BI/O CLK ENB F/F. Its Q output goes low immediately reseting the REG RST F/F and placing $\overline{\text{RST}}$ back in its normal state. RS and $\overline{\text{RS}}$ return to their normal state on the next falling edge of ØB.

The BI/O CLK ENB F/F is not reset during normal reception because the reset circuit has a built-in delay of one MCLOCK period. Even if a transition occurs at the end of the period shaded in the above timing diagram, there is enough time for the second transition to occur before the rising edge of ØB.

CLOCKS

The system master clock is applied to MCLOCK. This clock toggles theØCLK F/F which provides the two-phase (half-frequency, TTL level) clock for the CPU Clock Driver (mN640). MCLOCK is also differentially driven on the I/O bus (BM CLOCK) to provide synchronized timing for the peripheral Transceivers and Controllers.

CLEAR

The $\overline{\text{CLEAR}}$ pin is used during system reset. When asserted low, all the mN629's internal flip/flops are reset. If the Transceiver is in the Receive Mode, the I/O CLOCK pin is pulled low (resets the CPU, see mN601 or mN602). If the Transceiver is in the transmit mode, the I/O CLOCK pin is not affected by the assertion of $\overline{\text{CLEAR}}$ until the transmission is over and I/O INPUT is pulled high.

ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS

Supply voltage, V_{CC}	7V
Supply voltage, V_{EE}	-7V
Input voltage, V_I	5.5V
Transceiver output current, I_{OT}	80mA
All other output current, I_O	50mA
Operating free-air temperature range, T_A	O deg. to 70 deg.C
Storage temperature range, T_{STG}	-55 deg. to 125 deg.C

NOTE: Subjecting a circuit to conditions either outside these limits or at these limits for an extended period of time may cause irreparable damage to the circuit. These ratings are not intended to be used during the operation of the circuit.

RECOMMENDED OPERATING CONDITIONS

Supply Voltage V_{CC}	+5.0 ± 0.25V
Supply Voltage V_{EE}	-5.0 ± 0.25V
Average Power Dissipation	0.95 W
Operating free-air temperature range, TA	0 deg. to 70 deg. C

DC CHARACTERISTICS

SYMBOL	CHARACTERISTICS	CONDITIONS	MIN.	TYP.	MAX.	UNITS
V_{IH}	MCLOCK I/O INPUT I/O DATA1,I/O DATA2 $\overline{\text{CLEAR}}$, I/O CLOCK		2.0			V
	BI/O CLOCK,$\overline{\text{BI/O CLOCK}}$ BI/O DATA1,$\overline{\text{BI/O DATA1}}$ BI/O DATA2,$\overline{\text{BI/O DATA2}}$				5.25	V
V_{IL}	MCLOCK I/O CLOCK I/O INPUT I/O DATA1 I/O DATA2 $\overline{\text{CLEAR}}$				0.8	V
	BI/O CLOCK,$\overline{\text{BI/O CLOCK}}$ BI/O DATA1,$\overline{\text{BI/O DATA1}}$ BI/O DATA2,$\overline{\text{BI/O DATA2}}$		0.5			V

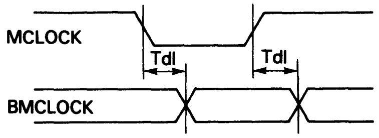
DC CHARACTERISTICS (CONT.)

SYMBOL	CHARACTERISTICS	CONDITIONS	MIN.	TYP.	MAX.	UNITS
$V_{ID} =$ $\lvert V_{IL} - V_{IH} \rvert$	(BI/O CLOCK,$\overline{\text{BI/O CLOCK}}$) (BI/O DATA1,$\overline{\text{BI/O DATA1}}$) (BI/O DATA2,$\overline{\text{BI/O DATA2}}$)	REQUIRED VALUE OF $\lvert V_{IL} - V_{IH} \rvert$ FOR VALID SIGNAL DETECTION	0.75	0.5		V
I_{IH}	MCLOCK	$V_I = 2.4V$		-0.22	-1.0	mA
	I/O INPUT	$V_I = 2.4V$		0	80	μA
	I/O DATA1 I/O DATA2 $\overline{\text{I/O CLOCK}}$ $\overline{\text{CLEAR}}$	$V_I = 2.4V$		0	40	μA
I_{IL}	MCLOCK	$V_I = 0.5\ V$		-2.2	-3.8	mA
	I/O CLOCK I/O INPUT I/O DATA1 $\overline{\text{I/O DATA2}}$ $\overline{\text{CLEAR}}$	$V_I = 0.5\ V$		-0.5	-1.6	mA
I_{ID}	BI/O CLOCK,$\overline{\text{BI/O CLOCK}}$ BI/O DATA1,$\overline{\text{BI/O DATA1}}$ BI/O DATA2,$\overline{\text{BI/O DATA2}}$	$0.55\ V \leq V_I \leq 2.0\ V$			500	μA
V_{OH}	$\overline{\text{I/O INPUT}}$	$I_O = -400uA$ $V_{CC} = 4.75\ V$	2.5	3.5		V
	I/O CLOCK I/O DATA1 I/O DATA2	$I_O = -20uA$ $V_{CC} = 4.75\ V$	3.0	3.8		V
	BI/O CLOCK (OPEN BI/O DATA1 EMITTER BI/O DATA2 OUTPUTS) BMCLOCK	$I_O = -35\ mA$ $V_{CC} = 5.25\ V$			3.2	V
		$I_O = -35\ mA$ $V_{CC} = 4.75\ V$	2.0	2.5		V
	ØA. ØB	$V_{CC} = 5.25,$ $RL = 1K$	4.95			V
V_{OL}	ØA. ØB	$I_O = 10mA$ $V_{CC} = 4.75\ V$		0.35	0.5	
	$\overline{\text{I/O INPUT}}$ I/O DATA1 I/O DATA2 I/O CLOCK	$I_O = 5mA$ $V_{CC} = 4.75\ V$		0.35	0.5	V
	$\overline{\text{BI/O CLOCK}}$ (OPEN $\overline{\text{BI/O DATA1}}$ COLLECTOR $\overline{\text{BI/O DATA2}}$ OUTPUTS) $\overline{\text{BMCLOCK}}$	$I_O = 35mA$ $V_{CC} = 4.75$		0.38	0.55	V
I_{CC}	TRANSMIT MODE	$V_{CC} = 5.25\ V$		130	170	mA
	RECEIVE MODE	$V_{EE} = -5.25\ V$		130	170	mA
I_{EE}		$V_{CC} = 5.25\ V$ $V_{EE} = -5.25\ V$		23		mA
C_I	ALL INPUTS				7	pf

NOTE: Positive current is into the pin.

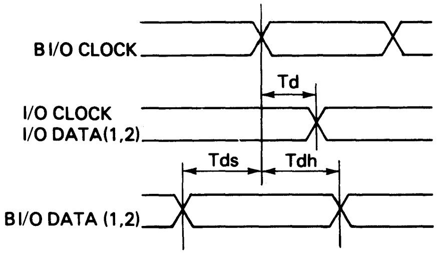
AC CHARACTERISTICS
SWITCHING DIAGRAMS

MCLOCK TRANSMITTER TIMING



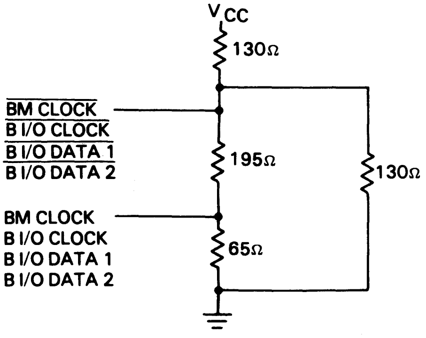
DG-04189

RECEIVER TIMING



DG-04130

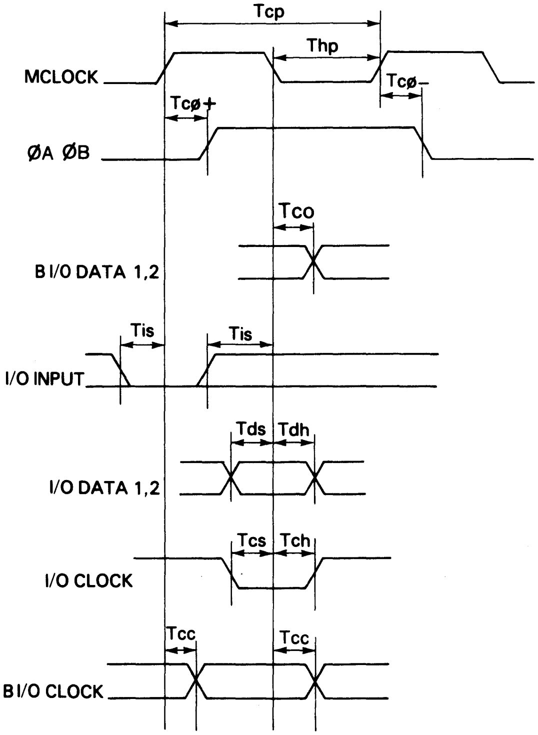
TEST CIRCUIT NO. 1



NOTE: ALL RESISTORS ARE ± 1%
ALL CAPACITORS ARE ± 5%
LOAD SHOULD BE CONNECTED TO
OUTPUT USING LESS THAN
2 INCHES OF WIRE.

DG-04192

TRANSMITTER TIMING

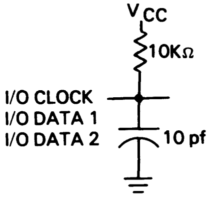


NOTES:

- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED FROM A 1.5 VOLT LEVEL.
- SEE TIMING TABLE ON PAGE 12 .

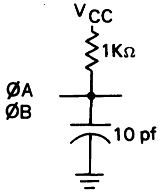
DG-04191

TEST CIRCUIT NO. 2



DG-04343

TEST CIRCUIT NO. 3



DG-04344

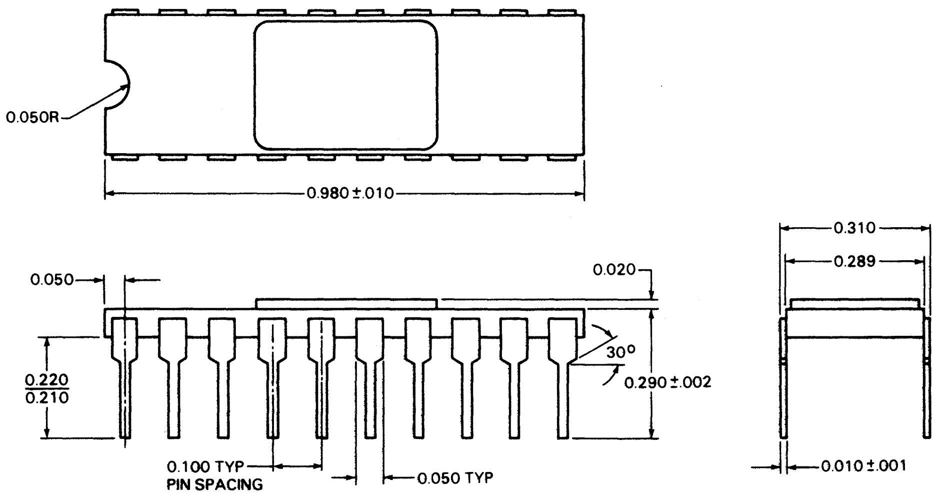
TRANSITION TIMING

SYMBOL	CHARACTERISTIC	MIN.	MAX.	UNIT	TEST CIRCUIT
T_{cp}	MASTER CLOCK PERIOD	99	125	ns	
T_{hp}	HALF CYCLE PERIOD	40	60	ns	
$T_{c\phi+}$	DELAY FROM MCLOCK TO ØA, ØB (RISING EDGE)	30	70	ns	NO. 3
$T_{c\phi-}$	DELAY FROM MCLOCK TO ØA, ØB (FALLING EDGE)	25	60	ns	NO. 3
T_{is-}	SET UP TIME FOR I/O INPUT	35	---	ns	
T_{is+}	SET UP TIME FOR I/O INPUT	20	---	ns	
T_{dl}	DELAY FROM MCLOCK TRANSITION TO BMCLOCK TRANSITION	5	40	ns	NO. 1
	TRANSMITTER TIMING				
T_{cc}	DELAY FROM MCLOCK TO BI/O CLOCK	15	55	ns	NO. 1
T_{co}	DELAY FROM MCLOCK TO BI/O DATA	25	65	ns	NO. 1
T_{ds}	DATA SETUP TIME	30	---	ns	NO. 1
T_{dh}	DATA HOLD TIME	10	---	ns	NO. 1
T_{cs}	I/O CLOCK SETUP TIME	30	---	ns	NO. 1
T_{ch}	I/O CLOCK HOLD TIME	10	---	ns	NO. 1
	RECEIVER TIMING				
T_d	DELAY FROM BI/O CLOCK TO I/O CLOCK AND I/O DATA(1,2)	30	80	ns	NO. 2
T_{ds}	DATA SETUP TIME	15	---	ns	NO. 2
T_{dh}	DATA HOLD TIME	10	---	ns	NO. 2

PACKAGE SPECIFICATIONS

The physical dimensions of the mN629 chip are given below.

DIMENSIONS OF CHIP



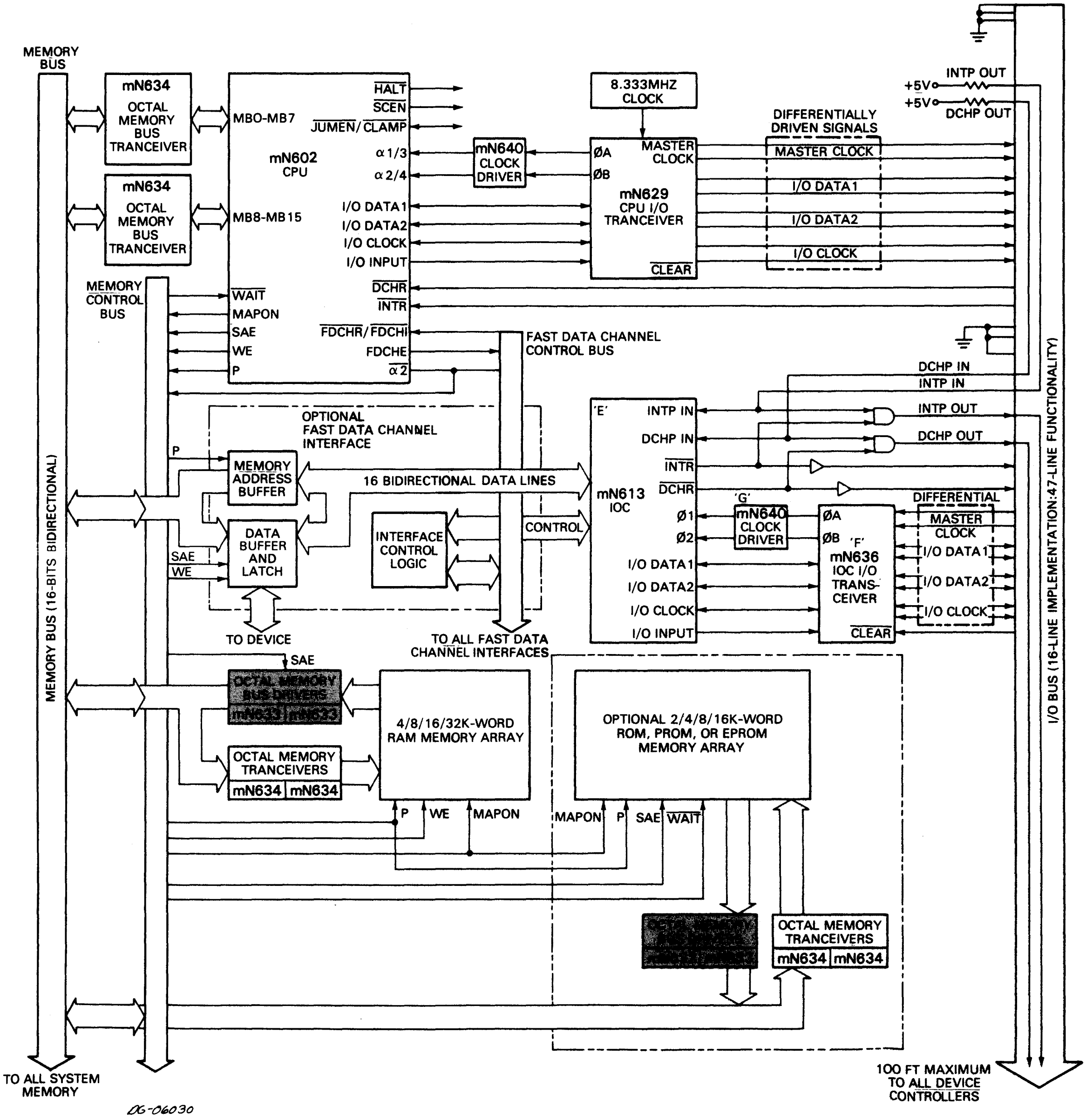
DG-04198

mN633
OCTAL DRIVER

8

CONTENTS

mN633
OCTAL DRIVER



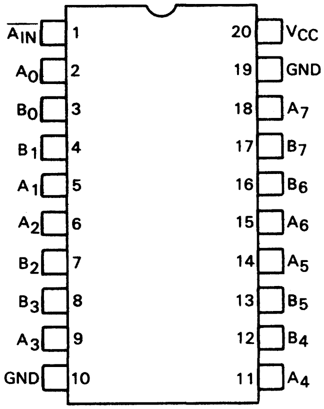
DG-06030

mN633
OCTAL DRIVER

FEATURES

- 8 DRIVERS IN A 20-PIN CERAMIC DUAL-IN-LINE PACKAGE.
- NON-INVERTING OPEN-COLLECTOR OUTPUTS.
- COMPLETELY ISOLATES TWO SETS OF LINES.
- SINKS 35mA per OUTPUT.

PACKAGE



DG-06207

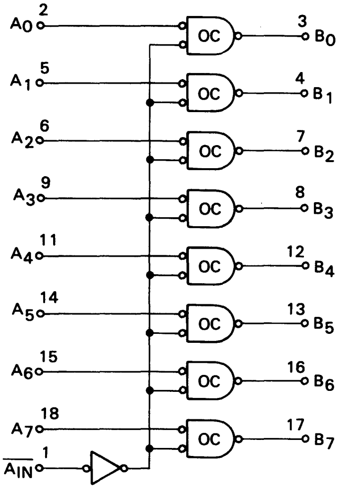
GENERAL DESCRIPTION

The mN633 Octal Driver gates lines to and isolates lines from a common bus. Eight bits are driven in one direction by the chip.

PIN DESCRIPTIONS

The diagram below shows the pin connections and the tables show the operation and function of the pins.

FUNCTIONAL PIN CONNECTION DIAGRAM



DG-06208

TRUTH TABLE

$\overline{A}_{IN}$	B_{IN}	OPERATION
LOW	LOW	A data → B data
HIGH	LOW	Isolate A from B

8

PIN DESCRIPTIONS

MNEMONIC	PIN #	I/O	FUNCTION
A_{0-7}	2,5,6, 9,11, 14,15, 18	IN	If A IN is low, these are the inputs of lines B_{0-7}. Else lines A_{0-7} are isolated from the values on B_{0-7}.
B_{0-7}	3,4,7, 8,12, 13,16, 17	OUT	If A IN is low, these are outputs of lines A_{0-7}. Else indeterminate.
$\overline{A}_{IN}$	1	IN	When asserted low, enables A pins as inputs. Else A pins are isolated from B pins.
V_{CC}	20		+5.0 ± 0.25 volts
GND	10,19		Ground

ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS

Supply voltage, V_{CC}	+7V
Input voltage, V_I	5.5V
Output Current, I_O	60mA
Operating free-air temperature range, T_A	0 deg. to 70 deg.C
Storage temperature range, T_{STG}	-55 deg. to 125 deg.C

NOTE: Subjecting a circuit to conditions either outside these limits or at these limits for an extended period of time may cause irreparable damage to the circuit. These ratings are not intended to be used during the operation of the circuit.

RECOMMENDED OPERATING CONDITIONS

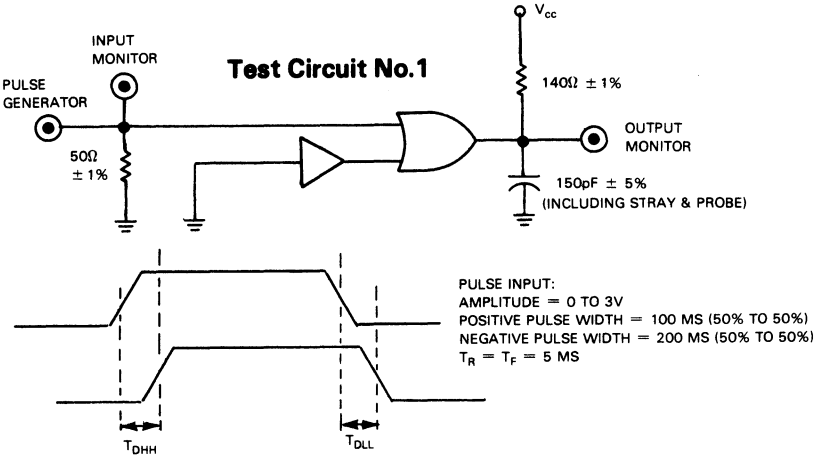
Supply voltage, V_{CC}	5.0 0.25V
Average power dissipation	1.25 W
Operating free-air temperature range, T_A	0 deg. to 70 deg.C

DC CHARACTERISTICS

SYMBOL	CHARACTERISTICS	CONDITIONS	MIN	MAX	UNITS
V_{IH}	A_{0-7} B_{0-7} $\overline{A_{IN}}$		2.0		V
V_{IL}	A_{0-7} B_{0-7} $\overline{A_{IN}}$			0.8	V
I_{IH}	$\overline{A_{IN}}$				
	A_{0-7}	$V_I = 2.5V$ $V_{CC} = 5.25V$		60	uA
		$V_I = 5.5V$ $V_{CC} = 5.25V$		1.0	mA
I_{IL}	$\overline{A_{IN}}$ A_{0-7}	$V_I = 0.5V$ $V_{CC} = 5.25V$		-1.6	mA
V_{OL}	B_{0-7}	$I_O = 40mA$ $V_{CC} = 4.75V$		0.5	V
I_{OH}	B_{0-7}	$V_O = 5.0V$ $V_{CC} = 4.75V$		60	uA
I_{CC}		$V_{CC} = 5.25V$ (INPUTS HIGH) $V_{CC} = 5.25V$ (INPUTS LOW)		97 140	mA mA
C_N				7	pF

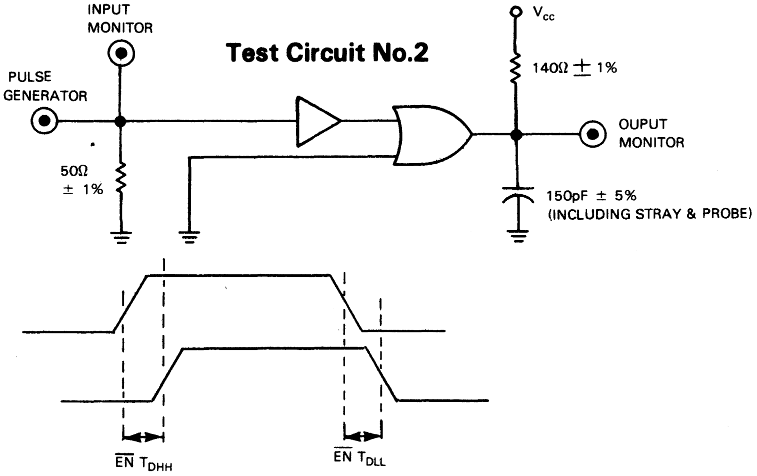
NOTE: Positive current is into the pin.

It is recommended not to exceed a 50% duty cycle for all outputs low.



PULSE INPUT:
AMPLITUDE = 0 TO 3V
POSITIVE PULSE WIDTH = 100 MS (50% TO 50%)
NEGATIVE PULSE WIDTH = 200 MS (50% TO 50%)
$T_R = T_F = 5$ MS

DG-08153


Test Circuit No.2
INPUT MONITOR
PULSE GENERATOR
50Ω ± 1%
Vcc
140Ω ± 1%
OUPUT MONITOR
150pF ± 5% (INCLUDING STRAY & PROBE)
$\overline{EN}$ T_{DHH}
$\overline{EN}$ T_{DLL}

DG-08154

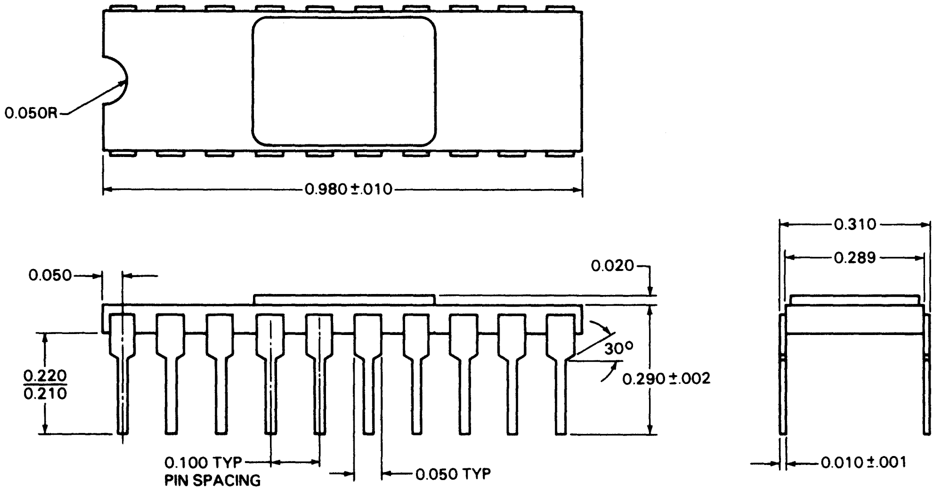
SYMBOL	CHARACTER-ISTIC	CONDITIONS	MIN	MAX	UNITS
T_{DHH}		TEST CIRCUIT #1 $V_{CC} = 5.0V$		45	ns
T_{DLL}				25	ns
$\overline{EN}$ T_{DHH}		TEST CIRCUIT #2 $V_{CC} = 5.0V$		55	ns
$\overline{EN}$ T_{DLL}				35	ns

NOTE: Times measured between 50% points.

PACKAGE SPECIFICATIONS

The physical dimensions of the mN633 chip are given below.

DIMENSIONS OF CHIP



DG-04198

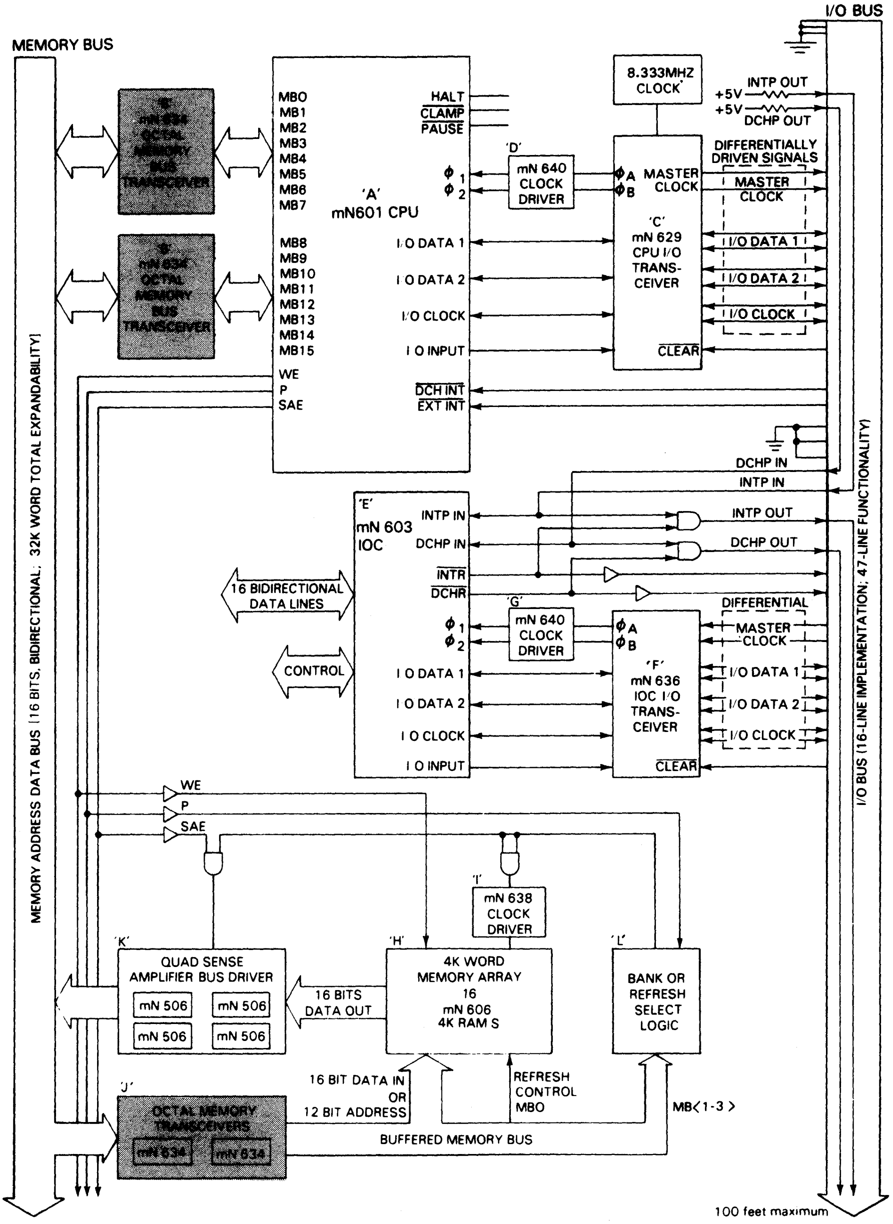
mN634
OCTAL MEMORY TRANSCEIVER

9

Data General
Data General Corporation, Westboro, Massachusetts 01581

CONTENTS

mN634
OCTAL MEMORY TRANSCEIVER



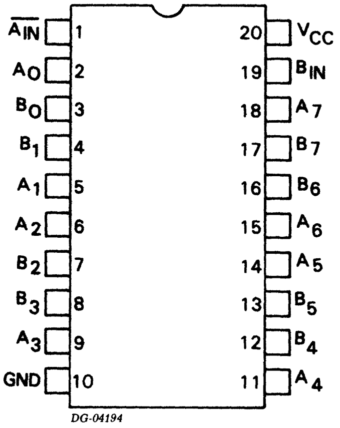
DG-04338

mN634
OCTAL MEMORY TRANSCEIVER

FEATURES

- 8 DRIVERS/RECEIVERS IN A 20-PIN CERAMIC DUAL-IN-LINE PACKAGE.
- NON-INVERTING OPEN-COLLECTOR OUTPUTS.
- SINKS 35mA per OUTPUT.

PACKAGE



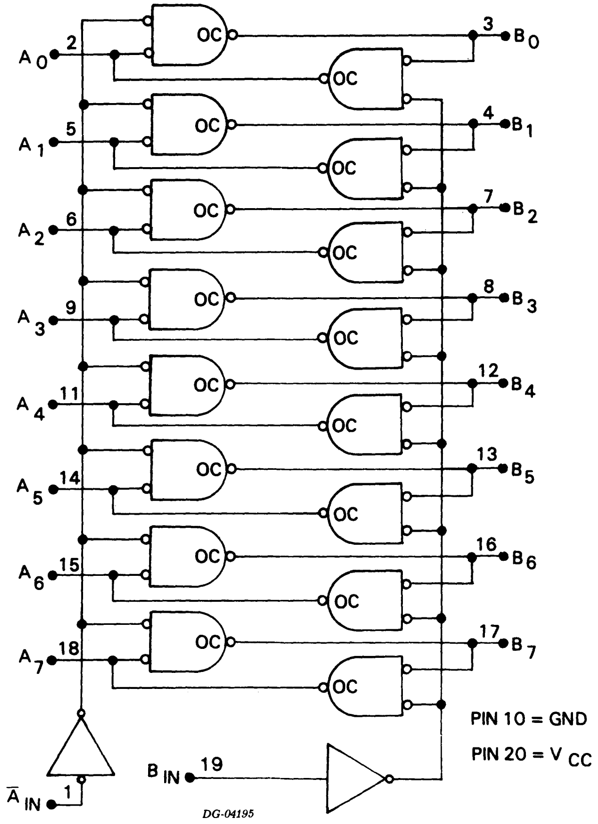
GENERAL DESCRIPTION

The mN634 Octal Memory Transceiver drives data to and from the microNOVA memory bus. Eight bits may be driven in either direction by the chip. It is used to interface the CPU to the memory bus as well as to drive the address and data lines for a memory array. See Application Notes.

PIN DESCRIPTIONS

The diagram and tables below show the connections, operations and functions of each pin in the mN634 chip.

FUNCTIONAL PIN CONNECTION DIAGRAM



TRUTH TABLE

$\overline{A}_{IN}$	B_{IN}	OPERATION
L	L	A data → B data
L	H	Low catching latch
H	L	Isolation
H	H	B data → A data

PIN FUNCTIONS

MNEMONIC	PIN NO.	IN/ OUT	FUNCTION
A_{0-7}	2,5,6,9, 11,14, 15,18	I/O	If Ain asserted low, A pins are inputs with corresponding B pins as outputs. If Bin asserted high, B pins are inputs with corresponding A pins as outputs.
B_{0-7}	3,4,7,8, 12,13 16,17	I/O	If neither Ain nor Bin is asserted, the A and B pins are open. If both Ain and Bin are asserted, the A and B outputs can be both high or latched low.
$\overline{\text{Ain}}$	1	In	When asserted low, enables A pins as inputs.
Bin	19	In	When asserted high, enables B pins as inputs.
V_{CC}	20		+5.0 ± 0.25 volts
GND	10		Ground

ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS

Supply voltage, V_{CC}	+7V
Input voltage, V_I	5.5V
Output Current, I_O	60mA
Storage temperature range, T_{STG}	-55 deg. to 125 deg.C

NOTE: Subjecting a circuit to conditions either outside these limits or at these limits for an extended period of time may cause irreparable damage to the circuit. These ratings are not intended to be used during the operation of the circuit.

RECOMMENDED OPERATING CONDITIONS

Supply voltage, V_{CC}	5.0 + 0.25V
Maximum power dissipation (averaged)	1.25 W
Operating free-air temperature range, T_A	0 deg. to 70 deg.C

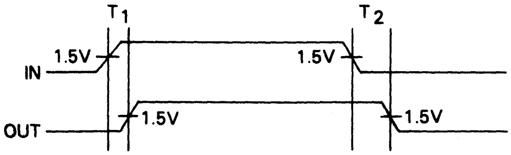
DC CHARACTERISTICS

SYMBOL	CHARACTERISTICS	CONDITIONS	MIN.	TYP.	MAX.	UNITS
V_{IH}	A_{0-7}, B_{0-7} $\overline{A}_{IN}, B_{IN}$		2.0			V
V_{IL}	A_{0-7}, B_{0-7} $\overline{A}_{IN}, B_{IN}$				.8	V
I_{IH}	$\overline{A}_{IN}, B_{IN}$	$V_I = 2.4$ V			0.06	mA
		$V_I = 5.5$ V			1	mA
	A_{0-7}, B_{0-7}	V_O=4V			0.06	mA
I_{IL}	A_{0-7}, B_{0-7} $\overline{A}_{IN}, B_{IN}$	$V_I = 0.5$ V			-2	mA
V_{OL}	A_{0-7}, B_{0-7}	I_O = 35mA			0.65	V
I_{CC}*		V_{CC} = 5.25V (INPUTS LOW)		217	280	mA
		V_{CC} = 5.25V (INPUTS HIGH)		115	170	mA
C_O		5 V OUTPUT, (OUTPUT OFF)		5		pF
		0 V OUTPUT, (OUTPUT OFF)		12		pF

NOTE: Positive current is into the pin.
*It is recommended not to exceed a 50% duty cycle for all outputs low.

AC CHARACTERISTICS
SWITCHING DIAGRAM

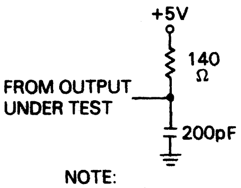
INPUT TO OUTPUT SWITCHING



DG-04196

TEST CONDITIONS

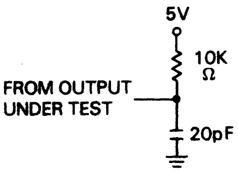
Test Circuit No.1



NOTE:
Load capacitors includes
jig and stray capacitance

DG-04197

Test Circuit No.2



DG-04341

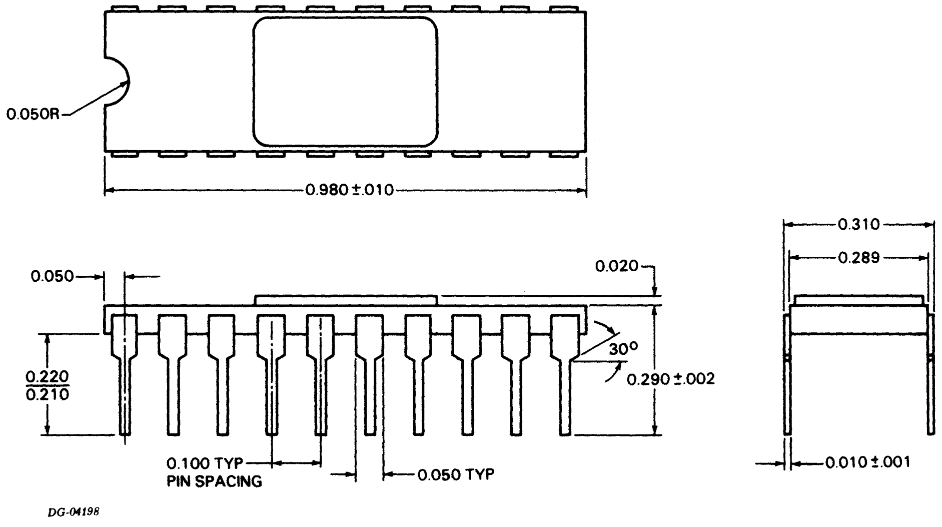
TRANSITION TIMING

SYMBOL	CHARACTERISTIC	CONDITIONS	MAX.	UNITS
T_1	INPUT TO OUTPUT DELAY FOR RISING TRANSITION	TEST CIRCUIT 1	35	ns
T_2	INPUT TO OUTPUT DELAY FOR FALLING TRANSITION	TEST CIRCUIT 1	25	ns
		TEST CIRCUIT 2	17	ns

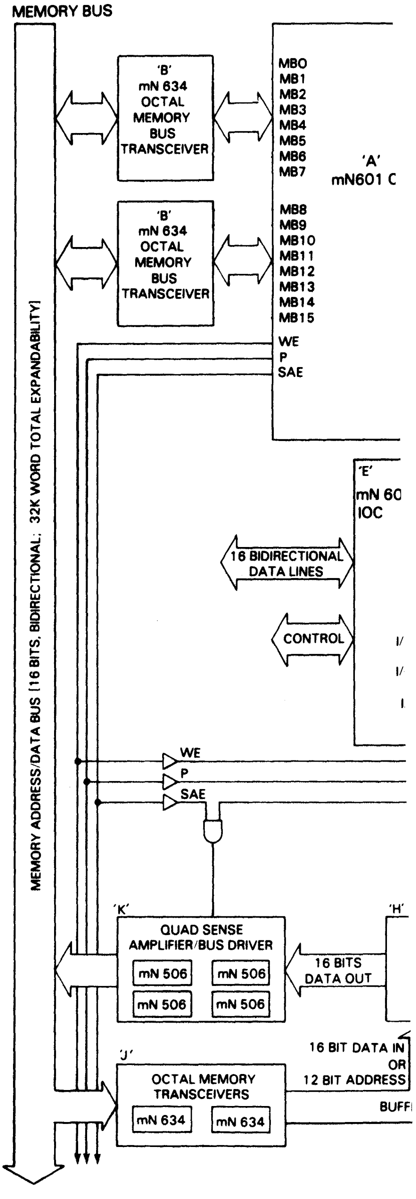
PACKAGE SPECIFICATIONS

The physical dimensions of the mN634 chip are given below.

DIMENSIONS OF CHIP



9


MEMORY BUS
'B'
mN 634
OCTAL
MEMORY
BUS
TRANSCEIVER
'B'
mN 634
OCTAL
MEMORY
BUS
TRANSCEIVER
MB0
MB1
MB2
MB3
MB4
MB5
MB6
MB7
MB8
MB9
MB10
MB11
MB12
MB13
MB14
MB15
WE
P
SAE
'A'
mN601 C
'E'
mN 60
IOC
16 BIDIRECTIONAL
DATA LINES
CONTROL
I/
I/
I
MEMORY ADDRESS/DATA BUS [16 BITS, BIDIRECTIONAL; 32K WORD TOTAL EXPANDABILITY]
WE
P
SAE
'K'
QUAD SENSE
AMPLIFIER/BUS DRIVER
mN 506
mN 506
mN 506
mN 506
16 BITS
DATA OUT
'H'
16 BIT DATA IN
OR
12 BIT ADDRESS
'J'
OCTAL MEMORY
TRANSCEIVERS
mN 634
mN 634
BUFF

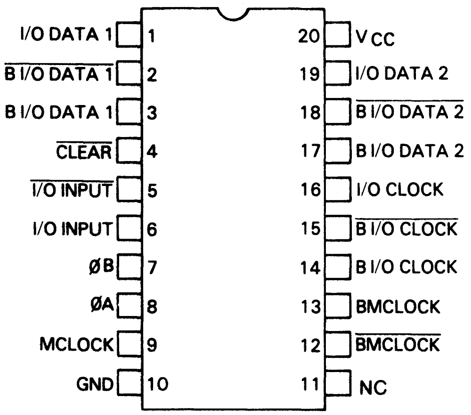
DG-04326

mN636
IOC I/O TRANSCEIVER

FEATURES

- I/O BUS RECEIVER AND TRANSMITTER IN A SINGLE 20-PIN CERAMIC PACKAGE.
- INTERFACES I/O CONTROLLER TO THE microNOVA DIFFERENTIAL I/O BUS.
- DIFFERENTIAL DRIVE CAPABILITIES:
  1. PROVIDES HIGH I/O BUS NOISE IMMUNITY.
  2. ALLOWS LONG DISTANCE APPLICATIONS (UP TO 100 FEET)
  3. ALLOWS THE USE OF INEXPENSIVE 16-CONDUCTOR RIBBON CABLE.
- SYNCHRONIZES THE IOC TO THE CPU

PACKAGE



DG-04199

GENERAL DESCRIPTION

The IOC I/O Transceiver chip interfaces the microNOVA I/O Controller chip with the bidirectional, differential I/O bus. It communicates with the CPU via a CPU I/O Transceiver. When the mN636 is in receive mode, it passes information to the I/O Controller. When it is in transmit mode, the IOC can send information to the CPU.

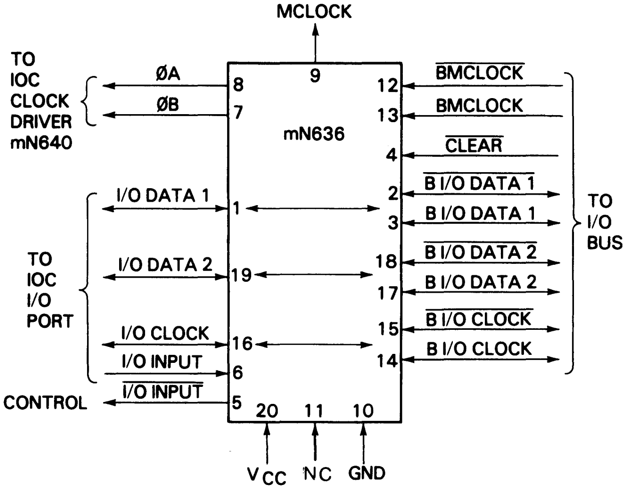
The mN636 uses differential drivers and receivers to communicate with the CPU Transceiver via the I/O bus. This allows high noise immunity over a long bus. An I/O bus length of up to 100 feet is possible using inexpensive 16-conductor ribbon cable.

In addition to passing data between an I/O Controller and the I/O bus, the mN636 receives the system master clock from the CPU Transceiver. This differential signal generates ϕA and ϕB which are used by the high voltage clock driver (mN640) to generate ϕ1 and ϕ2. These clocks provide synchronized timing for the associated I/O Controller.

PIN DESCRIPTIONS

The pin connections and their functions are given in the diagram and table below.

FUNCTIONAL PIN CONNECTION DIAGRAM


MCLOCK
TO IOC CLOCK DRIVER mN640
ØA
ØB
8
9
12
7
13
mN636
4
$\overline{\text{BMCLOCK}}$
BMCLOCK
$\overline{\text{CLEAR}}$
2
3
$\overline{\text{B I/O DATA 1}}$
B I/O DATA 1
TO I/O BUS
I/O DATA 1
1
TO IOC I/O PORT
I/O DATA 2
19
18
17
$\overline{\text{B I/O DATA 2}}$
B I/O DATA 2
I/O CLOCK
16
15
14
$\overline{\text{B I/O CLOCK}}$
B I/O CLOCK
I/O INPUT
6
CONTROL
$\overline{\text{I/O INPUT}}$
5
20
11
10
V_{CC}
NC
GND

DG-04200

PIN FUNCTIONS

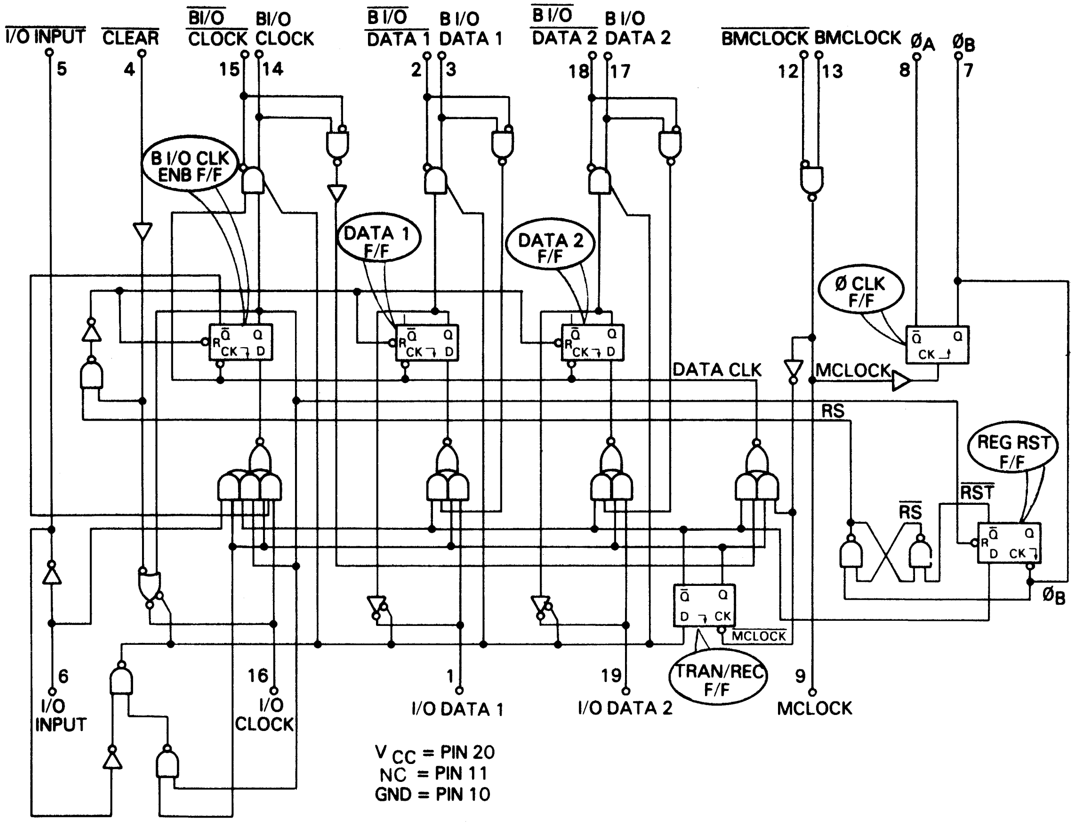
MNEMONIC	PIN NO.	IN/ OUT	FUNCTION
I/O DATA1 I/O DATA2	1 19	I/O I/O	Tri-state, bi-directional data port pins. Transfer serial data between Transceiver and IOC I/O port data pins.
I/O CLOCK	16	I/O	Tri-state, bi-directional clock line used to synchronize transmission and reception of data on I/O DATA1 and I/O DATA2 lines. Note: $\overline{\text{CLEAR}}$ asserted low pulls I/O CLOCK to its low state if Tranceiver is in Receive Mode.
I/O INPUT	6	IN	High = mN636 in receive mode. Low = mN636 in transmit mode.
$\overline{\text{I/O INPUT}}$	5	OUT	Inverted output of I/O INPUT.
$\overline{\text{CLEAR}}$	4	IN	When asserted, clears all internal flip/flops and pulls I/O CLOCK (pin 16) to the low state if the Transceiver is in the Receive Mode (used to reset associated mN603)
MCLOCK	9	OUT	MicroNOVA system master clock output.
ϕA ϕB	8 7	OUT OUT	Two-phase inverted clocks operating at one-half the BMCLOCK frequency. TTL level open collector outputs.

MNEMONIC	PIN NO.	IN/ OUT	FUNCTION
$\overline{\text{BMCLOCK}}$ BMCLOCK	12 13	IN IN	Differential pair, receives the system Master Clock transmitted by the CPU I/O Transceiver (mN629).
$\overline{\text{BI/O DATA1}}$ BI/O DATA1	2 3	I/O I/O	Open emitter, open collector differential pair. Transmit and receive serial data on I/O bus. RECEIVE MODE (I/O INPUT high) - receive differential signals from I/O bus. Clocked by BI/O CLOCK. TTL level signal output on I/O DATA1 pin. TRANSMIT MODE (I/O INPUT low) - TTL level input from I/O DATA1 transmitted differentially on I/O bus. Clocked by MCLOCK.
$\overline{\text{BI/O DATA2}}$ BI/O DATA2	18 17	I/O I/O	Open emitter, open collector differential pair. Transmit and receive serial data on I/O bus. RECEIVE MODE (I/O INPUT high) - receives differential signals from I/O bus. Clocked by BI/O CLOCK. TTL level signal output on I/O DATA2 pin. TRANSMIT MODE (I/O INPUT low) - TTL level input from I/O DATA2 transmitted differentially on I/O bus. Clocked by MCLOCK.
$\overline{\text{BI/O CLOCK}}$ BI/O CLOCK	15 14	I/O I/O	Open emitter, open collector differential pair. Used to synchronize transmission and reception of BI/O DATA <1,2> pairs. RECEIVE MODE (I/O INPUT high) - receive differential synchronizing clock from I/O bus. Clocks data in on BI/O DATA <1,2> differential pairs.TTL level signal output on I/O CLOCK pin. TRANSMIT MODE (I/O INPUT low) - TTL level input from I/O CLOCK pin used to generate a double frequency clock from MCLOCK; result is transmitted differentially to I/O bus.
			POWER
V_{CC}	20		+5.0 ± 0.25 volts
'NC	11		
GND	10		Ground

INTERNAL STRUCTURE

The mN636 consists of six flip/flops (F/F), a number of gates, three differential drivers, and four differential receivers. The I/O INPUT pin determines whether the Transceiver is in the Receive mode or the Transmit mode. When I/O INPUT is high, the Transceiver is in the receive mode. In this case, the I/O DATA1, I/O DATA2, and I/O CLOCK outputs are enabled. When I/O INPUT is low, these output drivers are disabled while the differential drivers are enabled. In this mode, the I/O DATA1, I/O DATA2, and I/O CLOCK inputs provide data to the DATA1 F/F, DATA2 F/F, and BI/O CLK ENB F/F.

INTERNAL BLOCK DIAGRAM

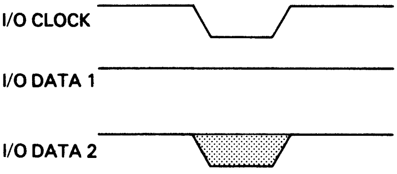


DG-04201

DATA TRANSMISSION AND RECEPTION

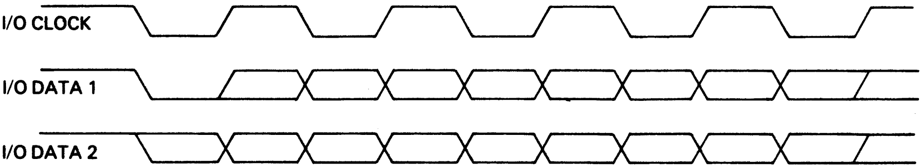
When connected to the microNOVA I/O bus, the mN636 performs two general types of data transfers; short and long as shown below:

SHORT TRANSFER



DG-04184

LONG TRANSFER



DG-04185

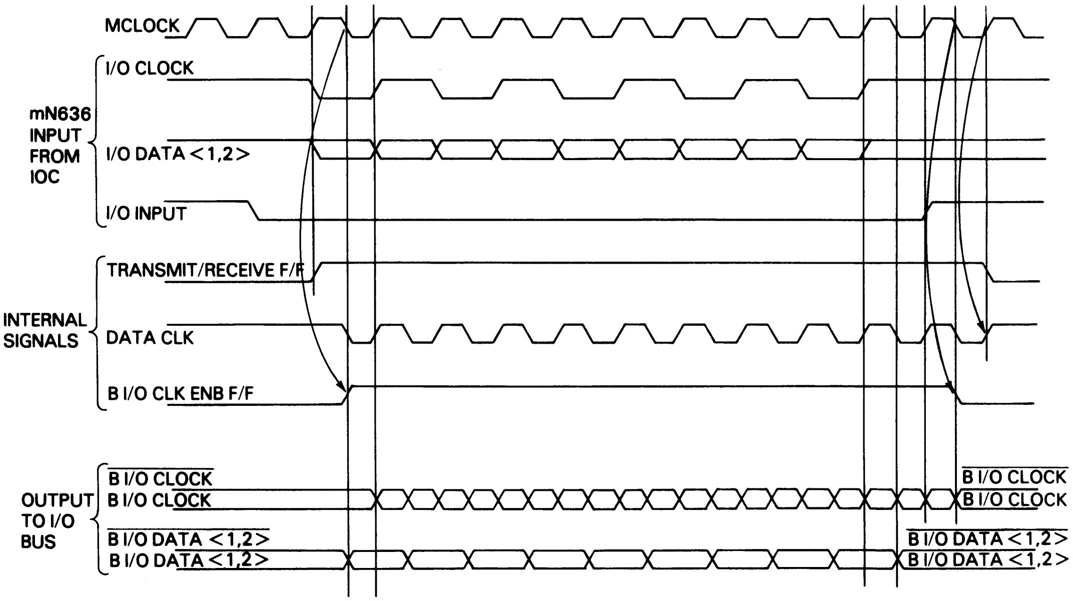
The short type consists of one I/O CLOCK pulse and transfers two bits. The long type consists of five I/O CLOCK pulses and transfers 18 bits. For the purpose of illustrating the operation of the Transceiver, the long transfer is used as an example. The one pulse transfer is simply a shortened version of the long transfer as far as the Transceiver's operation is concerned. See the Introduction and mN603/mN613, I/O Data Port for further explanation of the I/O protocols.

The Transceiver operates under two modes; transmit and receive. The input waveforms, some internal signals, output waveforms and a brief discussion of each mode are given below.

Transmit Mode

Transmit mode begins when I/O INPUT is asserted low (by an IOC). The next rising edge of MCLOCK sets the Transmit/Receive F/F allowing MCLOCK to clock the DATA CLK line. As shown in the block diagram, DATA CLK clocks data into the DATA1 F/F and DATA2 F/F. These flip/flops supply the data to the differential drivers. In addition, DATA CLK sets the BI/O CLK ENB F/F for the duration of the transmission. The high output of the BI/O CLK ENB F/F allows DATA CLK to be transmitted on the I/O bus via the BI/O CLOCK differential driver. A long transfer from the IOC to the I/O bus is shown in the following timing diagram.

TRANSMIT WAVEFORMS



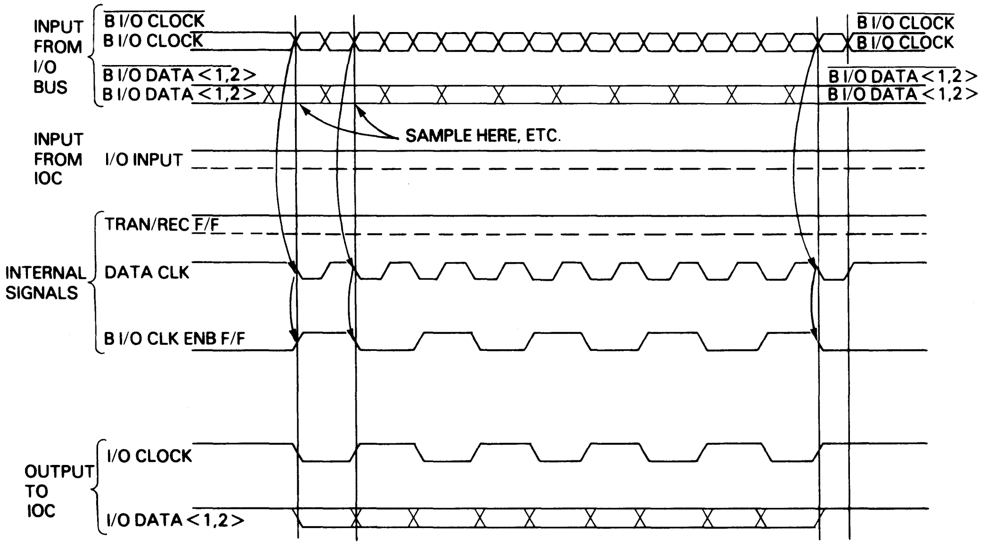
DG-04204

NOTE: MCLOCK and DATA CLK operate at twice the frequency of I/O CLOCK. This provides a transition of BI/O CLOCK near the middle of every BI/O DATA < 1,2 > pulse allowing the receiving transceiver to sample near the middle of each data bit.

Receive Mode

Receive mode begins when I/O INPUT is asserted high. The first BI/O CLOCK transition from the I/O bus initiates data reception. The signals received from the I/O bus are in the same form as those transmitted. The Transmit/Receive F/F is in the reset state allowing BI/O CLOCK to generate DATA CLK. Data is clocked into the DATA1 F/F and DATA2 F/F on every falling edge of DATA CLK. In addition, the BI/O CLK ENB F/F is toggled on each falling edge of DATA CLK. An inverted output of this flip/flop drives the I/O CLOCK pin. A long transfer from the I/O bus to the IOC is shown in the following timing diagram.

RECEIVE WAVEFORMS



DG-04205

NOTE: The last BI/O CLOCK transition causes the Transceiver to sample the differential data lines after the last bit of data has been received. This ensures that the DATA1 and DATA2 F/Fs are reset to their normal state at the completion of the transfer.

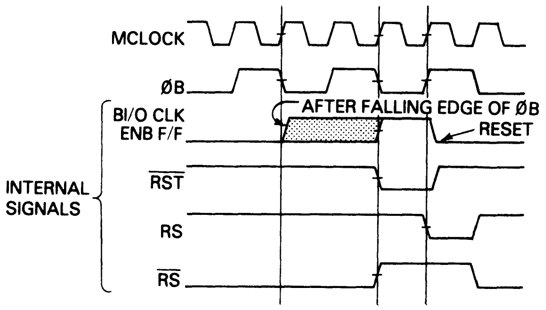
REGISTER RESET CIRCUIT

This circuit consists of the REG RST F/F, two NAND gates, and the associated control lines (see lower right corner of Internal Block Diagram). While in the receive mode, the Register Reset Circuit ensures that the BI/O CLK ENB F/F will be reset if set by a spurious transition of BI/O CLOCK.

Under normal operation, the BI/O CLK ENB F/F is clocked either twice or ten times depending on whether the Transceiver receives a short or a long type of transfer. Therefore, there should always be an even number of BI/O CLOCK transitions. If for some reason a single transition of BI/O CLOCK is received, the BI/O CLK ENB F/F must be reset to avoid holding I/O CLOCK low and resetting the associated IOC (see mN603/mN613 System Requirements).

If the BI/O CLK ENB F/F is set anywhere during the period shaded in the following timing diagram, it is reset after the rising edge of φB as described below.

B I/O CLK ENB F/F RESET TIMING



DG-04188

While the BI/O CLK ENB F/F is in the reset state, its Q output is 0. This forces the REG RST F/F to its reset state. If the BI/O CLK ENB F/F is clocked by a fraudulent transition of BI/O CLOCK, its Q output is no longer 0 and the REG RST F/F is no longer forced to the reset state. On the next falling edge of φB, the REG RST F/F is set placing RST low. Though $\overline{RST}$ is now low, RS remains high since φB is low. When φB goes high, RS goes low clearing the BI/O CLK ENB F/F. Its Q output goes low immediately resetting the REG RST F/F and placing $\overline{RST}$ back in its normal state. RS and $\overline{RS}$ return to their normal state on the next falling edge of φB.

The BI/O CLK ENB F/F is not reset during normal reception because the reset circuit has a built-in delay of one MCLOCK period. Even if a transition occurs at the end of the period shaded in the above timing diagram, there is enough time for the second transition to occur before the rising edge of φB.

CLOCKS

The system master clock is received via the BM CLOCK differential pair. This differential receiver clocks the φCLK F/F which provides the two-phase (half-frequency, TTL level) clock for the I/O Controller Clock Driver (mN640). A TTL level version of the system Master Clock is provided on the MCLOCK pin.

CLEAR

The $\overline{CLEAR}$ pin is used during system reset. When asserted low, all the mN636's internal flip/flops are reset. If the Transceiver is in the receive mode, the I/O CLOCK pin is pulled low (this resets the associated I/O Controller, see mN603). If the Transceiver is in the transmit mode, the I/O CLOCK pin is not affected by the assertion of $\overline{CLEAR}$ until the transmission is over and I/O INPUT is pulled high.

ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS

Supply voltage, V_{CC}	7V
Input voltage, V_I	5.5V
Transceiver output current, I_{OT}	80mA
All other output current, I_O	50mA
Operating free-air temperature range, T_A	O deg. to 70 deg.C
Storage temperature range, T_{STG}	-55 deg. to 125 deg.C

NOTE: Subjecting a circuit to conditions either outside these limits or at these limits for an extended period of time may cause irreparable damage to the circuit. These ratings are not intended to be used during the operation of the circuit.

RECOMMENDED OPERATING CONDITIONS

Supply voltage, V_{CC}	+5.0 ± 0.25V
Average power dissipation	0.95 W
Operating free-air temperature range, T_A	O deg. to 70 deg.C

DC CHARACTERISTICS

SYMBOL	CHARACTERISTICS	CONDITIONS	MIN.	TYP.	MAX.	UNITS
V_{IH}	I/O INPUT I/O DATA1,I/O DATA2 $\overline{\text{CLEAR}}$, I/O CLOCK		2.0			V
	BMCLOCK,$\overline{\text{BMCLOCK}}$ BI/O CLOCK,$\overline{\text{BI/O CLOCK}}$ BI/O DATA1,$\overline{\text{BI/O DATA1}}$ BI/O DATA2,$\overline{\text{BI/O DATA2}}$				5.25	V
V_{IL}	I/O CLOCK I/O INPUT I/O DATA1 I/O DATA2 $\overline{\text{CLEAR}}$				0.8	V
	BI/O CLOCK,$\overline{\text{BI/O CLOCK}}$ BI/O DATA1,$\overline{\text{BI/O DATA1}}$ BI/O DATA2,$\overline{\text{BI/O DATA2}}$ BMCLOCK,$\overline{\text{BMCLOCK}}$		-0.5			V

DC CHARACTERISTICS (CONT)

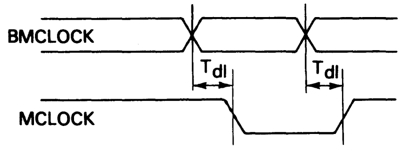
SYMBOL	CHARACTERISTICS	CONDITIONS	MIN.	TYP.	MAX.	UNITS
$V_{ID} = \lvert V_{IL.} - V_{IH} \rvert$	(BI/O CLOCK,$\overline{\text{BI/O CLOCK}}$) (BI/O DATA1,$\overline{\text{BI/O DATA1}}$) (BI/O DATA2,$\overline{\text{BI/O DATA2}}$) (BMCLOCK,$\overline{\text{BMCLOCK}}$)	REQUIRED VALUE OF $\lvert V_{IL}-V_{IH} \rvert$ FOR VALID SIGNAL DETECTION	0.75	0.5		V
I_{IH}	I/O INPUT	$V_I = 2.4V$		0	80	uA
	I/O DATA1 I/O DATA2 I/O CLOCK $\overline{\text{CLEAR}}$	$V_I = 2.4V$		0	40	uA
I_{IL}	I/O CLOCK I/O INPUT I/O DATA1 I/O DATA2 $\overline{\text{CLEAR}}$	$V_I = 0.5 V$		-0.5	-1.6	mA
I_{ID}	BI/O CLOCK,$\overline{\text{BI/O CLOCK}}$ BI/O DATA1,$\overline{\text{BI/O DATA1}}$ BI/O DATA2,$\overline{\text{BI/O DATA2}}$ BMCLOCK,$\overline{\text{BMCLOCK}}$	$0 55 V \leq V_I \leq 2 0 V$			500	uA
V_{OH}	$\overline{\text{I/O INPUT}}$	$I_O = -400uA$ $V_{CC} = 4.75 V$	2.5	3.5		V
	I/O CLOCK I/O DATA1 I/O DATA2	$I_O = -20uA$ $V_{CC} = 4.75 V$	3.0	3.8		V
	BI/O CLOCK (OPEN BI/O DATA1 EMITTER BI/O DATA2 OUTPUTS)	$I_O = -35 mA$ $V_{CC} = 5.25 V$			3.2	V
		$I_O = -35mA$ $V_{CC} = 4.75 V$	2.0	2.5		V
	MCLOCK	$V_{CC} = 4.75 V$ $I_{OH} = 400mA$	2.4	3.25		V
	ØA , ØB	$V_{CC} = 5.25V$, $RL = 1K$	4.95			V
V_{OL}	ØA , ØB	$I_O = 10mA$ $V_{CC} = 4.75 V$		0.35	0.5	V
	$\overline{\text{I/O INPUT}}$ I/O DATA1 I/O DATA2 I/O CLOCK, MCLOCK	$I_O = 5mA$ $V_{CC} = 4.75 V$		0.35	0.5	V
	$\overline{\text{BI/O CLOCK}}$ $\overline{\text{BI/O DATA1}}$ $\overline{\text{BI/O DATA2}}$	$I_O = 35mA$ $V_{CC} = 4.75V$		0.38	0.55	V
I_{CC}	TRANSMIT MODE	$V_{CC} = 5.25V$		130	170	mA
	RECEIVE MODE			130	170	mA
C_I	ALL INPUTS			7		pf

NOTE: Positive current is into the pin.

DG-04381

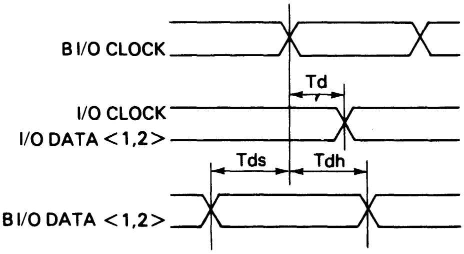
AC CHARACTERISTICS SWITCHING DIAGRAMS

BMCLOCK RECEIVER TIMING



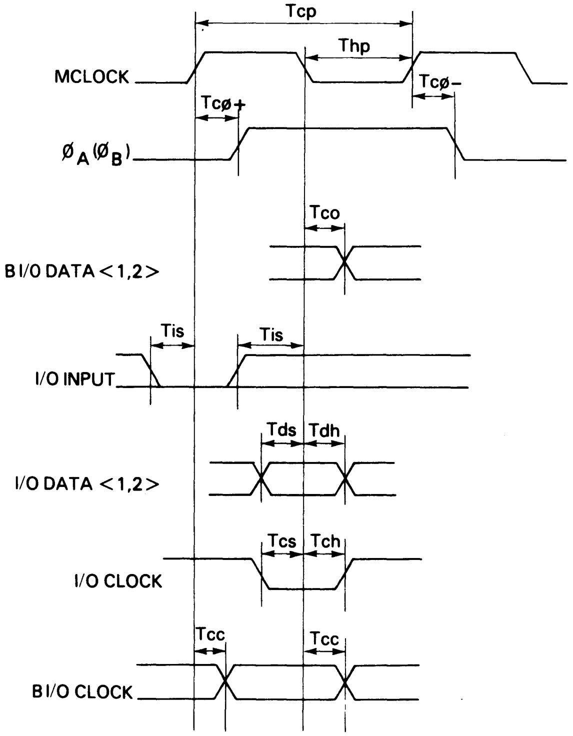
DG-04189

RECEIVER TIMING



DG-04190

TRANSMITTER TIMING

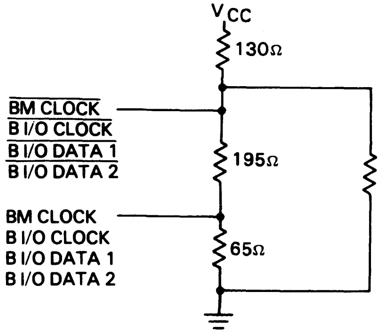


NOTES:

- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED FROM A 1.5 VOLT LEVEL.
- SEE TIMING TABLE ON PAGE 12 .

DG-04191

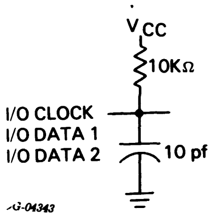
TEST CIRCUIT NO. 1



NOTE: ALL RESISTORS ARE ± 1%
ALL CAPACITORS ARE ± 5%
LOAD SHOULD BE CONNECTED TO
OUTPUT USING LESS THAN
2 INCHES OF WIRE.

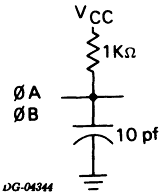
DG-04192

TEST CIRCUIT NO. 2



-G-04343

TEST CIRCUIT NO. 3



DG-04344

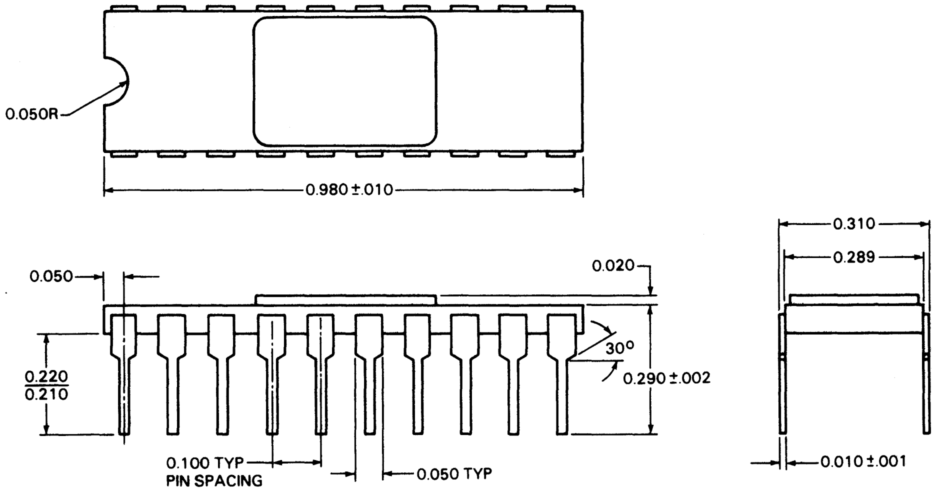
TRANSITION TIMING

SYMBOL	CHARACTERISTIC	MIN.	MAX.	UNIT	TEST CIRCUIT
T_{cp}	MASTER CLOCK PERIOD	99	125	ns	
T_{hp}	HALF CYCLE PERIOD	40	60	ns	
$T_{c\phi+}$	DELAY FROM MCLOCK TO ØA, ØB (RISING EDGE)	30	70	ns	NO. 3
$T_{c\phi-}$	DELAY FROM MCLOCK TO ØA, ØB (FALLING EDGE)	25	60	ns	NO. 3
T_{is-}	SET UP TIME FOR I/O INPUT	35	---	ns	
T_{is+}	SET UP TIME FOR I/O INPUT	20	---	ns	
T_{dl}	DELAY FROM MCLOCK TRANSITION TO BMCLOCK TRANSITION	5	40	ns	NO 1
	TRANSMITTER TIMING				
T_{cc}	DELAY FROM MCLOCK TO BI/O CLOCK	15	55	ns	NO. 1
T_{co}	DELAY FROM MCLOCK TO BI/O DATA	25	65	ns	NO 1
T_{ds}	DATA SETUP TIME	30	---	ns	NO 1
T_{dh}	DATA HOLD TIME	10	---	ns	NO. 1
T_{cs}	I/O CLOCK SETUP TIME	30	---	ns	NO 1
T_{ch}	I/O CLOCK HOLD TIME	10	---	ns	NO 1
	RECEIVER TIMING				
T_{d}	DELAY FROM BI/O CLOCK TO I/O CLOCK AND I/O DATA(1,2)	30	80	ns	NO. 2
T_{ds}	DATA SETUP TIME	15	---	ns	NO. 2
T_{dh}	DATA HOLD TIME	10	---	ns	NO. 2

PACKAGE SPECIFICATION

The physical dimensions of the mN636 chip are given below.

DIMENSIONS OF CHIP



DG-04198

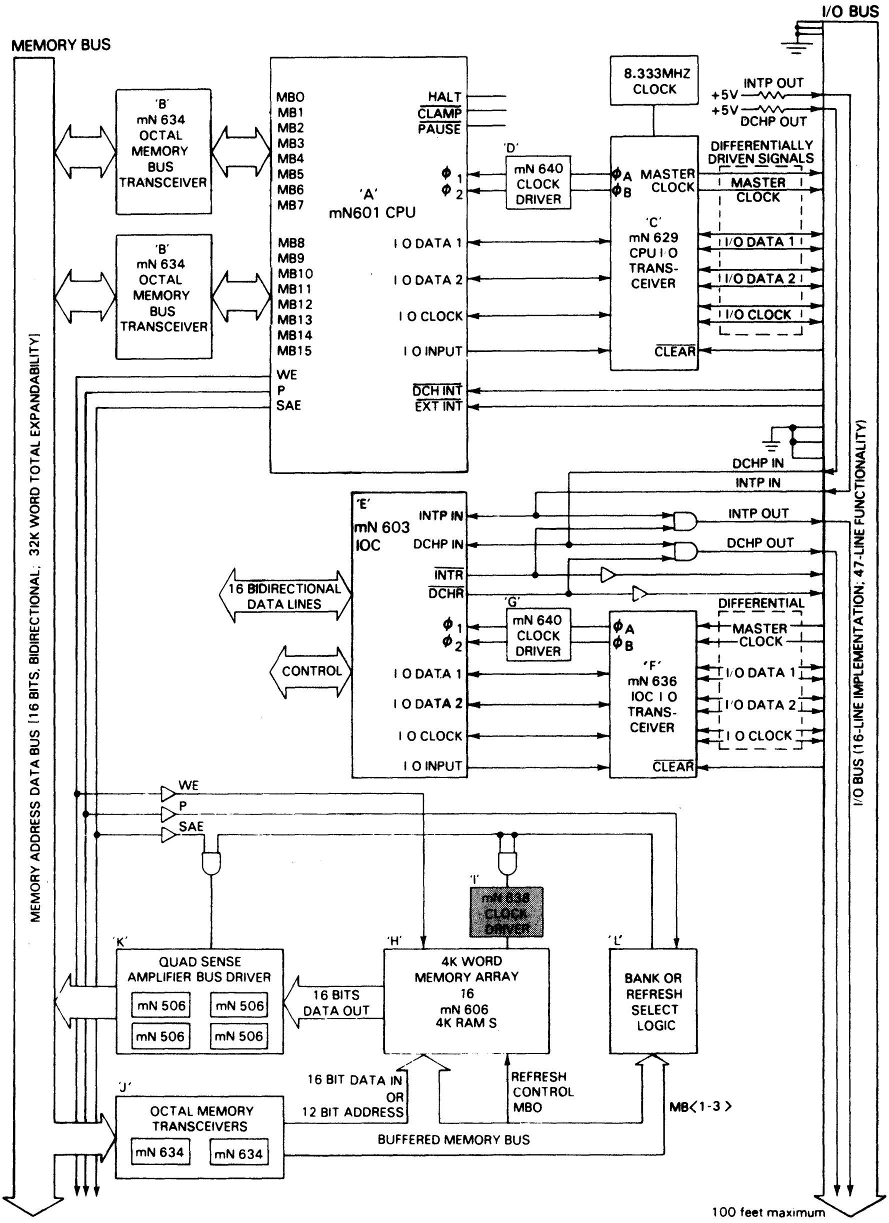
mN638
MEMORY CLOCK DRIVER

11

Data General

Data General Corporation, Westboro, Massachusetts 01581

mN638
MEMORY CLOCK DRIVER

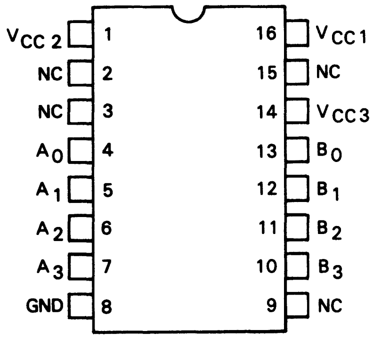

MEMORY BUS
I/O BUS
MEMORY ADDRESS DATA BUS (16 BITS, BIDIRECTIONAL; 32K WORD TOTAL EXPANDABILITY)
I/O BUS (16-LINE IMPLEMENTATION; 47-LINE FUNCTIONALITY)
'B' mN 634 OCTAL MEMORY BUS TRANSCEIVER
'B' mN 634 OCTAL MEMORY BUS TRANSCEIVER
'A' mN601 CPU
MB0 MB1 MB2 MB3 MB4 MB5 MB6 MB7 MB8 MB9 MB10 MB11 MB12 MB13 MB14 MB15
WE P SAE
HALT CLAMP PAUSE
ɸ 1 ɸ 2
I O DATA 1
I O DATA 2
I O CLOCK
I O INPUT
DCH INT
EXT INT
8.333MHZ CLOCK
'D' mN 640 CLOCK DRIVER
ɸA ɸB MASTER CLOCK
'C' mN 629 CPU I O TRANS-CEIVER
CLEAR
+5V INTP OUT
+5V DCHP OUT
DIFFERENTIALLY DRIVEN SIGNALS
MASTER CLOCK
I/O DATA 1
I/O DATA 2
I/O CLOCK
DCHP IN
INTP IN
'E' mN 603 IOC
INTP IN
DCHP IN
INTR
DCHR
INTP OUT
DCHP OUT
16 BIDIRECTIONAL DATA LINES
CONTROL
'G' mN 640 CLOCK DRIVER
ɸ 1 ɸ 2
ɸA ɸB
'F' mN 636 IOC I O TRANS-CEIVER
I O DATA 1
I O DATA 2
I O CLOCK
I O INPUT
CLEAR
DIFFERENTIAL
MASTER CLOCK
I/O DATA 1
I/O DATA 2
I O CLOCK
WE
P
SAE
'I' mN 638 CLOCK DRIVER
'K' QUAD SENSE AMPLIFIER BUS DRIVER
mN 506 mN 506 mN 506 mN 506
16 BITS DATA OUT
'H' 4K WORD MEMORY ARRAY 16 mN 606 4K RAM S
'L' BANK OR REFRESH SELECT LOGIC
16 BIT DATA IN OR 12 BIT ADDRESS
REFRESH CONTROL MBO
MB<1-3>
'J' OCTAL MEMORY TRANSCEIVERS
mN 634 mN 634
BUFFERED MEMORY BUS
100 feet maximum

DG-04342

FEATURES

- DUAL 3 INPUT BIPOLAR to MOS DRIVER IN A 16-PIN CERDIP PACKAGE.
- TTL COMPATIBLE INPUTS, HIGH VOLTAGE OUTPUTS COMPATIBLE WITH microNOVA mN606 MOS RANDOM ACCESS MEMORY CLOCK INPUTS.

PACKAGE



NC = no connection

DG-04211

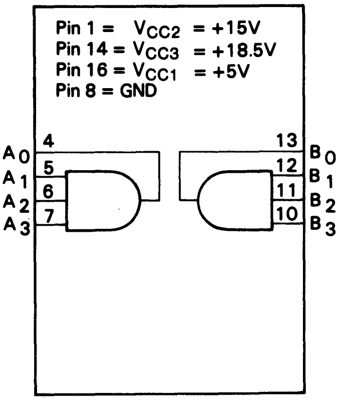
GENERAL DESCRIPTION

The mN638 Clock Driver chips contains two TTL level to high voltage converters. These may be used to drive the clock input of the mN606 Random Access Memory chip.

PIN DESCRIPTIONS

The pin connections and their functions are described in the diagram and table below.

FUNCTIONAL PIN CONNECTION DIAGRAM



DG-04212

PIN FUNCTIONS

MNEMONIC	PIN NO.	IN/ OUT	FUNCTION
A_1, A_2, A_3	5,6,7	In	TTL inputs to 3-input AND gate. A_O is high voltage output.
A_O	4	Out	High Voltage output of 3-input (A_1,A_2,A_3) AND gate.
B_1, B_2, B_3	12, 11, 10	In	TTL inputs to 3-input AND gate. B_O is high voltage output.
B_O	13	Out	High voltage output of 3-input (B_1,B_2,B_3) AND gate.
V_{CC1}	16		+5 ± 0.25 volts
V_{CC2}	1		+15 ± 1.0 volts
V_{CC3}	14		+18.5 ± 0.75 volts
GND	8		Ground

ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS

Supply voltage, V_{CC1}	7V
Supply voltage, V_{CC2}	18V
Supply voltage, V_{CC3}	21V
Input voltage, V_I	5.5V
Output current, I_O	60mA
Storage temperature range, T_{STG}	-55 deg. to 125 deg. C

NOTE: Subjecting a circuit to conditions either outside these limits or at these limits for an extended period of time may cause irreparable damage to the circuit. These ratings are not intended to be used during the operation of the circuit.

RECOMMENDED OPERATING CONDITIONS

Supply voltage, V_{CC1}	5.0 ± 0.25V
Supply voltage, V_{CC2}	15.0 ± 1V
Supply voltage, V_{CC3}	18.5 ± 0.75V
Average power dissipation	1.0 W
Operating free-air temperature range, T_A	0 deg. to 70 deg. C

DC CHARACTERISTICS

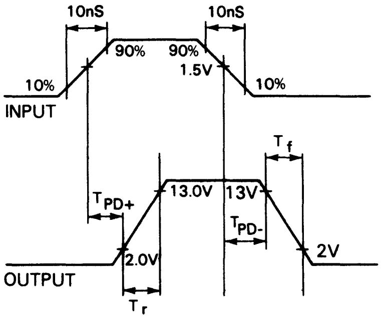
SYMBOL	CHARACTERISTICS	CONDITIONS	MIN.	MAX.	UNITS
V_{IH}	A_{1-3}, B 1-3		2.0		V
V_{IL}	A_{1-3}, B 1-3			0.8	V
I_{IH}	A_{1-3}, B 1-3	V_{CC1} = 5.25V V_I = 2.4V		40	µA
I_{IL}	A_{1-3}, B 1-3	V_{CC1} = 5.25V V_I = 0.5V		-1.6	mA

DC CHARACTERISTICS (CONT.)

SYMBOL	CHARACTERISTICS	CONDITIONS	MIN.	MAX.	UNITS
V_{OH}	A_O B_O	V_{CC1} = 4.75V V_{CC2} = 14V V_{CC3} = 17.75V I_O = -20mA	13		V
V_{OL}	A_O B_O	V_{CC1} = 4.75 V I_O = 25mA	0	0.5	V
I_{CC1H}	SUPPLY CURRENT	V_{CC1} = 5.5V		9	mA
I_{CC2H}	SUPPLY CURRENT	V_{CC2} = 16V	-2.7	.25	mA
I_{CC3H}	SUPPLY CURRENT	V_{CC3} = 19.25V		4	mA
I_{CC1L}	SUPPLY CURRENT	V_{CC1} = 5.5V		80	mA
I_{CC2L}	SUPPLY CURRENT	V_{CC2} = 16V		250	µA
I_{CC3H}	SUPPLY CURRENT	V_{CC3} = 19.25V		20	mA

NOTE: Positive current is into the pin.

AC CHARACTERISTICS
SWITCHING DIAGRAM

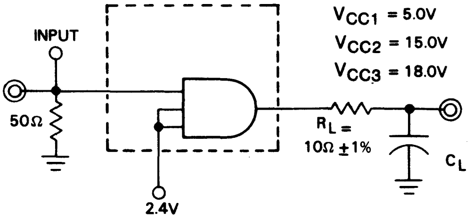


DG-04213

TRANSITION TIMING

SYMBOL	CHARACTERISTIC	CONDITIONS	MIN.	MAX.	UNITS
T_r	OUTPUT RISE TIME (A_O, B_O)	C_L = 15pF	5		ns
		C_L = 470pF		25	ns
T_f	OUTPUT FALL TIME (A_O, B_O)	C_L = 15pF	5		ns
		C_L = 470pF		25	ns
T_{PD+}	PROPAGATION DELAY	C_L = 15pF	3		ns
T_{PD-}		C_L = 470pF		20	ns

TEST CONDITIONS



V_{CC1} = 5.0V
V_{CC2} = 15.0V
V_{CC3} = 18.0V

1. Input Pulse: Amplitude = 0 to .3v
Width = 400nS (50% to 50%)
t r = t f = 10nS
Freq. = 1.67MHz

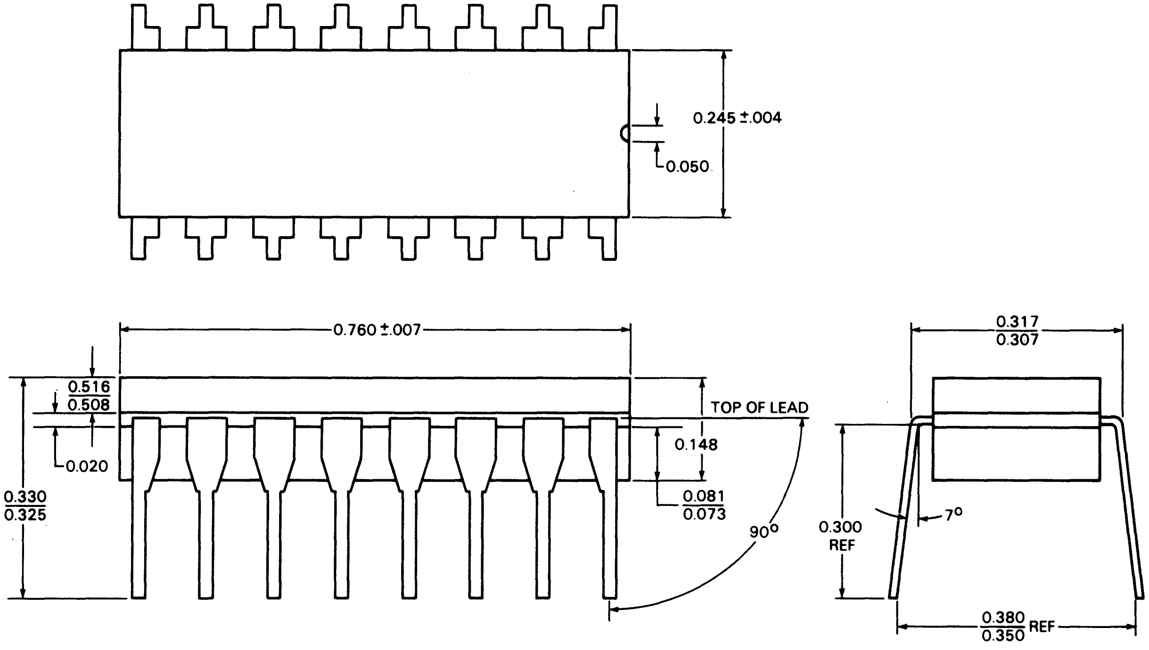
NOTE: LOAD CAPACITOR INCLUDES JIG AND WIRING CAPACITANCE.

DG-04214

PACKAGE SPECIFICATIONS

The physical dimensions of the mN638 chip are given below.

DIMENSIONS OF CHIP



DG-04240

mN640
CPU/IOC CLOCK DRIVER

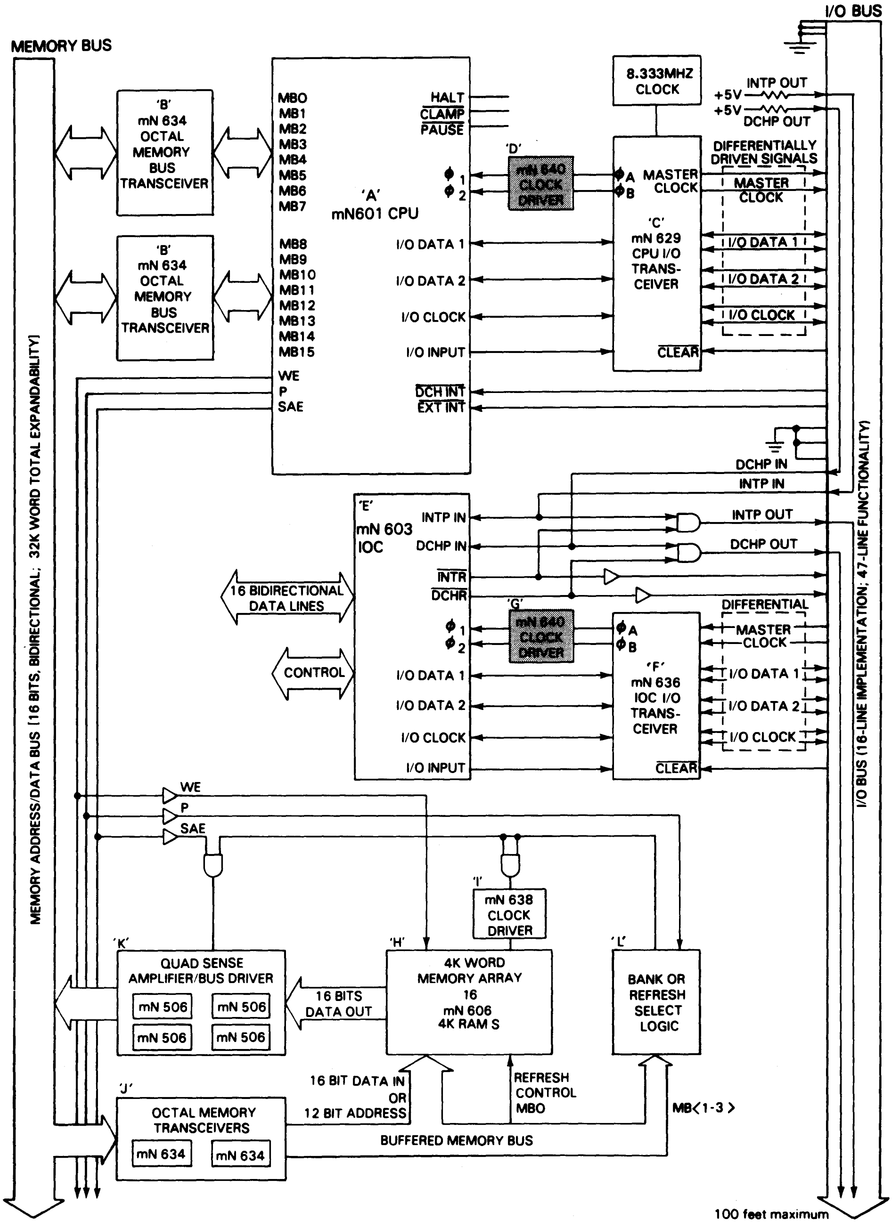
12

Data General

Data General Corporation, Westboro, Massachusetts 01581

mN640
CONTENTS

mN640
CPU/IOC CLOCK DRIVER



DG-04339

Data General

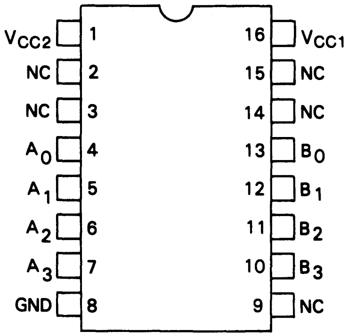
Data General Corporation, Westboro, Massachusetts 01581

mN640
CPU AND IOC CLOCK DRIVER

FEATURES

- DUAL 3 INPUT BIPOLAR TO MOS DRIVER IN A 16-PIN CERDIP PACKAGE.
- TTL COMPATIBLE INPUTS, HIGH VOLTAGE OUTPUTS COMPATIBLE WITH microNOVA CPU AND I/O CONTROLLER MOS CLOCK INPUTS.

PACKAGE



NC = no connection

DG-04215

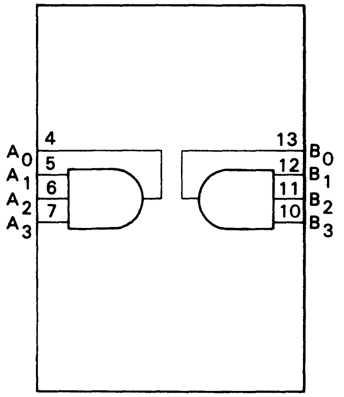
GENERAL DESCRIPTION

The mN640 Clock Driver chip contains two TTL level to high voltage converter gates. These are used to drive the two-phase clock inputs of the CPU and IOC. Additional signal shaping circuitry, required to shape the non-overlapping clock signals, is described in the Design Notes.

PIN DESCRIPTIONS

The pin connections and their functions are described in the diagram and table below.

FUNCTIONAL PIN CONNECTION DIAGRAM



DG-04216

PIN FUNCTIONS

MNEMONIC	PIN NO.	IN/ OUT	FUNCTION
A1,A2,A3	5,6,7	In	TTL inputs to 3-input AND gate. A_O is high voltage output.
A_O	4	Out	High voltage output of 3-input (A_1,A_2,A_3) AND gate.
B1,B2,B3	12,11, 10	In	TTL inputs to 3-input AND gate. B_O is high voltage output.
B_O	13	Out	High voltage output of 3-input (B_1,B_2,B_3) AND gate.
V_{CC1}	16		5 ± 0.25 volts
V_{CC2}	1		15 ± 1.0 volts
GND	8		Ground

ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS

Supply voltage, V_{CC1}	7V
Supply voltage, V_{CC2}	18V
Input voltage, V_I	5.5V
Output current, I_O	60mA
Storage temperature range, T_{STG}	-55 deg. to 125 deg.C

NOTE: Subjecting a circuit to conditions either outside these limits or at these limits for an extended period of time may cause irreparable damage to the circuit. These ratings are not intended to be used during the operation of the circuit.

RECOMMENDED OPERATING CONDITIONS

Supply voltage, V_{CC1}	5.0 ± 0.25V
Supply voltage, V_{CC2}	15.0 ± 1.0V
Average power dissipation	1.0 W
Operating free-air temperature range T_A	0 deg.C to 70 deg.C

DC CHARACTERISTICS

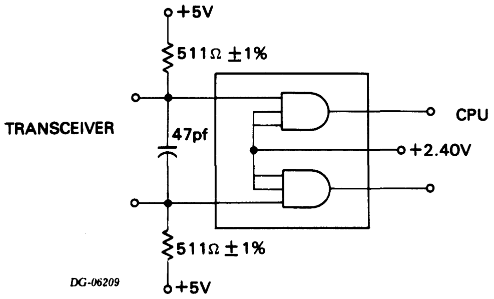
SYMBOL	CHARACTERISTICS	CONDITIONS	MIN.	MAX.	UNITS
V_{IH}	A_{1-3}, B_{1-3}		2.0		V
V_{IL}	A_{1-3}, B_{1-3}			0.8	V
I_{IH}	A_{1-3}, B_{1-3}	V_{CC1} = 5.25V V_I = 2.4V		40	μA
I_{IL}	A_{1-3}, B_{1-3}	V_{CC1} = 5.25V V_I = 0.4V		-1.6	mA

DC CHARACTERISTICS (CONT.)

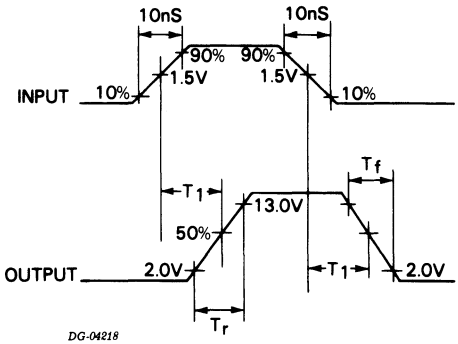
SYMBOL	CHARACTERISTICS	CONDITIONS	MIN.	MAX.	UNITS
V_{OH}	A_O, B_O	V_{CC1} = 4.75V V_{CC2} = 14V I_O = 500 µA	13		V
V_{OL}	A_O, B_O	V_{CC1} = 4.75 V I_O = 25mA	0	0.5	V
I_{CC1H} + I_{CC2H}	SUPPLY CURRENT	V_{CC1} = 5.5V		9	mA
	SUPPLY CURRENT	V_{CC2} = 16V		250	µA
I_{CC1L} + I_{CC2L}	SUPPLY CURRENT	V_{CC1} = 5.5V		60	mA
	SUPPLY CURRENT	V_{CC2} = 16V		42	mA

NOTE: Positive current is into the pin.

RECOMMENDED CONFIGURATION



AC CHARACTERISTICS SWITCHING DIAGRAM

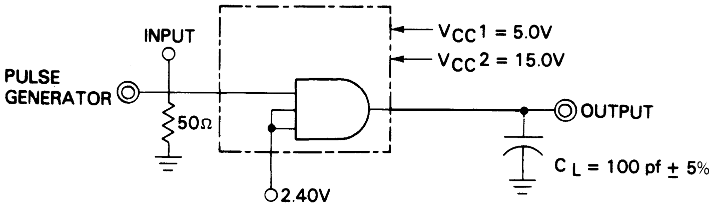


DG-04218

TRANSITION TIMING

SYMBOL	CHARACTERISTICS	CONDITIONS	MIN	MAX	UNITS
T_r	OUTPUT RISE TIME	V_{CC1} = 5.0V V_{CC2} = 15.0V TEST CIRCUIT	4	20	ns
T_f	OUTPUT FALL TIME	V_{CC1} = 5.0V V_{CC2} = 15.0V TEST CIRCUIT	4	20	ns
T_1	PROPAGATION DELAY	V_{CC1} = 5.0V V_{CC2} = 15.0V TEST CIRCUIT	15	35	ns

TEST CONDITIONS



1. Test each input separately
2. Input Pulse: Amplitude = 0 to 3V
   Width = 100nS (50% to 50%)
   $t_r = t_f$ = 10nS
   Freq. = 5MHz

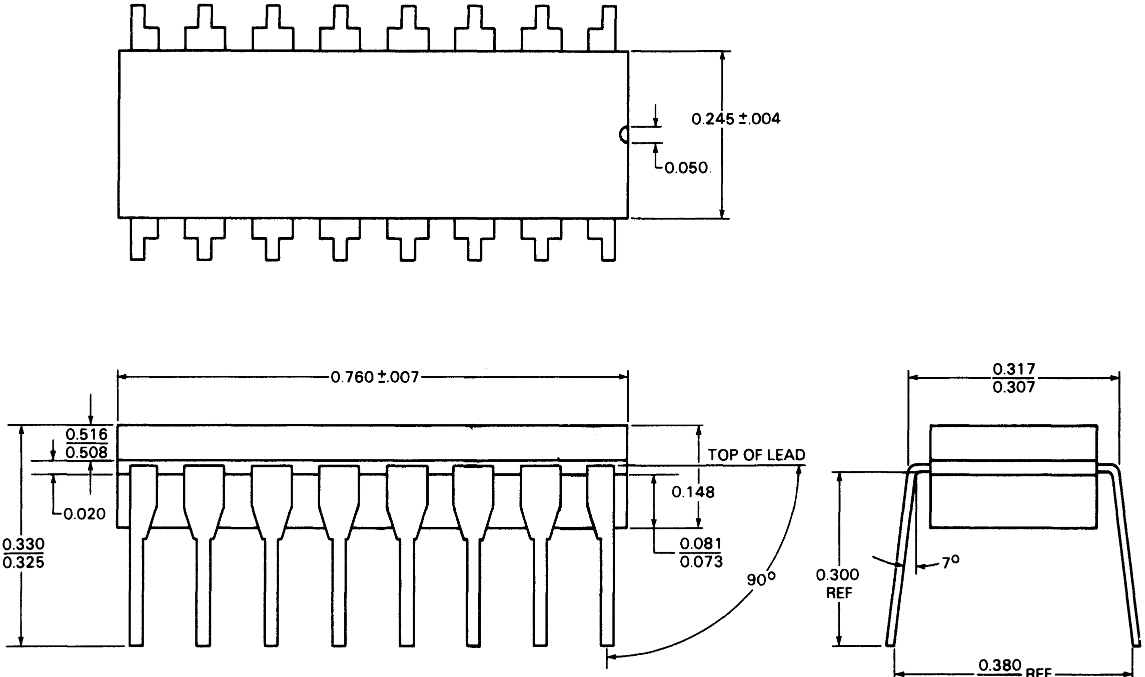
NOTE: C_L includes jig and wiring capacitance.

DG-04219

PACKAGE SPECIFICATIONS

The physical dimensions of the mN640 chip are given below.

DIMENSIONS OF CHIP



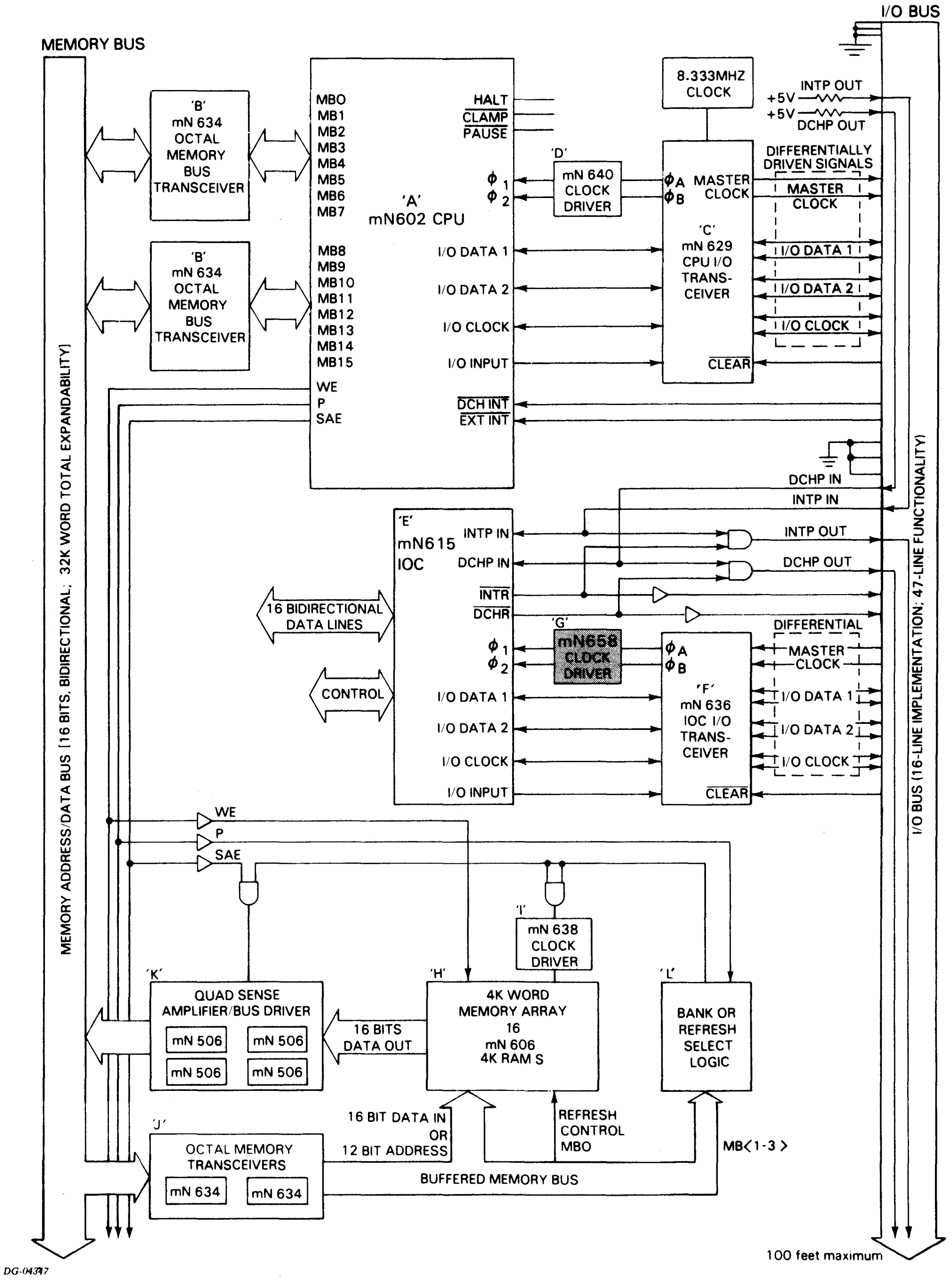
DG-04214

12

mN658
CLOCK DRIVER

13

mN658
CLOCK DRIVER



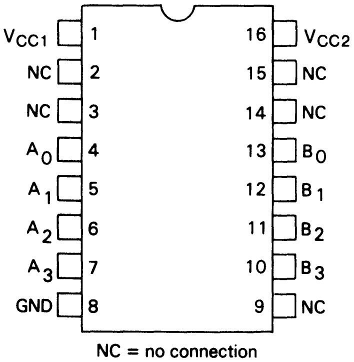
Data General
Data General Corporation, Westboro, Massachusetts 01581

mN658
IOC CLOCK DRIVER

FEATURES

- DUAL 3 INPUT BIPOLAR TO MOS DRIVER IN A 16 PIN CERDIP PACKAGE
- TTL COMPATIBLE INPUTS HIGH VOLTAGE OUTPUTS COMPATIBLE mN615 MOS CLOCK INPUTS.

PACKAGE



DG-08155

GENERAL DESCRIPTION

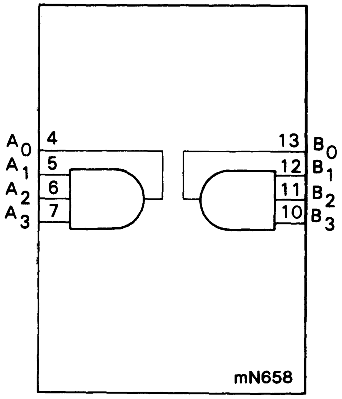
The mN658 Clock Driver chip contains two TTL level to high voltage converter gates. These are used to drive the two-phase clock inputs of the mN615. Additional signal shaping circuitry, required to shape the non-overlapping clock signals, is described in the Design Notes.

13

PIN DESCRIPTIONS

The pin connections and there functions are described in the diagram and table below.

FUNCTIONAL PIN CONNECTION DIAGRAM



DG-08156

PIN FUNCTIONS

MNEMONIC	PIN NO.	IN/OUT	FUNCTION
A1,A2,A3	5,6,7	IN	TTL inputs to 3-input AND gate.
A0	4	OUT	High voltage output of 3-input (A1, A2, A3) AND gate.
B1,B2,B3	12,11,10	IN	TTL inputs to 3-input AND gate.
B0	13	OUT	High voltage output of 3-input (B1, B2, B3) AND gate.
V_{CC1}	1		+12V ± 1V or +15 ± 1V
V_{CC2}	16		+5V ± 0.25V
GND	8		Ground

ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS

Supply voltage, V_{CC1}	17V
Supply voltage, V_{CC2}	7V
Input voltage	5.5V
Output voltage	60 mA
Storage temperature range	-55 deg to 125 deg C

SPECIFIED OPERATING RANGES

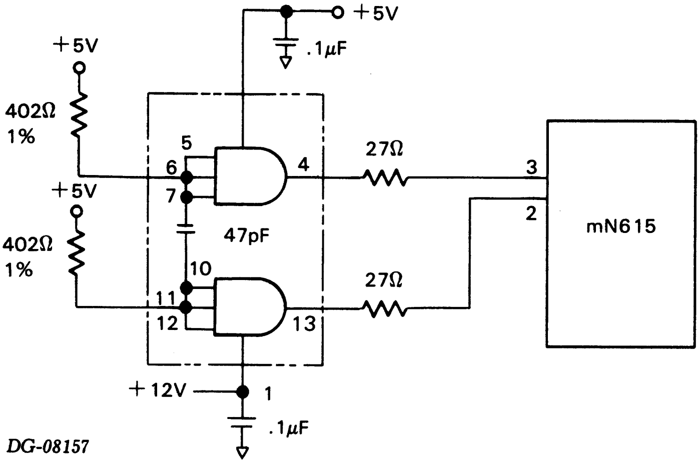
Supply voltage, V_{CC1}	+11 to +16V
Supply voltage, V_{CC2}	+.5 ± 0.25V
Average power dissipation	1.0 W
Operating free-air temperature range	0 deg to 70 deg C

DC CHARACTERISTICS

SYMBOL	CHARACTERISTICS	CONDITIONS	MIN	MAX	UNITS
V_{IH}	A_{1-3} B_{1-3}		2.0		V
V_{IL}	A_{1-3} B_{1-3}			0.8	V
I_{IH}	A_{1-3} B_{1-3}	V_{CC1} = 16V V_{CC2} = 5.25V V_I = 2.4V		40	μA
		V_{CC1} = 16V V_{CC2} = 5.25V V_I = 5.5V		1	mA
I_{IL}	A_{1-3} B_{1-3}	V_{CC1} = 16V V_{CC2} = 5.25V V_I = 0.4V		-1.6	mA
V_{OH}	A_O B_O	V_{CC1} = 11V V_{CC2} = 4.75V I_O = -100μA	4.4		V
V_{OL}	A_O B_O	V_{CC1} = 11V V_{CC2} = 4.75V I_O = 10μA		0.25	V
I_{CC1}		V_{CC1} = 16V V_{CC2} = 5.25V (INPUTS LOW)		15	mA
I_{CC2}		V_{CC1} = 16V V_{CC2} = 5.25V (INPUTS HIGH)		10	mA
		V_{CC1} = 16V V_{CC2} = 5.25V (INPUTS LOW)		100	mA

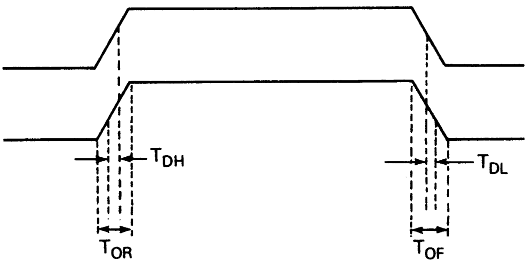
13

RECOMMENDED CONFIGURATION



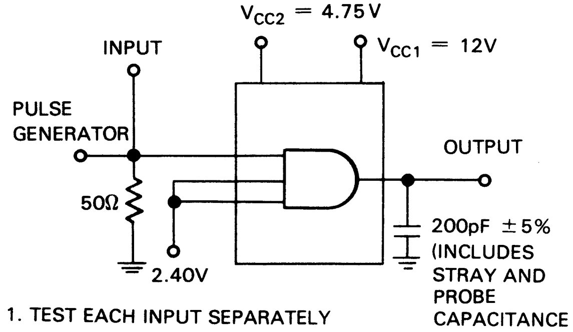
DG-08157

AC CHARACTERISTICS/SWITCHING DIAGRAM



DG-08158

TEST CONDITIONS



1. TEST EACH INPUT SEPARATELY
2. INPUT PULSE
   AMPLITUDE = 0 TO 3V
   WIDTH = 100NS (50% TO 50%)
   $T_R = T_F = 10$ N.S
   FREQ = 5MHz

TRANSITION TIMING

SYMBOL	CHARACTERISTICS	CONDITIONS	MIN	MAX	UNITS
T_{DH}	RISING OUTPUT DELAY	SEE ABOVE	15	35	ns
T_{DL}	FALLING OUTPUT DELAY	SEE ABOVE	15	35	ns
T_{OR}	OUTPUT RISE TIME	SEE ABOVE	8	25	ns
T_{OF}	OUTPUT FALL TIME	SEE ABOVE	8	25	ns

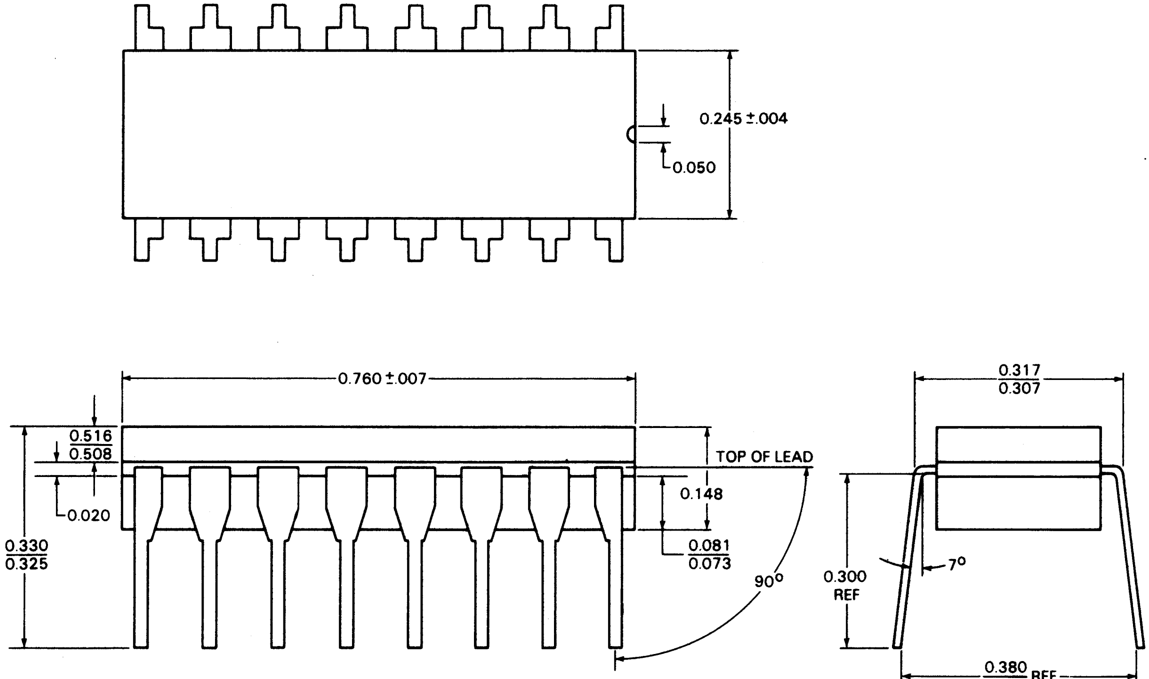
NOTE: T_{DH} and T_{DL} measured between 50% points.

T_{OR} and T_{OF} measured between 10% and 90% points.

PACKAGE SPECIFICATIONS

The physical dimensions of the mN640 chip are given below.

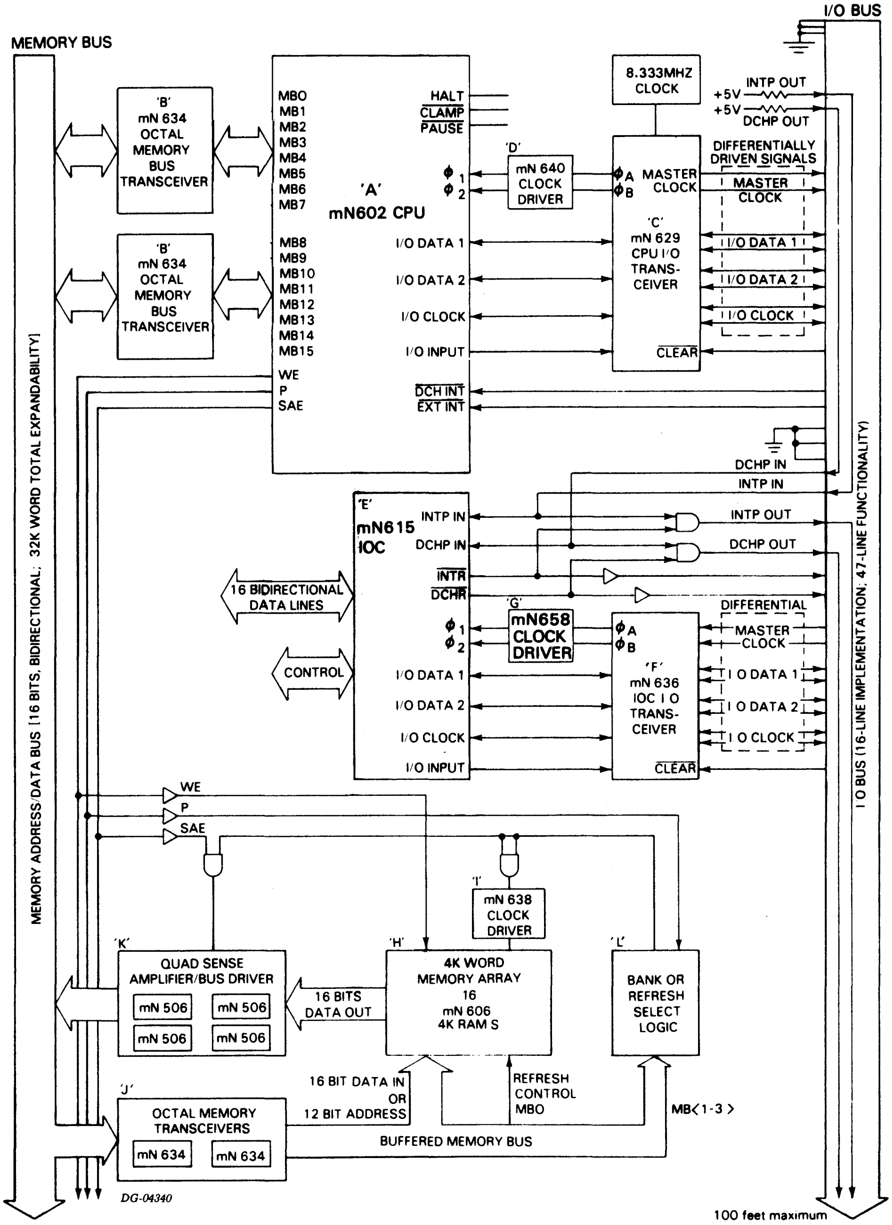
DIMENSIONS OF CHIP



DG-04214

DESIGN NOTES

DESIGN NOTES
SYSTEM BLOCK DIAGRAM



INTRODUCTION TO DESIGN NOTES

This portion of the data manual contains engineering notes which are aids to the design engineer using the microNOVA chip set. The notes are compiled in sections, each devoted to a particular facet of design implementation. Each circuit shown has been thoroughly tested, and the text accompanying each schematic suggests proven methods for circuit layout. The six design notes are:

- Typical CPU module design
- General purpose I/O interface design
- mN606 RAM array design
- Using the memory bus to address I/O devices
- Generating the non-overlapping clocks required by the mN601 and mN603
- 12V to 15V power supply conversion

The design notes covering each of the above subjects describe conservative engineering practices suitable for a wide range of applications. The logic schematics show suggested circuitry, values of discrete components and, where applicable, manufacturer's part numbers for integrated circuits not included in Data General's microcomputer chip set.

SIMPLIFIED CPU MODULE

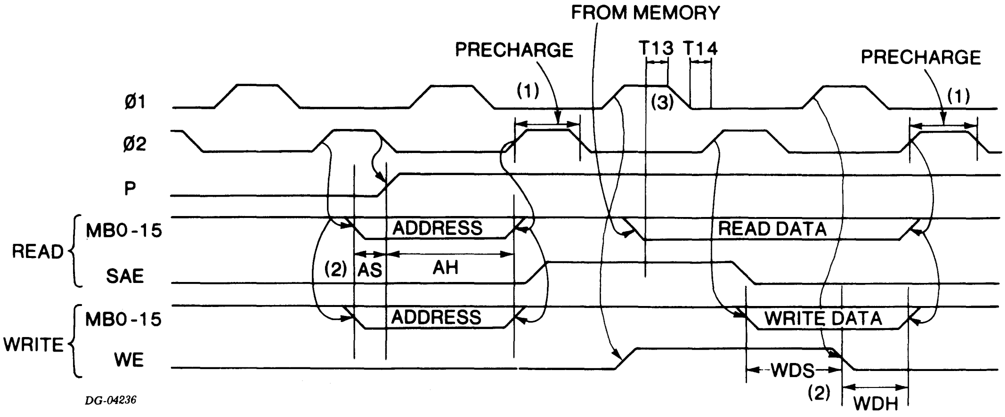
In a typical microNOVA CPU configuration, the mN601 or mN602 microprocessor is interfaced to both the microNOVA I/O bus and memory bus as shown in the CPU logic schematic on the following page. When designing a CPU module, there are several practical considerations that should be taken into account; these are discussed below.

MEMORY BUS

The memory bus, MB <0-15> , is precharged internally by the CPU during every other ø2 clock pulse, regardless of the operation being performed. During precharging, care should be exercised to ensure that memory bus lines are not driven low, thereby causing excess current on the lines.

After the memory bus lines are precharged, data is placed on the bus by discharging those lines which carry 0's and leaving charged those lines which carry 1's. As shown in the diagram below, the memory bus is precharged once during either a read or write cycle (and at least twice during a read-modify-write cycle - see mN601 circuit description, Memory Operations and mN602 circuit description, Memory Operations). Thus, discharging the memory bus before information is either read by the CPU during a read operation or read by memory during a write operation can result in data loss. Similarly, discharging the memory bus before the memory address is read at address time (when the memory control signal P is asserted) can result in incorrect memory addressing. The interline capacitance in the bus should be limited to a 10pf to eliminate crosstalk.

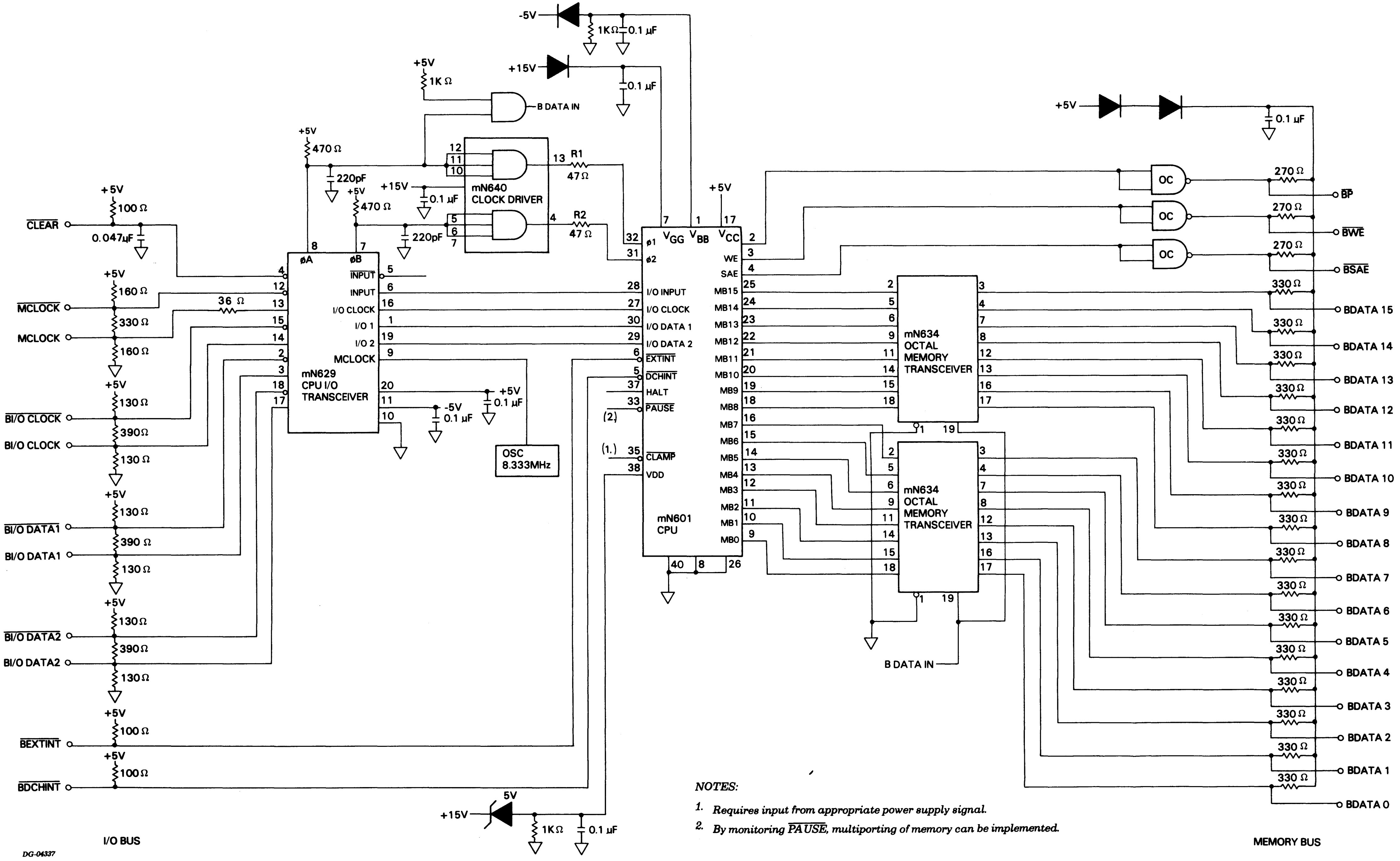
MEMORY BUS PRECHARGING



NOTES:

1. *During precharge time, MB lines should not be driven low because high current spikes may cause damage to the mN601.*
2. *During memory address or data set-up time, the MB lines should not be driven low, since incorrect addressing or data loss will result. See mN601, Memory Timing Table.*
3. *In order to read information from the memory, MB lines carrying 0's must be discharged by T13, before the falling edge of ø1 (see mN601, Switching Diagrams). Once discharged, the MB lines remain discharged until the next precharge.*

TYPICAL microNOVA CPU MODULE



NOTES:

1. *Requires input from appropriate power supply signal.*
2. *By monitoring PAUSE, multiporting of memory can be implemented.*

mN602/mN602 SYSTEM CLOCK

The non-overlapping clocks required to generate the microprocessor's internal four-phase clock are described in th Non-overlapping Clocks note. This circuitry is also shown in the CPU logic schematic.

CLOCK OVERSHOOT

Care should be exercised in minimizing inductance on the clock lines. An overshoot on the clock waveform caused b inductance in the connections between the mN640 clock driver and the mN601 will not create a problem providing its amplitud is limited to 1V or less. This overshoot is dependent upon the clock driver rise and fall times. The value of the overshoot can b controlled by changing the values of the resistors, R1 and R2 (see schematic), in series with the outputs of the mN640 cloc driver.

mN601 GROUND PADS

Noise voltage developed between the three grounds pads (VSS, pins 8, 26, and 40) as a result of current in the etch will result i reduced CPU operating margins. For this reason, it is recommended that these pads be interconnected with short and heavy high conductivity connections.

mN601/mN602 DECOUPLING

It is recommended that decoupling of the mN601's or mN602's supply voltage connections (V_{BB}, V_{CC}, V_{DD} and V_{GG}) b made as close as possible to the mN601 or mN602 chip and the nearest ground connected to it.

Suggested values for decoupling capacitors are:

Supply Connection	Capacitor Value
V_{BB}	0.1µf EIA # Z5V dielectric, or equivalent
V_{CC}	0.1µf EIA # Z5V dielectric, or equivalent
V_{DD}	0.1µf EIA # Z5V dielectric, or equivalent
V_{GG}	0.1µf EIA # Z5V dielectric, or equivalent

Overall board decoupling at low frequency should be incorporated according to standard engineering practices.

GENERAL PURPOSE I/O INTERFACE

The logic schematic on the following page illustrates a typical general purpose microNOVA I/O interface. In the drawing, the mN615 I/O controller is interfaced to the I/O bus via an mN658 IOC I/O transceiver and to the device via the 16-bit data bus (D<0-15> H or D<0-15> L) and a 4-to-16 line decoder.

mN615 SYSTEM CLOCK

The non-overlapping clocks required to generate the mN603, mN613, and mN615 internal four-phase clock are described in the Non-overlapping Clocks note. The circuitry for the mN615 is also shown in the microNOVA I/O interface schematic.

CLOCK OVERSHOOT

Care should be exercised in minimizing inductance on the clock lines. An overshoot on the clock waveform caused by inductance in the connections between the mN640 clock driver and the mN603 will not create a problem providing its amplitude is limited to 1V or less. This overshoot is dependent upon the clock driver rise and fall times. The value of the overshoot can be controlled by changing the values of the resistors, R1 and R2 (see schematic), in series with the outputs of the mN640 clock driver.

mN603/mN613 DECOUPLING

It is recommended that decoupling of the mN603's and mN613 supply voltage connections (V_{BB}, V_{CC}, V_{DD} and V_{GG}) be made as close as possible to the mN603 chip and the nearest ground connection.

Suggested values for the decoupling capacitors are:

Supply Connection	Capacitor Value
V_{BB}	0.1µf EIA # Z5V dielectric, or equivalent
V_{CC}	0.1µf EIA # Z5V dielectric, or equivalent
V_{DD}	0.1µf EIA # Z5V dielectric, or equivalent
V_{GG}	0.1µf EIA# Z5V dielectric, or equivalent

mN615 DECOUPLING

It is recommended that decoupling of the mN615's supply voltage connections (V_{CC} and V_{BB}) be made as close as possible to the mN615 chip and the nearest ground connection.

Suggested values for the decoupling capacitors are

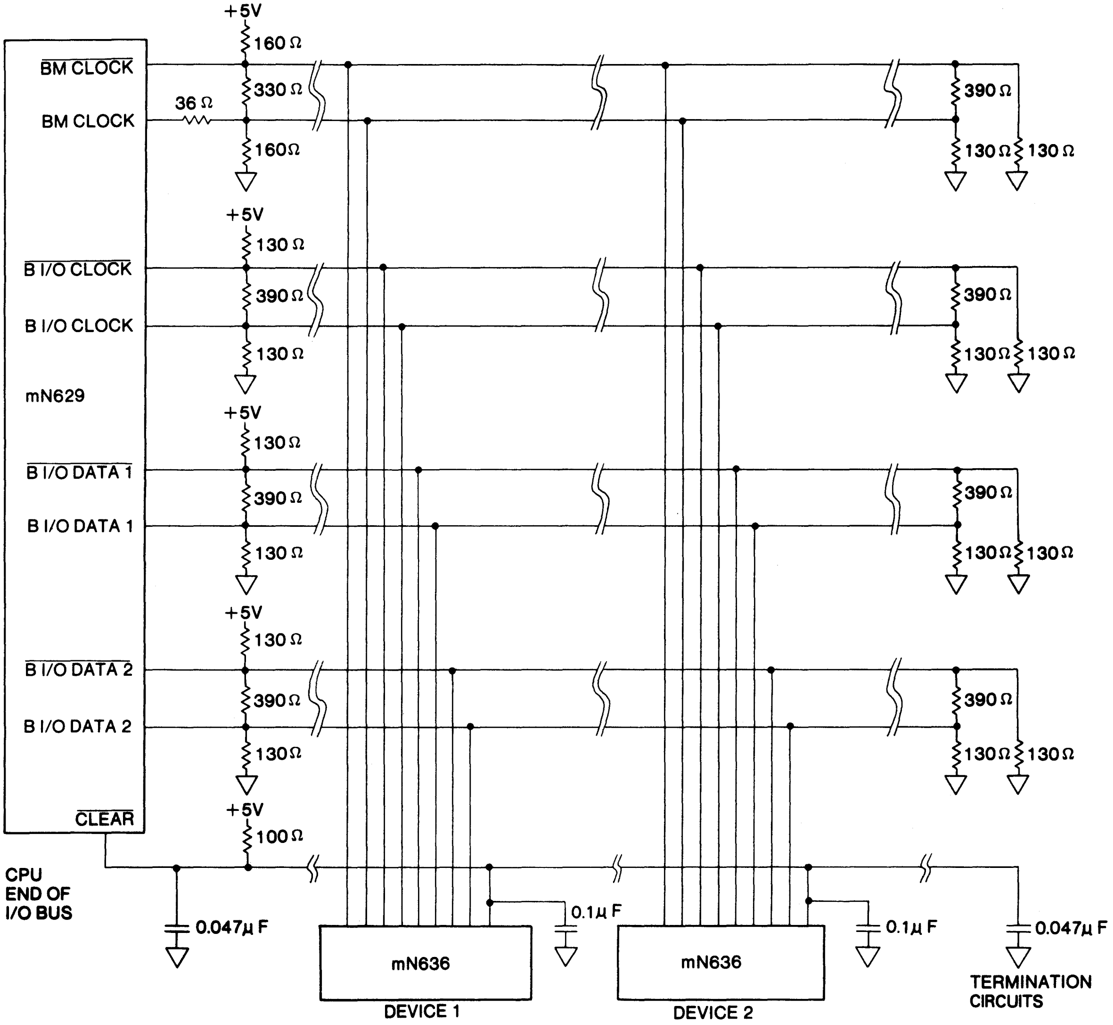
Supply Connection	Capacitor Value
V_{BB}	0.1µf EIA # Z5V dielectric, or equivalent
V_{CC}	0.1µf EIA # Z5V dielectric, or equivalent

Overall board decoupling at low frequency should be incorporated according to standard engineering practices.

I/O BUS TERMINATION

A suggested method of terminating the differentially driven I/O bus signals is shown on the following page. Non-differential I/O bus signals do not require termination; however, the $\overline{\text{CLEAR}}$ line should contain a 0.1µf capacitor into each device and a 0.047µf capacitor in the termination circuits.

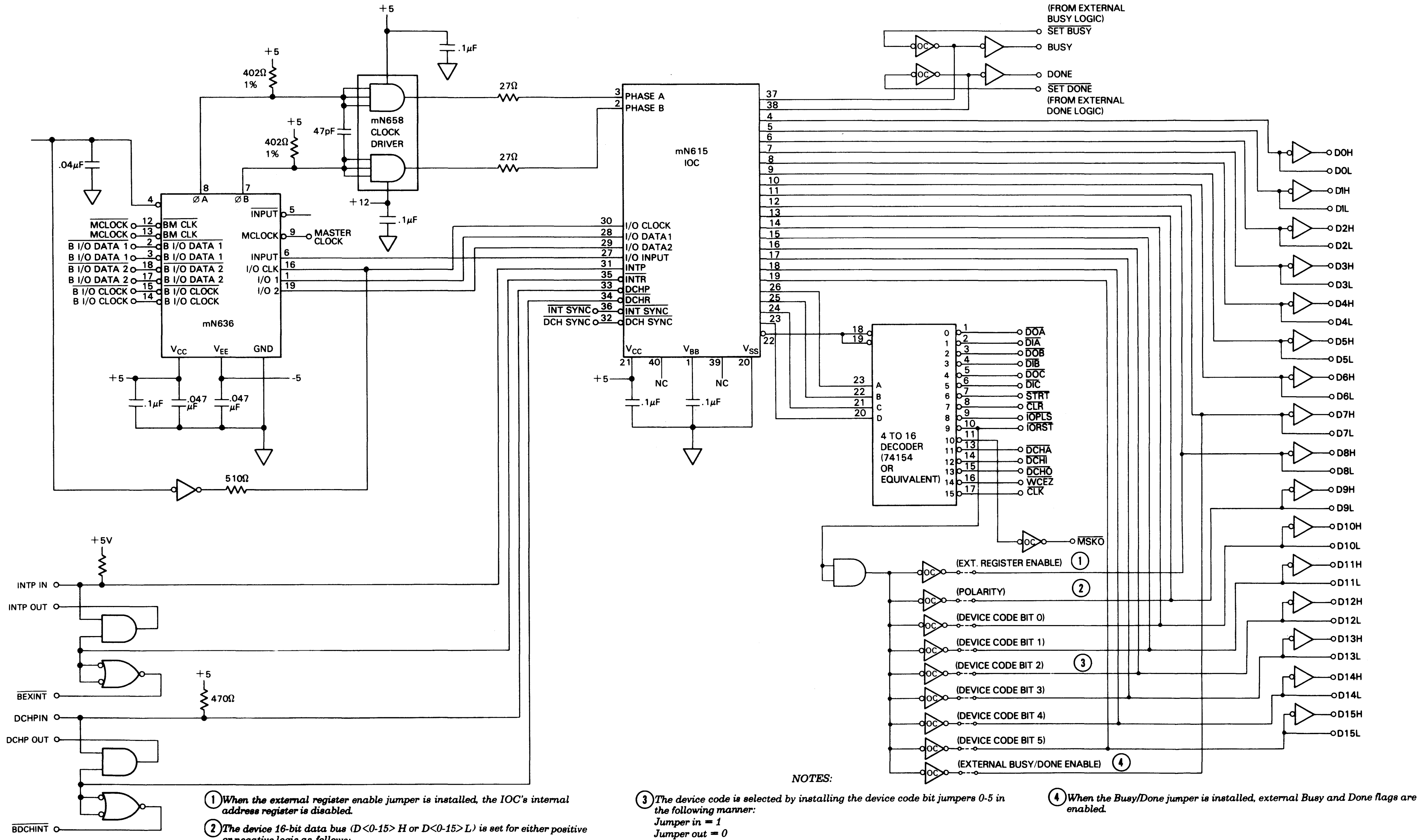
I/O BUS TERMINATION



NOTE:
The values of the resistors shown in the diagram are applicable when Scotchflex cable, part No. 3365/16, or equivalent, is used. With other cables, the values of the resistors may vary; but, the current driving capabilities of the transceivers must not be exceeded.

DG-04274

GENERAL PURPOSE INTERFACE



DG-08643

NOTES:

(1) *When the external register enable jumper is installed, the IOC's internal address register is disabled.*

(2) *The device 16-bit data bus (D<0-15> H or D<0-15> L) is set for either positive or negative logic as follows:*

Polarity Bit Jumper	*Out*	*In*
D<0-15>H	*H = 1, L = 0*	*H = 0, L = 1*
D<0-15>L	*H = 0, L = 1*	*H = 1, L = 0*

(3) *The device code is selected by installing the device code bit jumpers 0-5 in the following manner:*
Jumper in = 1
Jumper out = 0
Example of jumpering for device code 15_8:

Device code bit jumpers	*0*	*1*	*2*	*3*	*4*	*5*
Jumper in/out	*out*	*out*	*in*	*in*	*out*	*in*

$\underbrace{0\ 0\ 1}_{1}\ \underbrace{1\ 0\ 1}_{5} = 15_8$

(4) *When the Busy/Done jumper is installed, external Busy and Done flags are enabled.*

mN606 RAM ARRAY DESIGN

The following notes discuss basic concepts which should be considered when designing a printed circuit board suitable for mN606 MOS RAMs.

BOARD CONSTRUCTION/VOLTAGE DISTRIBUTION

Two-layer printed circuit board construction provides an effective method of distributing voltages and ground on a high-speed memory card such as a dynamic RAM board. On a two-layer board, the normal method is to grid or mesh ground and power distribution lines. While these grids may be irregular, each grid should be a rectangle, approximately 1 by 2 inches. The grid distribution is constructed by connecting the horizontal voltage or ground traces on one side of the board with their vertical counterparts on the opposite side of the board (see the printed circuit board diagram). This type of construction reduces signal return distances and equalizes current flow along the buses. Equalizing current flow reduces supply voltage drops, thereby reducing voltage and ground variations between memory devices.

mN606 RAM DECOUPLING

It is necessary to decouple the mN606 RAM supply connections to prevent power spikes which will force the RAM to operate out of specification. MOS memories exhibit small standby currents but, when accessed, draw relatively large, short duration transient currents from the power supply. These high transient currents can be reduced by placing decoupling capacitors a short distance from the memory device. The decoupling capacitors average the current pulse over the interval of 'on time' to 'off time', essentially decreasing the frequency of the current components presented to the main power distribution lines and supplies.

Two types of capacitors are normally used for decoupling each voltage. The first type provides the high frequency pulse currents required by the RAMs. The second type provides storage for the first type and further reduces the magnitude of the low frequency currents that are presented to the power supply.

The first type (designated as C1 and C2 on the diagram) is usually ceramic and is located close to the mN606. These capacitors must have sufficient capacity and a frequency response that is high enough to supply the pulse currents to the RAMs without discharging more than an acceptable amount. The capacitance is determined by both the voltage drop that can be tolerated at the mN606 and the peak transient current. The leading and trailing edge current spikes contain high frequency components which dictate the frequency response characteristics of the capacitor. These high frequency response capacitors should be located adjacent to each mN606; or, if the array is designed with the RAMs close together, they can be located near every other chip.

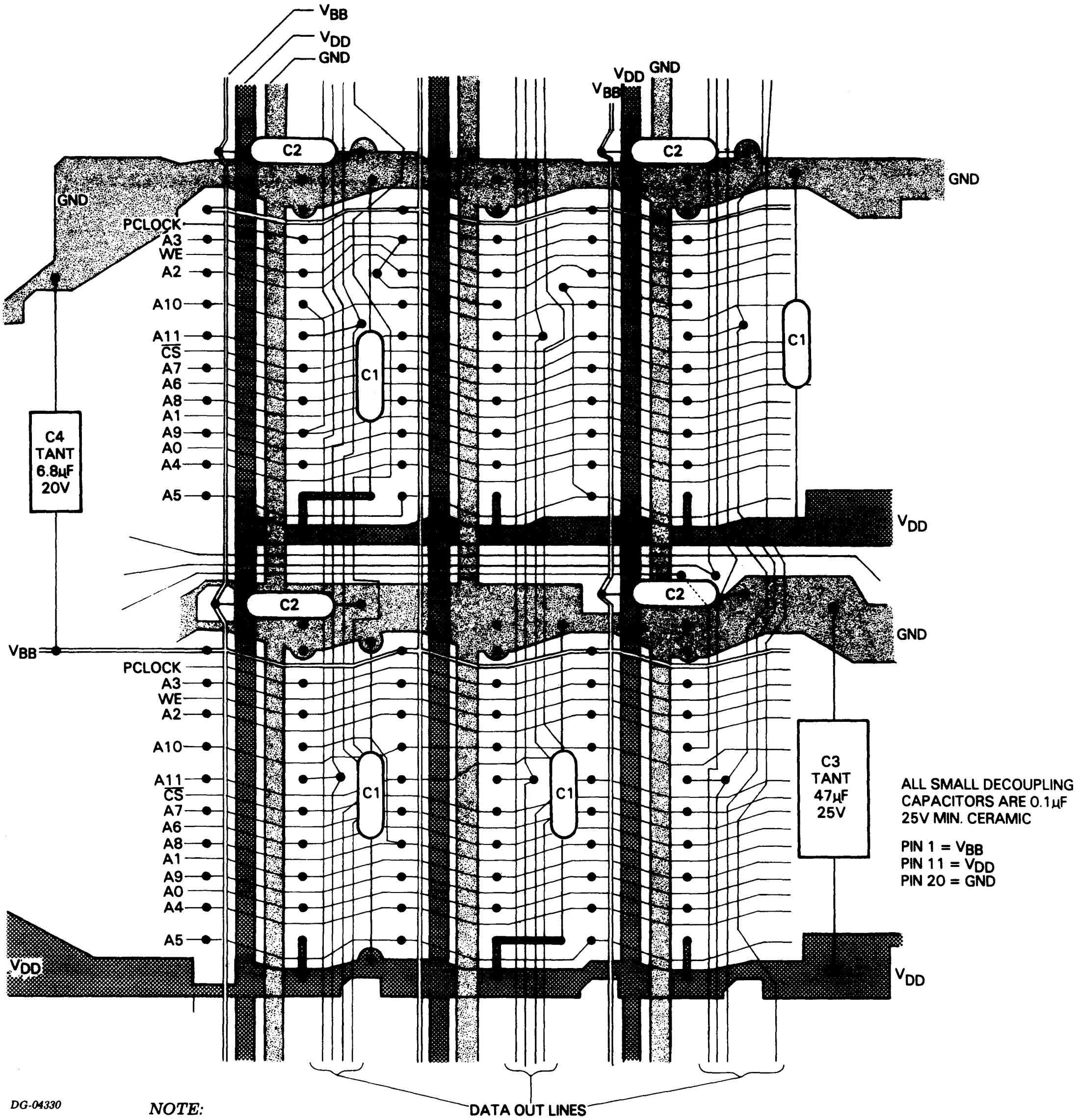
For a 64 memory device array (16K of memory), a typical value for V_{DD} decoupling (C1) is 0.05µf per 4K RAM, or approximately thirty-two 0.1µf capacitors distributed between alternate chips. Similarly, the typical value for V_{BB} decoupling (C2) is 0.05µf per mN606, or approximately thirty-two 0.1µf capacitors distributed between alternate chips.

The second type (designated as C3 and C4 on the diagram) is usually a tubular tantalum capacitor which is used to integrate the cycle current requirements. The value of capacitors C3 and C4 is related to the difference between the cycle current and the standby current. For a 64 memory device array, the typical total value for C3 (V_{DD} decoupling) is 100µf divided between two or three capacitors located near the outer or connector edge of the array. Similarly, the typical value for C4 (V_{BB} decoupling) is 5µf.

mN638 CLOCK DRIVER DECOUPLING

Decoupling the V_{CC2} (+15V) connection of each mN638 memory clock driver is recommended because of the relatively high driver peak currents encountered. V_{CC2} should be decoupled by both a 0.1µf and a 6.8µf capacitor. The 0.1µf capacitor should be located as close to each mN638 chip as possible while one 6.8µf capacitor can serve up to four mN638 chips.

Both the V_{CC3} and V_{CC1} connections of the clock driver should also be decoupled. In this case, a 0.1µf capacitor should be used to decouple each connection and the two capacitors should be located as close to the chip as possible.



DG-04330

NOTE:

In a 4K memory array, the same bus is used for both address and data inputs. Thus, in most cases, the data input line of each chip is tied to one of the address lines, and one address line can serve only one chip; i.e., one chip per data bit. Since the mN606 has only 12 address lines, some chips receive their data input from a memory bus line that is not otherwise connected to the chip (see diagram). The data outputs of each chip are independent. They are fed as inputs to sense amp/bus drivers (see 4K memory schematic).

In an 8K memory array, the data inputs to the same bits can be connected in parallel. Similarly, the data outputs of the same bits can also be connected in parallel.

mN506 SENSE AMP/BUS DRIVER DECOUPLING

It is recommended that the V_{CC1}, V_{CC2} and V_{EE} connections of each mN506 chip be decoupled using a 0.1µf capacitor for each connection. These capacitors should be located as close to the mN506 chip as possible. One capacitor can serve the V_{CC1} connection of two chips, one capacitor can serve the V_{CC2} connection of two chips, and one capacitor can serve the V_{EE} connection of four chips.

MEMORY BUS RECEIVERS

mN634 memory bus drivers can be used successfully to drive the mN606 memory array. Since the mN634 is an open-collector output device, it requires pull-up resistors to supply the high input requirement (+4V) of the RAM. The value of the pull-up resistors is computed from the timing requirements of the CPU. The memory bus is precharged by the CPU prior to the time the input lines are sampled by the RAMs. Thus, there is a period of approximately 240ns during which the input lines can be brought to a high of 4V. In the schematic of a typical 4K memory shown on the following page, the pull-up resistors have a value of 510 ohms.

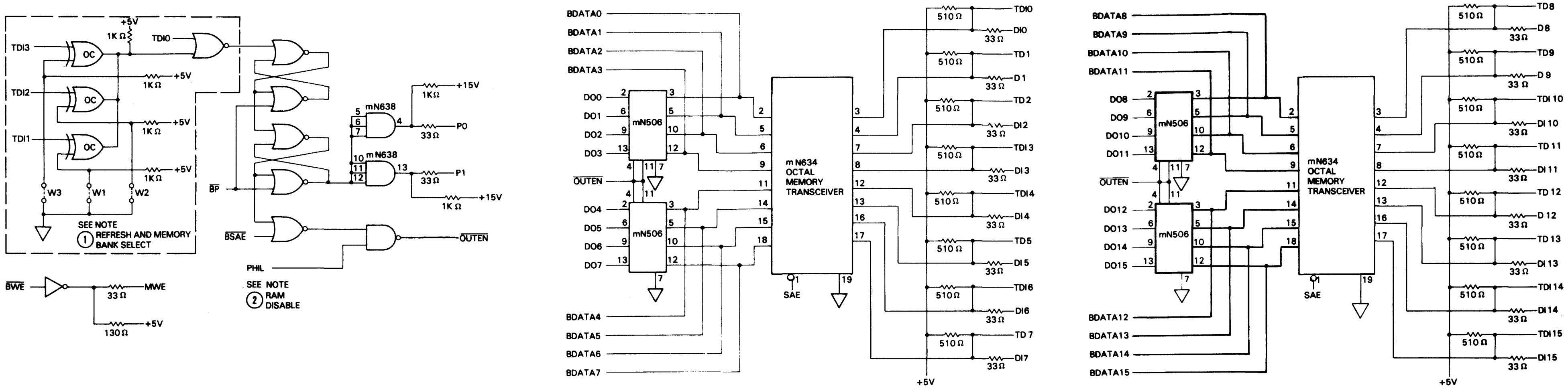
Because of the inductance of the RAM array, there may be an inductive undershoot in the input lines when they are pulled low by the mN634. This can drive the low input below the recommended specifications of the mN606 RAM. To reduce the undershoot pulse, a resistor in series with the input line can be used. While the value of these resistors (one per input line) is layout dependent, the computed value of the series resistors in the typical 4K memory schematic is 33 ohms.

Devices other than mN634's can be used to drive the input lines of the RAM array. When TTL drivers with totem pole outputs are used, a pull-up resistor, as described above, is required to ensure that the input threshold requirement of the RAM (+4V) is satisfied. Schottky gates with totem pole outputs are not recommended.

PROM/RAM

When both PROM and RAM elements are included in a microNOVA computer system, the PROM is given priority over the RAM. This is accomplished using the PHIL signal, as shown in the logic schematic on the following page.

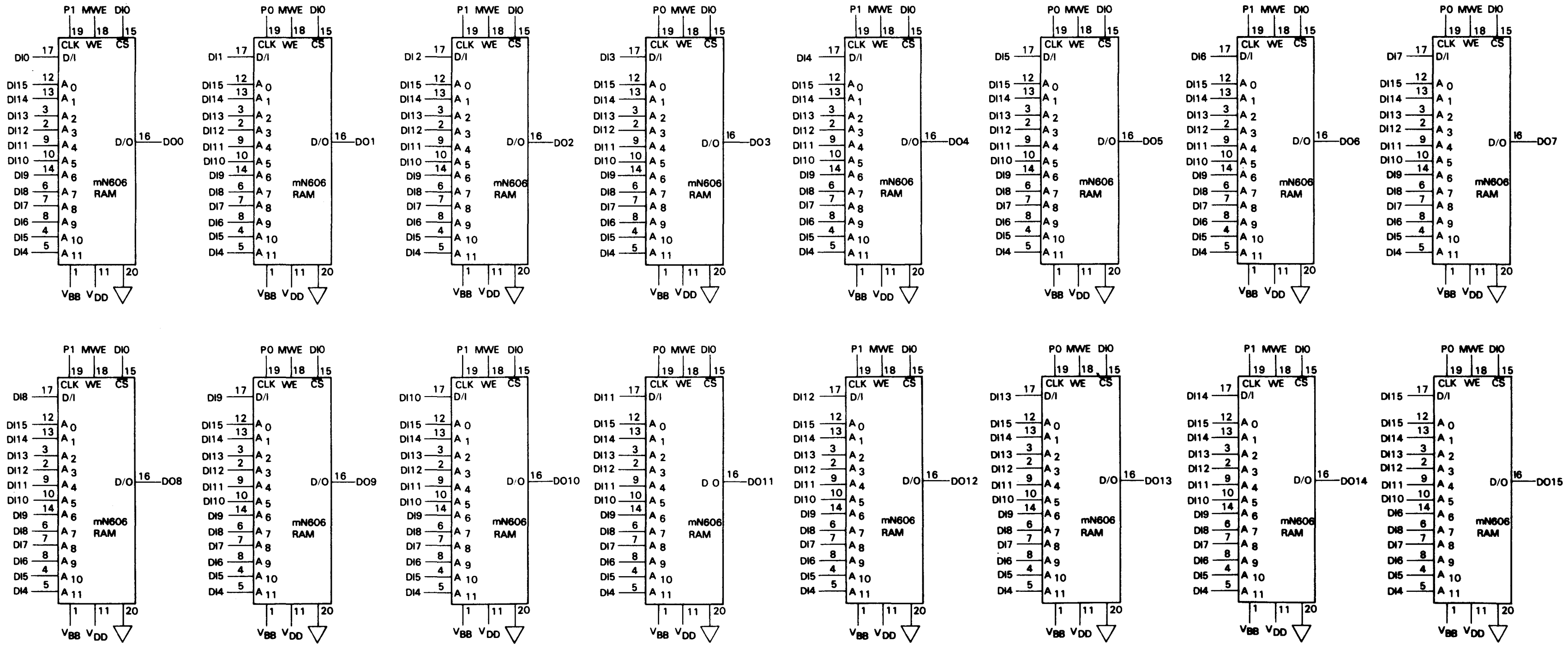
4K RAM MODULE



NOTES:

1. *The bank select logic consists of three open-collector EXCLUSIVE-OR gates, a pull-up resistor to +5Vdc, and three jumpers, designated W1, W2, and W3. The 4Kx16-bit RAM memory array is selected when both the value of the address lines (TDI1-TD13) matches the value selected by the jumper settings and TDI0 is low. When this occurs, the output of the EXCLUSIVE-OR gates goes to a high logic level, and the RAM array is enabled for either read, write, or read-modify-write operations when the memory control signal $\overline{BP}$ is asserted. When a refresh cycle is initiated, TDI0 is high, thereby selecting the memory, indpendent of TDI1-TDI3, when $\overline{BP}$ is asserted. Also, DI0 (input to $\overline{CS}$ pin of RAMs) goes high so that a refresh operation will be performed by the mN606's.*

2. *When a PROM element is addressed, the PHIL signal is pulled low and $\overline{OUTEN}$ cannot go low. When this occurs the RAM outputs are disconnected from the memory bus.*



DG-04227

ADDRESSING I/O DEVICES VIA THE MEMORY BUS

In microNOVA systems it is possible to make an active register behave as a memory register, thus allowing I/O interfaces to be accessed via the microNOVA memory bus.

A suggested method of implementing this technique is shown in the following logic schematic. It is assumed in the drawing that the memory bus is driven as depicted in the CPU logic schematic contained in the CPU Notes.

ADDRESSING

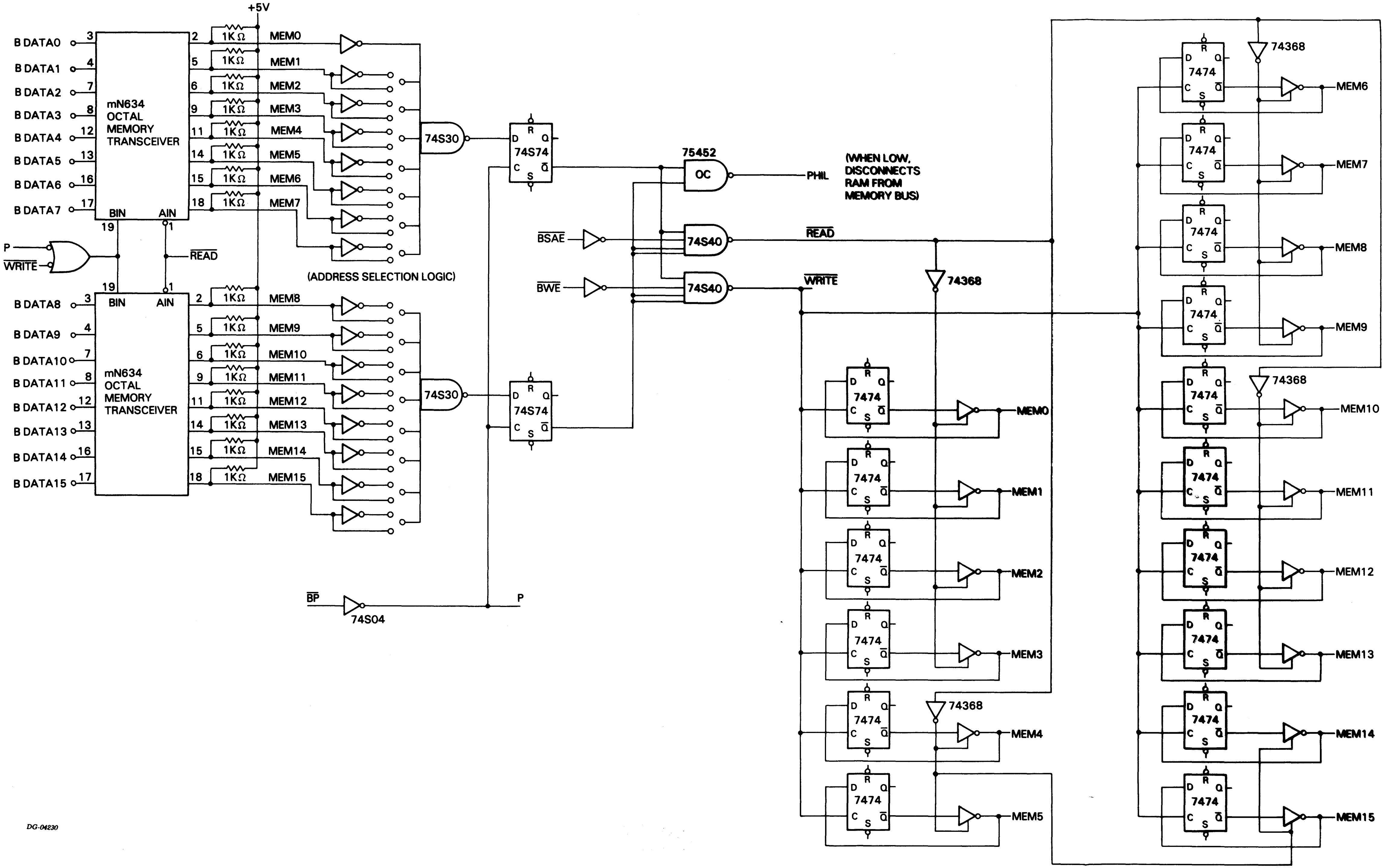
The memory register can be addressed by using jumpers on the memory input lines (MEM <0-15> , see schematic). When all the input lines to the address select NAND gates are at a high logic level, the memory register is enabled for either reading or writing.

MEMORY REGISTER

In the drawing, the register is comprised of sixteen edge-triggered flip-flops with independent clear and reset lines. The outputs of the register are connected to the memory output lines (MEM <0-15>) via tri-state gates that are enabled at memory read time.

RAM DISABLE

The PHIL signal (see schematic) is used to disable any RAMs in the system which respond to the same address as that selected for the memory register. When this condition does not exist, the PHIL circuitry has no function.


+5V
B DATA0
B DATA1
B DATA2
B DATA3
B DATA4
B DATA5
B DATA6
B DATA7
mN634
OCTAL
MEMORY
TRANSCEIVER
BIN
AIN
1KΩ
MEM0
MEM1
MEM2
MEM3
MEM4
MEM5
MEM6
MEM7
P
WRITE
READ
(ADDRESS SELECTION LOGIC)
B DATA8
B DATA9
B DATA10
B DATA11
B DATA12
B DATA13
B DATA14
B DATA15
MEM8
MEM9
MEM10
MEM11
MEM12
MEM13
MEM14
MEM15
74S30
74S74
D
R
Q
C
S
Q̄
75452
OC
PHIL
(WHEN LOW, DISCONNECTS RAM FROM MEMORY BUS)
BSAE
BWE
74S40
READ
WRITE
74368
7474
BP
74S04
P

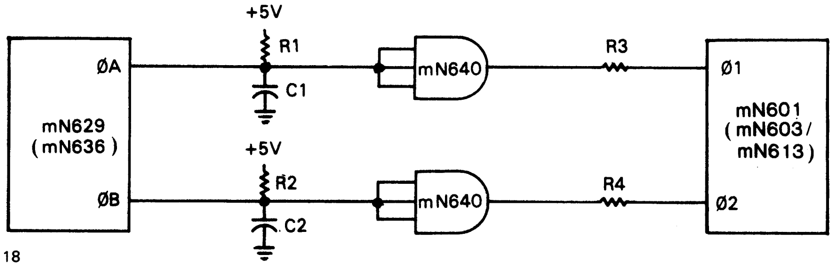
DG-04230

mN601 and mN603/mN613 NON-OVERLAPPING CLOCKS

The mN601 microprocessor and the mN603/mN613 I/O controllers require two non-overlapping clock signals to generate their internal four-phase system clock. The logic schematic below illustrates a suggested method of producing these MOS clock inputs.

As shown, external circuitry is used to shape the clock signals generated by the appropriate transceivers before they are supplied to the mN601 or mN603/mN613 via the mN640 clock driver (shown as two AND gates).

EXTERNAL CLOCK SHAPING
(mN601 and mN603/mN615)



DG-08618

NOTES:

1. The recommended values for the discrete components shown in the diagram are

Reference Designators	*Values*
R1, R2	*470Ω, 5%*
C1, C2	*220 pf, 5%. This value may vary, depending on circuit layout.*
R3,, R4	*Typical value for the mN601 is 47Ω and for the mN603 or mN613, 100Ω. These values may vary when required to reduce overshoot in the clock signals.*

2. Ground connections between C1, C2 and the mN640 clock driver should be as short and heavy as possible.

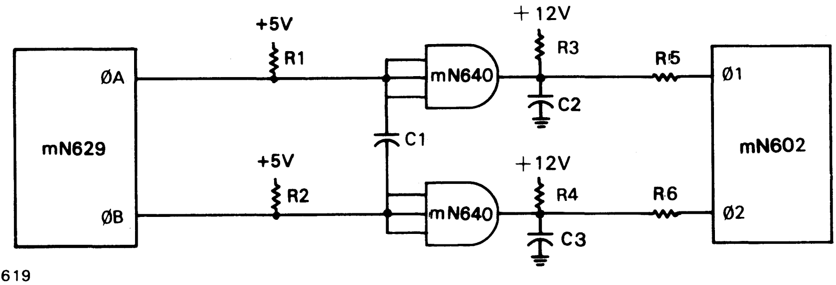
3. It is recommended that decoupling of the mN640 clock driver supply voltage connections (V_{CC1} and V_{CC2}) be made as close to the mN640 as possible.

mN602 NON-OVERLAPPING CLOCKS

The mN602 microprocessor requires two non-overlapping clock signals to generate its internal four-phase system clock. The logic schematic below illustrates a suggested method of producing these MOS clock inputs.

As shown, external circuitry is used to shape the clock signals generated by the mN629 transceiver before they are supplied to the mN602 via the mN640 clock driver (shown as two AND gates).

EXTERNAL CLOCK SHAPING
mN602



dg-08619

NOTES:

1. The recommended values of the discrete components shown in the diagram are

Reference Designators	*Values*
R1, R2	*402Ω, 1%*
C1	*47 pf, 5%*
R3, R4	*1Kpi(9), 5%*
C2, C3	*47 pf, 5%. This value may vary, depending on circuit layout.*
R5, R6	*Typical value is 0Ω. This value may vary when required to reduce overshoot in the dock signals. An overshoot of 1V or less is acceptable.*

2. Ground connections between C2, C3 and the mN640 clock driver should be as short and heavy as possible.

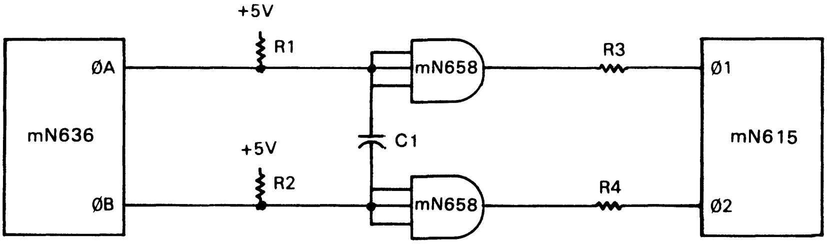
3. It is reommended that decoupling of the mN640 clock driver supply voltage connections (V_{CC1} and V_{CC2}) be made as close to the mN640 as possible.

mN615 NON-OVERLAPPING CLOCKS

The mN615 I/O controller requires two non-overlapping clock signals to generate their internal four-phase system clock. The logic schematic below illustrates a suggested method of producing these MOS clock inputs.

As shown, external circuitry is used to shape the clock signals generated by the mN658 transceiver before they are supplied to the mN615 via the mN658 clock driver (shown as two AND gates).

EXTERNAL CLOCK SHAPING
mN615



dg-8620

NOTES:

1. The recommended values of the discrete components shown in the diagram are

Reference Designators	*Values*
R1, R2	*402Ω, 1%*
C1	*47 pf, 5%*
R3, R4	*27Ω This value may vary when required to reduce overshoot*

2. It is recommended that decoupling of the mN658 clock driver supply voltage connectors (V_{CC1} and V_{CC2}) be made as close to the mN658 chip as possible.

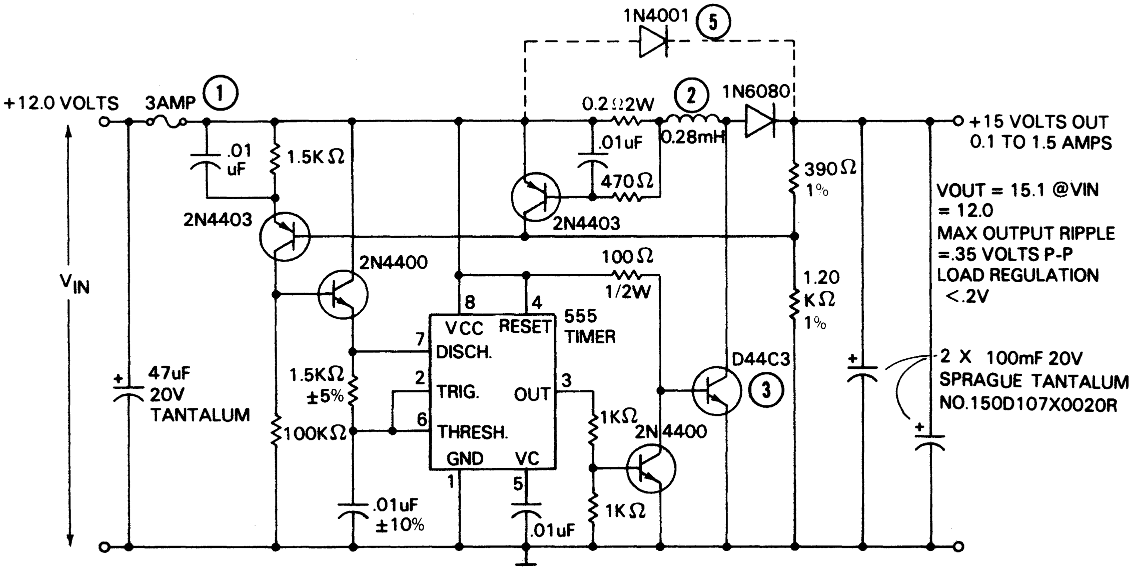
POWER SUPPLY
12V TO 15V CONVERSION

To accommodate systems running with a 12V power supply, logic schematics for two 12V to 15V converters are shown. These converters provide the 15V supply required to operate Data General's microNOVA integrated circuits.

The first voltage converter schematic produces a supply voltage of 15V at 1.5amps. This voltage converter supports a full microNOVA chip set, including 32K of memory.

The second schematic produces a supply voltage of 15.2V at 0.15amps. This will support either an mN601 or mN603 connection.

+12 TO +15 VOLT CONVERTER
FOR microNOVA CHIP SET



DG-04238

NOTES:

(1) *Fuse or +12volt current limit needed to avoid overheating 0.2Ω resistor and 1N6080 diode with permanent output short.*

(2) *0.28mH Inductor: 41 turns NO.20 wire on Arnold Magnetics toroid core NO.A-201163.*

(3) *D44C3 transistor on time = 10μSEC. Off time varies with load. Max frequency is 25KHz.*

(4) *Max component height 0.34" above board.*

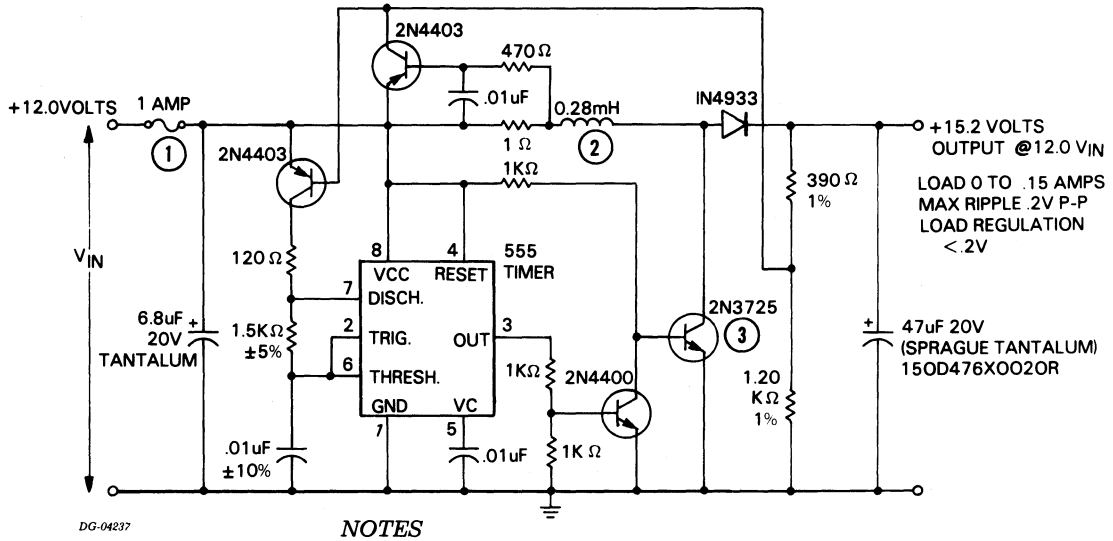
(5) *1N4001 diode may be necessary to prevent overshoot of output if V IN turn on time is fast.*

(6) *Efficiency: 80%, typical.*

DESIGN NOTES

POWER SUPPLY 12V TO 15V CONVERSION

+12 TO +15 VOLT CONVERTER
FOR microNOVA CHIPS



DG-04237

NOTES

1. *Fuse or +12 current limit necessary to avoid overheating 1Ω resistor and 1N4933 diode, in the event of permanent output shorts.*
2. *0.28MH: 71 turns NO.24 wire on Arnold Magnetics toroid core NO.A-05056.*
3. *2N3725 on time is 10μs. Output load varies frequency.*
4. *Max. component height 0.34".*
5. *Efficiency: 70%, typical.*

14

14

APPENDICES

14

APPENDICES
CONTENTS

NOTES:

① I/O COMMAND OR DATA SENT FROM mN601 MICROPROCESSOR TO THE CPU I/O TRANSCEIVER.

② CPU I/O COMMAND OR DATA DIFFERENTIALLY DRIVEN ON THE I/O BUS BY THE CPU I/O TRANSCEIVER.

③ DIFFERENTIALLY DRIVEN I/O COMMAND OR DATA RETRIEVED FROM THE I/O BUS BY THE IOC I/O TRANSCEIVER.

④ CPU I/O COMMAND OR DATA RECEIVED BY THE mN603 I/O CONTROLLER.

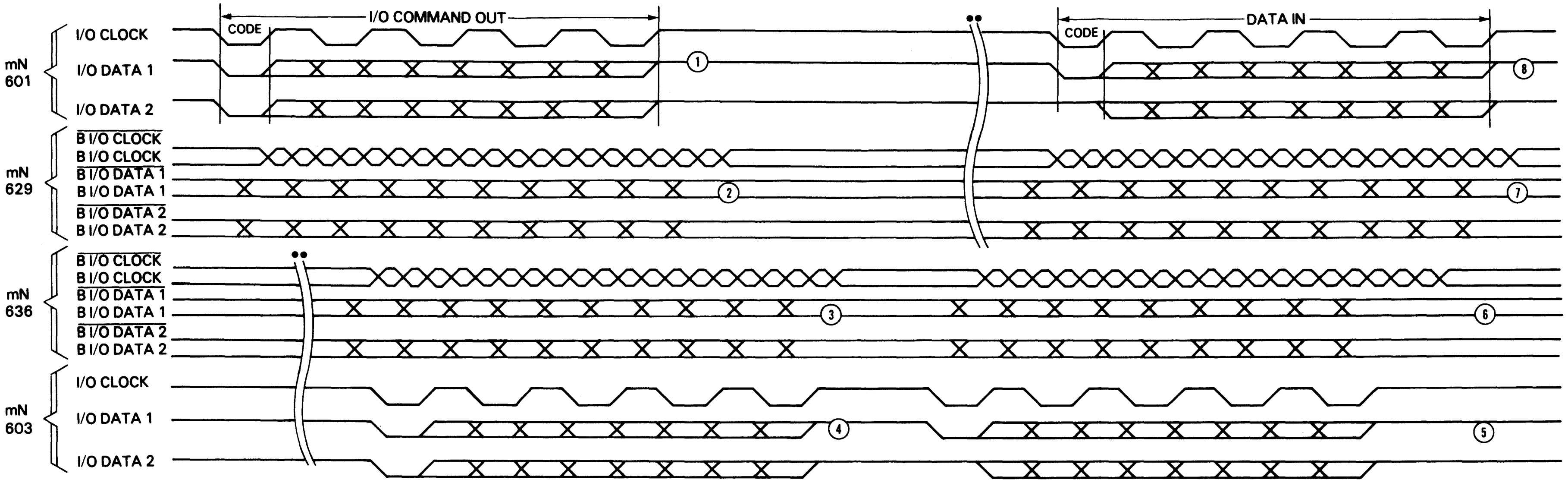
⑤ DATA SENT FROM THE mN603 TO THE IOC I/O TRANSCEIVER.

⑥ IOC DATA DIFFERENTIALLY DRIVEN ON THE I/O BUS BY THE IOC I/O TRANSCEIVER.

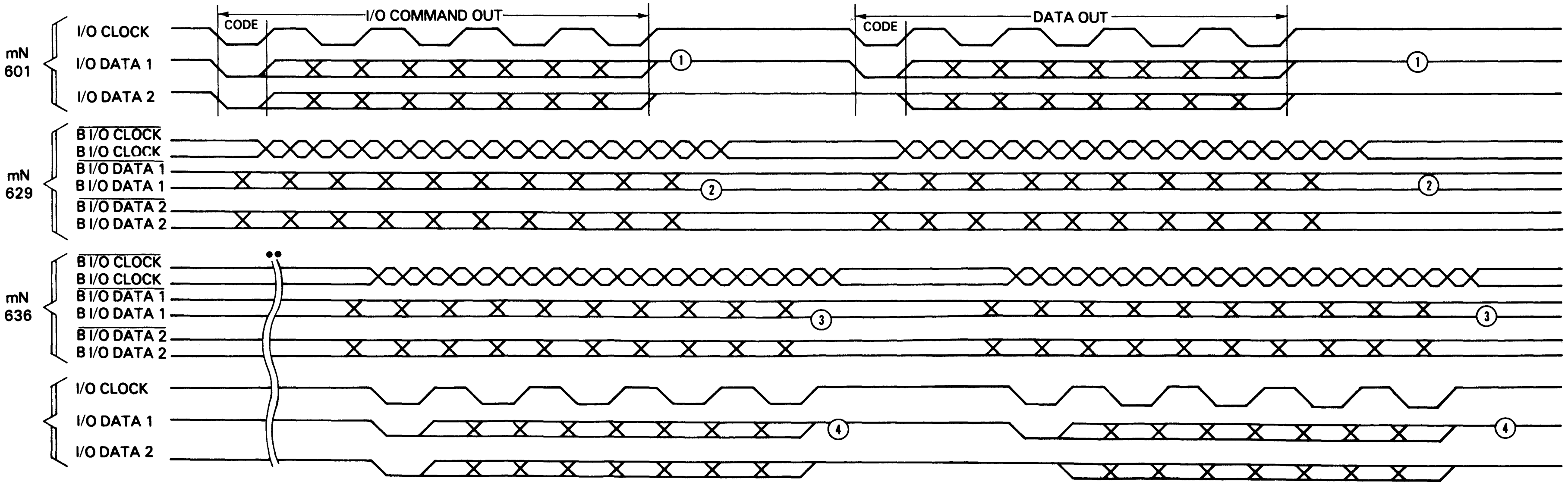
⑦ DIFFERENTIALLY DRIVEN DATA RETRIEVED FROM THE I/O BUS BY THE CPU I/O TRANSCEIVER.

⑧ IOC DATA RECEIVED BY THE mN601.

PROGRAMMED I/O DATA IN TIMING DIAGRAM



PROGRAMMED I/O DATA OUT TIMING DIAGRAM



NOTE:
SEE mN601 I/O OPERATIONS AND I/O TIMING TABLE FOR DETAILED INFORMATION.
•• TIME IS DEPENDENT UPON THE LENGTH OF THE I/O BUS BETWEEN THE mN601 AND THE mN603.

DG-04232

NOTES :

① DATA CHANNEL ADDRESS REQUEST SENT FROM THE mN601 TO THE CPU I/O TRANSCEIVER.

② DATA CHANNEL ADDRESS REQUEST DIFFERENTIALLY DRIVEN ON THE I/O BUS BY THE CPU I/O TRANSCEIVER.

③ DIFFERENTIALLY DRIVEN DATA CHANNEL ADDRESS REQUEST RETRIEVED FROM THE I/O BUS BY THE IOC I/O TRANSCEIVER.

④ DATA CHANNEL ADDRESS REQUEST RECEIVED BY THE mN603.

⑤ MEMORY ADDRESS AND DIRECTION MODE BIT SENT FROM THE mN603 TO THE IOC I/O TRANSCEIVER.

⑥ MEMORY ADDRESS AND DIRECTION MODE BIT DIFFERENTIALLY DRIVEN ON THE I/O BUS BY THE IOC I/O TRANSCEIVER.

⑦ DIFFERENTIALLY DRIVEN MEMORY ADDRESS AND DIRECTION MODE BIT RETRIEVED FROM THE I/O BUS BY THE CPU I/O TRANSCEIVER.

⑧ MEMORY ADDRESS AND MODE BIT RECEIVED BY THE mN601 AND READ-MODIFY-WRITE MEMORY OPERATION INITIATED.

DATA CHANNEL DATA OUT OPERATION - MEMORY DATA TO I/O CONTROLLER

⑨ DATA RETRIEVED BY CPU DURING READ PORTION OF MEMORY OPERATION IN PROGRESS. WRITE PORTION OF OPERATION NOT EXECUTED.

⑩ MEMORY DATA SENT FROM THE mN601 TO THE CPU I/O TRANSCEIVER.

⑪ MEMORY DATA DIFFERENTIALLY DRIVEN ON THE I/O BUS BY THE CPU I/O TRANSCEIVER.

⑫ DIFFERENTIALLY DRIVEN MEMORY DATA RETRIEVED FROM THE I/O BUS BY THE IOC I/O TRANSCEIVER.

⑬ MEMORY DATA RECEIVED BY THE mN603.

DATA CHANNEL DATA IN OPERATION - DATA FROM I/O CONTROLLER TO MEMORY

⑭ DATA SENT FROM THE mN603 TO THE IOC I/O TRANSCEIVER.

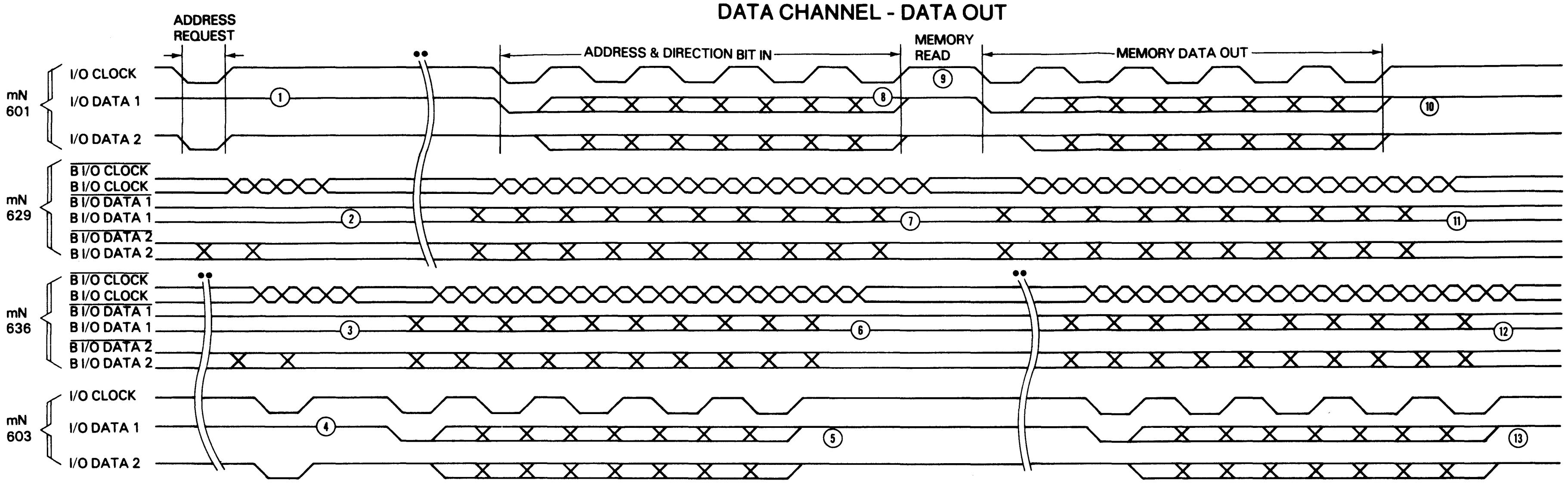
⑮ DATA DIFFERENTIALLY DRIVEN ON THE I/O BUS BY THE IOC I/O TRANSCEIVER.

⑯ DIFFERENTIALLY DRIVEN DATA RETRIEVED FROM THE I/O BUS BY THE CPU I/O TRANSCEIVER.

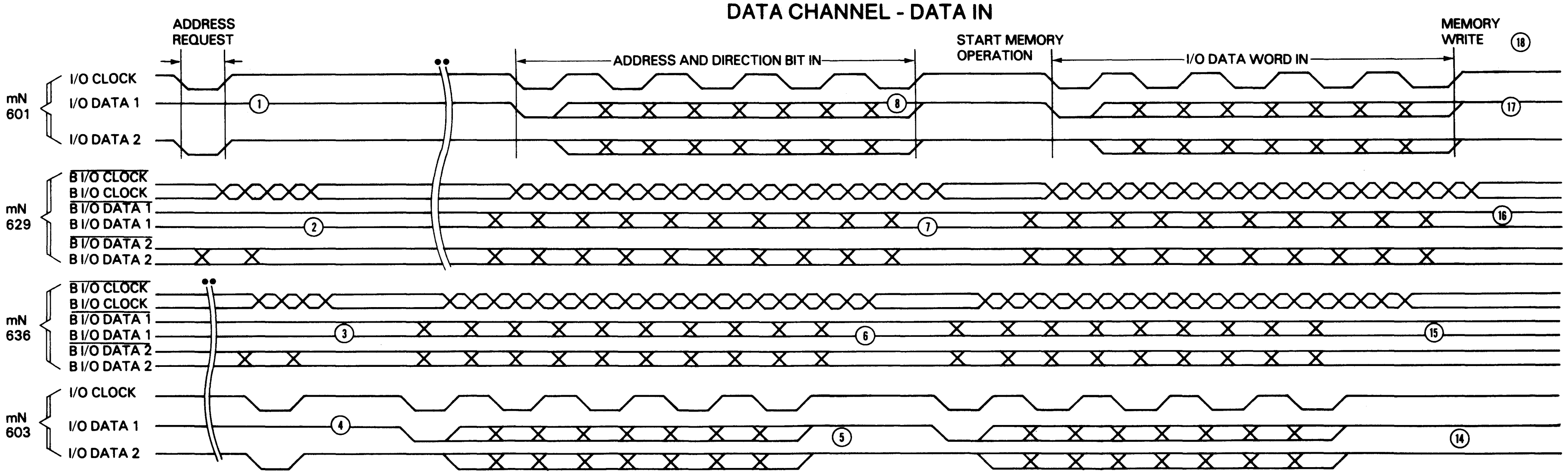
⑰ DATA RECEIVED BY THE mN601.

⑱ DATA WRITTEN IN MEMORY DURING WRITE PORTION OF MEMORY OPERATION IN PROGRESS.

DATA CHANNEL - DATA OUT



DATA CHANNEL - DATA IN



DG-04233

NOTE:
SEE mN601 I/O OPERATIONS AND I/O TIMING TABLE FOR DETAILED INFORMATION.
●●TIME IS DEPENDENT UPON THE LENGTH OF THE I/O BUS BETWEEN THE mN601 AND THE mN603.

APPENDIX B
LOGIC CONVENTIONS

LOGIC CONVENTIONS

Data General logic schematics are drawn in close accordance with Mil-Std-806C. Using this convention, logical functions are drawn as physically implemented; i.e., where discrete gates are used to implement a function, the gates are shown. On the other hand, where a more complex integrated circuit is used, e.g., a memory transceiver, that function is shown as a rectangular box instead of the gates comprising the function.

Throughout this manual, a distinction is frequently made between electrical levels and logical values. Electrical levels are referred to as high or low; logical values, as 1 or 0.

Signal Names

The voltage level at which a signal is said to be 'asserted' (true) is a matter of definition. To distinguish between signals that are asserted high (0=L, 1=H) and those that are asserted low (0=H, 1=L), a naming convention has been adopted in Data General's documentation which defines the relationship between the logical value and electrical level of a signal. Signal names which include a horizontal bar over the name, e.g., $\overline{\text{WRITE}}$, are asserted when they are at a low electrical level; signal names without the bar, e.g., WRITE, are asserted when they are at a high electrical level.

Logical functions often require more than one binary signal. For instance, three lines are required to express an octal digit. In general, these closely related signals are individually identified by effectively subscripting a common label. For example, MB0 through MB15 are all required to completely specify a function. All or part of this group of signals is identified by placing brackets around the range of subscripts included, as MB < 0-15 > . In this case, the suffix carries the information that there are sixteen MB lines under discussion, from MB0 through MB15, inclusive.

DG OFFICES

NORTH AMERICAN OFFICES

Alabama: Birmingham

Arizona: Phoenix, Tucson

Arkansas: Little Rock

California: Anaheim, El Segundo, Fresno, Los Angeles, Oakland, Palo Alto, Riverside, Sacramento, San Diego, San Francisco, Santa Barbara, Sunnyvale, Van Nuys

Colorado: Colorado Springs, Denver

Connecticut: North Branford, Norwalk

Florida: Ft. Lauderdale, Orlando, Tampa

Georgia: Norcross

Idaho: Boise

Iowa: Bettendorf, Des Moines

Illinois: Arlington Heights, Champaign, Chicago, Peoria, Rockford

Indiana: Indianapolis

Kentucky: Louisville

Louisiana: Baton Rouge, Metairie

Maine: Portland, Westbrook

Maryland: Baltimore

Massachusetts: Cambridge, Framingham, Southboro, Waltham, Wellesley, Westboro, West Springfield, Worcester

Michigan: Grand Rapids, Southfield

Minnesota: Richfield

Missouri: Creve Coeur, Kansas City

Mississippi: Jackson

Montana: Billings

Nebraska: Omaha

Nevada: Reno

New Hampshire: Bedford, Portsmouth

New Jersey: Cherry Hill, Somerset, Wayne

New Mexico: Albuquerque

New York: Buffalo, Lake Success, Latham, Liverpool, Melville, New York City, Rochester, White Plains

North Carolina: Charlotte, Greensboro, Greenville, Raleigh, Research Triangle Park

Ohio: Brooklyn Heights, Cincinnati, Columbus, Dayton

Oklahoma: Oklahoma City, Tulsa

Oregon: Lake Oswego

Pennsylvania: Blue Bell, Lancaster, Philadelphia, Pittsburgh

Rhode Island: Providence

South Carolina: Columbia

Tennessee: Knoxville, Memphis, Nashville

Texas: Austin, Dallas, El Paso, Ft. Worth, Houston, San Antonio

Utah: Salt Lake City

Virginia: McLean, Norfolk, Richmond, Salem

Washington: Bellevue, Richland, Spokane

West Virginia: Charleston

Wisconsin: Brookfield, Grand Chute, Madison

DG-04976

INTERNATIONAL OFFICES

Argentina: Buenos Aires

Australia: Adelaide, Brisbane, Hobart, Melbourne, Newcastle, Perth, Sydney

Austria: Vienna

Belgium: Brussels

Bolivia: La Paz

Brazil: Sao Paulo

Canada: Calgary, Edmonton, Montreal, Ottawa, Quebec, Toronto, Vancouver, Winnipeg

Chile: Santiago

Columbia: Bogato

Costa Rica: San Jose

Denmark: Copenhagen

Ecuador: Quito

Egypt: Cairo

Finland: Helsinki

France: Le Plessis-Robinson, Lille, Lyon, Nantes, Paris, Saint Denis, Strasbourg

Guatemala: Guatemala City

Hong Kong

India: Bombay

Indonesia: Jakarta, Pusat

Ireland: Dublin

Israel: Tel Aviv

Italy: Bologna, Florence, Milan, Padua, Rome, Tourin

Japan: Fukuoka, Hiroshima, Nagoya, Osaka, Tokyo, Tsukuba

Jordan: Amman

Korea: Seoul

Kuwait: Kuwait

Lebanon: Beirut

Malaysia: Kuala Lumpur

Mexico: Mexico City, Monterrey

Morocco: Casablanca

The Netherlands: Amsterdam, Rijswijk

New Zealand: Auckland, Wellington

Nicaragua: Managua

Nigeria: Ibadan, Lagos

Norway: Oslo

Paraguay: Asuncion

Peru: Lima

Philippine Islands: Manila

Portugal: Lisbon

Puerto Rico: Hato Rey

Saudi Arabia: Jeddah, Riyadh

Singapore

South Africa: Cape Town, Durban, Johannesburg, Pretoria

Spain: Barcelona, Bibao, Madrid

Sweden: Gothenburg, Malmo, Stockholm

Switzerland: Lausanne, Zurich

Taiwan: Taipei

Thailand: Bangkok

Turkey: Ankara

United Kingdom: Birmingham, Bristol, Glasgow, Hounslow, London, Manchester

Uruguay: Montevideo

USSR: Espoo

Venezuela: Maracaibo

West Germany: Dusseldorf, Frankfurt, Hamburg, Hannover, Munich, Nuremburg, Stuttgart

DataGeneral
Data General Corporation, Westboro, Massachusetts 01581

CUT ALONG DOTTED LINE

Data General

Engineering Publications Comment Form

Please help us improve our future publications by answering the questions below. Use the space provided for your comments.

Title: ______________________________

Document No. 014-000074-04

Yes	No			
□	□	Is this manual easy to read?	○ You (can,cannot) find things easily. ○ Language (is, is not) appropriate. ○ Technical terms (are, are not) defined as needed.	○ Other:
		In what ways do you find this manual useful?	○ Learning to use the equipment ○ As a reference ○ As an introduction to the product	○ To instruct a class. ○ Other:
□	□	Do the illustrations help you?	○ Visuals (are, are not) well designed. ○ Labels and captions (are, are not) clear. ○ Other:	
□	□	Does the manual tell you all you need to know? What additional information would you like?		
□	□	Is the information accurate? (If not please specify with page number and paragraph.)		

Name: ______________________________ Title: ______________________________

Company: ______________________________ Division: ______________________________

Address: ______________________________ City: ______________________________

State: ____________ Zip: ____________ Telephone: ____________ Date: ____________

TAPE TAPE

FOLD FOLD

FOLD FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

Data General
ATTN: Engineering Publications (C-138)
4400 Computer Drive
Westboro, MA 01581