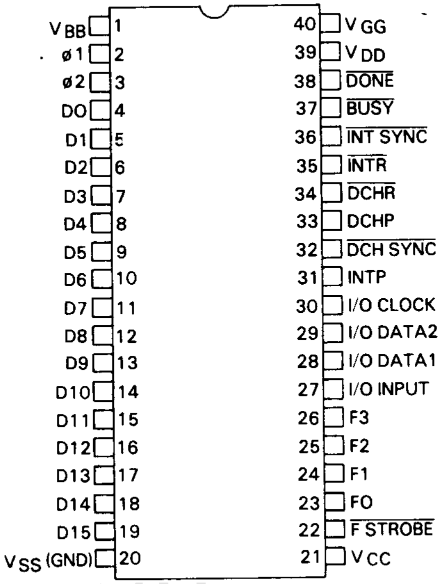
mN603
I/O CONTROLLER

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

PACKAGE



DG-04137

GENERAL DESCRIPTION

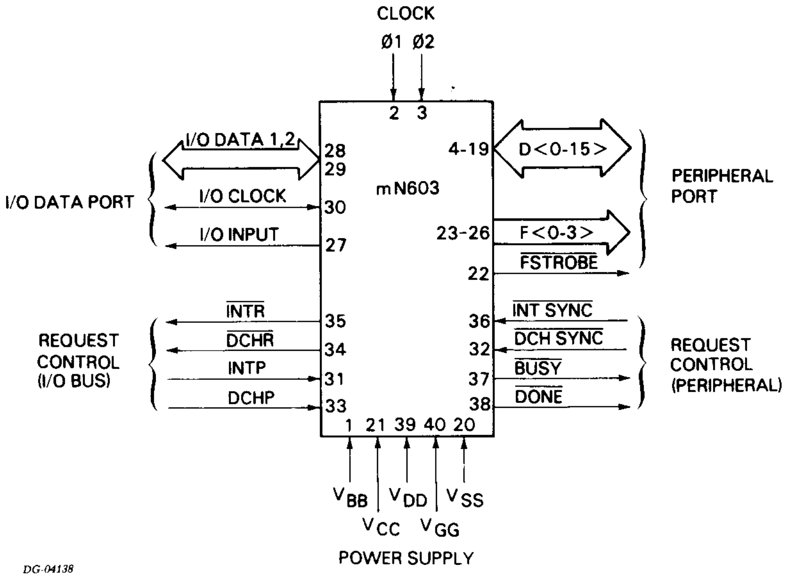
The mN603 is an I/O controller (IOC) that provides the full functional capability of Data General's 47-line NOVA I/O bus. The IOC decodes an encoded data stream from the CPU and presents a parallel 16-bit bidirectional interface, four encoded function bits, and a function strobe, to the peripheral for simple interfacing.

The 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. Those necessary are:

1) an mN640 Clock Driver
2) an mN636 IOC I/O Tranceiver
3) interrupt and data channel priority circuitry
4) decode logic for the IOC's four encoded function bits
5) additional gating to interface a peripheral and buffer the IOC

FUNCTIONAL PIN CONNECTION DIAGRAM



DG-04138

The following table describes the function of each pin shown in the pin connection diagram:

PIN DESCRIPTIONS

MNEMONIC	PIN NO.	IN/ OUT	FUNCTION
			CLOCKS
Ø1 Ø2	2 3	IN IN	Two-phase non-overlapping clock. Operates between 0 and 14V amplitude. 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 DESCRIPTIONS (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 DESCRIPTIONS (CONT.)

MNEMONIC	PIN NO.	IN/ OUT	POWER
V_{BB}	1		-4.25 ± 0.5 V
V_{CC}	21		+5 ± 0.25 V
V_{DD}	39		+10 ± 1.0 V
V_{GG}	40		+14 ± 1.0 V
V_{SS}	20		GROUND

NOTE: * MASTER CLOCK refers to the system clock, see Overview Section of the microNOVA Line Integrated Circuits.

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 compatible with 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 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 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 (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 mN603. 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:

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

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. Likewise, no specific function code is generated by a No I/O Transfer 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

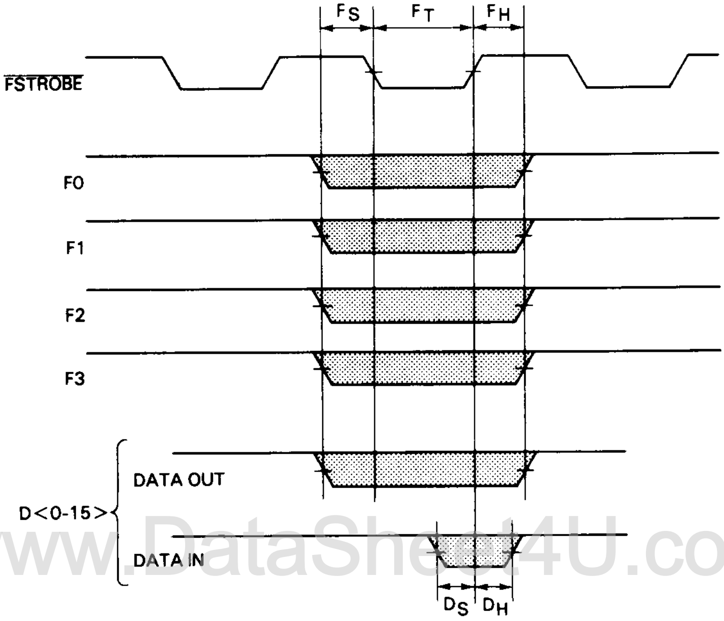
FUNCTION CODE F<0-3>	LABEL	FUNCTION
0000	DOA	Load the A register with data from IOC data lines D <0-15>.
0001	DIA	Gate data into IOC from the A register via data lines D <0-15>.
0010	DOB	Load the B register with data from IOC data lines D <0-15> (see Internal Structure, Data Channel Registers).
0011	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 <0-15> = 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 D0. If external registers are being used, gate external Address Register into 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).

DG-04411

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:



NOTES:

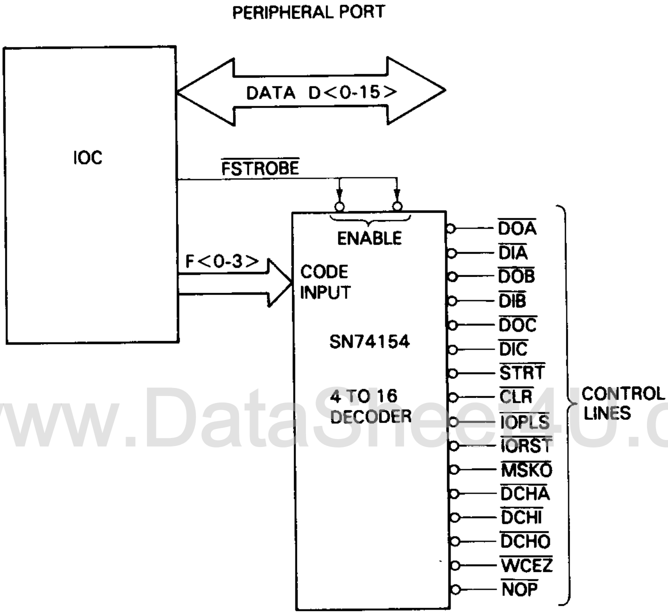
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE ON PAGE 43.
- D<0-15> MUST NOT BE PULLED LOW AT ANY TIME OTHER THAN THAT INDICATED BY D_S AND D_H IN THE DIAGRAM ABOVE.

DG-04139

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 $\overline{\text{FSTROBE}}$:

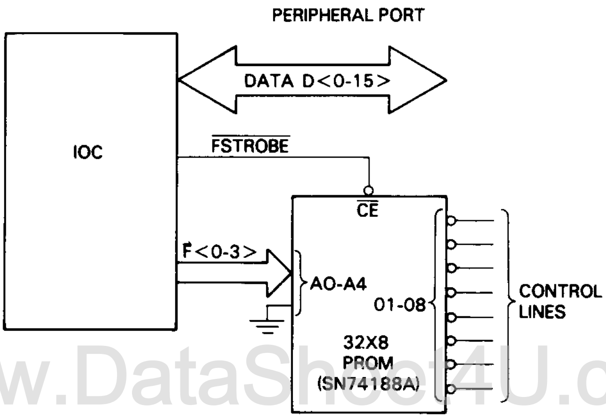


DG-04140

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.

PROM DECODER



DG-04141

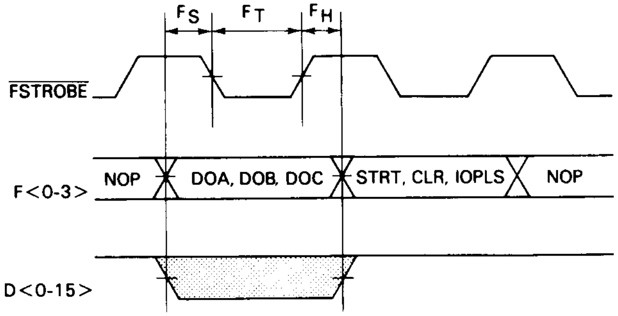
Most of the control lines resulting from the decoding of F <0-3> are used to control the flow of the data to and from the Peripheral Port Data lines. These data lines are used when an IOC executes certain programmed I/O instructions or when it performs Data Channel Breaks. The following sections explain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



NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE ON PAGE 43.

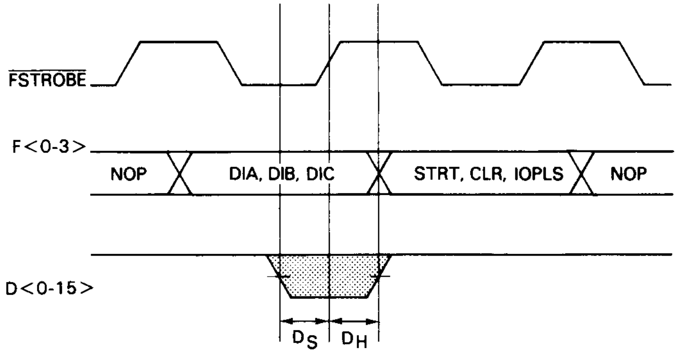
DG-04142

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).

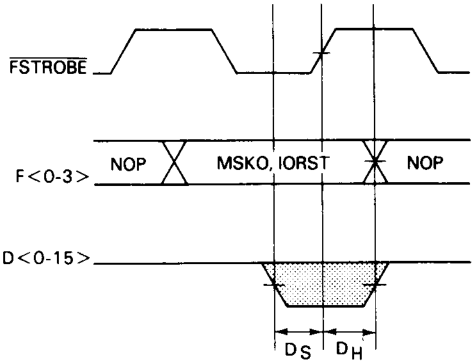


NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE ON PAGE 43.

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).



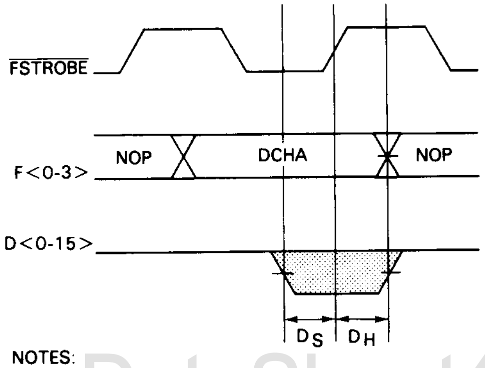
NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE ON PAGE 43.

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).



NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE ON PAGE 43.

DG-04145

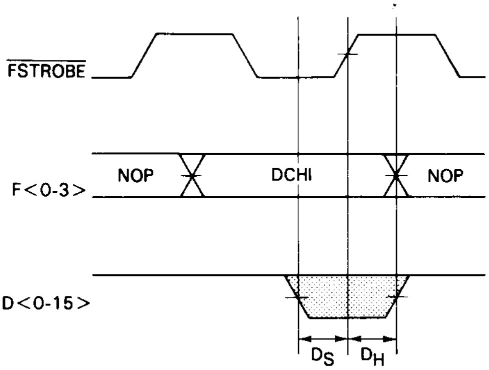
If the direction bit is a 0, a Data Channel Out Transaction is performed. This transfers a 16 bit word "out" from the memory location specified by the Data Channel Address to the peripheral device. If the direction bit is a 1, a Data Channel In Transaction is performed. In this case, a 16 bit word is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

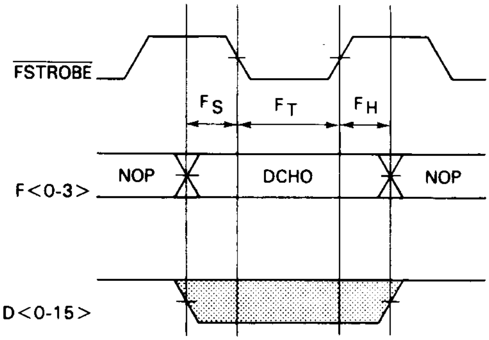
If the first transfer specifies a Data Channel In Transaction, the DCHI code is issued during the second transfer. This code may be used to strobe the data word onto D < 0-15 >. The IOC samples these lines on the rising edge of $\overline{\text{FSTROBE}}$ when the DCHI code is valid.



NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE ON PAGE 43.

DG-04146

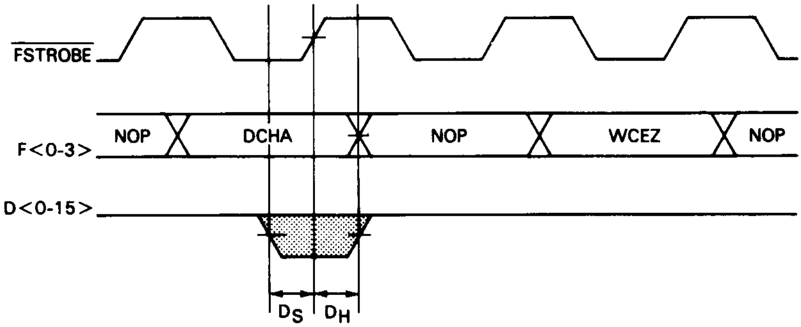
If the first transfer specifies a Data Channel Out Transaction, the DCHO code is issued during the second transfer. The IOC places the contents of the second data transfer on D < 0-15 > when the DCHO code is valid. This code may be used to strobe the data word into the appropriate peripheral register.



NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE ON PAGE 43.

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.



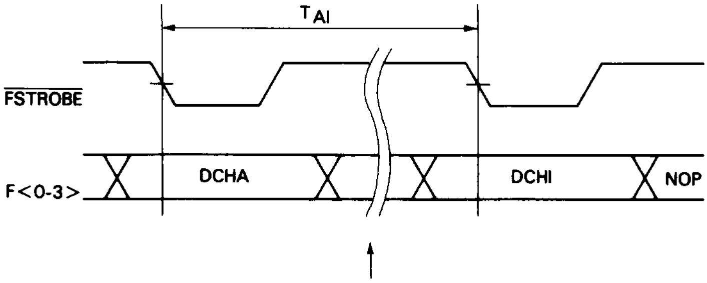
NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE ON PAGE 43.

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

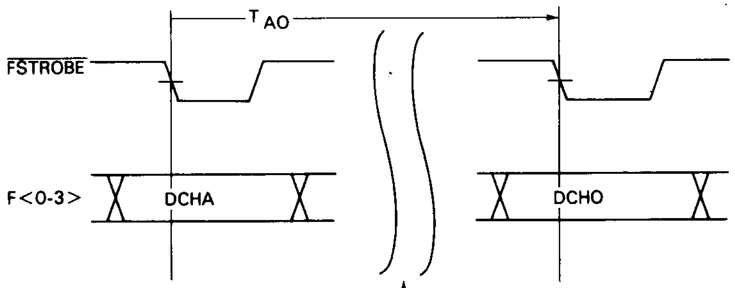


NOTE:
ANOTHER DCHA OR OTHER FUNCTION CODE MAY OCCUR HERE, SEE INTERNAL STRUCTURE, DATA CHANNEL LOGIC

NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE ON PAGE 43.

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 ON PAGE 43.

DG-04150

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 the mN601 (CPU) instruction execution times is given under that chip's section.

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 (mN601) 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.

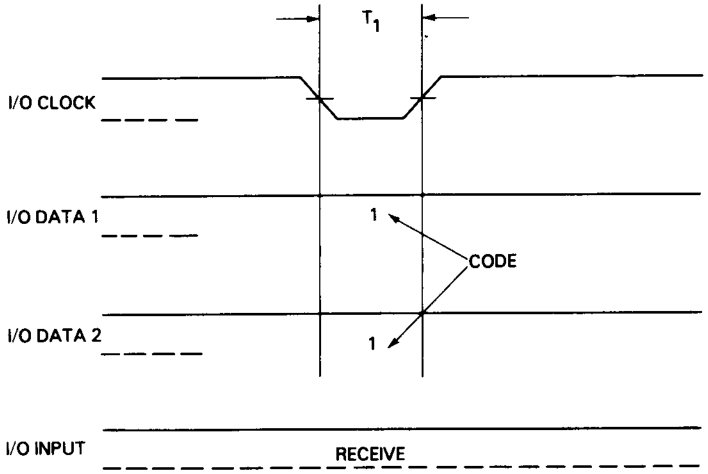
These information types are transferred in one of two formats; short and long. Both Request Enable and Data Channel Address Request use the short format which requires only one I/O CLOCK pulse. I/O Instructions and Data use the long format which requires five I/O CLOCK pulses. In all cases, the first 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:

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).

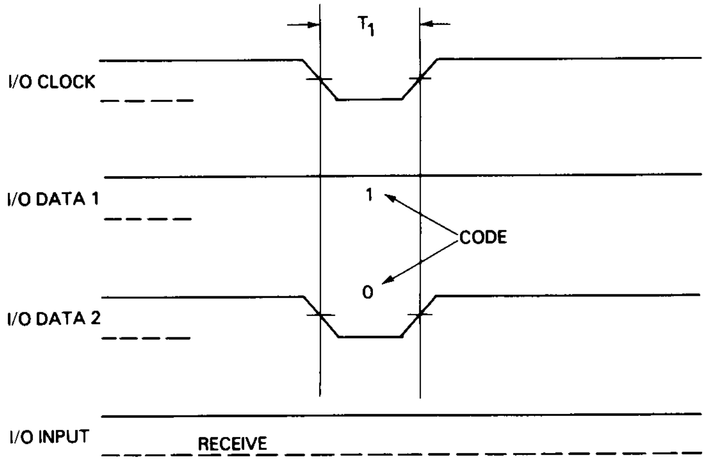


NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE ON PAGE 43.

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).



NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE ON PAGE 43.

DG-04152

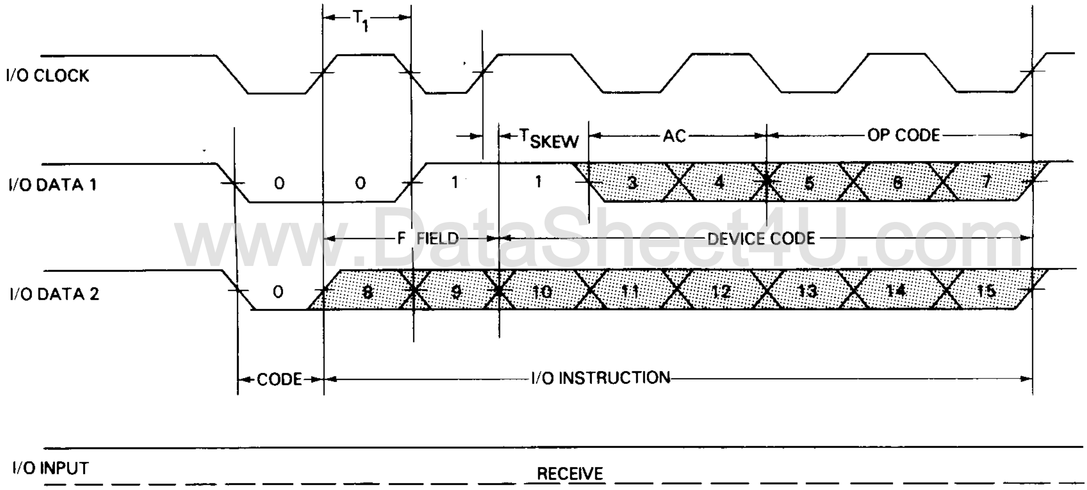
I/O Instructions

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.

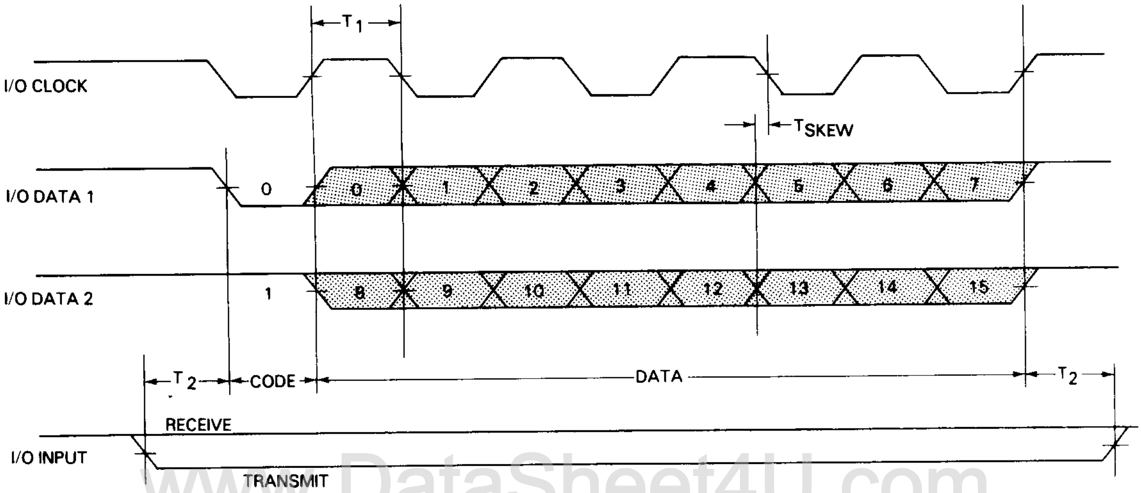


NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE ON PAGE 43.

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.



NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE ON PAGE 43.

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.

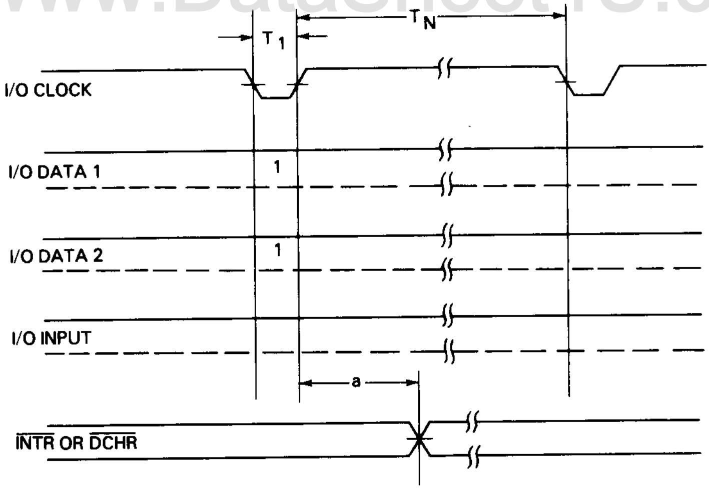
Request Enable Transactions

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).



NOTES:

- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE ON PAGE 43.

DG-04155

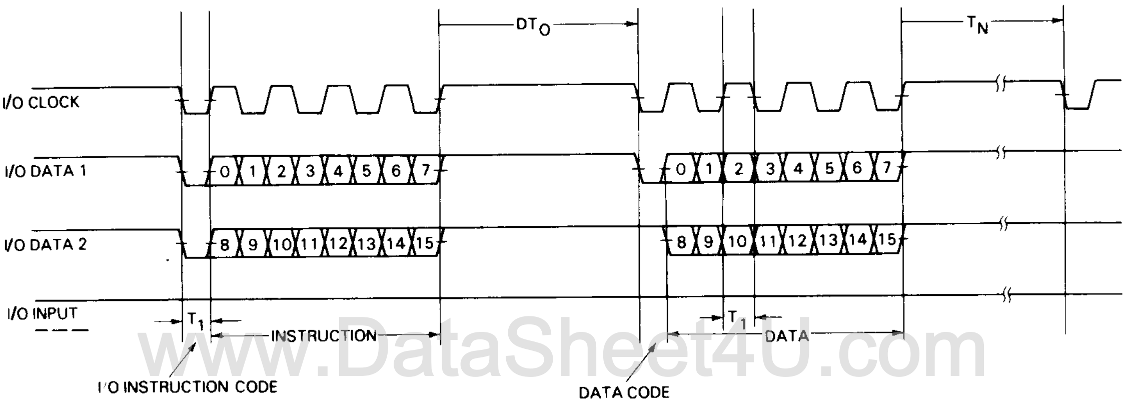
Programmed I/O Transactions

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.



NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE ON PAGE 43.

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 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 = see mN601, CPU
F = see Internal Structure
Op Code = see below

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. 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

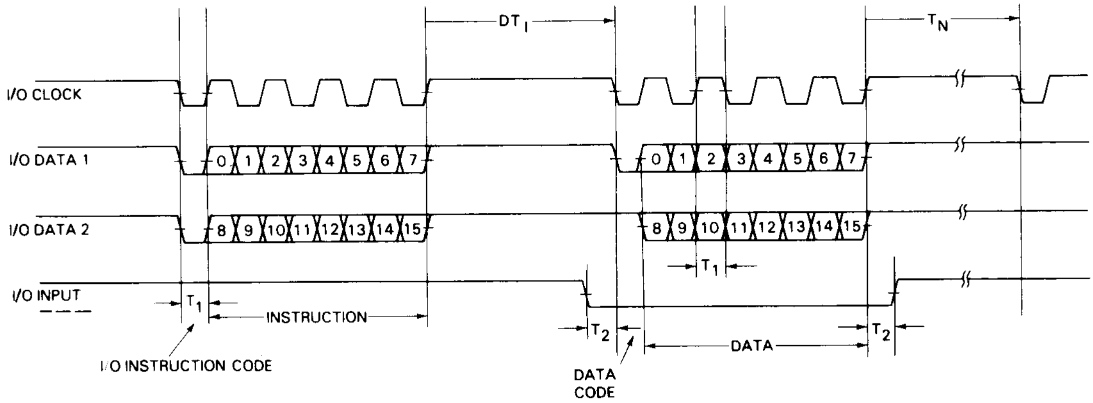
F = see mN601, CPU

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.

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.



NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE ON PAGE 43.

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

AC = see mN601, CPU
F = see Internal Structure
Op Code = see below

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

AC, F = see mN601, CPU

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

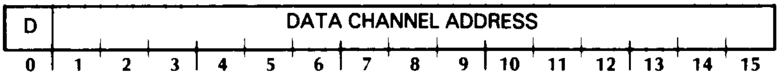
There are two types of Data Channel Transactions:
1) CPU to IOC Data Channel Out Transactions
2) IOC to CPU Data Channel In Transactions

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.

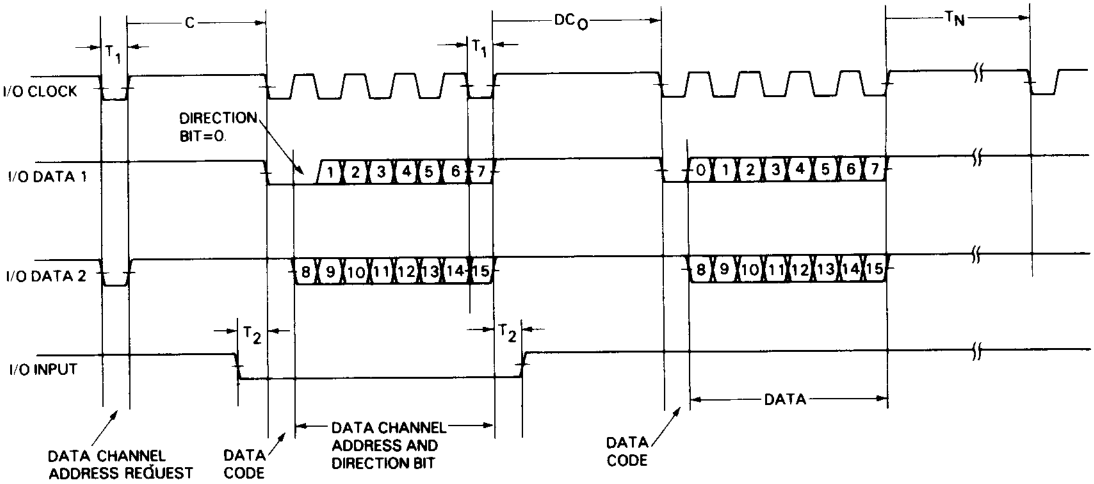


D = direction bit:
0 = Data Channel Out Transaction
1 = Data Channel In Transaction

NOTE: See Internal Structure, Data Channel Logic.

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 .

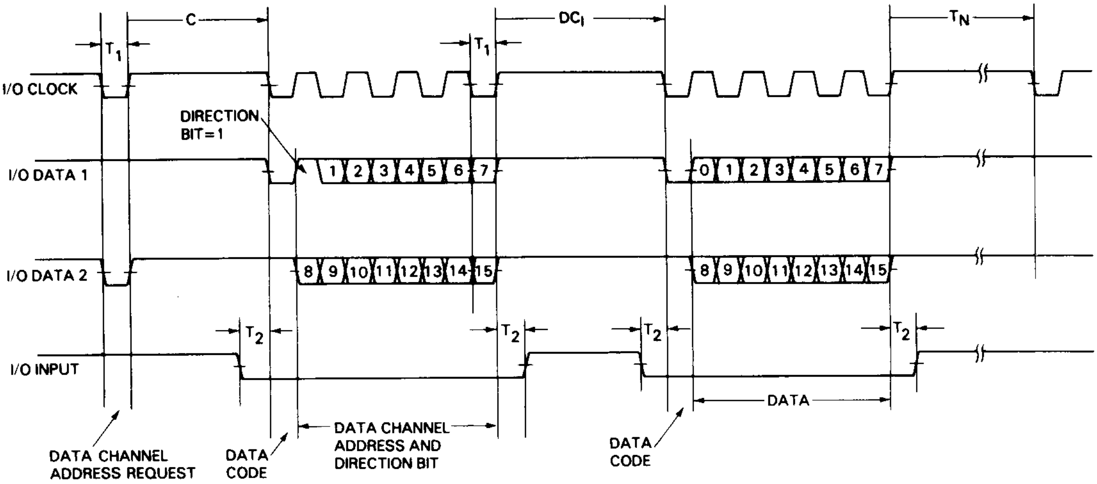


NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE ON PAGE 43.

DG-04158

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.

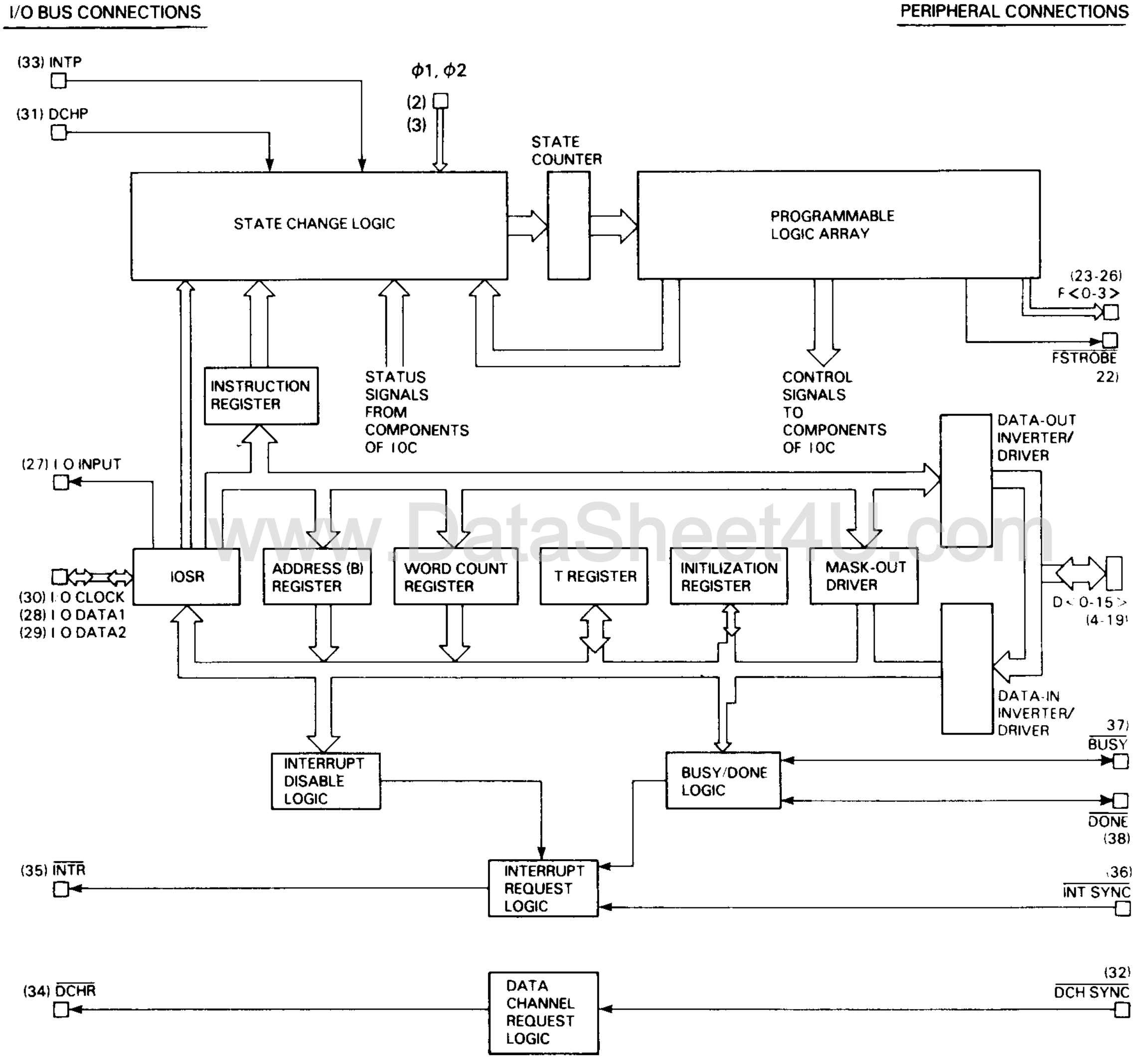


NOTES
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE ON PAGE 43.

DG-04159

INTERNAL STRUCTURE

In order to make 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

mN603
INTERNAL STRUCTURE

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

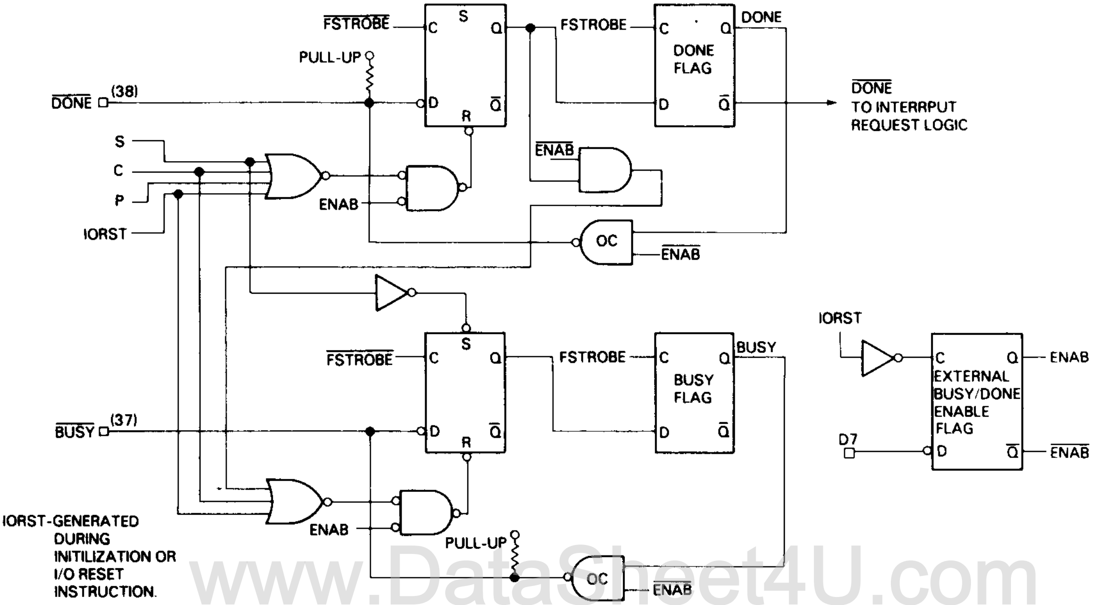
The BUSY and DONE flags indicate the state of the peripheral device. In normal use, the BUSY flag being set to 1 indicates the peripheral device is performing an operation; the DONE flag being set to 1 indicates the peripheral device has completed an operation.

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

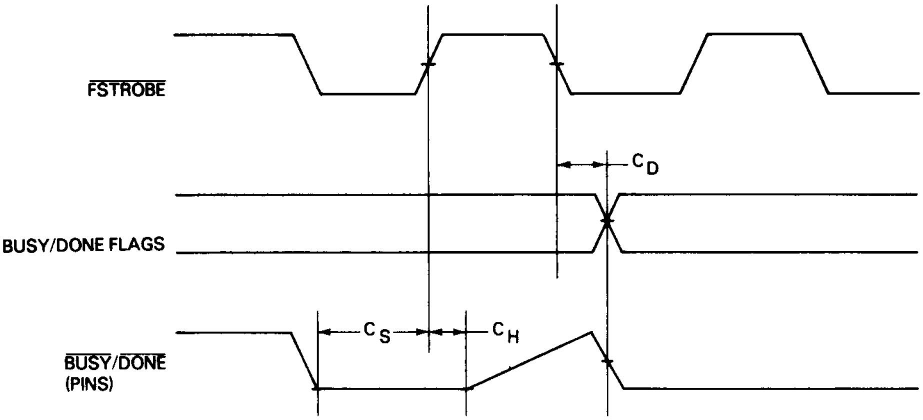
DG-04417

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:


$\overline{\text{FSTROBE}}$
C
S
Q
D
R
$\overline{\text{Q}}$
PULL-UP
$\overline{\text{DONE}}$ (38)
FSTROBE
DONE FLAG
DONE
$\overline{\text{DONE}}$ TO INTERRPUT REQUEST LOGIC
S
C
P
IORST
ENAB
$\overline{\text{ENAB}}$
OC
$\overline{\text{ENAB}}$
BUSY FLAG
BUSY
$\overline{\text{BUSY}}$ (37)
IORST
EXTERNAL BUSY/DONE ENABLE FLAG
ENAB
D7
$\overline{\text{ENAB}}$
IORST-GENERATED DURING INITILIZATION OR I/O RESET INSTRUCTION.
ENAB
PULL-UP
OC
$\overline{\text{ENAB}}$

DG-04161

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.


$\overline{\text{FSTROBE}}$
C_D
BUSY/DONE FLAGS
C_S
C_H
$\overline{\text{BUSY}}$/$\overline{\text{DONE}}$ (PINS)

NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED BETWEEN POINTS AT 1.5V.
- SEE TIMING TABLE ON PAGE 43.

DG-04162

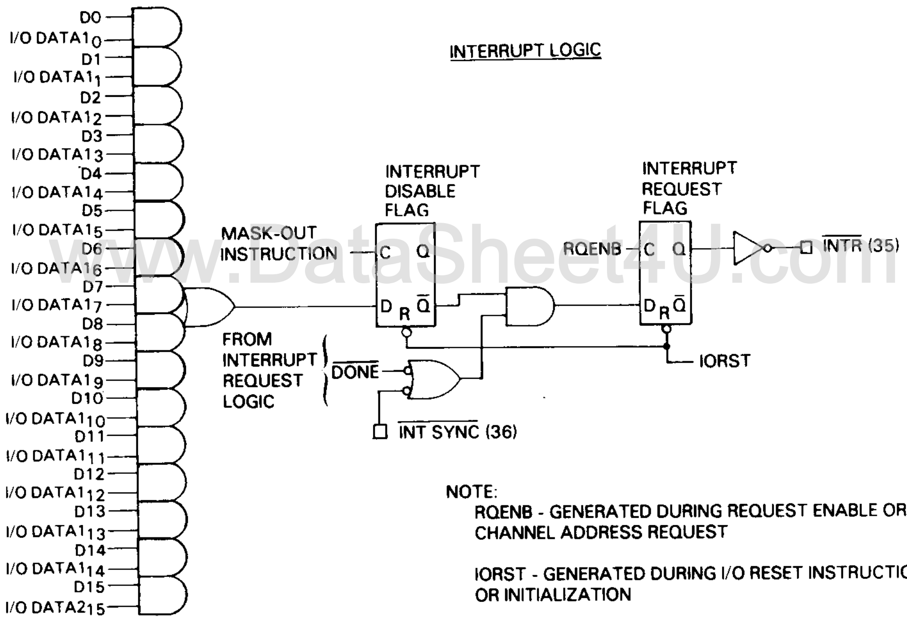
INTERRUPT REQUEST LOGIC

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.

Shown below is a representation of the Interrupt Request logic:

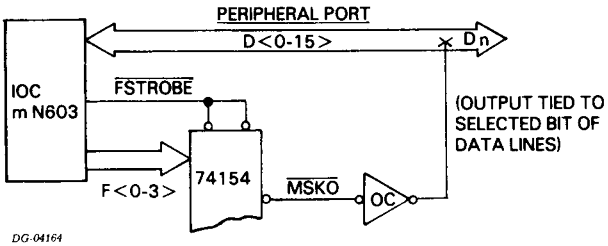


NOTE:
RQENB - GENERATED DURING REQUEST ENABLE OR DATA CHANNEL ADDRESS REQUEST

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, Program Interrupts)



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 DCH SYNC line. On the Reqest Enable or Data Channel Address Request following the assertion of DCH SYNC, the IOC's Data Channel Request Flag is set to 1. This flag controls the state of the data channel request line (DCHR). When the flag is set to one, the DCHR line is asserted requesting service from the CPU. Synchronizing the request in this way ensures the stability of the DCHR line when it is sampled by the CPU.

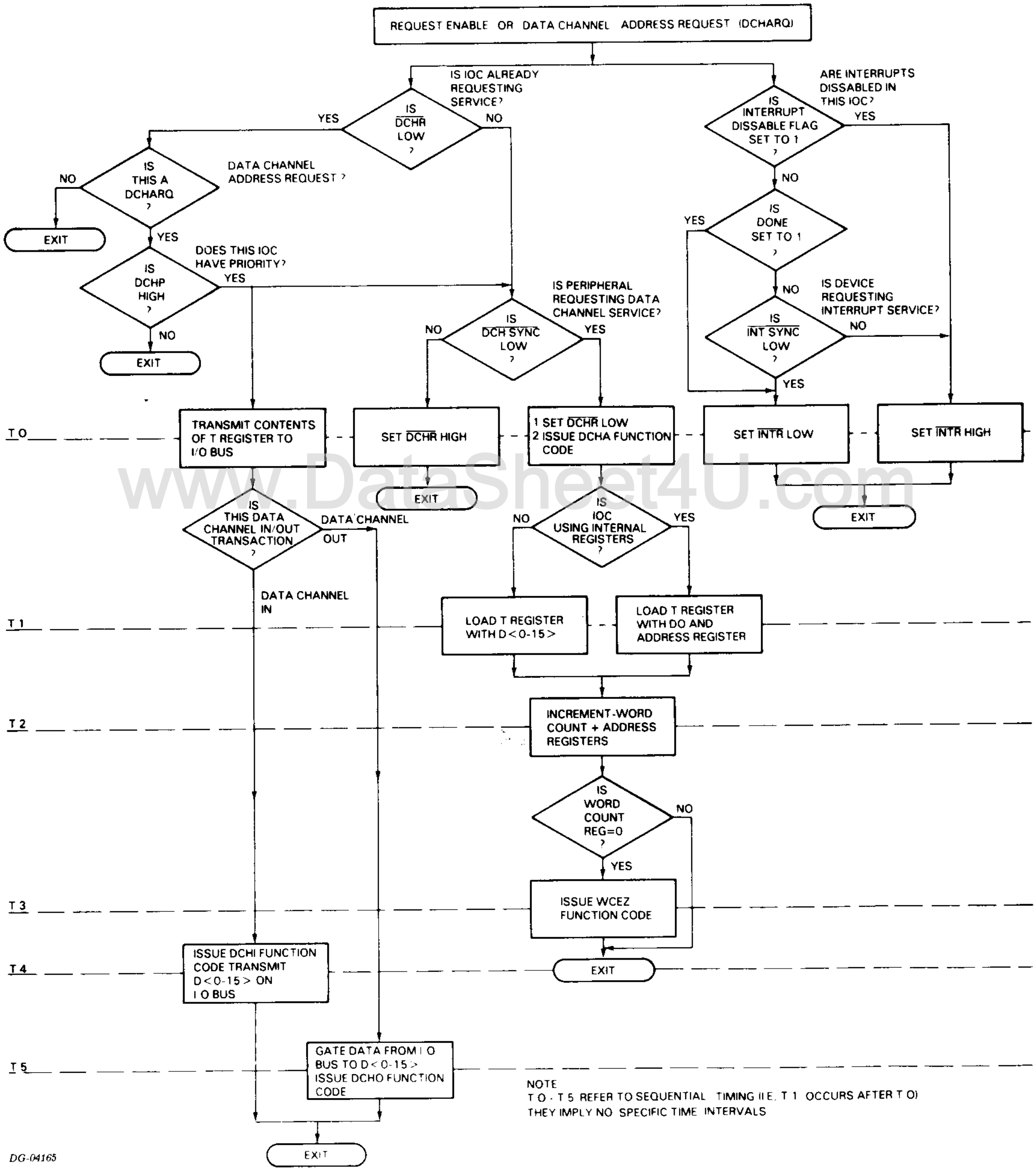
Upon receiving data channel service, an IOC must transmit control information about the transfer to the CPU (see I/O Data Port, Data Channel Transactions). If the control information is ready for transmission when service is received, the entire transaction can be accomplished more quickly. Therefore, 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 (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 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 (DCHR is released).

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 DISSABLED IN THIS IOC?
IS INTERRUPT DISSABLE FLAG SET TO 1 ?
YES
NO
IS THIS A DCHARQ ?
DATA CHANNEL ADDRESS REQUEST ?
NO
EXIT
YES
IS DONE SET TO 1 ?
YES
NO
DOES THIS IOC HAVE PRIORITY?
IS DCHP HIGH ?
YES
NO
EXIT
IS PERIPHERAL REQUESTING DATA CHANNEL SERVICE?
IS DCH SYNC LOW ?
NO
YES
IS DEVICE REQUESTING INTERRUPT SERVICE?
IS INT SYNC LOW ?
NO
YES
T0
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
T1
LOAD T REGISTER WITH D<0-15>
LOAD T REGISTER WITH DO AND ADDRESS REGISTER
T2
INCREMENT-WORD COUNT + ADDRESS REGISTERS
IS WORD COUNT REG=0 ?
NO
YES
T3
ISSUE WCEZ FUNCTION CODE
T4
ISSUE DCHI FUNCTION CODE TRANSMIT D<0-15> ON I O BUS
EXIT
T5
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

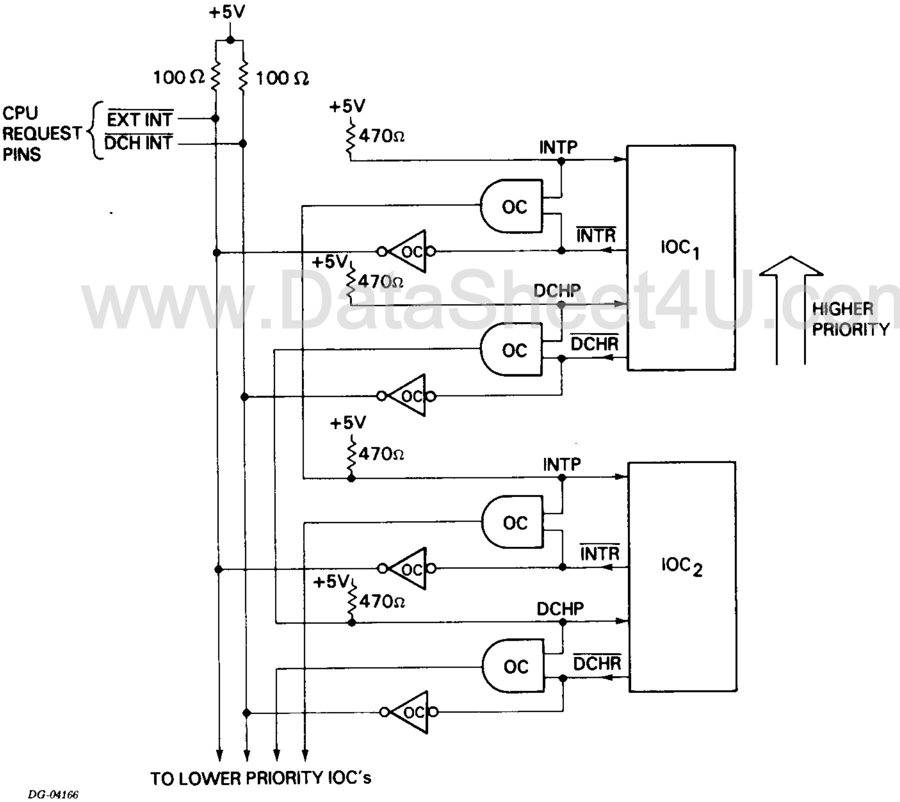
REQUEST CONTROL

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)



DG-04166

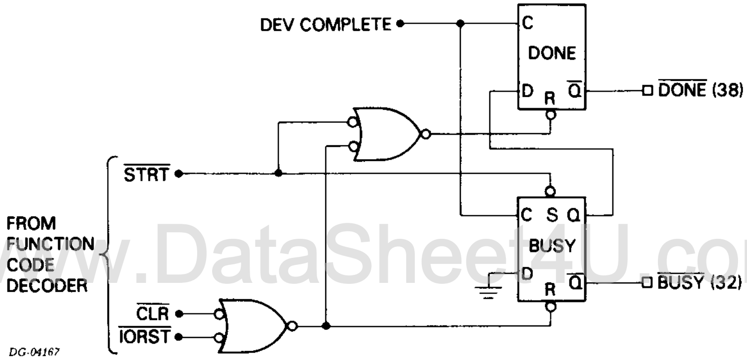
INTERRUPT REQUESTS

An IOC requests a program interrupt if either its DONE flag is set to one or the peripheral device asserts the IOC's $\overline{\text{INT SYNC}}$ 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 $\overline{\text{DONE}}$ or the assertion of $\overline{\text{INT SYNC}}$, the IOC requests service (asserts $\overline{\text{INTR}}$). The service request should remain in effect ($\overline{\text{DONE}}$ set or $\overline{\text{INT SYNC}}$ 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 $\overline{\text{INT SYNC}}$ being asserted, the instruction should specify a function code which the peripheral should use to release $\overline{\text{INT SYNC}}$. In both cases, the IOC stops requesting interrupt service (asserting $\overline{\text{INTR}}$) on the Request Enable or Data Channel Address following reception of the I/O Instruction honoring the request.

EXTERNAL BUSY/DONE LOGIC

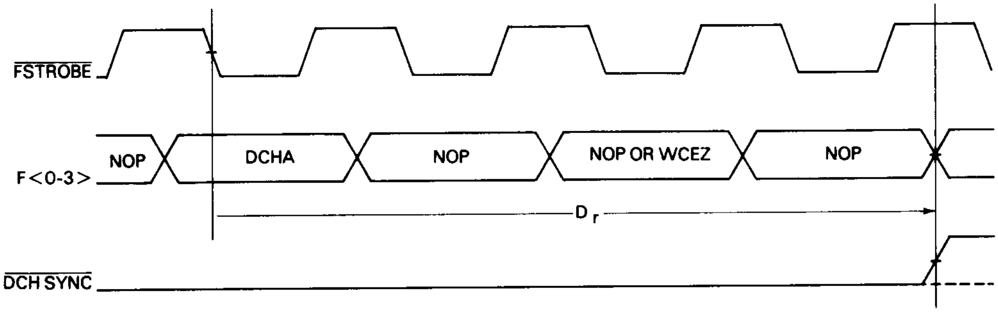
The following circuit is an example of how external BUSY/DONE flags may be connected to an IOC. In this case, the External BUSY/DONE Enable flag must be set to 1 (see Internal Structure and System Requirements, Initialization).



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 DCH SYNC line. On the first Request Enable or Data Channel Address Request following the assertion of DCH SYNC, the 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 DCH SYNC line unless another data channel transfer is desired. The interval Dr in the following diagram indicates the time in which DCH SYNC must be removed to prevent prompting extraneous data channel requests:



NOTES:
- UNLESS OTHERWISE NOTED, TIME INTERVALS ARE MEASURED FROM A 1.5 VOLT LEVEL.
- SEE TIMING TABLE ON PAGE 43.

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

On power-up, V_{BB} must be brought within its specified operating range before the clocks $\phi1$ and $\phi2$ 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 $\phi1$ and $\phi2$ 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>:

NOTE: The contents of the polarity bit does not take effect until after Initialization is complete.

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

CLOCKS

The two phase, non-overlapping clock $\phi1$ and $\phi2$ generates the IOC's four phase internal clock. $\phi1$ and $\phi2$ 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

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	O 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. 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

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.

DC CHARACTERISTICS

SYMBOL	CHARACTERISTICS	CONDITIONS	MIN.	MAX.	UNITS
V_{IH}	Ø1,Ø2		13	15	V
	D ⟨0-15⟩ BUSY, DONE INTP, DCHP INT SYNC, 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⟩ BUSY, DONE INTP, DCHP INT SYNC, DCH SYNC		-1	+0.8	V
	I/O CLOCK I/O DATA1,I/O DATA2		-0.5	+0.5	V

DC CHARACTERISTICS (CONT'D)

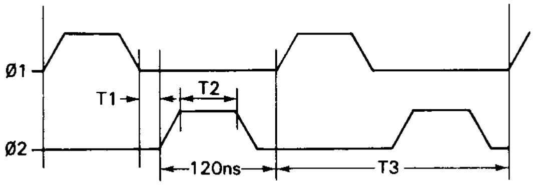
SYMBOL	CHARACTERISTICS	CONDITIONS	MIN.	MAX.	UNITS
I_{IH}	Ø1,Ø2	V_I = 15V		+.01	mA
	D<0-15> $\overline{\text{BUSY}}$, $\overline{\text{DONE}}$ INTP, DCHP $\overline{\text{INT SYNC}}$, $\overline{\text{DCH SYNC}}$	V_I = 4V		+70	µA
	I/O CLOCK I/O DATA1, I/O DATA2	V_I = 2.7V		+70	µA
I_{IL}	φ1,φ2	V_I = 0.8V		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 = 0V		-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>, $\overline{\text{FSTROBE}}$ $\overline{\text{DCHR}}$, $\overline{\text{INTR}}$ 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
I_{GG}	MAX AVERAGE SUPPLY CURRENT	V_{GG} = 14.0 ± 1.0V		20	mA
C_I	Ø1, Ø2 $\overline{\text{BUSY}}$, $\overline{\text{DONE}}$			50	pF
	D<0-15> I/O CLOCK I/O DATA1, I/O DATA2 INTP, DCHP $\overline{\text{INT SYNC}}$, $\overline{\text{DCH SYNC}}$			30	pF

NOTE: Positive current is into the pin.

AC CHARACTERISTICS

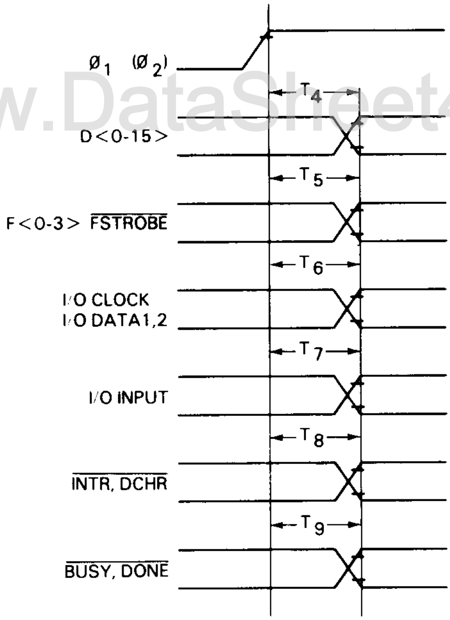
Switching Diagrams

IOC CLOCK TIMING



DG-04023

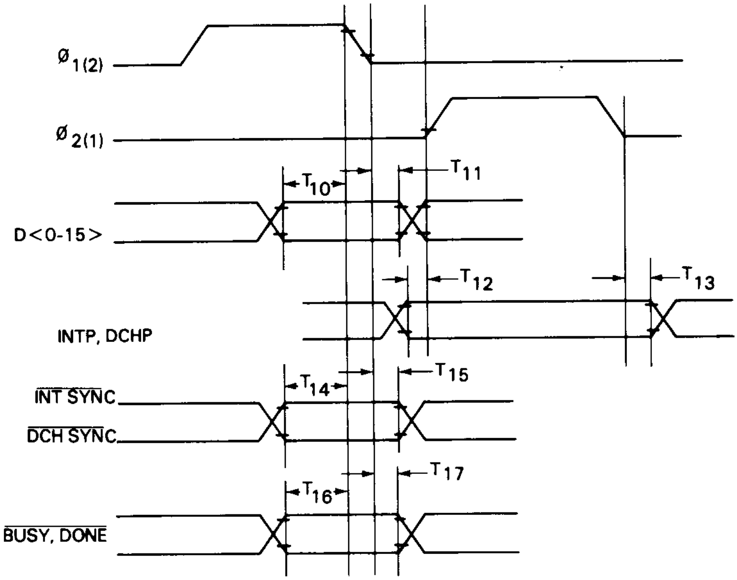
DATA PINS - OUTPUT MODE



NOTE: TIME INTERVALS ARE MEASURED BETWEEN 10% AND/OR 90% POINTS, UNLESS OTHERWISE SPECIFIED.

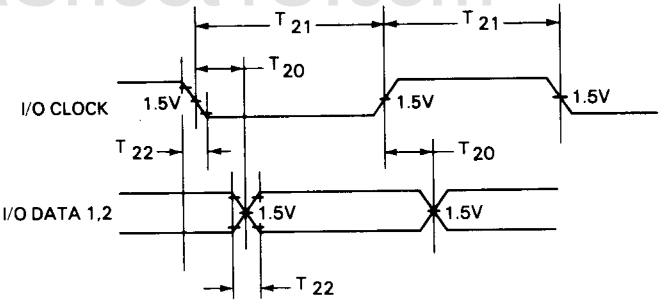
DG-04173

DATA PINS - INPUT MODE



DG-04172

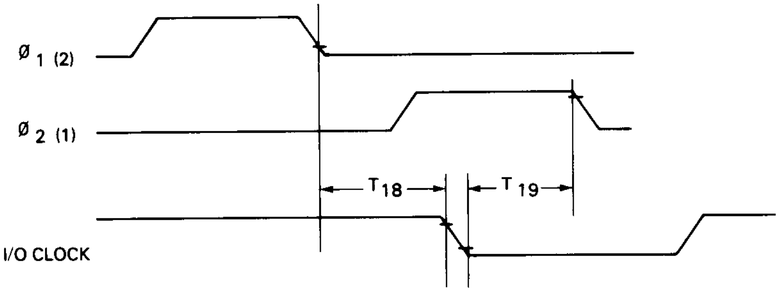
I/O PORT - INPUT MODE



DG-04171

NOTE: TIME INTERVALS ARE MEASURED BETWEEN 10% AND/OR 90% POINTS, UNLESS OTHERWISE SPECIFIED.

I/O CLOCK - OUTPUT MODE



DG-04170

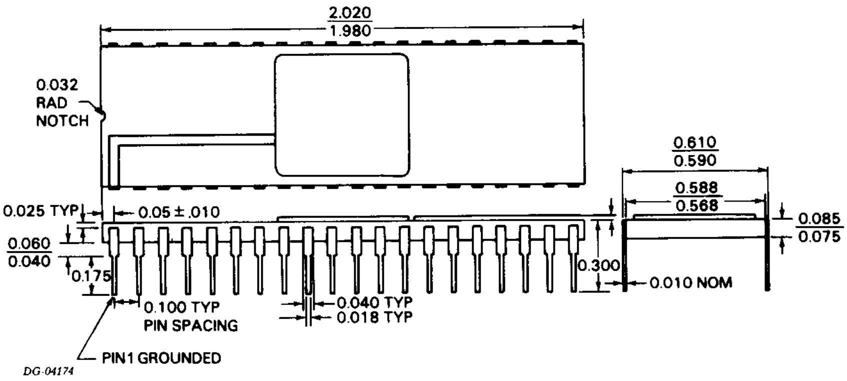
Transition Timing

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

PACKAGE SPECIFICATION



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

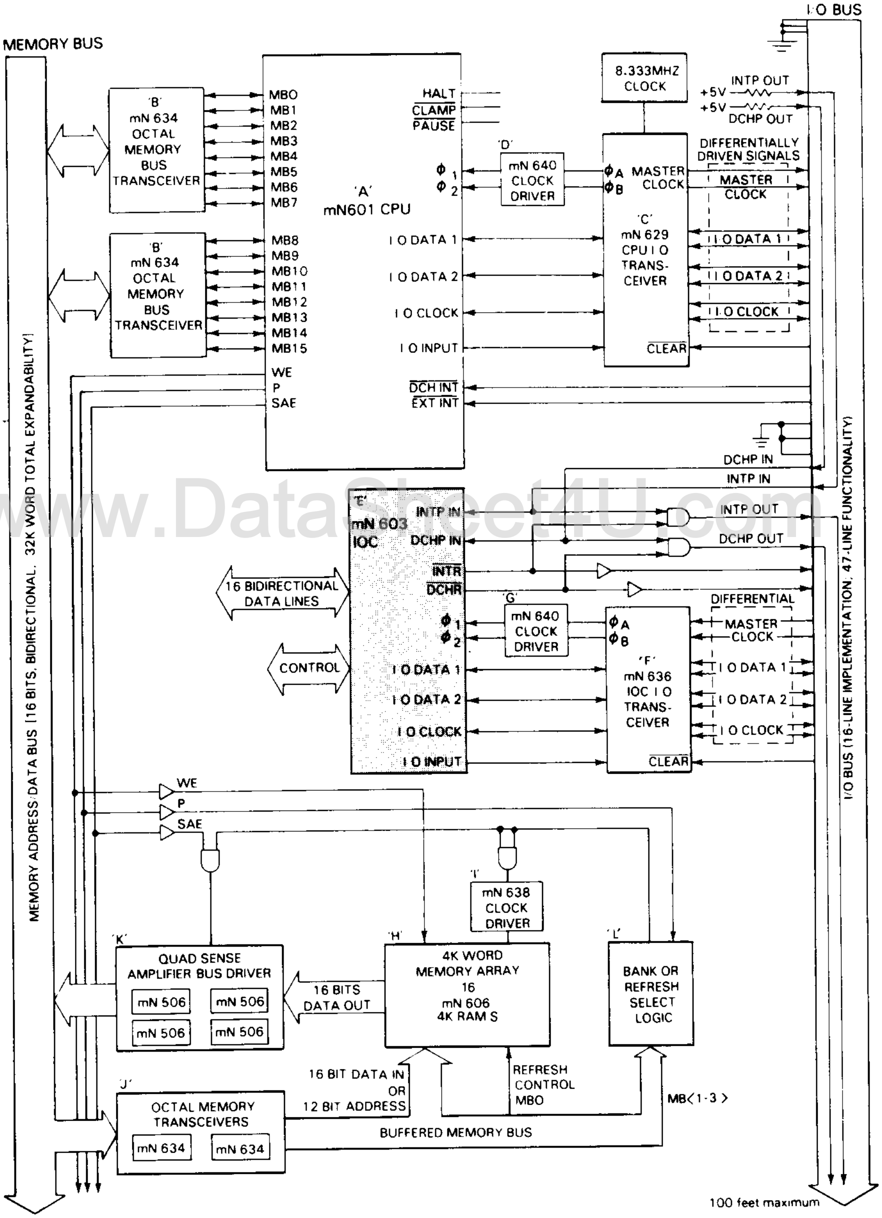
SYNCHRONIZATION TIMING TABLE

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

mN603 I/O CONTROLLER



DG-04326