PRELIMINARY

TEMPO I

INTERFACE REFERENCE MANUAL

TA-1000-469.1

TABLE OF CONTENTS

Figures

Tables

I. INTRODUCTION

1.1 General Description

The TEMPO I is a very high speed, parallel, general-purpose digital computer. It incorporates the latest in technological achievement with organizational features that make it applicable to a wide range of applications. It is equally well suited for the most demanding large scale processing as it is for on-line control applications.

In general its characteristics are:

- Memory cycle time of 900 nanoseconds
- 16-bit word length (parity optional)
- Modular construction
- Memory expandable to 65, 536 words - all addressable
- Flexible addressing modes
- Register oriented - one Accumulator Register, seven Accumulator/Index Registers (eight additional Accumulator/ Index Registers optional). Nine other internal registers.
- Flexible I/O - Asynchronous (hand-shaking) operation. Up to 700, 000 words per second and seven high rate channels optional.
- Software includes Real-Time Fortran IV as well as assembler and program and diagnostic aids.
- Ruggedized mechanical design which is modularly expandable.
- Module size 12-1/4 inches by 17 inches by 24 inches - contains up to 8K of memory, power supply, control processor and controllers.
- Expansion module can contain up to 16K of memory.

In addition to the features listed above, the TEMPO I has many options which allow the system to be tailored for a specific application in an optimum manner.

1.2 Purpose of the Manual

This manual is intended to provide all the necessary information to interface the TEMPO I computer to other system components. Included herein are circuit and timing information as well as design and interface constraints. Cabling and grounding information is also provided.

While a detailed knowledge of the internal working of the TEMPO I system is not necessary in order to interface the equipment it is recommended that the user familiarize himself with the total system concept and machine organization. A brief description of the machine structure is included here but a more detailed treatise may be found in the System Reference Manual.

1.3 Processor Organization

1.3.1 Introduction

The systems organization of the TEMPO I Control Processor (CPU) is shown in Figure 1-1. The CPU consists of a register group, a control section, and arithmetic section and the console and I/O sections.

1.3.2 Register Descriptions

Programmable Registers

Accumulators: There are effectively eight accumulators and seven index registers in the basic system (denoted as X-Registers, X_0 through X_7). Registers X_0 and X_1 are used as A and B registers for most operations.

Program Counter (P): The Program Counter is a register which contains the address of the next instruction word. The programmer may control this register through the use of Branch instructions. Hardware control normally causes automatic increment of this register.

Status Register (S) (Optional): The S-Register contains the various flags set as the result of the execution of instructions. These flags are: Less Than; Greater Than; Equal; Overflow; Master Mode (optional); Program Flags 1, 2, 4, and 8 (optional); and seven High Rate I/O Enable Flags (optional). This register may be loaded from and stored into memory. This is particularly useful for servicing interrupts with subroutines which may alter the state of the control flags.

Register Memory (X) (Optional): These are eight additional general-purpose registers which may be used for accumulators, indexing and high rate I/O.

Interrupt Mask (N) (Optional): This is a 16-bit register used to selectively enable/disable the external interrupts. The register can be loaded from memory or stored into memory.

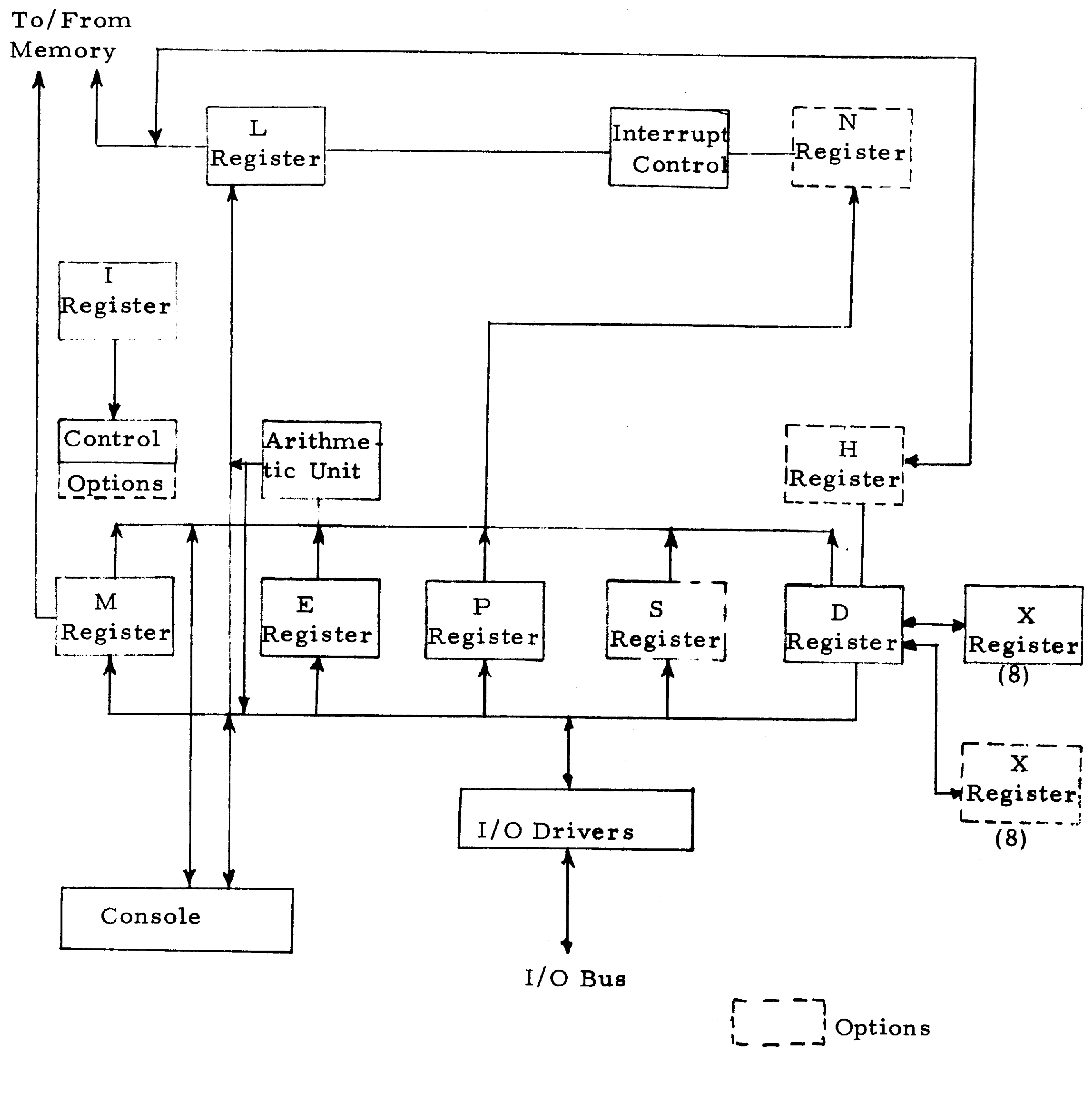


Figure 1-1. Control Processor Organization

Non-Programmable Registers

Instruction Register (I): The I-Register accepts instruction words from memory during the instruction fetch cycle of the machine and holds them for interpretation by the control section during the execute cycle.

M-Register (M): This register contains memory address and operand data. It is used for information transfer to and from the memory. In addition it is used to store intermediate results during the execution of instructions.

L-Register (L): The L-Register holds the effective address as calculated by the address modification sequence of the current instruction. The L-Register is the register through which all memory addresses pass; it drives the memory address bus.

D-Register: A buffer register used to fetch and store data in the X-Registers.

H-Register: A buffer register required for high rate input/output.

E-Register: Used for double length operations in conjunction with the D-Register.

Control Section: The Control Section is the origin of all timing and data path functions for the CPU.

Arithmetic Section: The Arithmetic Section contains all the elements for arithmetic and logical operations on the data and for program modification of instruction words and indexing.

Console Section: This section contains the necessary switches and gating elements to transfer data from the console panel into the CPU. All programmable registers and sense functions are accessible via the console.

Input/Output Section: The transmission of control and data words between the CPU and external devices is accomplished by the I/O section. Each peripheral device option includes a controller which controls one or more devices and interfaces with the peripheral bus. The general characteristics of the Input/Output interface are:

- 64 device addresses available
- Asynchronous
- Four standard interrupts
- Interrupts expandable to a maximum of 256
- Single word or block transfer
- Up to 700, 000 words per second
- DTL compatible

II. INPUT/OUTPUT OPERATIONS

2.1 General

The input/output interface of the Control Processor is shown in Figure 2.1. The communication is between the Control Processor and the various devices. A standardized interface (the I/O bus) is the key system concept. This bus provides a simple, regular interface for attaching various I/O devices to the system. The standardization is carried through the logic, circuit, and mechanical designs.

Between the standard I/O bus and the various devices with their associated electronics (device oriented electronics - DOE) are device controllers. The purpose of these controllers is to adapt the diverse requirements of the devices to the standard interface (the I/O bus). Note that controllers may handle multiple devices. Many of these controllers are standard products of Tempo Computers, Inc.; those that are not can be easily designed as special products due to the straightforward logic of the I/O bus.

The I/O concept is that of asynchronous priority interrupts. A particular peripheral or set of peripherals (devices and associated controllers) is started. No further attention is required of the CPU, thereby allowing processing to proceed until the device has come up to speed and either has data ready for input or is ready to accept information. Upon reaching this ready state the controller activates its associated interrupt without regard to either the state of the CPU or other controllers. The priority interrupt logic of the CPU will choose the appropriate time to interrupt the processing (generally at the end of the current instruction in process) and will choose the highest priority interrupt for service.

Actual transfer of data is done either between the A-Register (Word Transfer Command) or the memory (High Rate I/O feature) and the peripheral upon acknowledgment of the interrupt.

Three basic classes of instructions are provided: 1) Execute Device Function (examples: start reader, set read mode, stop write); 2) Word Transfer (transfer one word into A-Register from peripheral, transfer one word from A-Register to peripheral); 3) Status Examination (read a word of status information from peripheral to A-Register, interrogate interrupt status of common interrupt line). Appendix A contains a complete description of I/O instructions as well as typical I/O rates on the standard bus.

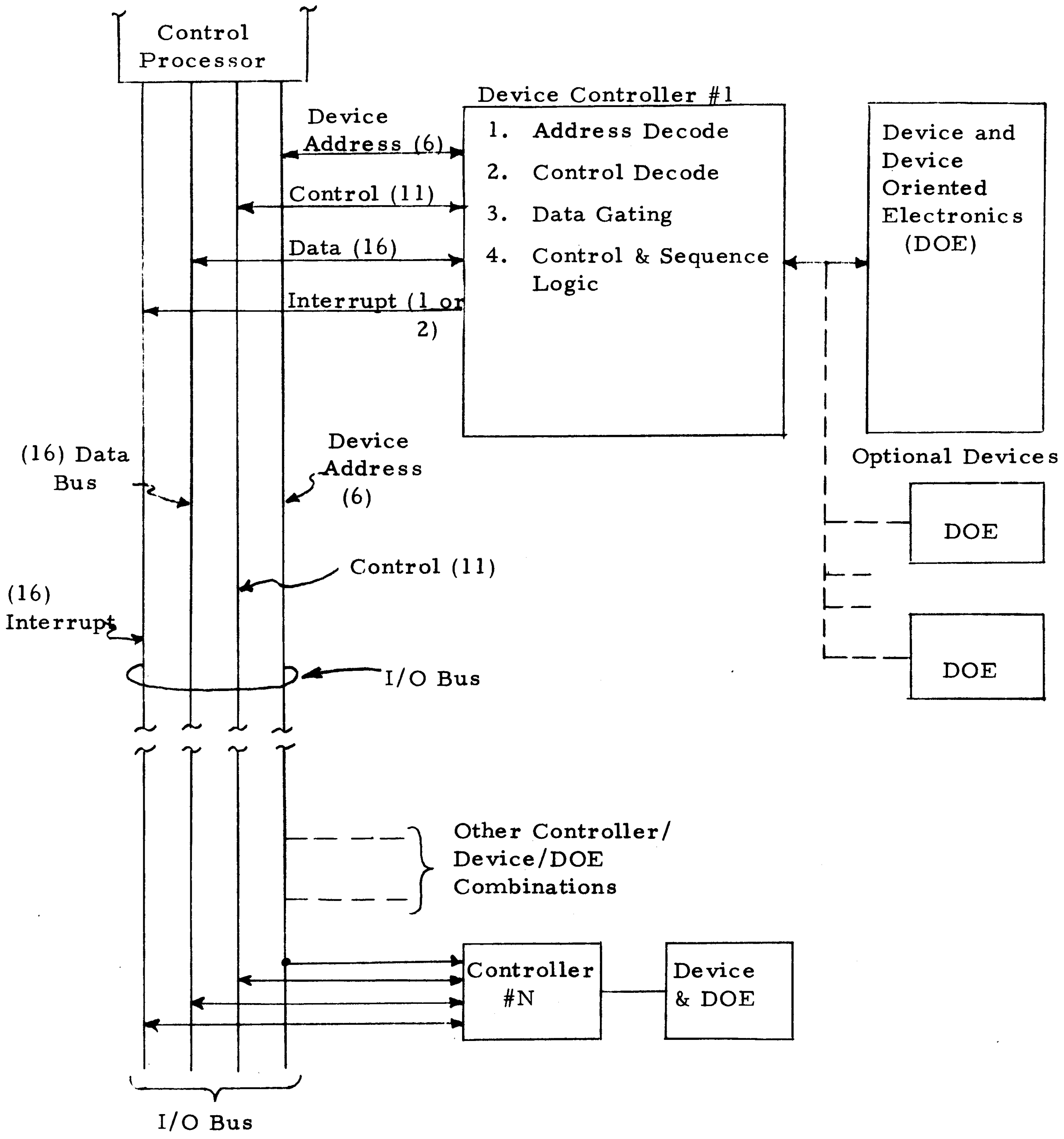
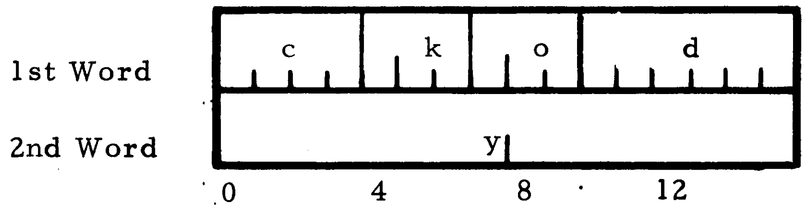


Figure 2-1. Input/Output Interface

2.2 Input/Output Commands

General

In order to properly communicate I/O commands between the CPU and peripheral controllers, both units require information about the instruction to be performed. The format of all the I/O instructions is as follows:



The c-field is used only by the CPU to specify that the instruction is an I/O instruction; all I/O instructions have the c-field equal to 15.

The k-field is used by both the CPU and the controllers to determine the sequence which each must execute for the various commands. See explanation of each below. Note that, in general, the k-field describes a class of command which defines the CPU sequencing required. The peripheral needs the information contained in the k-field but it is not sufficient to describe the operation for the controller. In addition, it needs the information in the o-field.

The o-field contains information which expands the meaning of the k-field for the selected controller. The CPU does not use this information. Each controller interprets the o-field in its own way. In other words, a card reader controller would interpret an o-field code differently than a magnetic tape controller would upon receiving the same code. The A-Register data is also gated onto the I/O bus data lines during the execution of EDF commands and is used to expand the orders to the peripheral controllers (if the controller is designed to use this information).

The d-field is used only by the controllers and provides the device selection address.

The y-field is used only by the CPU as the branch address in the event that an I/O command is rejected. A timer in the CPU control logic is started when the I/O command is sent out. If a response is not received from the controller (EKO) before the timer has run out the command is defined as being rejected. The y-field is used to execute a branch. The program counter is stored in the memory location addressed by y and the next instruction is fetched from y + 1.

There are three basic classes of I/O commands: Execute Function, Word Transfer, and Status Examination. The next sections describe these commands in detail.

Execute Functions

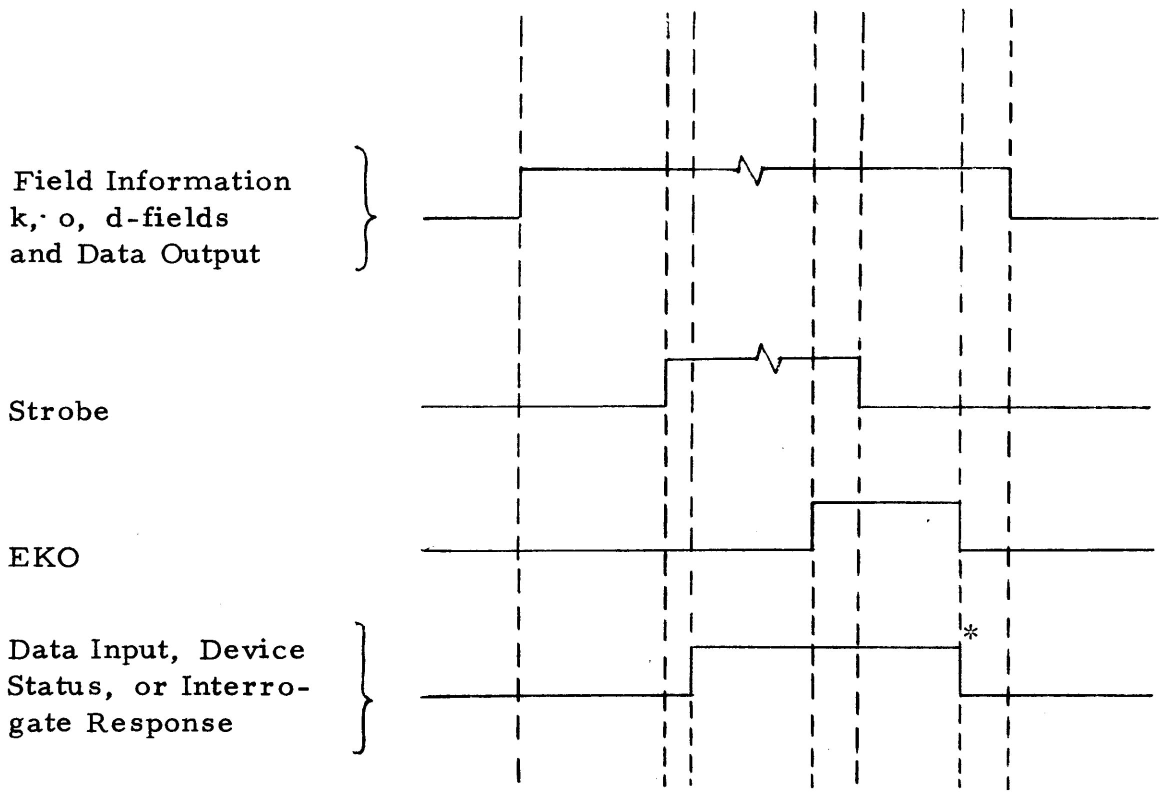
These are the Execute Device Function (EDF) commands which cause a controller and associated device to start or stop or go into a new mode. As with most I/O commands, failure to receive an EKO signal at the CPU within the timer limit will result in a reject branch.

Figure 2. 2 shows the timing sequence for these commands. The EDF sequence proceeds as follows: a) the k, o, and d-fields and A-Register (data output) information is gated into the appropriate I/O busses; b) after an appropriate delay the strobe line is made true and the reject timer is started; c) when the selected (d-field code) controller receives the strobe it transfers the appropriate information into its state register and sends the EKO pulse. When the CPU receives the EKO pulse it drops the k, o, and d-field, data output information and the strobe line. The turn off of these lines happens as soon as the EKO is received but an overlap is shown to indicate the line and driver/receiver delays. The EDF command is now complete.

Word Transfer

This class consists of Word Transfer In (WTI) and Word Transfer Out (WTO) commands. The WTO command causes 16-bit data to be transferred from the A-Register to the selected controller. The WTI command transfers 16-bits of data from the controller to the A-Register. Some controllers do not supply a full 16-bits (example: card reader 12-bits) but this is a function of the gating of the controller. The WTI and WTO commands transfer in whatever is put on the bus and always gates the full 16-bits of the A-Register onto the data lines. The sequence of operations for the WTO is the same as that described for the EDF command. For the WTI command the sequence is the same with the following exceptions: a) the A-Register is not gated onto the data bus along with the field information; b) the information from the controller is gated onto the data lines just prior to the EKO pulse and remains true throughout the duration of EKO.

During the High Rate I/O sequence a simulated WTO and WTI is executed. Viewed from the controller the sequences are identical to those described for WTI/WTO; however, the instructions are not inserted into the I-Register and the data is transferred to or from memory rather than the A-Register. High Rate I/O operations do not affect the CPU status and control states.



*Applicable only to RDS, WTI and ICI commands.

Figure 2-2. Timing Sequence for all I/O Commands

Status Examination

This class of I/O commands include: Request Device Status (RDS), and Interrogate Common Interrupts (ICI). All of these commands are involved in examining the status of selected controllers. It should be noted that the execution of any Status Examination command does not reset the status flags in the controllers. Execution of an EDF command does reset the status flags.

The RDS command causes a word (up to 16-bits) of status information to be transferred from the selected controller (d-field) to the A-Register. The timing sequence, shown in Figure 2.2, is the same as that described for the WTI instruction. The reject of the command will be detected and the branch will be executed to the y-field address as previously described.

The Interrogate Common Interrupts (ICI) command causes a word of information to be transferred from the data lines to the A-Register. The ICI command applies to multiple controllers which are sharing a common interrupt. Each controller on the common interrupt line recognizes the interrogate by the code in the k-field. If the controller requires service it will make true at EKO time the data line corresponding to its assigned device code. The ICI command does not reset the interrupt.

2.3 I/O Bus

The I/O bus is a true functional bus in that only one of the controllers may be communicating at a time. The bus may be classified into four types of signals: address, control, data and interrupts. These lines and their interface into the CPU are shown in Figure 2.3.

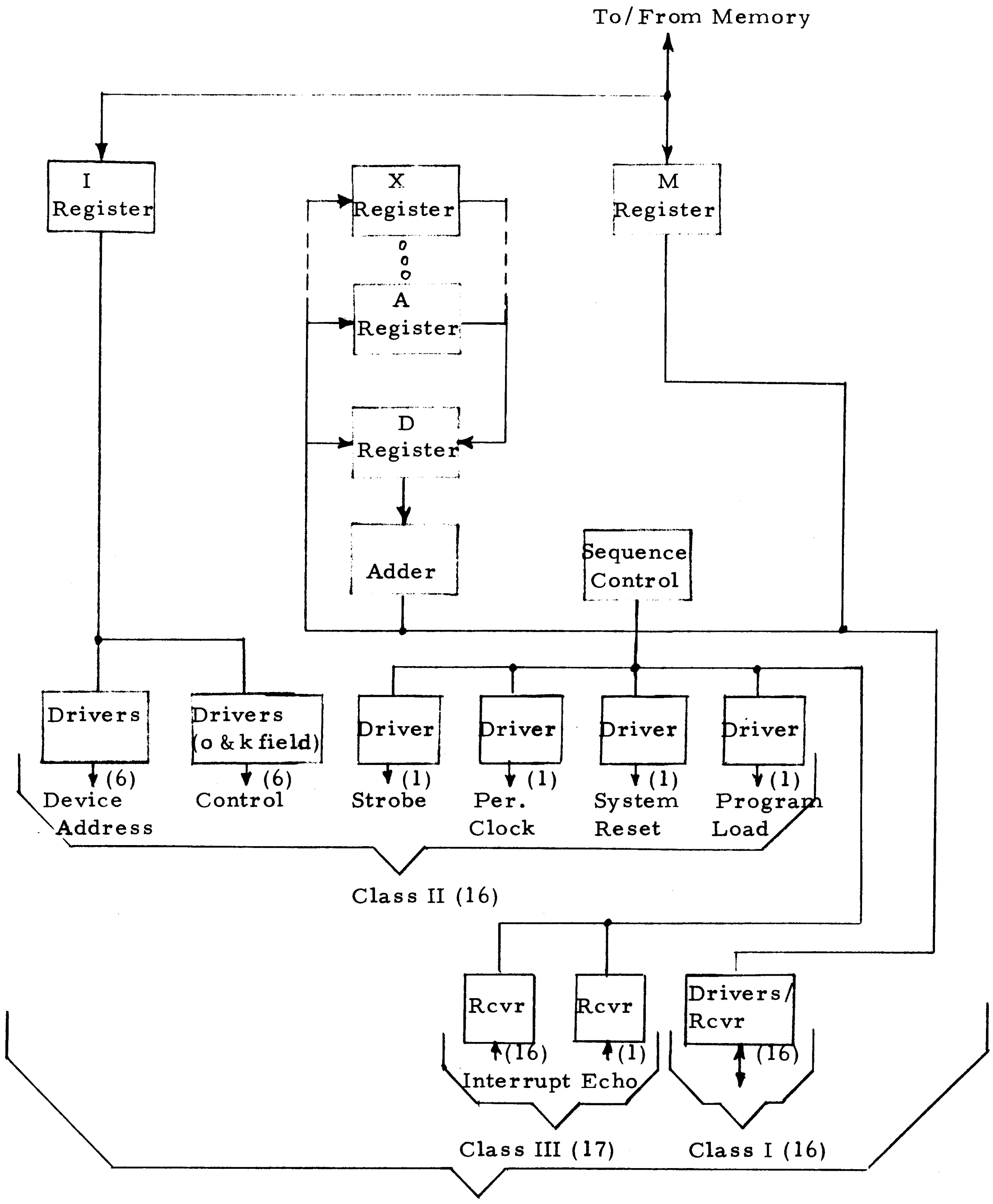
Device Address Lines (Output)

Six lines transmit a binary code of 0 to 63 to the controllers for selection of device and/or controller. These lines are used in conjunction with the strobe line for all I/O commands. Only that controller whose device code is true will respond; all others remain quiescent.

Control Lines (Input and Output)

These lines provide control signals between the CPU and the peripheral controllers. They are:

a) <u>o-field</u>: Three binary coded lines which describe the order being sent to the selected peripheral controller. These lines are interpreted by the particular controller only if it has been selected by the address lines having the unique address associated with it on the bus at strobe time. Note that the A-Register is also made



I/O Bus - 49 Lines

Class I - Bidirectional
Class II - Output
Class III - Input

Figure 2-3. Block Diagram of I/O Structure

available on the data lines to expand the order set during Execute Device Function (EDF) commands.

b) <u>k-field:</u> (output) Three binary coded lines which describe the class of instruction to be performed by the selected peripherals. These lines may be interpreted by the peripheral controller only during strobe time and only if the device address contains the unique address associated with the controller.

k-field	code	
000	EDF	Execute Device Function
001	WTO	Word Transfer Out
010	WTI	Word Transfer In
011	RDS	Request Device Status
100	ICI	Interrogate Common Interrupts
101		Reserve
110		Reserve
111		Reserve

Note that all controllers must fully decode these lines.

c) <u>Strobe:</u> (output) This line is used by the controllers to examine various output lines (to controllers) at a time when their condition has settled to a valid state. The output lines pertinent to strobe are: data output, control and address lines.

d) <u>Echo (EKO):</u> (input) This line is sent to the CPU from the peripheral controllers in response to the strobe. This signal notifies the CPU that the selected controller has received a valid order (k-field and/or A-Register) and is responding. The strobe is terminated upon receipt of the EKO pulse. In addition, the EKO is used to strobe input data from the I/O bus into the CPU during Word Transfer In commands.

e) <u>Program Load:</u> (output) This line is effective only if the Program Load option is installed. It signals the selected peripheral, when initiated by the Program Load action from the console, to execute a special Read One Record (EDF) sequence. This sequence causes a single physical record to be read. Each time a word or character is ready, the associated interrupt line is raised and a transfer takes place. At the end of the physical record the device is halted and the controller returns to the quiescent state.

f) System Reset: (output) This line causes all peripheral controllers to cease any current activity and return to an idle state. All status flags (except those requiring operator attention) are reset, all mechanical devices come to a halt and the controllers become ready to accept commands (unless operator intervention is required). The System Reset switch on the console initiates this reset pulse.

g) Peripheral Clock: (output) This line carries a free running 4 MHz clock for use in the logic design of peripheral controllers. It is not in synchronization with the CPU clock, nor can it be guaranteed to be in phase with itself in different controllers. It is merely provided to avoid duplication of clock oscillators in the various peripherals.

Data Lines (Input/Output)

Sixteen bidirectional lines which transmit information either from the selected controller to the A-Register (I/O commands) or M-Register (High Rate I/O) or from the A or M-Registers to the selected controller.

For byte transfers the data input to the computer must appear on lines 8 through 15. All data output from the CPU will appear on lines 0 through 7.

Interrupts (Input)

Sixteen lines which transmit interrupt signals from the controllers to the CPU. Generally only one controller attaches to each interrupt line but up to 16 may be "ORed" to any interrupt line.

2.4 Interrupt Description

General

The interrupt design has the following general characteristics: a) asynchronous, b) multiple priority level and c) maskable.

The interrupts are asynchronous in that they may be set by their respective controllers or control logic without regard to the CPU clock or control state or the state of any other controller. Generally the controller has a flag which is made true when the controller requires service, timed by the controller clock. The flag is reset when the service has been provided by the CPU via an I/O command.

There are various "levels" of priority which are generally established by the classes: a) internal, b) system, c) high-rate, and d) external.

The priority logic of the system selects the appropriate interrupt for servicing when two or more interrupts are pending. In addition, this logic permits higher level interrupts to interrupt lower level interrupts in process of being serviced.

All external interrupts are maskable. By various means they may be selectively disabled under program control. This masking operation does not reset the flag in the controller, it merely prevents the priority logic from considering the disabled interrupt at priority selection time. Interrupts which are disabled (masked) and are set true by the controller will be "remembered" and serviced when the mask is removed.

Number

There are six optional internal interrupts. These are in order of priority:

1. Power Fail Restart
2. Parity Error
3. Instruction Trap
4. Memory Protect Error
5. Privileged Instruction Error
6. System

The four standard interrupts are expandable in groups of four to a maximum of 16. If the High-Rate I/O option is selected the first four interrupts must be in the machine as the four standard interrupts are numbers 12 through 15. The priority of the interrupt is related to its number with 0 having the highest priority*. A maximum of seven high rate channels may be obtained. The interrupt mask (N-Register) is optional in groups of four.

Rates

The rate at which the external and internal interrupts may be processed depends on the length of the subroutine required to process each interrupt. Because the High-Rate interrupts are processed by a hardware generated sequence they are not program dependent for rate. Each High-Rate interrupt can be serviced at a maximum of 700K words/second. The internal and system interrupts are primarily for error processing routines and a typical rate is not pertinent.

*Each of these interrupts may be used as "common" interrupts. Up to 16 device controllers may "OR" onto the same interrupt line and the Interrogate Common Interrupts (ICI) command is used to bring their interrupt status into the CPU for analysis. The device code field of all I/O instructions can be used to address up to 64 devices, while it is possible to attach 16 x 16 or 256 external interrupts to the system. Where greater than 64 devices must be addressed, the A-Register may also be used to provide additional selection information to controllers which select multiple devices.

Priority Assignments

The priority for servicing of interrupts is determined, as mentioned previously, by the number or name of the interrupt line. Associated with the interrupt number or name are two other addresses: interrupt location (address) and device address. The interrupt location is wired into the logic of the CPU while the device address and priority number may be altered in the controller by means of a plugboard. This plugboard provides reassignment of priority without rewiring. It is not necessary that the interrupt number be the same as the device number but for general consistency and use of standard I/O subroutines they will correspond in the standard system. Deviations from the assignments shown in Table 2.1 may be accommodated on special request.

The "common" interrupts which are those devices sharing a common external interrupt can be considered as having a level of priority within a level. Their priority is determined first by the relative position of the shared interrupt and second by the priority established by the subroutine which services the shared interrupt. For those controllers which can share an interrupt a plugboard arrangement is provided in the controller to select which data line (implies the priority to the software) is to respond to the ICI I/O command. Device codes for "common" interrupts are assigned 16_{10} through 63_{10}.

The priority of the High-Rate interrupts is the same as the correspondingly numbered external interrupts - 0 through 7. Priority within the High-Rate group is consistent with the other external interrupt numbering scheme in that the highest priority is 0 and the lowest is 7. The device addresses for the High-Rate interrupts carry the same number as the High-Rate interrupt. When the count associated with the High-Rate hardware sequence overflows an external interrupt is set and the High-Rate interrupt is reset. The external interrupt thus set carries the same number as that of the High-Rate interrupt.

<u>External Interrupts</u>: Assume that the machine is not in a Halt state, i.e. it is in either a Run or Wait state.

The operation of the external interrupts is as follows:

a) The controller interrupt logic asynchronously turns on a flag in the controller to indicate that it requires service. This service may be a requirement to transfer data (in or out), to indicate an error condition, or merely to indicate the occurrence of an event (button depressed, relay closure, timer clock, etc.). In any event, the flag is set by a relatively short pulse and is to remain set until a response from the CPU is received.

Interrupt Location (Memory) (Hex)	Interrupt Name/Number -	Device Address (Decimal)	Remarks
000A	PFR	0	Power Fail Restart (Internal)
000B	PAER	1	Parity Error (Internal)
000C	INTR	2	Instruction Trap (Internal)
000D	MEMP	3	Memory Protect Error
000E	PRIE	4	Privileged Instruction Error
000F	SYS	5	System Component (Internal)
0010	0**	0	External Interrupt
0011	1**	1	External Interrupt
0012	2**	2	External Interrupt
0013	3**	3	External Interrupt
0014	4**	4	External Interrupt
0015	5**	5	External Interrupt
0016	6**	6	External Interrupt
0017	7	7	External Interrupt
0018	8	8	External Interrupt
0019	9	9	External Interrupt
001A	10	10	External Interrupt
001B	11	11	External Interrupt
001C	12*	12	External Interrupt
001D	13*	13	External Interrupt
001E	14*	14	TTY/CPU Channel (External)
001F	15*	15	External Interrupt

Table 2.1 Priority Assignments (from highest to lowest: top to bottom)

*These are the four standard interrupts.

**These external interrupts used by High Rate I/O also.

b) At an appropriate time, which is generally at the completion of an instruction execution and just prior to the next instruction fetch, the CPU priority logic will examine the state of all interrupt input lines.

All interrupt lines in the true state will be set into the CPU interrupt flip-flops. See Figure 2.4 which is a simplified logic diagram of the flip-flops and logic involved per external interrupt level. The priority logic will select the highest priority interrupt which is true and is not masked out (disabled) by the corresponding N-Register bit. If the High-Rate Interrupt Flag is set, the priority logic will generate the selected X-Register pair address and the High-Rate I/O sequencer will be started. If the High-Rate I/O flag is not set the priority logic will generate the memory address of the interrupt location for the selected interrupt and the interrupt control logic will be started. Of course, if no interrupts are true or if all which are true are masked out, no interrupt sequence will occur and the instruction fetch cycle will take place.

c) If the High-Rate I/O flag is not set, the interrupt will be fetched from memory. The contents of this memory cell will be used to address a cell in memory into which the program counter is stored. The address used to store the program counter is incremented by one and is set into the program counter. During this hardware sequence no interrupt sample is allowed until after the new program counter value has been set.

d) Prior to the next Instruction Fetch cycle, another examination of the interrupt lines will be made. If an interrupt of higher priority than the highest "current" (previously selected) interrupt is found then step c is repeated.

e) If no higher interrupt is found the interrupt servicing subroutine proceeds. At the end of each instruction execution the subroutine looks for an interrupt of higher priority than the one currently selected.

f) At some point in the interrupt subroutine an I/O command is sent to the controller, resetting the controller interrupt flip-flop. This does not reset the CPU interrupt flip-flop, however, and further processing of the interrupt may proceed (with interrupts to higher levels possibly occuring). Eventually the subroutine for the interrupt is concluded by executing a Branch and Reset External Interrupt (BRE) indirectly through the memory location where the old program counter value was stored. Execution of this BRE instruction resets the CPU interrupt flip-flop specified in the BRE which is generally that of the current interrupt.

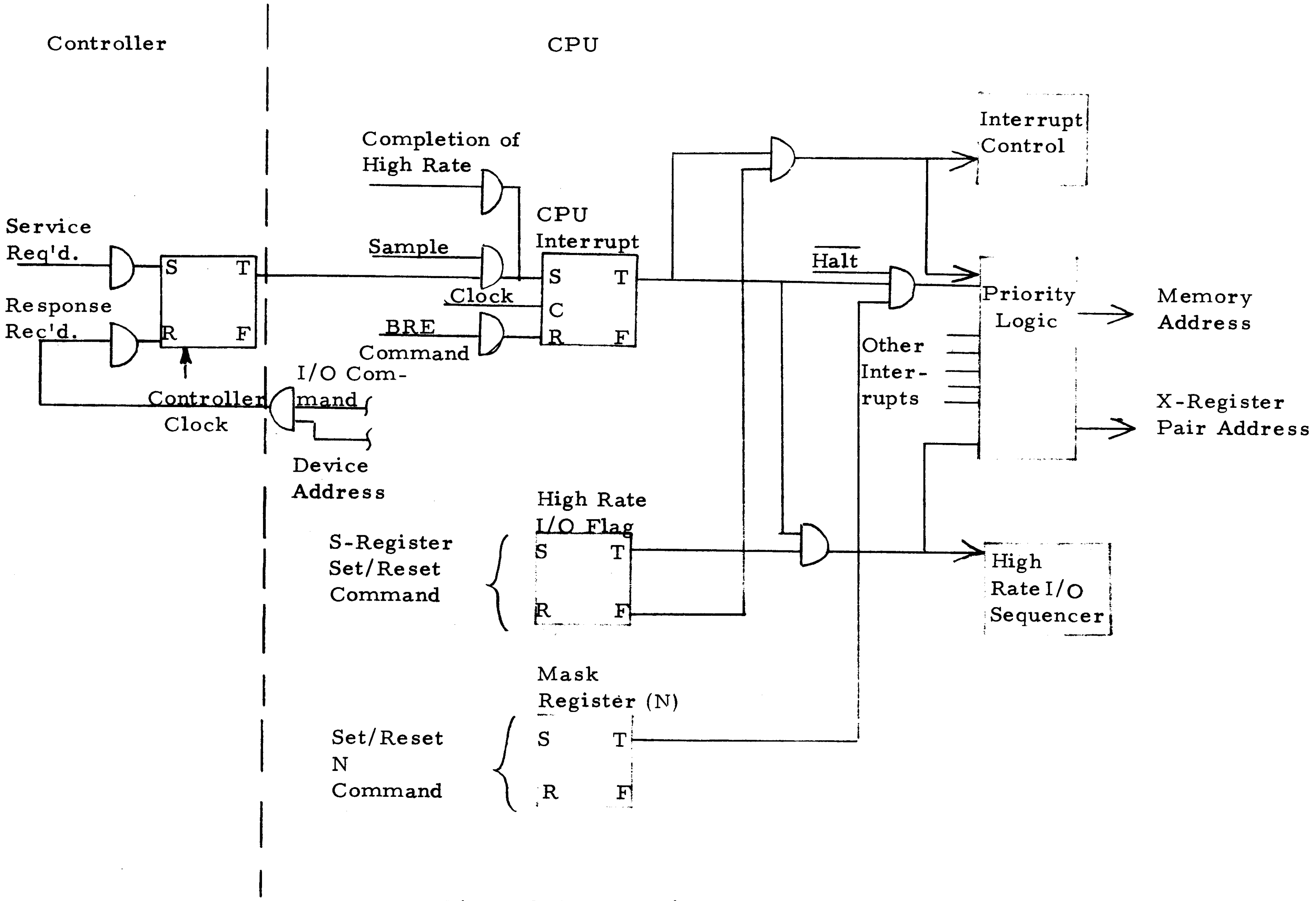


Figure 2-4. Simplified Logic for External Interrupt

Common Interrupts

The same hardware previously described for the external interrupts is used for the common interrupts; the only difference being that the controller interrupts are "ORed" onto the same interrupt line. Thus any of the controllers can hold the interrupt line true regardless of the state of the other controllers.

The sampling of interrupt I/O bus lines, selecting of highest priority and subroutine entry is identical to that described for external interrupts above. However, the subroutine must be written to include an examination of interrupt status and a selection of the highest priority device requesting service through the common interrupt. The ICI instruction is used to bring the interrupt status of all controllers sharing the common interrupt into the A-Register. Generally a shift left and test for negative sign loop will be used to find the higher priority device. Having stored the shift count which produced the first "one" in the A-Register the subroutine can use this to address the highest priority controller.

As in the external interrupt description, the servicing of the addressed controller will reset its interrupt flip-flop and the use of the BRE command to terminate the subroutine will reset the CPU interrupt flip-flop. Of course, if other common interrupts are pending, the CPU interrupt flip-flop will be set again at the completion of the BRE command and therefore will be considered by the priority selection logic prior to executing another instruction fetch cycle.

III. I/O BUS DESCRIPTION

3.1 Introduction

This section describes the I/O bus technique employed in the TEMPO I system. The I/O bus provides communication between the peripheral controllers and the CPU. It is restricted to servicing no more than 16 controllers all of which must be located such that the physical bus does not exceed 5 feet in length. Note that up to eight controllers may be packaged within 12-1/4 inches of rack space using Tempo standard expansion chassis. If devices must be located further than 5 feet, then the High-Drive I/O Bus option should be used.

A generalized block diagram of the I/O bus configuration is shown in Figure 3.1. There are, in reality, two busses; an internal and an external bus as shown in the figure. The internal bus in the CPU is electrically identical to the internal bus in the expansion chassis and all controller designs are made to interface with this bus. The external bus is designed to provide higher noise rejection and drive capability than the internal bus.

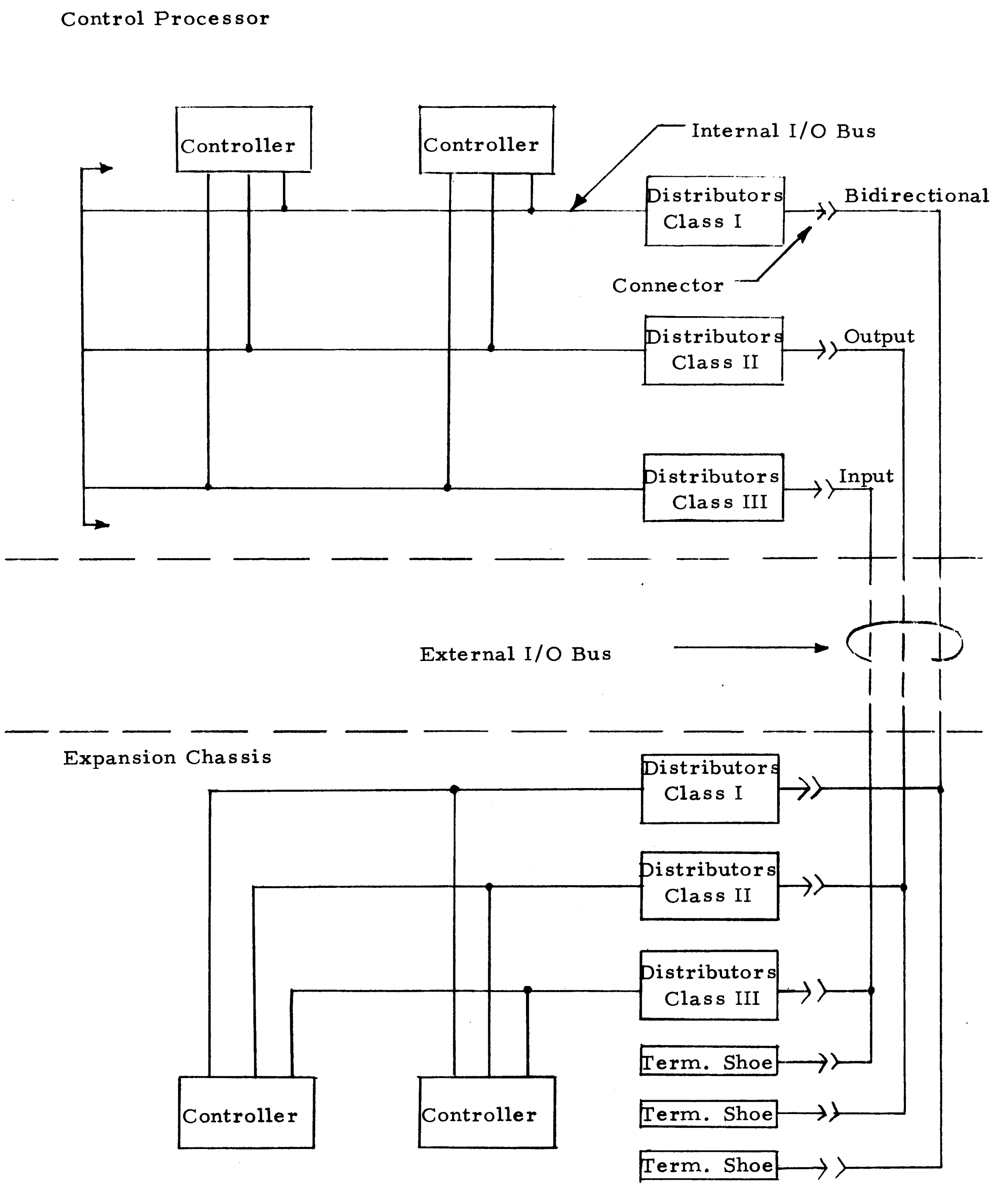


Figure 3-1. TEMPO I I/O Bus Configuration

The transition from the internal to the external bus is accomplished by the use of distributors whose design is a function of the type or class of line. The classes are:

Class I

These lines carry data both out of and into the CPU. A driver and receiver are both connected to the line at the CPU and in the expansion chassis. The line is terminated in the circuit within the CPU and by a termination shoe located at the end of the line.

Class II

These lines are output lines which have a driver at the CPU end and a receiver in each expansion chassis. The bus is terminated by the termination shoe at the end of the line.

Class III

These lines are input lines which have a receiver at the CPU and a driver at the expansion chassis. These lines are terminated the same as the Class I lines.

In order to have a standard interface the distributors are included in the CPU and in each expansion chassis. Thus, all controller design is to the requirements of the internal bus. The internal bus is the only one described in detail in this section. Each distributor is capable of interfacing with up to eight loads in each direction, thus eight expansion chassis with eight controllers each is the limit of expansion as long as the physical restraints mentioned earlier are observed.

If it is necessary to use controllers outside the expansion chassis then the distributor circuit (or its equivalent) should be used as the input circuit to the controller.

Table 3. 1 gives the list of lines in the I/O bus and their associated mnemonics by class.

3. 2 I/O Timing

3. 2. 1 Introduction

In the descriptions to follow the worst case conditions and the constraints which the controller must satisfy will be given. Figure 2. 2 has shown the general timing for I/O functions.

Class	Mnemonic	Description	
I	DB00-	Input/Output Data Bit 00	H*
(16)	.	.	
	.	.	
	.	.	
	DB15-	Input/Output Data Bit 15	L
	DA00-	Device Address Bit 00	H
	.	.	
	.	.	
	.	.	
	DA05-	Device Address Bit 05	L
II	KF0-	K-Field Bit 0	H
	KF1-	K-Field Bit 1	
	KF2-	K-Field Bit 2	L
(16)	OF0-	O-Field Bit 0	H
	OF1-	O-Field Bit 1	
	OF2-	O-Field Bit 2	L
	STRB-	Strobe	
	PCLK-	Peripheral Clock	
	SRST-	System Reset	
	PRLD-	Program Load	
III	INT00-	Interrupt 0	
(17)	.	.	
	.	.	
	.	.	
	INT15-	Interrupt 15	
	EKO	Echo	

Table 3.1 I/O Bus Lines and Mnemonics

*H = High Order Bit of the Field
L = Low Order Bit of the Field

The basic timing is derived in the CPU and consists of a Synchronizing Clock (an 8 megahertz square wave) and a Peripheral Clock (a 4 megahertz square wave) derived from the Synchronizing Clock. Both phases of the Peripheral Clock are derived. The negative of the Peripheral Clock appears in the I/O bus for use by the controller (if desired). The Tx pulse (shown in Figure 3. 2a) initiates the I/O sequence.

All I/O transfers are started the same way, that is, data is placed on the Control Lines at t_2 and the Synchronizing Clock and Peripheral Clock are used to logically generate the strobe which is a bistable output.

The worst case timing sequence occurs when the Tx pulse coincides with the Synchronizing Clock and the Peripheral Clock as shown in Figure 3. 2a.

Word Transfer Out

The Word Transfer Out (WTO) sequence in the CPU is as follows:

1. Information appears on control bus.
2. Data is placed on data bus.
3. The strobe line is brought true.

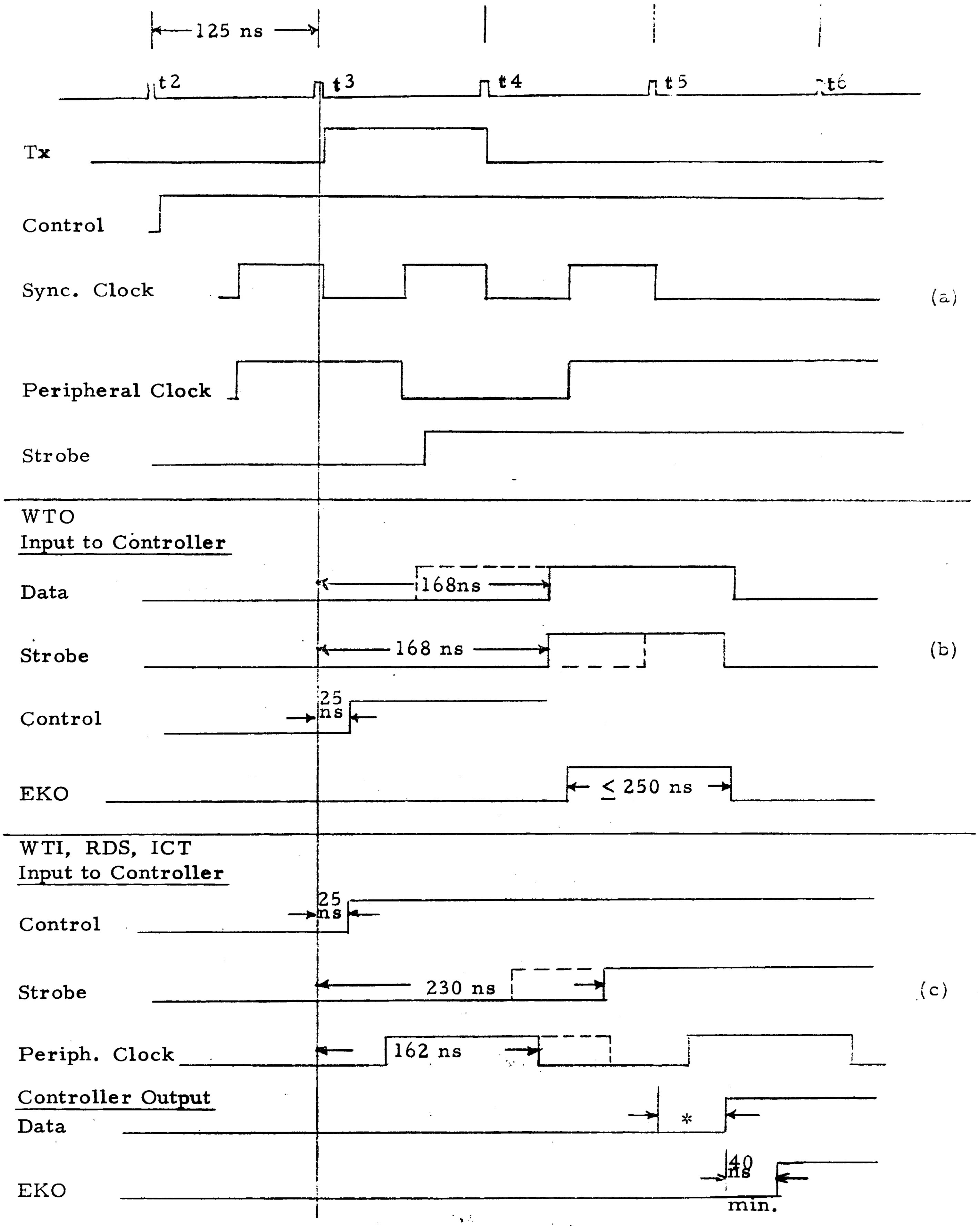
The controller responds as follows:

1. Decodes the control lines to achieve device selection and function interpretation.
2. If selected the strobe is used to transfer data into the controller and generates an echo pulse (EKO). The EKO pulse should not have a pulse width in excess of 250 ns. The Peripheral Clock may be used for aid in generating the EKO.

The sequence ends in the CPU by:

1. The strobe line is brought false by the EKO pulse.
2. The data and control lines are brought false.

Note that in all I/O operations all lines are dropped with EKO.



*Data should be placed on-line as soon after receipt of Strobe as possible since any delay adds directly to the High Rate timing.

Figure 3.2. I/O Timing

Figure 3.2b shows the timing requirements for WTO. Note that the Control signals arrive at the controller a minimum of 125 nanoseconds prior to Strobe time, thus, all decoding (Device Address and Function) must be accomplished within that period so that the controller can select the Strobe for proper routing. This is the worst case timing for the Control signals.

Under worst case conditions the Data and Strobe could arrive simultaneously, therefore, it is necessary that the Strobe have a longer timing path, within the controller, than the data path in order to assure reliable gating. (See Figure 3.6 for a recommended controller input section design.) Upon receipt of the Strobe the controller must place an EKO pulse on the line to signal the CPU that the device has accepted the data. The timing of the EKO is not critical in this mode.

For maximum data transfer rates (as in High-Rate I/O) care should be taken in the timing paths so as not to delay the strobing of the data into the controller.

Word Transfer In, Request Device Status, Interrogate Common Interrupts

For Word Transfer In (WTI), Request Device Status (RDS) and Interrogate Common Interrupts (ICI), the following sequence takes place:

The CPU

1. Places the necessary information on the control lines.
2. The Strobe line is brought true.

The Controller responds

1. Decodes the Control lines to provide device selection and function interpretation.
2. If selected, places data on data lines when Strobe is received.
3. Generates EKO.

The CPU then

1. Gates data into CPU upon receipt of EKO.
2. Drops Strobe line and Control lines.

The Controller

1. Drops data lines with trailing edge of EKO.

Figure 3.2c shows the timing for WTI, RDS and ICI. In this instance it is necessary that the data be placed on the bus in the minimum possible time after Strobe is received so that the EKO and data are in the proper phase at the CPU. See Figure 3.6 for the suggested circuit for the controller.

Note that the Data should be placed on the line as soon as possible after receipt of Strobe since the delay adds directly to the turn-around time for High-Rate I/O. The EKO must be delayed from Data for reliable gating at the CPU.

It should be emphasized that the timing given here is for the achieving of the shortest cycle times in the I/O system. If slower rates are acceptable then it is only necessary to maintain the phasing shown in Figure 2.2.

If the CPU does not receive the EKO it will, as described in Section 2, execute a branch at the end of two microseconds from the time the I/O sequence starts.

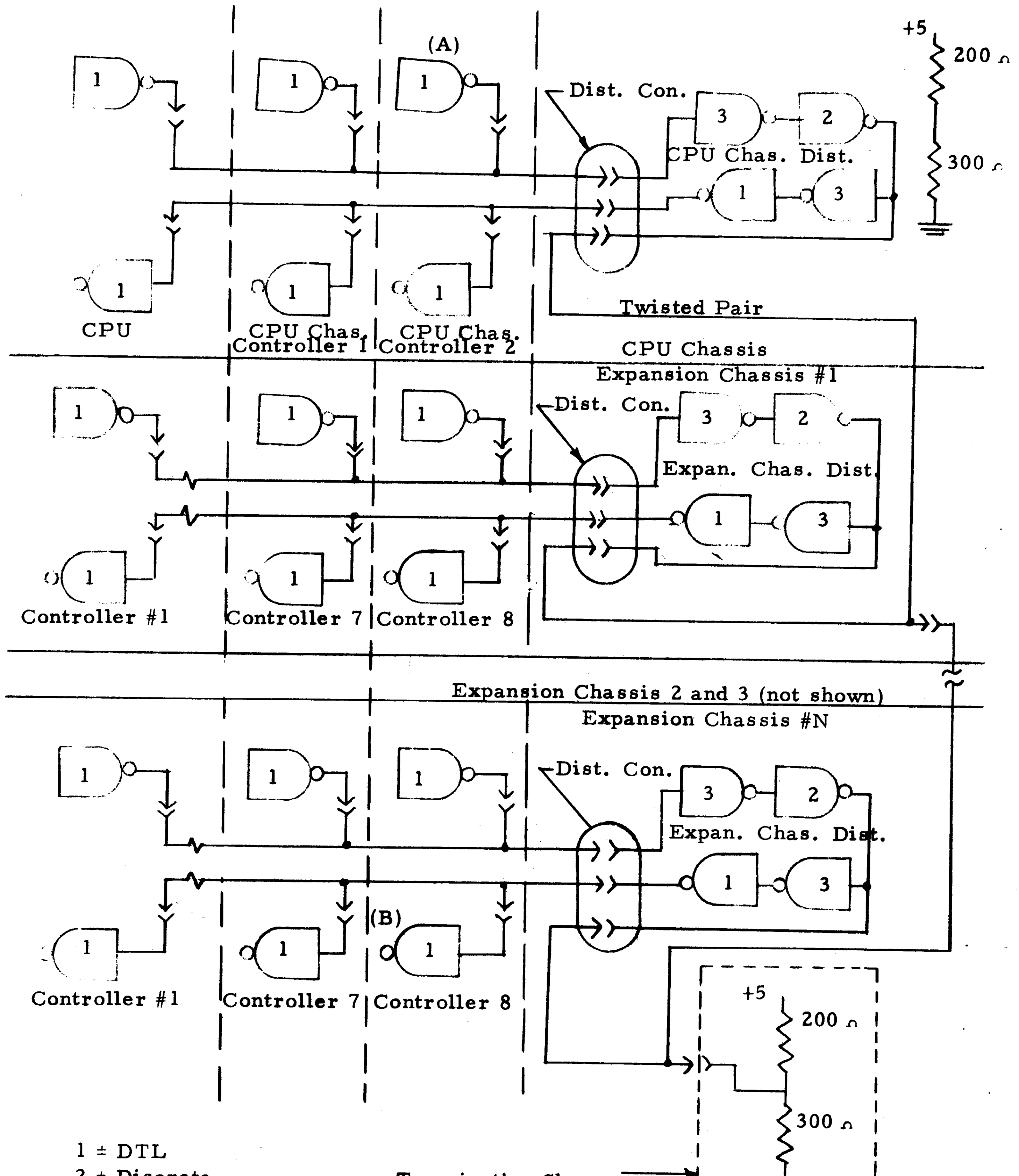
3.3 Electrical Interface

As described earlier, the I/O bus structure of the TEMPO I can be considered two distinct busses, an external and an internal bus. The internal bus is driven by distributors in each chassis and is the one which the controller designer must interface with. The external bus consists of twisted pair wires between chassis which are terminated at the end by a termination shoe. The termination results in a 120 ohm impedance to +3 volts.

The configuration of the three classes of lines are shown in Figures 3.3, 3.4, and 3.5. Note that all controller inputs and outputs are required to be at DTL levels with the exception of the peripheral clock which is a TTL circuit.

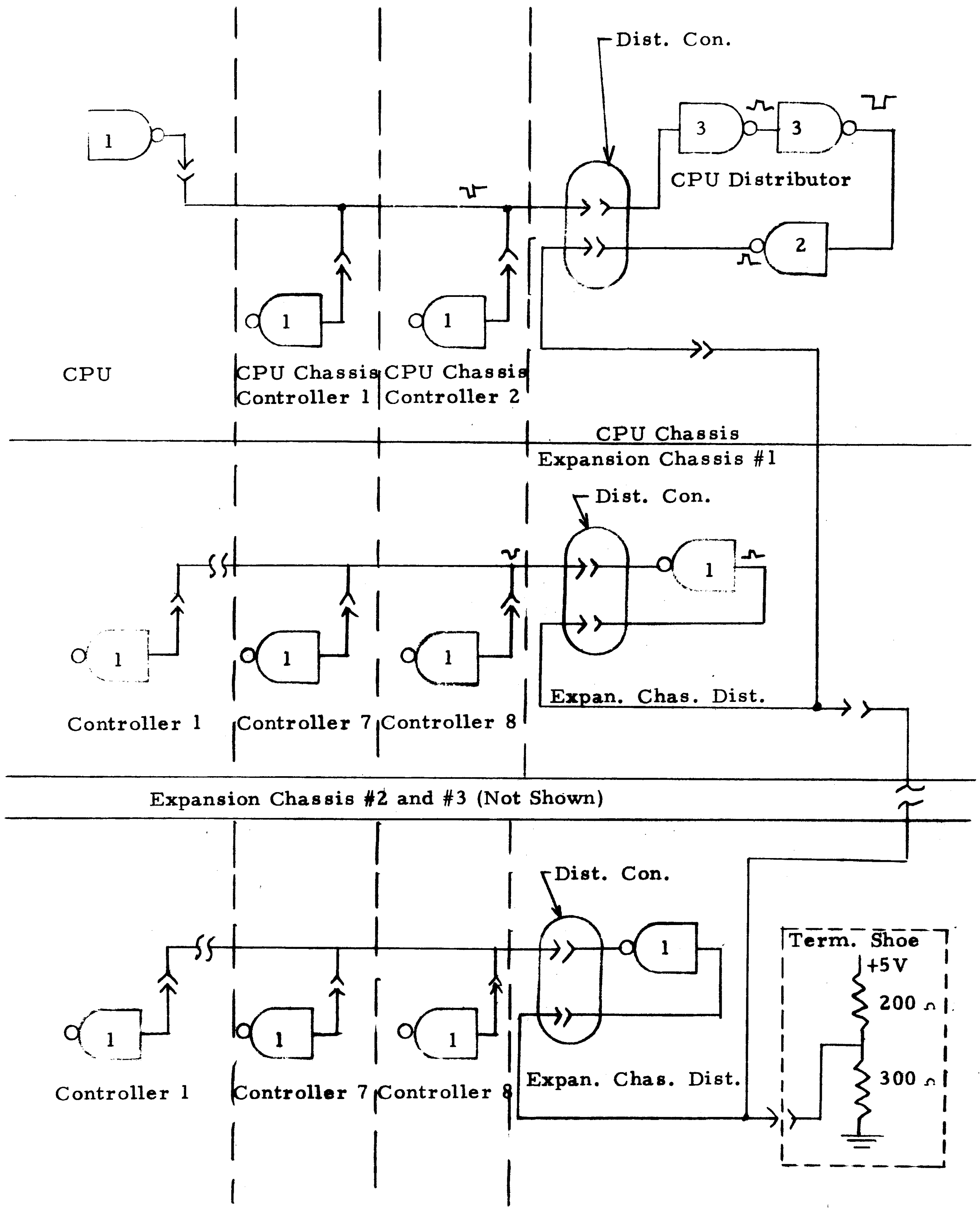
The design of the distributor is contained in Appendix D. All Tempo controllers and expansion chassis and power supplies conform to the following:

Power Supply	5 volts $\pm 5\%$ with both line and load variations referred to local controller ground
Ground	DC ground level less than 0.1 volt difference between chassis. All controllers must be designed with floating ground and minimum impedance between ground points to avoid transient noise and ground loops. For external controllers conforming to restraints given earlier, #26 stranded or #30 solid twisted pair should be used for interconnecting.



1 = DTL
2 = Discrete
3 = TTL

Figure 3-3. Class I Lines (Bidirectional) - 16 Lines



NOTE: Peripheral clock line requires TTL device.

Figure 3-4. Class II Lines (Output from CPU) - 17 Lines

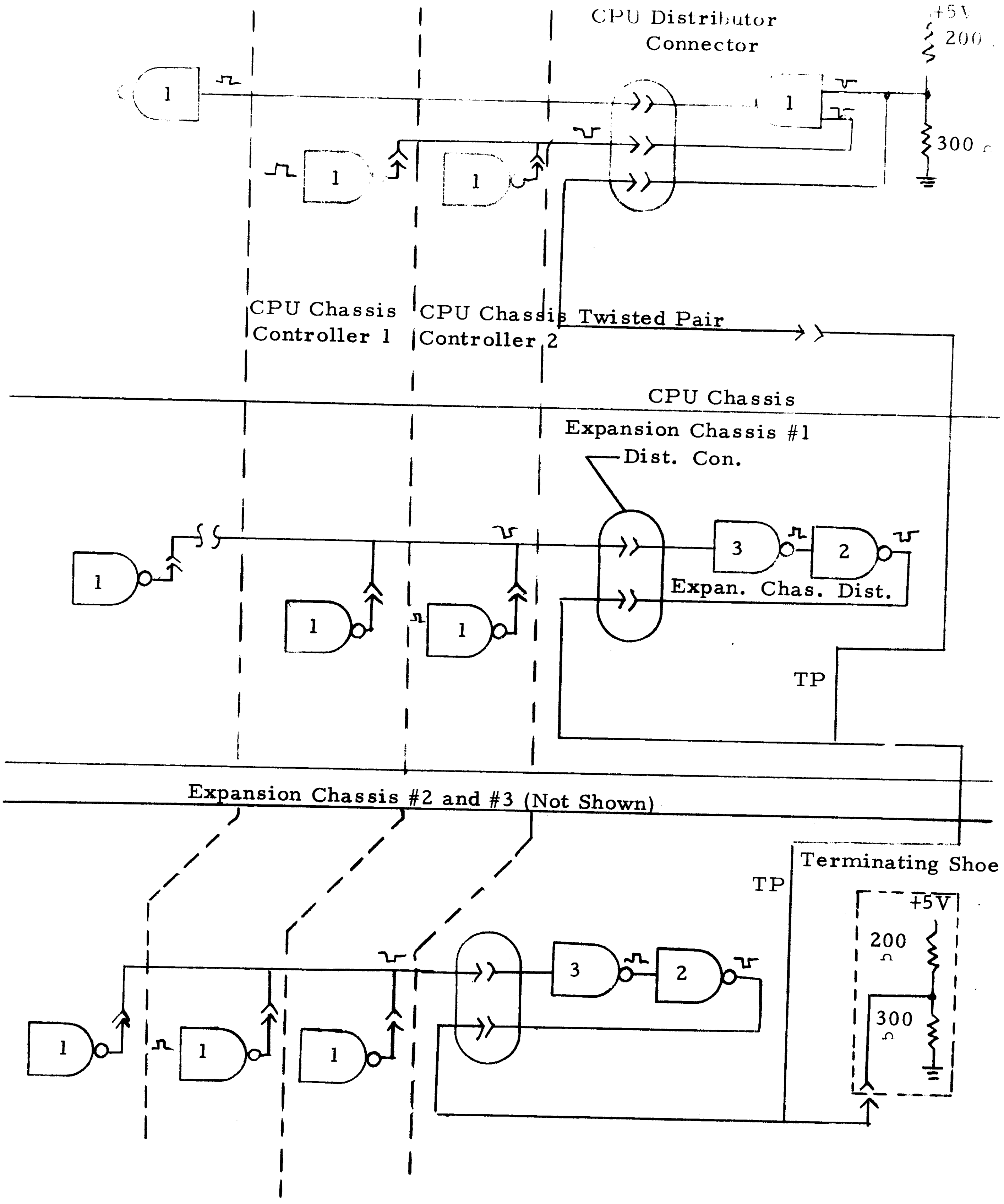


Figure 3-5. Class III Lines (Input to CPU) - 17 Signals

Controller Load	One DTL load per line (1.5 MA max.)
Maximum Number of Loads	Two per line in CPU, eight per line in expansion chassis.

Since the signals are effectively reshaped by the distributor the degradation of rise and fall times is not of importance. For the customer designing his own controllers it is recommended that the standard Tempo interface be used. In any event, all the parameters above must be met to assure reliable performance.

3.4 Recommended Controller Interface

In order to achieve the timing as indicated previously, and particularly if the High Rate I/O option is incorporated, the circuit of Figure 3.6 is recommended for the front end of any controller designed by the user. Tempo will provide detailed schematics if desired or will, under special engineering request, provide assistance in adapting the circuit to the user's physical requirements.

The circuit accomplishes the following:

1. Minimizes the time from receipt of strobe until the data appears on the line for WTI, RDS and ICI functions.

2. Provides the necessary delay between data and strobe receipt to assure reliable gating of data into the controller.

The distributors are only shown functionally.

If maximum throughput rate is not required then any circuit which provides the timing required will suffice as long as the DTL interface is maintained.

IV. HIGH RATE INPUT/OUTPUT

A preprequisite for this option is the installation of at least the four highest priority external interrupts (0 through 3). Interrupts 12 through 15 are standard. The maximum number of high rate channels available is seven. To achieve this requires four additional interrupts and the additional X-Registers (X_8 - X_{15}), since they are used in pairs and X_0 and X_1 are reserved for normal operations.

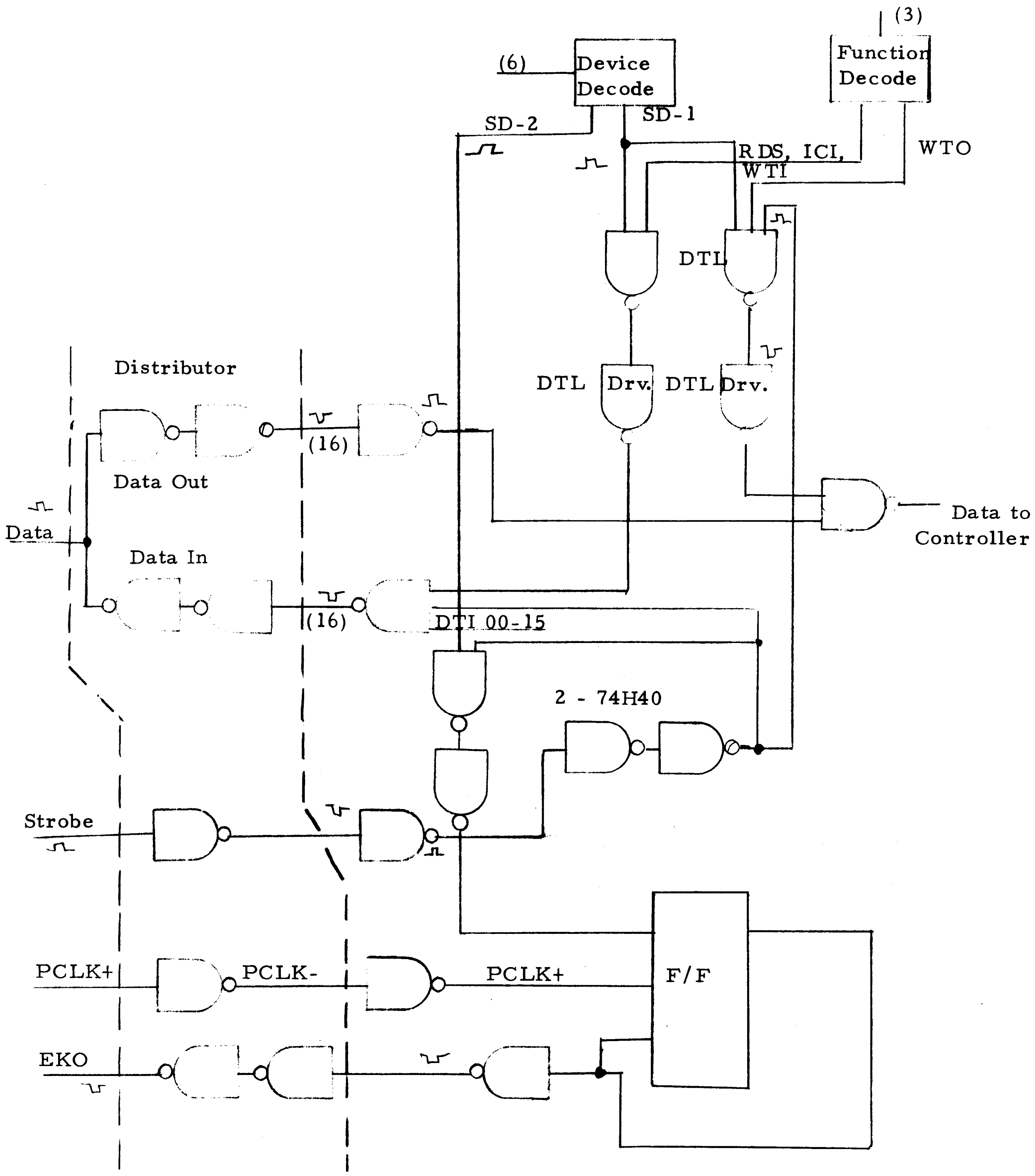
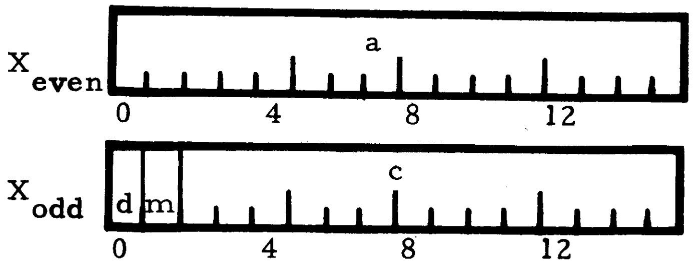


Figure 3.6 - Recommended Controller Interface

This option provides high rate block transfers between memory and the I/O bus for up to seven I/O controllers operating simultaneously. The transfers occur on a memory cycle stealing basis in response to interrupts and do not disturb the programmable registers (other than the X-Registers being used as control words) or flags of the CPU. Maximum throughput is approximately 700, 000 16-bit words or bytes per second.

This option consists of: a) seven high rate interrupt flags used in the interrupt control; b) control logic to provide the required sequences; c) overflow detect logic to detect end of block transfer; and d) a 16-bit address register.

The X-Registers (X_2 - X_{15}) are used in pairs to provide the following functions with formats as shown:



Field	Function
a	Address which I/O bus data will be transferred to or from. Range: 0 to 65, 535$_{10}$. The transfer is made and then the register is incremented. If m = 0 (byte mode) even addresses refer to the high order 8-bits (0 through 7) of the memory words and odd addresses refer to the low order bits (8 through 15) of the memory words.
d	Direction of transfer. "1" = input from I/O bus to memory; "0" = output from memory to I/O bus.
c	"Count" of words of bytes (depending on state of m). Since the count is updated by incrementing the count through the adder and the end of block transfer is detected by looking for an overflow, the "count" is set to the two's complement of the number of words or bytes required to be transferred, i.e. initial setting of c = 2^{14} -n, where n is the number of words to be transferred.
m	Mode of transfer, byte or word, if m = 0 the transfer takes place a byte per interrupt; if m = 1 a word per interrupt. In the byte mode all data input to the computer is on lines DB08 through DB15 while data output from the computer is on lines DB00 through DB07.

During the high rate interrupt sequence addresses are presented to memory through the H-Register. The H-Register (see Figure 1.1) is a 16-bit non-programmable register that receives address data from the X-Registers by way of the D-Register. It connects to the address bus thus providing an address to memory.

The main adder of the CPU is time-shared to provide for incrementing and replacing of the D-Register in one clock period. Detection logic is provided to indicate when an overflow has occurred. Note that this is not the same overflow flag associated with CPU arithmetic operations· high rate I/O operations do not affect any of the CPU flags.

High rate interrupt flags are provided in the interrupt logic. If one of these is set, an interrupt occurring at the corresponding position will cause the high rate interrupt sequence to be entered instead of the normal sequence. These flags are set or reset by the Set High Rate Interrupt Flags (SIF) and Reset High Rate Interrupt Flags (RIF) instructions. In addition, detection of overflow of count (in X odd) will cause the high rate flag to be reset (and the interrupt to be set).

Logic is provided to generate the controller address corresponding to the interrupt position, send the I/O strobe, and use the EKO to strobe data from the I/O bus to the M-Register.

The High Rate I/O option operation can be divided into three sections: Setup, Transfers, and Completion Servicing.

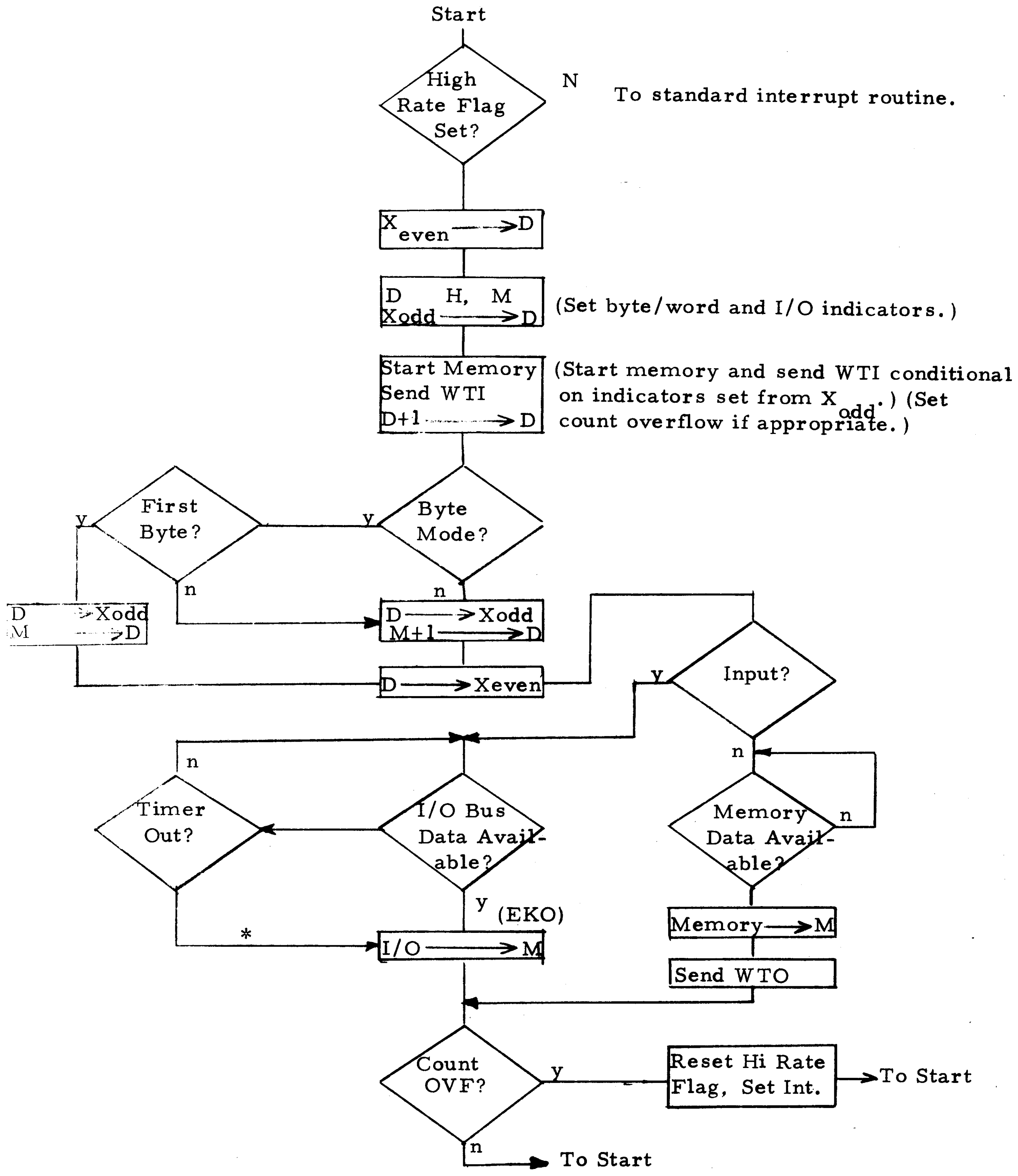
The Setup is accomplished as follows through the use of an appropriate software routine:

A. The X-Registers are loaded with the starting address, count of transfers, mode of transfer and direction of transfer (two LDX).

B. The high rate interrupt flag is set (SIF).

C. The corresponding controller is started (EDF).

The transfers then proceed starting with the interrupt from the controller. The flow chart shown in Figure 4.1 describes the hardware sequence that occurs each time an interrupt occurs.



*If this path is taken the I/O bus information probably is invalid.

Figure 4-1. High Rate I/O Interrupt Sequence

In summary, the address stored in X_{even} is used to address memory, the address is incremented (conditional on byte address if byte mode is specified), the count (X_{odd}) is incremented, the transfer between the I/O bus and memory is made, a check for count overflow is made and the proper exit is made. The Setup (steps A and B) varies from 3 to 5 memory cycles dependent upon whether X is loaded from the registers or memory respectively.

Completion of transfer can occur either when the count has overflowed or when the physical record being read at the controller has ended. When the count overflows the high rate interrupt flag is reset and the interrupt is set. The interrupt with no high rate flag condition causes a normal interrupt. The user will have stored at this interrupt location the address of an appropriate subroutine for processing the completion of block transfer.

If the physical record is shorter than the count stored in X_{odd}, the high rate interrupts will cease before the count has overflowed. For devices that can have variable length records it is suggested that a separate interrupt (low priority) be used to signal that physical end of record has been reached. All TEMPO I controllers of this nature will provide this Program Interrupt. The programmer can easily resolve any ambiguities by examination of the high rate interrupt flag or the count in the X_{odd} registers.

APPENDICES

V. DESIGN NOTES

The following points of importance should be understood for all controller design.

5.1 Each controller must decode device address and select the strobe.

5.2 Each controller must respond with an EKO pulse within two micro-seconds.

5.3 For high rate the timing of the sequence given must be adhered to.

5.4 Priority assignments are plugboard changeable in the controller.

5.5 All controllers should have DTL interfaces.

5.6 Lines longer than 5 feet total are not allowed with the basic bus.

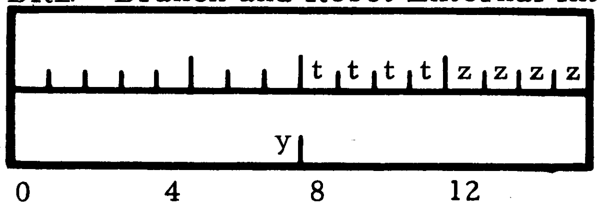
5.7 All byte transfers <u>into</u> the CPU must be on data line DB08 through DB15. All byte transfers <u>out</u> of the CPU must appear on lines DB00 through DB07.

5.8 If a peripheral device operating through a high rate channel can have a physical record shorter than the count set into the X-Registers, a program interrupt should be provided in addition to the high priority interrupt associated with the device. This second interrupt may be shared among several high rate devices.

APPENDIX A

INPUT/OUTPUT INSTRUCTIONS AND SAMPLE I/O SEQUENCES

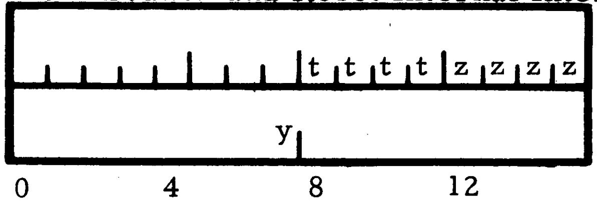
BRE Branch and Reset External Interrupt Timing



The interrupt latch specified by bits 12-15 of the instruction is reset. Then the memory address y' is placed in the P-Register, allowing complete memory addressing capability for the fetch of the next instruction.

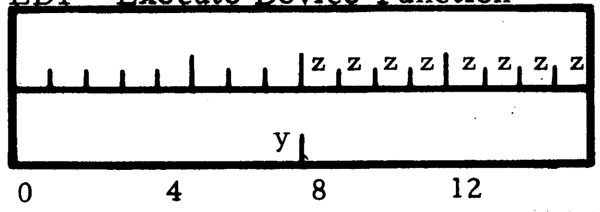
If the Read Only Memory option is implemented, this instruction will operate as described only when an interrupt is being serviced. At any other time the ROM mode flag will be reset after which the memory address y' is placed in the P-Register, allowing complete memory addressing capability for the fetch of the next instruction.

BRI Branch and Reset Internal Interrupt Timing



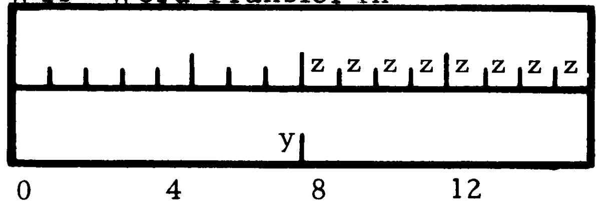
The internal interrupt latch specified by bits 12-15 of the instruction is reset. Then the memory address y' is placed in the P-Register, allowing complete memory addressing capability for the fetch of the next instruction.

EDF Execute Device Function



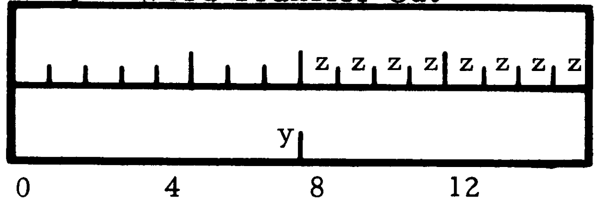
The action specified by bits 7-9 of the instruction is performed by the device specified by bits 10-15. In addition to bits 7-9 of the instruction, the A-Register is available to the controller for further instruction coding, providing a total of 19-bits for specifying an action by an external device. If the EDF command is rejected, a similated BSP to memory location y is affected. A detailed description of how the EDF command applies to each device is given in Sections

WTI Word Transfer In



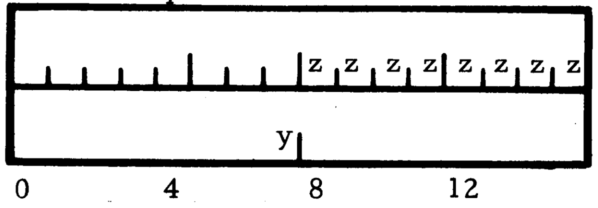
One word (up to 16-bits of information) is transferred into the A-Register from the device specified by bits 10-15 of the instruction. Bits 7-9 represent the order code and are available for special coding at the system level. The interrupt line in the controller specified by bits 10-15 is reset. If the WTI command is rejected, a simulated BSP operation occurs to memory location y.

WTO Word Transfer Out



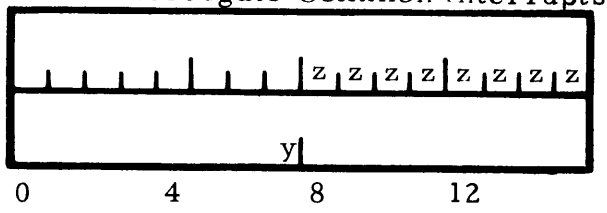
One word (up to 16-bits of information) is transferred from the A-Register to the device specified by bits 10-15 of the instruction. Bits 7-9 represent the order code and are available for special coding at the system level. The interrupt line in the controller specified by bits 10-15 is reset. If the WTO command is rejected, a simulated BSP operation occurs to memory location y.

RDS Request Device Status



The status of the device specified by bits 10-15 is transferred into the A-Register. If the RDS command is rejected, a simulated BSP operation occurs to memory location y. Bits 7-9 represent the order code and are available for special coding at the system level; if not used, these bits should be zero.

ICI Interrogate Common Interrupts



The service-required status of the devices (up to 16) that share the common interrupt line (line 15) is transferred into the A-Register. Device 16 corresponds to bit position zero, and each succeeding device up to 31 corresponds to each succeeding bit position up to 15. A one in any of the 16-bit positions indicates that the corresponding device requires service. The execution of the ICI instruction resets all service-required indicators, so that status received must be saved until all devices requiring service have been processed. The ICI instruction is a two word instruction with no device code required since it applies only to the common interrupt line. Bits 7-9 represent the order code and are available for special coding at the system level; if not used, these bits should be zero. If the ICI command is rejected, a simulated BSP operation occurs to memory location y.

TYPICAL I/O INTERRUPT SUBROUTINE WHICH ALLOWS PROCESSING BETWEEN INTERRUPTS (USING X-REGISTER)

ASSUMPTIONS

a) The CPU is processing when the interrupt occurs, i.e. its status must be stored and reloaded.
b) The output table address and table length have been given in the I/O subroutine calling sequence and saved by the I/O subroutine initialization.
c) The interrupt mask is set to allow the interrupt.
d) The EDF has been given to start the output device.
e) The interrupt is processed as soon as it is set true (no latency time).
f) Three X-Registers (X2, X3 and X4) are dedicated to this interrupt.

SEQUENCE

```
LABEL  COMMAND          OPERAND        COMMENTS                        CYCLES

       BSP (simulated)  INT                                              2
INT    CON              0
       TRF              A, 2           Save A-Register                   1
       LDA              0, X, 3        Pick-Up Output Data               3
       WTO              DEV, REJECT    Output                            1.5
       IRT              3, 3           Increment Index                   1
       IRT              4, 4           Increment Table Length            1
       BIR              E, DONE        If output complete, branch.       1
CONT   TRF              2, A           Restore A-Register                1
       BRE              DEV, INT, I    Return to Caller                  3
                                                                       ____
                                                                       14.5

DONE   BSP              TERM           Call Terminate Routine            *
       BAR              GE, CONT       If terminate, continue.           *
       EDF              DEV, REJECT    Stop I/O                          *
       BUR              CONT                                             *
```

Assuming 0.9 µs memory $\frac{1}{0.9 \text{ X } 14.5}$ = 77K words/second.

*Terminate branch only.

TYPICAL I/O INTERRUPT SUBROUTINE WHICH ALLOWS PROCESSING BETWEEN INTERRUPTS (NOT USING X-REGISTER)

ASSUMPTIONS

a) The CPU is processing when the interrupt occurs; i.e., its status must be stored and reloaded.
b) The output table address and table length have been given in the I/O subroutine calling sequence and saved by the I/O subroutine initialization.
c) The interrupt mask is set to allow the interrupt.
d) The EDF has been given to start the output device.
e) The interrupt is processed as soon as it is set true (no latency time).

SEQUENCE

LABEL	COMMAND	OPERAND	COMMENTS	CYCLES
	BSP (simulated)	INT		2
INT	CON	0		
	STA	TEMP1	Save A-Register	2
	LDA	TEMP2, I	Pick-Up Output Data	4
	WTO	DEV, REJECT	Output	1.5
	INC	TEMP2	Increment Output Table Address	2
	INC	TEMP3	Increment Table Length	2
	BIR	E, DONE	If output complete, branch.	1
CONT	LDA	TEMP1	Restore A-Register	2
	BRE	DEV, INC, I	Return to Caller	3
				19.5
DONE	BSP	TERM	Call Terminate Routine	*
	BAR	GE, CONT	If terminate, continue.	*
	EDF	DEV, REJECT	Stop I/O	*
	BUR	CONT		*

Assuming 1 μs memory $\frac{1}{0.9 \text{ X } 19.5}$ = 57.0K words/second.

NOTE: If it is not required to store/load CPU status, the rate is $\frac{1}{0.9 \text{ X } 15.5}$ = 71.8K words/second.

*Terminate branch only.