TEXAS INSTRUMENTS

TMS9914A General Purpose Interface Bus (GPIB) Controller

Data Manual

1989

Microprocessor Peripheral Devices

TMS9914A General Purpose Interface Bus (GPIB) Controller Data Manual

TEXAS INSTRUMENTS

TABLE OF CONTENTS

SECTION PAGE

1. INTRODUCTION 1-1
   - 1.1 Description 1-1
   - 1.2 Key Features 1-1
   - 1.3 Relationship to the TMS9914 1-1
   - 1.4 Introduction to the IEEE-488 1975/78 Interface Bus 1-1
   - 1.5 Typical Applications 1-2

2. ARCHITECTURE 2-1
   - 2.1 Registers 2-2
     - 2.1.1 Interrupt Mask and Status Registers 0 2-2
     - 2.1.2 Interrupt Mask and Status Registers 1 2-4
     - 2.1.3 Address Status Register 2-6
     - 2.1.4 Address Register 2-6
     - 2.1.5 Auxiliary Command Register 2-6
     - 2.1.6 Description of Auxiliary Commands 2-7
     - 2.1.7 Bus Status Register 2-10
     - 2.1.8 Serial Poll Register 2-10
     - 2.1.9 Command Pass Through Register 2-11
     - 2.1.10 Parallel Poll Register 2-11
     - 2.1.11 Data In Register 2-11
     - 2.1.12 Data Out Register 2-12
   - 2.2 Direct Memory Access 2-12
   - 2.3 Terminal Assignments and Functions 2-12
   - 2.4 Transceiver Connections 2-15

3. STATE DIAGRAM IMPLEMENTATION 3-1
   - 3.1 Auxiliary Commands 3-1
   - 3.2 Acceptor Handshake 3-2
   - 3.3 Source Handshake 3-4
   - 3.4 Talker and Listener Functions 3-6
   - 3.5 Service Request Function 3-10
   - 3.6 Remote/Local Function 3-12
   - 3.7 Parallel Poll Function 3-13
     - 3.7.1 Remotely Configured Parallel Poll 3-13
   - 3.8 Controller Function 3-14
     - 3.8.1 Controller Self Addressing 3-14
     - 3.8.2 Passing Control 3-16
     - 3.8.3 System Controller 3-17

4. TMS9914A ELECTRICAL SPECIFICATIONS 4-1
   - 4.1 Absolute Maximum Ratings Over Operating Free-Air Temperature Range 4-1
   - 4.2 Recommended Operating Conditions 4-1
   - 4.3 Electrical Characteristics Over Full Range of Recommended Operating Conditions 4-1
   - 4.4 Timing Characteristics and Requirements 4-1
     - 4.4.1 Clock and Host Interface Timing Requirements 4-2
     - 4.4.2 Host Interface Timing Characteristics 4-2
     - 4.4.3 Source Handshake Timing Characteristics 4-2
     - 4.4.4 Acceptor Handshake Timing Characteristics 4-3
     - 4.4.5 ATN, EOI, and IFC Timing Characteristics 4-3
     - 4.4.6 Controller Timing Characteristics 4-4

5. MECHANICAL SPECIFICATIONS 5-1
   - 5.1 TMS9914A JL – 40-Pin Ceramic Package 5-1
   - 5.2 TMS9914A NL – 40-Pin Plastic Package 5-1
   - **5.3 TMS9914A FN—44-Pin PLCC Package 5-2**

LIST OF APPENDICES

APPENDIX — PAGE

A IEEE-488 Standard Connector A-1
B Typical Sequences of Events for the Controller B-1
C SN75160/161/162 Data Sheets C-1
D Example Software D-1

LIST OF ILLUSTRATIONS

FIGURE — PAGE

1-1 Typical TMS9914A Application 1-2
2-1 Simplified Block Diagram 2-1
2-2 DMA Configuration 2-13
2-3 Transceiver Connections 2-16
3-1 TMS9914A Auxiliary Command State Diagram 3-2
3-2 TMS9914A Acceptor Handshake State Diagram 3-3
3-3 TMS9914A Source Handshake State Diagram 3-5
3-4 TMS9914A Listener State Diagram 3-7
3-5 TMS9914A Talker State Diagram 3-8
3-6 Service Request State Diagram 3-11
3-7 TMS9914A Remote Local State Diagram 3-12
3-8 TMS9914A Parallel Poll State Diagram 3-14
3-9 TMS9914A Controller State Diagrams 3-15
3-10 Passing Control Between TMS9914s 3-17
3-11 IFC and REN Pins 3-18
4-1 TMS9914A Clock Cycle Timing 4-4
4-2 TMS9914A Read Cycle Timing 4-5
4-3 TMS9914A Write Cycle Timing 4-5
4-4 TMS9914A DMA Read Operation 4-6
4-5 TMS9914A DMA Write Operation 4-6
4-6 TMS9914A Source and Acceptor Handshake Timing(s) 4-7
4-7 TMS9914A Acceptor Handshake Timing ''ATN'' True 4-8
4-8 TMS9914A Response to ''ATN'' and ''EOI'' 4-9
4-9 TMS9914A Controller Timing 4-10

LIST OF TABLES

TABLE — PAGE

2-1 TMS9914A Read Registers 2-2
2-2 TMS9914A Write Registers 2-2
2-3 Auxiliary Commands 2-7
2-4 Software Reset Conditions 2-8
3-1 Auxiliary Command State Diagram Mnemonics 3-2
3-2 Acceptor Handshake Mnemonics 3-3
3-3 Acceptor Handshake Message Outputs 3-4
3-4 Source Handshake Mnemonics 3-5
3-5 Source Handshake Message Outputs 3-5
3-6 Talker and Listener Mnemonics 3-10
3-7 Talker Function Message Outputs 3-10
3-8 Service Request Mnemonics 3-12
3-9 Service Request Message Outputs 3-12
3-10 Remote/Local Mnemonics 3-13
3-11 Parallel Poll Mnemonics 3-14
3-12 Parallel Poll Message Outputs 3-14
3-13 Controller Function Mnemonics 3-15
3-14 Controller Function Message Outputs 3-16
3-15 Multiline Interface Messages 3-19

1. INTRODUCTION

1.1 DESCRIPTION

The TMS9914A provides an interface between a Microprocessor System and the General Purpose Interface Bus (GPIB) specified in the IEEE-488 1975/78 standards and the IEEE-488A 1980 supplement. The device is controlled and configured through 8-bit memory mapped registers and enables all aspects of the standards to be implemented, including talker, listener and controller.

1.2 KEY FEATURES

- Handles all IEEE-488 1975/78 functions
- Compatible with IEEE-488A 1980 supplement
- Talker and listener function (T,TE,L,LE)
- Automatic source and acceptor handshakes (SH,AH)
- Controller with pass control
- System Controller capabilities
- Device trigger and device clear capabilities (DT,DC)
- Optional automatically cleared 'request service bit'
- Parallel and serial poll facilities (PP)
- Remote/local function with local lockout (RL)
- Single or dual primary addressing
- Secondary address capabilities
- Direct interface to SN75160/161/162 bus transceivers with no additional logic
- Compatible with most microprocessors
- Direct memory access facilities
- Memory-mapped microprocessor interface

1.3 RELATIONSHIP TO THE TMS9914

The TMS9914A is compatible with the TMS9914 and may replace it in any application without software alterations. New features are included on the TMS9914A which increase the flexibility of the device. These features are disabled at power-up and must be programmed by the user as needed.

TMS9914A VS. TMS9914:

1) Byte Output interrupt modification (see Section 2.1.1 and Appendix B)

2) Reduced bus settling time T1 (see Section 2.1.6 and Section 3.3)

3) Modification to the Service Request Function (see Section 3.5)

4) Addition of a second request service (rsv) bit which is automatically cleared (see Section 2.1.6 and Section 3.5)

1.4 INTRODUCTION TO THE IEEE-488 1975/78 INTERFACE BUS

The GPIB is designed to allow up to 15 instruments within a localized area to communicate with each other over a common bus. Each device has a unique address, read from external switches at power-on, to which it responds. Information is transmitted in byte serial bit parallel format and may consist of either device-dependent data or interface messages, commonly referred to as data or commands, respectively.

Device data may be sent by any one device (the talker) and received by a number of other devices (listeners). Instructions, such as select range, select function, or measurement data for processing or printout, may be sent in this way.

One of the devices on the bus, designated the Controller in charge (Controller), may send interface control messages. Devices can be assigned to the bus as listeners or talkers by sending their unique talk or listen addresses and may be switched between remote and local control.

The bus itself consists of a 24-wire shielded cable. Eight lines carry data; 8 are control lines; 8 are signal and system grounds. A diagram showing the IEEE bus configuration is given in Appendix A.

Three of the management lines operate as a three-line handshake between talker (or controller) and listeners. No new data is sent until each device addressed to listen has received the last byte and is ready for the next. This method of asynchronous communication ensures that the data rate is suited to the slowest active listener, as well as ensuring compatibility over a wide range of devices.

The remainder of this manual assumes working familiarity with the IEEE-488 1975/78 standards. Terminology and abbreviations defined within these standards are freely used throughout. The IEEE convention of lower case for local messages and upper case for remote messages (received via the interface) is used in all acronyms.

1.5 TYPICAL APPLICATIONS

The TMS9914A is used when an intelligent instrument is required to communicate with an IEEE-488 General Purpose Interface Bus (GPIB). It performs the interface function between the microprocessor and bus and relieves the processor of the task of maintaining the IEEE protocol. By utilizing the interrupt capabilities of the device, the bus does not have to be continually polled, and fast responses to changes in the interface configuration can be achieved.

A block diagram showing the TMS9914A in a typical application is given in Figure 1-1.

The GPIB input/output pins are connected to the IEEE-488 bus via bus transceivers. The direction of data flow is controlled by the TE and $\overline{\text{CONT}}$ outputs generated on the TMS9914A. The SN75160, 75161 and 75162 (see Appendix C) are designed specifically for use with a GPIB interface. The TE and $\overline{\text{CONT}}$ signals are routed within the devices so that the buffers on particular lines are controlled as required by the TMS9914A. Other buffers may be used, but they may require a small amount of external logic, particularly around the EOI line buffer.

Communication between the microprocessor and TMS9914A is carried out via memory-mapped registers. There are 13 registers within the TMS9914A, 6 of which are read and 7 write. These registers both pass control data to and get status information from the device.

The three least significant address lines from the MPU are connected to the register select lines RS0, RS1, and RS2 and determine the particular register selected. The high order address lines are decoded by external logic to cause the $\overline{\text{CE}}$ input to the TMS9914A to be pulled low when any one of eight consecutive addresses are selected. Thus the internal registers appear to be situated at eight consecutive locations within the MPU address space. Reading or writing to these locations transfers information between the TMS9914A and the microprocessor. Note that reading and writing to the same location will not access the same register within the TMS9914A since they are either read-only or write-only registers. For example, a read operation with RS2-RS0 = 011 gives the current status of the GPIB interface control lines, whereas a write to this location loads the auxiliary command register.

Each device on the bus interface is given a 5-bit address enabling it to be addressed as a talker or listener. This address is set on an external DIP switch (usually at the rear of an instrument) before power-on.

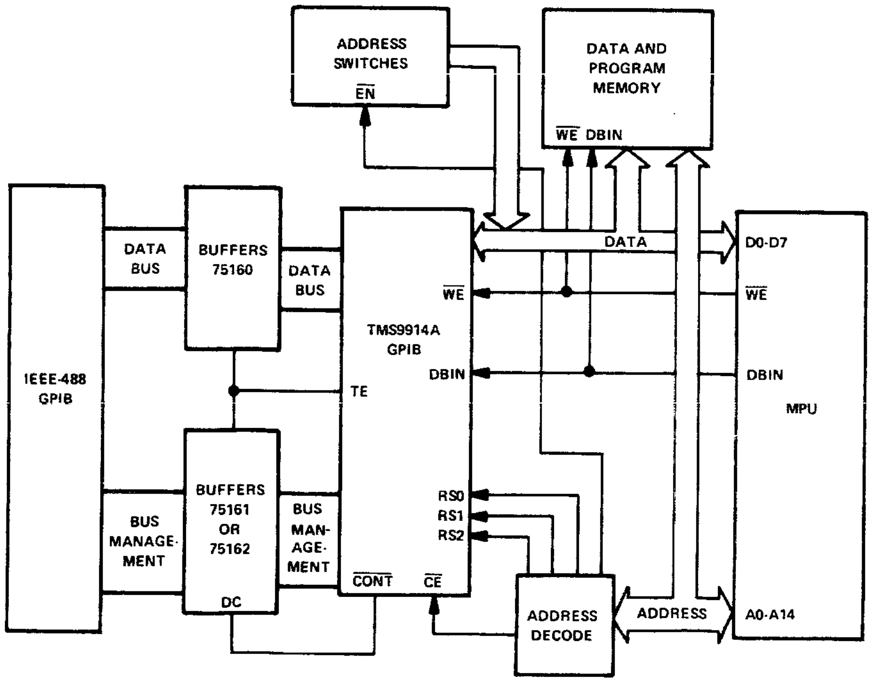


FIGURE 1-1 – TYPICAL TMS9914A APPLICATION

Typical TMS9914A configuration utilizes registers 100 or 101 as an address switch register (see Table 2-1). This register may consist of a DIP switch which drives the data lines via tristate buffers when one of these addresses is read. This allows the host MPU to read a device address which is manually set and write this address into the address register of the TMS9914A for device identification on the bus. The TMS9914A responds by causing a My Address (MA) interrupt and entering the required addressed state when this address is detected on the GPIB data lines.

2. ARCHITECTURE

The block diagram of the internal architecture of the TMS9914A is given in Figure 2-1. As previously stated, there are 13 MPU accessible registers of which 6 are read and 7 are write. These registers handle all communication between the IEEE-488 1975/78 bus and microprocessor.

Each register is accessed by putting the relevant address on lines RS0, RS1 AND RS2 and performing a memory read ($\overline{WE}$ = 1 DBIN = 1) or memory write ($\overline{WE}$ = 0 DBIN = 0) operation. The register addresses and use of each bit is shown in Table 2-1 for the read registers and Table 2-2 for the write registers. A full description of each register is given in the following paragraphs.

Implementation of the functions described by the state diagrams of the IEEE-488 standard is carried out in the IEEE-488 state diagram block. Information is received from the IEEE bus and from the internal registers and is combined with the current status of the device (for example, Talker Active State, TACS) to produce the control signals to load registers or handle the handshake or bus management lines.

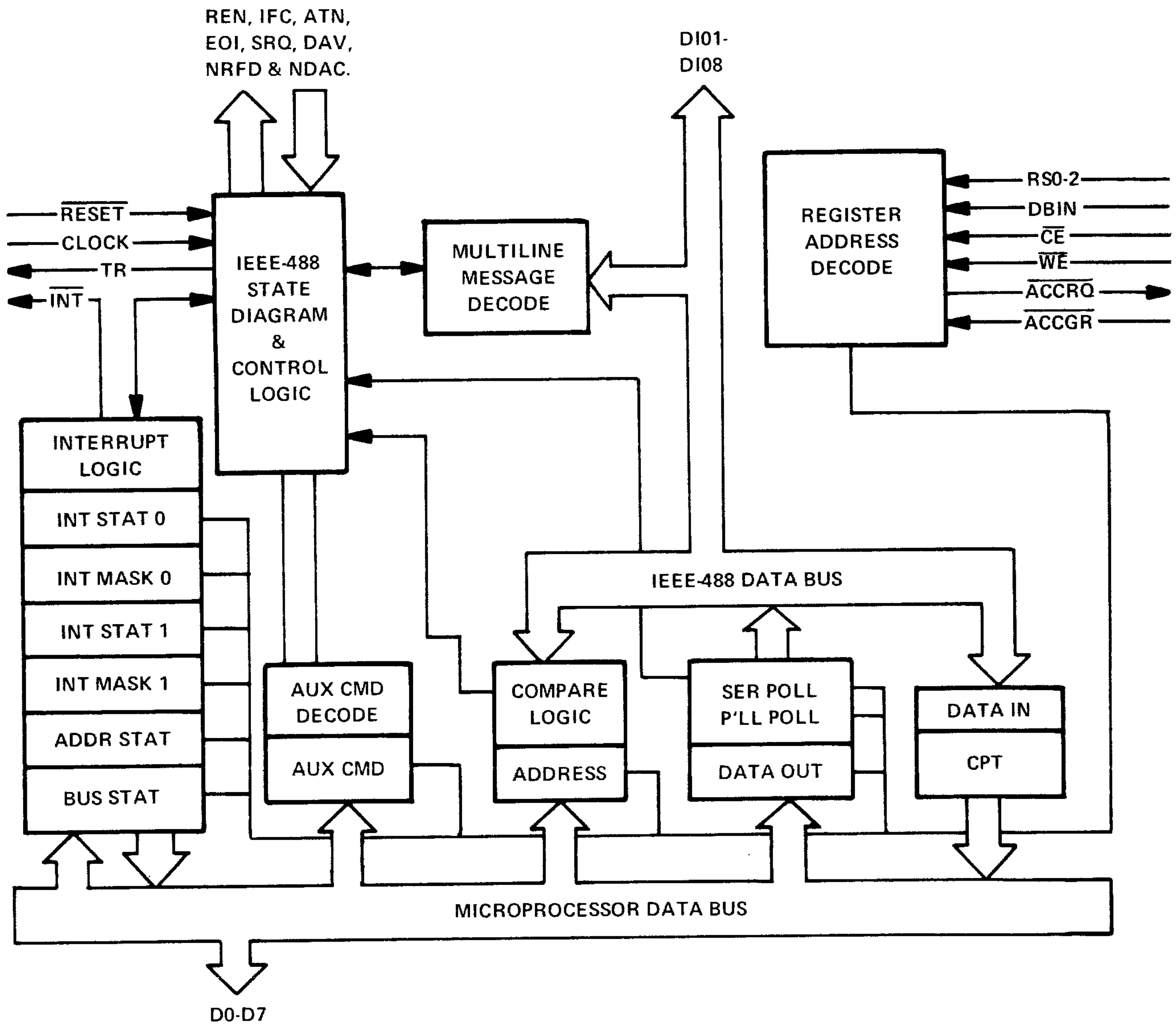


FIGURE 2-1 – SIMPLIFIED BLOCK DIAGRAM

TABLE 2-1 – TMS9914A READ REGISTERS

ADDRESS			REGISTER NAME	BIT ASSIGNMENT							
RS2	RS1	RS0		D0	D1	D2	D3	D4	D5	D6	D7
0	0	0	Int Status 0	INT0	INT1	BI	BO	END	SPAS	RLC	MAC
0	0	1	Int Status 1	GET	ERR	UNC	APT	DCAS	MA	SRQ	IFC
0	1	0	Address Status	REM	LLO	ATN	LPAS	TPAS	LADS	TADS	ulpa
0	1	1	Bus Status	ATN	DAV	NDAC	NRFD	EOI	SRQ	IFC	REN
1	0	0	*								
1	0	1	*								
1	1	0	Cmd Pass Thru	DIO8	DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1
1	1	1	Data In	DIO8	DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1

*The TMS9914A host interface data lines will remain in the high impedance state when these register locations are addressed. An Address Switch Register may therefore be included in the address space of the device at these locations (see Section 1.5).

TABLE 2-2 – TMS9914A WRITE REGISTERS

ADDRESS			REGISTER NAME	BIT ASSIGNMENT							
RS2	RS1	RS0		D0	D1	D2	D3	D4	D5	D6	D7
0	0	0	Int Mask 0			BI	BO	END	SPAS	RLC	MAC
0	0	1	Int Mask 1	GET	ERR	UNC	APT	DCAS	MA	SRQ	IFC
0	1	0	*	xx	xx	xx	xx	xx	xx	xx	xx
0	1	1	Auxiliary Cmd	cs	xx	xx	f4	f3	f2	f1	f0
1	0	0	Address	edpa	dal	dat	A5	A4	A3	A2	A1
1	0	1	Serial Poll	S8	rsvl	S6	S5	S4	S3	S2	S1
1	1	0	Parallel Poll	PP8	PP7	PP6	PP5	PP4	PP3	PP2	PP1
1	1	1	Data Out	DIO8	DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1

*This address is not decoded by the TMS 9914A. A write to this location will have no effect on the device, as if a write had not occurred.

2.1 REGISTERS

2.1.1 Interrupt Mask and Status Registers 0

The Interrupt Mask and Interrupt Status registers operate independently of each other. The status bits will always be set when the appropriate events occur regardless of the state of the corresponding mask bit.

All interrupt bits, with the exception of INT0 and INT1 which are not storage bits, are edge triggered and are set when the appropriate condition becomes true. The storage bits are cleared immediately after the corresponding Interrupt Status Register is read by the host MPU. If an interrupt condition becomes true during this read operation, then the event is stored. The corresponding bit is set when the read operation ends, hence no interrupts are lost. In addition to being cleared by a read operation, the BO interrupt is also cleared by writing to the Data Out Register, and the BI interrupt is cleared by reading the Data In Register.

The interrupt status bits are cleared and held in the 0 condition while Software Reset (swrst) is set.

The corresponding bit of the Interrupt Mask register must be set to a 1 if an interrupt status bit is to cause an external interrupt ($\overline{\text{INT}}$ Low) when it is set (i.e., $\overline{\text{INT}} = \overline{\text{INT STATUS.INT MASK}}$). The mask register is not cleared by 'swrst' or the Hardware Reset pin ($\overline{\text{RESET}}$) and will power on in a random state. It must, therefore, be written to by the host MPU before 'swrst' is cleared to avoid extraneous interrupts (see Section 2.1.6 for operation of 'swrst').

The INT0 and INT1 bits of the Interrupt Status Register are not true status bits. INT1 will be true if there are any unmasked interrupt status bits set to a 1 in Interrupt Status Register 1. INT0 will be true if any of bits 2-7 of Interrupt Status Register 0 are unmasked and set to a 1. If either INT1 or INT0 is true, then the external interrupt pin ($\overline{\text{INT}}$) will be pulled low provided that the Disable All Interrupts feature (dai) has not been set.

The individual bits of Interrupt Status and Interrupt Mask Register 0 are described are in the following paragraphs. The conditions which set these bits, shown in parentheses, are given in terms of the state diagrams described in Section 3. Each bit is set on the rising edge of the condition shown.

INTERRUPT MASK/STATUS REGISTER 0

xx	xx	BI	BO	END	SPAS	RLC	MAC	INT MASK 0
INT0	INT1	BI	BO	END	SPAS	RLC	MAC	INT STATUS 0
D0	D1	D2	D3	D4	D5	D6	D7	MPU BUS

NOTE: A 0 masks and a 1 unmasks the bits in the interrupt mask registers.

INT1	This will be a 1 when an unmasked status bit in Interrupt Status Register 1 is set to a 1.
INT0	This will be a 1 when any of bits 2-7 of Interrupt Status Register 0 is unmasked and set to a 1.

BI Byte In. A data byte has been received in the Data In register. If the mask bit is not set, then no interrupt is generated but a RFD holdoff will still occur before the next data byte is accepted. If the Shadow Handshake feature is used, then this status bit will not be set. This bit is cleared by reading the Data In Register as well as after Interrupt Status Register 0 has been read. (Set On: ACDS1.LACS)

BO Byte Out. This is set when the Data Out Register is available to send a byte over the GPIB. This byte may be either a command if the device is a controller or data if the device is a talker. It is set when the device becomes an active talker or controller but will not occur if the Data Out register has been loaded with a byte which has not been sent. Subsequently, it will occur after each byte has been sent and the TMS9914A returns to SGNS. This bit is cleared by writing to the Data Out Register as well as by reading Interrupt Status Register 0.
(Set On: SGNS.CACS+SGNS.TACS.$\overline{\text{SHFS}}$)

NOTE

When a controller addresses itself as a talker and then goes to standby, there will be a momentary transition of the source handshake into SIDS before TACS becomes true and it reenters SGNS. Under these circumstances, the TMS9914A is guaranteed to give a BO interrupt on reentering 'SGNS'. The TMS9914, however, may not, and a controller going to standby as a talker should write the first byte of data into the Data Out Register immediately after writing the gts auxiliary command, without waiting for a BO interrupt (see Appendix B for details).

END This indicates that a byte just received by a listener was the last byte in a string, that is, it was received with the EOI line true. It is set at the same time as the BI interrupt. (Set On: (ACDS1.LACS.EOI)

NOTE

The END flag should be set at the same time as the BI flag. However, the END flag rises slightly before BI by about 5 to 15 ns. If the system is being used in an Interrupt driven mode, this delay is very unlikely to cause problems. If the interrupt flags are polled, there is the possibility that the Host CPU could read the Interrupt Status 0 register at such a time to see the END flag set, but not the BI flag. In this case, the BI flag will be held and loaded into the Interrupt Status 0 register when the current read cycle ends.

A typical Listener polling algorithm continually reads the Interrupt Status 0 register to check for the BI flag to be set. Once found, the END flag would then be checked to see if the byte in the DATA IN register is the last byte expected in the present string of data. However, with the END flag being set slightly before the BI flag, the following possibility of missing the END flag exists:

1. **Read Interrupt Status Register 0 to see if the BI flag is set. (Condition: END is set, but BI is not.)**
   a. **The read operation clears the contents of Status register 0.**
   b. **Since routine checks first for BI, there is no reason to check for the END flag if the BI flag is not set.**

2. Read Interrupt Status Register 0 again. (Condition: Previous BI flag is now set, END flag has been cleared.)

   a. Check to see if BI is set.
   b. Yes, Check END flag to see if last byte in string.
   c. END not set, expect more data to come.

3. Continue to read Status Register 0.

A solution is to have the Interrupt service routine also check for the END flag, even if the BI flag is not set. If the END flag is set, then also assume that BI is also set and act accordingly. If the BI flag is set and the END flag is not set, then read the DATA IN register to get the byte of data.

One problem with this is that if Interrupt Status 0 is read with the END flag set and the BI flag not set, then the pending Bi flag will be held until after the read operation is complete. On the next read of the Interrupt Status register, it will be very difficult to tell whether this BI flag is the result of the previous 'Pending' BI flag or is it the beginning of the next received message. The way around this problem is to ensure that after a BI or END 'interrupt', the information in the DATA IN Register is read BEFORE reading the Interrupt Status 0 register again. This will clear the present BI flag. The program flow for this suggested solution is as follows:

1. Read Interrupt Status Register 0 to see if the BI flag or the END flag is set. (Condition: UNKNOWN)

   a. The read operation clears the contents of Status register 0.
   b. Check for BI flag. If set, check END flag to see if last byte in string. If BI flag is set and END flag is not, then the byte of data is not the last in the string. Go to step 2.
   c. If BI flag is not set, still check END flag to see if set. If neither flag is set, repeat step 1.
   d. If BI flag is not set, but the END flag is set, continue with step 2.

2. Read the DATA IN register to get the last byte of data in the string. This will also clear any pending BI flag which may not have been seen in the Interrupt Status 0 register at the time an END flag was seen.

3. Continue to read Status Register 0. (Repeat Step 1.)

SPAS This indicates the the TMS9914A has requested service via rsv1 or rsv2 (in the Serial Poll Register or Auxiliary Command Register) and has been polled in a serial poll. It is set on the false transition of STRS when the serial poll status byte is sent. (Set On: $\overline{\text{STRS}}\cdot\text{SPAS}\cdot(\text{APRS}+\text{APRS2})$)

RLC Remote/Local Change. This is set by any transition between local and remote states in the Remote/Local function.
(Set On: (LOCS-REMS) + (REMS-LOCS) + (LWLS-RWLS) + (RWLS-LWLS)

MAC My Address Change. This indicates that a command has been received from the GPIB which has resulted in the addressed state of the TMS9914A to change. It will not occur if secondary addressing is being used, nor indicate that the TMS9914A has been readdressed on its other primary address.
(Set On: $\text{ACDS1}\cdot(\text{MTA}\cdot\overline{\text{TADS}}+\text{UNT}\cdot\text{TADS}+\text{OTA}\cdot\text{TADS}+\text{MLA}\cdot\overline{\text{LADS}}+\text{UN}\cdot\text{LADS})$

2.1.2 Interrupt Mask and Status Registers 1

The operation of Interrupt Mask and Status Register 1 is similar to that of Interrupt Mask and Status Register 0 except that all bits are true storage bits. The status bits are cleared only following the register being read and by 'swrst'.

There is one distinct group of interrupts in this register: GET, UNC, APT, DCAS, MA. These are all set in response to commands received over the bus and if unmasked, a Data Accepted (DAC) holdoff will occur when the interrupt in question is set. It may be released with a 'dacr' auxiliary command. This is further discussed in Section 3.2.

The mask bit of the APT interrupt is further used in the talker and listener functions. When the interrupt is unmasked, the talker and listener functions of the TMS9914A implement the extended talker and extended listener functions of IEEE-488. Otherwise these functions implement the talker and listener functions of IEEE-488.

The individual bits of Interrupt Status and Interrupt Mask Register 1 are described below. The conditions which set these bits, shown in parentheses, are given in terms of the state diagrams described in Section 3.

INTERRUPT MASK/STATUS REGISTER 1

GET	ERR	UNC	APT	DCAS	MA	SRQ	IFC	INT MASK 1
GET	ERR	UNC	APT	DCAS	MA	SRQ	IFC	INT STATUS 1
D0	D1	D3	D3	D4	D5	D6	D7	MPU BUS

GET	This is set if a Group Execute Trigger command is received. A DAC holdoff occurs if the interrupt is unmasked. The TR pin becomes high when this command is received and persists high for the duration of a DAC holdoff if one occurs. If the interrupt is masked, the TR pin becomes high for approximately five clock cycles. (Set On: GET.LADS.ACDS1)
ERR	Error. This is set if the source handshake becomes active and finds that the NDAC and NRFD lines are both high. This indicates that, for whatever reason, there are no acceptors on the bus. (Set On: SERS)
UNC	**Unrecognized Command. This is set if a command that has no meaning to the TMS9914A has been received. Unrecognized addressed commands will only cause this interrupt if the device is LADS except for TCT which will only interrupt in TADS. Secondary commands will only cause this interrupt if the 'pts' auxiliary command has been set previously. A DAC holdoff will occur if this interrupt is unmasked which effectively enables the command pass through feature. Unrecognized commands may be inspected in the Command Pass Through Register before this holdoff is released. (Set On: ACDS1·(UCG·$\overline{\text{LLO}}$·$\overline{\text{SPE}}$·$\overline{\text{SPD}}$·$\overline{\text{DCL}}$ + ACG·$\overline{\text{GET}}$·$\overline{\text{GTL}}$·$\overline{\text{SDC}}$·$\overline{\text{TCT}}$·LADS + TCT·TADS + SCG·pts))**
APT	Address Pass Through. Unmasking this interrupt enables secondary addressing. It is set if a secondary command is received provided that the last primary command received was a primary talk or listen address of the TMS9914A. A DAC holdoff will occur and the secondary address may be read from the Command Pass Through Register. The holdoff may be released by a 'dacr' auxiliary command and the 'cs' bit of the Auxiliary Command Register is used to indicate that a valid (cs = 1) or an invalid (cs = 0) secondary has been identified by the host MPU. (Set On: ACDS1.SCG.(LPAS + TPAS))
DCAS	Device Clear Active State. This is set when a device clear command (DCL) is received or when a selected device clear (SDC) is received with the TMS9914A in LADS. This will cause a DAC holdoff if unmasked. (Set On: ACDS1.(DCL + SDC.LADS))
SRQ	Service Request. This is provided for the benefit of the controller which should execute a serial poll in response to this interrupt. It is set when the SRQ line becomes true. (Set On: SRQ.($\overline{\text{CIDS + CADS}}$))
MA	**My Address. This is set when the TMS9914A recognizes its primary talk or listen address. A DAC holdoff will occur if this is unmasked. (Set On: (MLA + MTA)·$\overline{\text{SPMS·aptmk}}$)**

IFC	Interface Clear. This is provided for the benefit of devices which are not the System Controller. It is set when the IFC line becomes true and indicates that the TMS9914A has been returned to an idle state. If the device is the System Controller, then the IFC interrupt is not set. (Set On: IFCIN)

2.1.3 Address Status Register

REM	LLO	ATN	LPAS	TPAS	LADS	TADS	ulpa	
D0	D1	D2	D3	D4	D5	D6	D7	MPU BUS

REM	The device is in the remote state
LLO	Local lockout is in operation
ATN	The attention line is low (true) on the bus
LPAS	TMS9914A is in the listener primary addressed state
TPAS	TMS9914A is the talker primary addressed state
LADS(or LACS)	The device is addressed to listen
TADS(or TACS)	The device is addressed to talk
ulpa	This bit shows the LSB of the last address recognized by the TMS9914A.

2.1.4 Address Register

edpa	dal	dat	A5	A4	A3	A2	A1
D0	D1	D2	D3	D4	D5	D6	D7

edpa	Enable dual primary addressing mode
dal	Disable listener function
dat	Disable talker function
A5-A1	Primary address of the TMS9914A

Bits A5-A1 of this register contain the primary address of the device (denoted AAAAA in Table 3-15). IEEE-488 1975/78 does not allow a device to be assigned the value 11111 for bits A5-A1. When 'swrst' is true at power-up or if set by the host MPU, the TMS9914A is held in an idle state. During this time the host MPU may load the primary address of the device into these bits. Often this will be read from an Address Switch Register (see Section 1.5).

The 'edpa' bit is used to enable the dual addressing mode of the TMS9914A. It causes the LSB of the address to be ignored by the address comparator giving two consecutive primary addresses for the device. The address by which the TMS9914A was selected is indicated by the 'ulpa' bit of the Address Status Register.

The Address Register is not cleared by 'swrst' or hardware reset.

2.1.5 Auxiliary Command Register (see Section 3.1 for Auxiliary Command State Diagram)

cs	xx	xx	F4	F3	F2	F1	F0
D0	D1	D2	D3	D4	D5	D6	D7

f4-f0	Auxiliary command select (see Table 2-3)
cs	Clear or set the feature (where applicable)

Auxiliary commands are used to enable and disable most of the selectable features of the TMS9914A and to initiate many of the actions of the device. The desired feature is selected by writing a byte to this register with the appropriate value in bits f4-f0. These values are given in Table 2-3.

The 'cs' bit is used in most cases when the feature selected by f4-f0 is of the clear/set type. Tne feature is enabled if 'cs' = '1' and disabled if 'cs' = '0'. The holdoff on all data (hdfa) feature is an example of such a feature. Other auxiliary commands initiate an action of the TMS9914A, such as release RFD holdoff (rhdf). In most cases, the 'cs' bit is unused and ignored by these commands.

All the clear/set auxiliary commands are cleared by the hardware RESET pin except 'swrst,' which is set true by RESET.

The force group execute trigger (fget) and return to local (rtl) auxiliary commands have a clear/set mode of operation and a pulsed mode of operation. They behave as normal clear/set features, but if they are written with 'cs' = '0' when they have not been previously set, then they will pulse true. Using the 'fget' command in this manner will produce a pulse of approximately 1 μs at the TR pin (with a 5 MHz clock). The 'rtl' command used in this way will cause a return to one of the local states (assuming local lockout is not in force) but the TMS9914A may reenter the remote state next time the listen address occurs (see Section 3.6).

TABLE 2-3 – AUXILIARY COMMANDS

c/s	f4	f3	f2	f1	f0	MNEMONIC	FEATURES
0/1	0	0	0	0	0	swrst	Software reset
0/1	0	0	0	0	1	dacr	Release DAC holdoff
na	0	0	0	1	0	rhdf	Release RFD holdoff
0/1	0	0	0	1	1	hdfa	Holdoff on all data
0/1	0	0	1	0	0	hdfe	Holdoff on EOI only
na	0	0	1	0	1	nbaf	New byte available false
0/1	0	0	1	1	0	fget	Force group execute trigger
0/1	0	0	1	1	1	rtl	Return to local
na	0	1	0	0	0	feoi	Send EOI with next byte
0/1	0	1	0	0	1	lon	Listen only
0/1	0	1	0	1	0	ton	Talk only
na	0	1	0	1	1	gts	Go to standby
na	0	1	1	0	0	tca	Take control asynchronously
na	0	1	1	0	1	tcs	Take control synchronously
0/1	0	1	1	1	0	rpp	Request parallel poll
0/1	0	1	1	1	1	sic	Send interface clear
0/1	1	0	0	0	0	sre	Send remote enable
na	1	0	0	0	1	rqc	Request control
na	1	0	0	1	0	rlc	Release control
0/1	1	0	0	1	1	dai	Disable all interrupts
na	1	0	1	0	0	pts	Pass through next secondary
0/1	1	0	1	0	1	stdl	Short TI settling time
0/1	1	0	1	1	0	shdw	Shadow handshake
0/1	1	0	1	1	1	vstdl	Very short T1 delay
0/1	1	1	0	0	0	rsv2	Request Service Bit 2

2.1.6 Description of Auxiliary Commands

Software Reset (swrst) 0/1xx00000

Setting this command causes the TMS9914A to be returned to a known idle state during which it will not take part in any activity on the GPIB. This auxiliary command is set by the power-on RESET and the chip should be configured while 'swrst' is set. Configuration should include writing the address of the device into the Address Register, writing mask values into the Interrupt Mask Registers and selecting the desired features in the Auxiliary Command Register and Address Register. After this, 'swrst' may be cleared at which point the device becomes logically existent on the GPIB. The Serial Poll Register and Parallel Poll Registers may also be written in this period but this is not necessary if there is no status to report as both of these are cleared by the power-on RESET pin. Table 2-4 lists the various states and other conditions forced by 'swrst'.

TABLE 2-4—SOFTWARE RESET CONDITIONS

MNEMONIC	DESCRIPTION
SIDS	Source idle state
AIDS	Acceptor idle state
TODS	Talker idle state
TPIS	Talker primary idle state
LIDS	Listener idle state
LPIS	Listener primary idle state
NPRS	Negative poll response state
LOCS	Local state
CIDS	Controller idle state
SPIS	Serial poll idle state
PPSS	Parallel poll standby state
ADHS	DAC holdoff state
AEHS	RFD holdoff on end state
SHFS	Source holdoff state
ENIS	END idle state

NOTES: 1. See Section 3 for definition of above.
2. All interrupt status bits are held in a 0 state, but interrupt mask bits are not affected.

Release DAC Holdoff (dacr)0/1xx00001

The Data Accepted (DAC) holdoff allows time for the host microprocessor to respond to unrecognized commands, secondary addresses, and device trigger or device clear commands. The holdoff is released by the MPU when the required action has been taken. Normally the command is loaded with the clear/set bit at zero; however, when used with the address pass through feature CS is set to one if the secondary address was valid or to zero if invalid (see APT interrupt in Section 2.1.2).

Release RFD Holdoff (rhdf)naxx00010

Any Ready For Data (RFD) holdoff caused by a 'hdfa' or 'hdfe' is released.

Holdoff on All Data (hdfa) 0/1xx00011

A Ready For Data (RFD) holdoff is caused on every data byte until the command is loaded with CS set to zero. The handshake must be completed after each byte has been received by the MPU using the 'rhdf' command.

Holdoff on End (hdfe)0/1xx00100

A RFD holdoff will occur when an end of data string message (EOI true with ATN false) is received over the interface. This holdoff must be released using 'rhdf'.

Set New Byte Available False (nbaf)naxx00101

If a talker is interrupted before the byte just stored in the data out register is sent over the interface, this byte will normally be transmitted as soon as the ATN line returns to the false state. If, as a result of the interrupt, this byte is no longer required, its transmission may be suppressed using the 'nbaf' command.

Force Group Execute Trigger (fget)0/1xx00110

The state of the TR output from the TMS9914A is affected when this command is executed. If the CS bit is zero, the line is pulsed high for approximately 5 clock cycles (1 μs at 5 MHz). If CS is one, the TR line goes high until 'fget' is sent with CS equal to zero. No interrupts or handshakes are initiated.

Return to Local (rtl)0/1xx00111

Provided the local lockout (LLO) has not been enabled, the remote/local status bit is reset, and an interrupt is generated (if enabled) to inform the host microprocessor that it should respond to the front panel controls. If the CS bit is set to one the 'rtl' command must be cleared (CS = 0) before the device is able to return to remote control. If CS is set to zero, the device may return to remote without first clearing 'rtl'.

Force End or Identify (feoi) naxx01000

This command causes the EOI message to be sent with the next data byte. The EOI line is then reset.

Listen Only (lon) 0/1xx01001

The listener state is activated until the command is sent with CS set to 0 or until deactivated by a bus command.

Talk Only (ton) 0/1xx01010

The talker state is activated until the command is sent with CS set to 0 or until deactivated by a bus command.

NOTE

> 'ton' and 'lon' are included for use in systems without a controller. However, where the TMS9914A is being used as a controller, it utilizes the 'lon' and 'ton' functions to set itself up as a listener or talker, respectively. Care must therefore be taken to ensure these functions are reset if sending UNL or OTA.

Go to Standby (gts)naxx01011

Issued by the controller in charge to set the ATN line false.

Take Control Synchronously (tcs)naxx01101

Control is again taken by the controller in charge, and ATN is asserted. If the controller is not a true listener, the shadow handshake command must be used to monitor the handshake lines so that the TMS9914A is synchronous with the talker/ listeners and only sends ATN true at the end of byte transfer. This ensures that no data is lost or corrupted.

Request Parallel Poll (rpp)0/1xx01110

This is executed by the controller in charge to send the parallel poll command over the interface (the TMS9914A must be in the Controller Active State so that the Attention line is asserted). The poll is completed by reading the Command Pass Through Register to obtain the status bits, then sending 'rpp' with the CS bit at zero.

Take Control Asynchronously (tca)naxx01100

This command is used by the controller in charge to set the attention line true and to gain control of the interface. The command is executed immediately and data corruption or loss may occur if a talker/listener is in the process of transferring a data byte.

Send Interface Clear (sic)0/1xx01111

The IFC line is set true when this command is sent with CS set to one. This must only be sent by the system controller and should be reset (CS = 0) after the IEEE minimum time for IFC has elapsed (100 μs). The system controller is put into the controller active state.

Send Remote Enable (sre)0/1xx10000

Issued by the system controller to set the REN line true and send the remote enable message over the interface, REN is set false by sending 'sre' with CS at zero.

Request Control (rqc)naxx10001

When the TCT command has been recognized via the unidentified command pass through, this command is sent by the MPU. The TMS9914A waits for the ATN line to go false and then enters the controller active state (CACS).

Release Control (rlc)naxx10010

This command is used after TCT has been sent and handshake completed to release the ATN line and pass control to another device.

Disable All Interrupts (dai)0/1xx10011

The $\overline{\text{INT}}$ line is disabled, but the interrupt registers and any holdoffs selected are not affected.

Pass Through Next Secondary (pts)naxx10100

This feature may be used to carry out a remote configuration of a parallel poll. The parallel poll configure command (PPC) is passed through the TMS9914A as an unrecognized addressed command and is identified by the MPU. The 'pts' command is loaded, and the next byte received by the TMS9914A is passed through via the Command Pass Through Register. This would be the parallel poll enable (PPE), which is read by the microprocessor.

Set T1 Delay (std1)1xx10101

The T1 delay time can be set to 6 clock cycles (1.2 μs at 5 MHz) if this command is sent with the CS bit at one. The Tl delay time is 11 clock cycles (2.2 μs at 5 MHz) following a power-on reset or if the command is sent with CS set to zero.

Shadow Handshake (shdw)0/1xx10110

This feature enables the controller in charge to carry out the listener handshake without participating in a data transfer. The Data Accepted line (DAC) is pulled true a maximum of 3 clock cycles after Data Valid (DAV) is received, and Not Ready For Data (NRFD) is allowed to go false as soon as DAV is removed.

The shadow handshake function allows the 'tcs' command to be synchronized with the Acceptor Not Ready State (ANRS) so that ATN can be re-asserted without causing the loss or corruption of data byte. The END interrupt can also be received and causes a RFD holdoff to be generated.

Very Short T1 Delay (vstd1)0/1xx10111

If this feature is enabled, the GPIB settling time (T1) will be reduced to 3 clock cycles (600 ns at 5 MHz) on the second and subsequent data bytes when ATN is false. Otherwise, the GPIB settling time is determined by the std1 feature.

Request Service Bit 2 (rsv2)0/1xx11000

The rsv2 bit performs the same function as the rsv1 bit (see Section 2.1.8) but provides a means of requesting service which is independent of the Serial Poll Register.

This allows minor updates to be made to the Serial Poll Register without affecting the state of the request service.

In addition, rsv2 is cleared when the serial poll status byte is sent to the controller during a serial poll. It is therefore used in situations where a service request is simply a request from an instrument for the controller to poll its status. As soon as this happens, rsv2 is cleared since the reason for requesting service has been satisfied. This eliminates the burden of clearing the bit from the host MPU but also guarantees that rsv2 is cleared before another serial poll can occur. If this were not so, there would be a possibility of a second status byte being sent with the RQS message true, which could result in confusion for the controller. (rsv2 is cleared on: SPAS.(APRS1 + APRS2).STRS). It should be noted that the vstd1 and rsv2 features were not present on the TMS9914.

2.1.7 Bus Status Register

ATN	DAV	NDAC	NRFD	EOI	SRQ	IFC	REN	
D0	D1	D2	D3	D4	D5	D6	D7	MPU BUS

The host MPU may examine the status of the GPIB management lines at the time of reading.

The IFC bit of this register does not indicate a true value if the device is a system controller using the 'sic' auxiliary command.

2.1.8 Serial Poll Register

S8	rsv1	S6	S5	S4	S3	S2	S1	
DIO8	DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	GPIB
D0	D1	D2	D3	D4	D5	D6	D7	MPU BUS

S8, S6-S0	Device status
rsv1	Request service bit 1

Bits S8, S6-S1 of this register are sent out over the GPIB when the device is addressed during a serial poll. They are cleared by a hardware reset but not by 'swrst' and may therefore be set up during configuration of the chip. These bits are fully double buffered and if the register is written to while the device is addressed during a serial poll (serial poll active state, SPAS), the value written is saved, and these bits are updated when SPAS is terminated.

The rsv1 bit provides an input to the service request function of the TMS9914A and is used to instruct this to request that the controller service the device. When rsv1 is set true, the SRQ line is pulled true on the GPIB, and the controller typically responds by setting up a serial poll to obtain the status of all instruments on the bus that may require service. When the TMS9914A is addressed to send its status byte, SRQ is set false, and the status byte is sent with the SRQ message true on DIO7. The rsv1 bit must then be cleared and set true again if service is to be requested a second time. The SPAS interrupt is set immediately following the status byte being sent.

The rev1 bit is also cleared by the hardware reset pin but not by 'swrst'. It is not double-buffered but the service request function comprehends changes in the state of rsv1 while the device is in SPAS. The Serial Poll Register may therefore be written to any time. Section 3.5 contains more information on the service request function. Note that the rev1 bit of the TMS9914 was simply referred to as 'rsv' since this device did not have an rsv2 auxiliary command.

2.1.9 Command Pass Through Register

DIO8	DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	GPIB
D0	D1	D2	D3	D4	D5	D6	D7	MPU BUS

This provides a means of directly inspecting the GPIB data lines (DIO(8-1)). It has no storage and should only be used when the data lines are known to be in a steady state such as will occur during a DAC holdoff or in CPWS during a parallel poll. It is used to read unrecognized commands and secondaries following a UNC interrupt or to read secondary addresses following an APT interrupt. In addition, an active controller uses this register to read the results of a parallel poll at least 2μs after setting the 'rpp' auxiliary command.

2.1.10 Parallel Poll Register

PP8	PP7	PP6	PP5	PP4	PP3	PP2	PP1	
DIO8	DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	GPIB
D0	D1	D2	D3	D4	D5	D6	D7	MPU BUS

When a controller initiates a parallel poll, the contents of this register are presented to the GPIB data lines. If all bits of the register are cleared, then none of the lines DIO(8-1) will be pulled low during a parallel poll which corresponds to the Parallel Poll Idle State (PPIS) of IEEE-488. If it is desired to participate in a parallel poll, then the bit corresponding to the desired parallel poll response is set to a 1.

The Parallel Poll Register is fully double buffered. If it is written to during a parallel poll, the new value is held until the parallel poll ends, at which point the register is updated. This permits the host MPU to update the parallel poll response completely asynchronously to the GPIB.

If this register is cleared by the hardware RESET pin but not by 'swrst,' it may be loaded while the chip is being configured with 'swrst' set.

2.1.11 Data In Register

DIO8	DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	GPIB
D0	D1	D2	D3	D4	D5	D6	D7	MPU BUS

This register is used to hold data received by the TMS9914A when it is a listener. It is loaded during Accept Data State (ACDS1) and, following this, an RFD holdoff will occur. This will normally be released when the byte is read by the host MPU, but if the Holdoff On All Data (hdfa) feature is selected, this holdoff must be released by the 'rhdf' auxiliary command.

If the Holdoff On End (hdfe) feature is selected, the RFD holdoff will be released by reading the Data In Register. But if the EOI line is true when the byte is received, reading the data byte will not release the holdoff and rhdf must be used.

As the Data In Register is loaded, the BI interrupt is set. The END interrupt is set simultaneously if the byte is accompanied by a true EOI line.

2.1.12 Data Out Register

DIO8	DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	GPIB
D0	D1	D2	D3	D4	D5	D6	D7	MPU BUS

The Data Out register is used by a controller or talker for sending interface messages and device dependent messages. When the TMS9914A enters the Talker Active State (TACS) or the Controller Active State (CACS), the contents of the Data Out Register are presented to the GPIB data lines (DIO(8-1)), and the byte is sent over the bus under the control of the Source Handshake. Each time a byte is written, the source handshake is enabled, and the byte is sent. If the handshake is interrupted before the byte can be sent, then it will be sent next time the Source Handshake becomes active unless a new byte available false (nbaf) auxiliary command is written. This has the effect of clearing an unsent byte from the Data Out Register, and although the register itself is not cleared, the TMS9914A behaves as if it had not been loaded.

Each time the source handshake becomes active and there is no unsent byte in the Data Out Register, a BO interrupt will occur informing the host MPU that the Data Out Register is available for use.

The Data In Register and Data Out Register operate independently. The Data Out Register is not double buffered, and its contents are output directly to the data lines of the GPIB.

2.2 DIRECT MEMORY ACCESS

The TMS9914A can operate in DMA using the $\overline{ACCRQ}$ (DMA request) and $\overline{ACCGR}$ (DMA grant) DMA handshake lines. The operation is automatic within the TMS9914A and needs no 'mpu' configuration.

The $\overline{ACCRQ}$ signal is set by $(BO.\overline{CACS}+BI)$ and can therefore not be used by a controller while ATN is asserted. It is reset by 'swrst' readin data in register, writing to the data out register and $\overline{ACCGR}$. It is not cleared by reading interrupt status register 0.

If using DMA, the internal CE and addressing is disabled by the $\overline{ACCGR}$ signal going low and $\overline{ACCGR}$ will automatically address either the data in register (DBIN = 0) or the data out register (DBIN = 1).

NOTE

The sense of DBIN is inverted for DMA operation.

At the end of a DMA read from memory sequence, the $\overline{ACCRQ}$ will be left low (also BO bit set). It may be necessary for the 'mpu' to clear this in some circumstances, e.g., starting DMA write to memory sequence.

In DMA it is recommended that the MA interrupt be unmasked to prevent errors due to interrupted data streams.

If DMA is not being utilized, the $\overline{ACCGR}$ signal must be held high. In this case, the $\overline{ACCRQ}$ signal can be used as a separate interrupt line for BO and BI. This allows faster 'mpu' transfers to take place as it is not necessary to read the interrupt register to find the cause of the interrupt. Figure 2-2 shows a typical DMA configuration.

2.3 TERMINAL ASSIGNMENTS AND FUNCTIONS

The IEEE-488 standard uses the negative logic convention for the GPIB lines. The FALSE state (0) is represented by a high voltage (> 2.0 V); the TRUE state (1) is represented by a low voltage (> 0.8 V). The GPIB terminations of the TMS9914A are in agreement with this convention. For example, if Data Valid is true (1), the DAV line is pulled low by the device. These terminations are connected to the bus via noninverting buffers to obtain the correct signal polarity.

Note that the terminations on the microprocessor side of the device are in positive logic (true state (1) = high voltage : false state (0) = low voltage). This is in agreement with the logic convention used by most microprocessors. Thus if:

D0(MSB) D7(LSB)

0	1	1	0	1	0	0	1

is written into the data out register, it will appear as:

DIO8(MSB) DIO1(LSB)

HIGH	LOW	LOW	HIGH	LOW	HIGH	HIGH	LOW

on the IEEE-488 D10 lines.

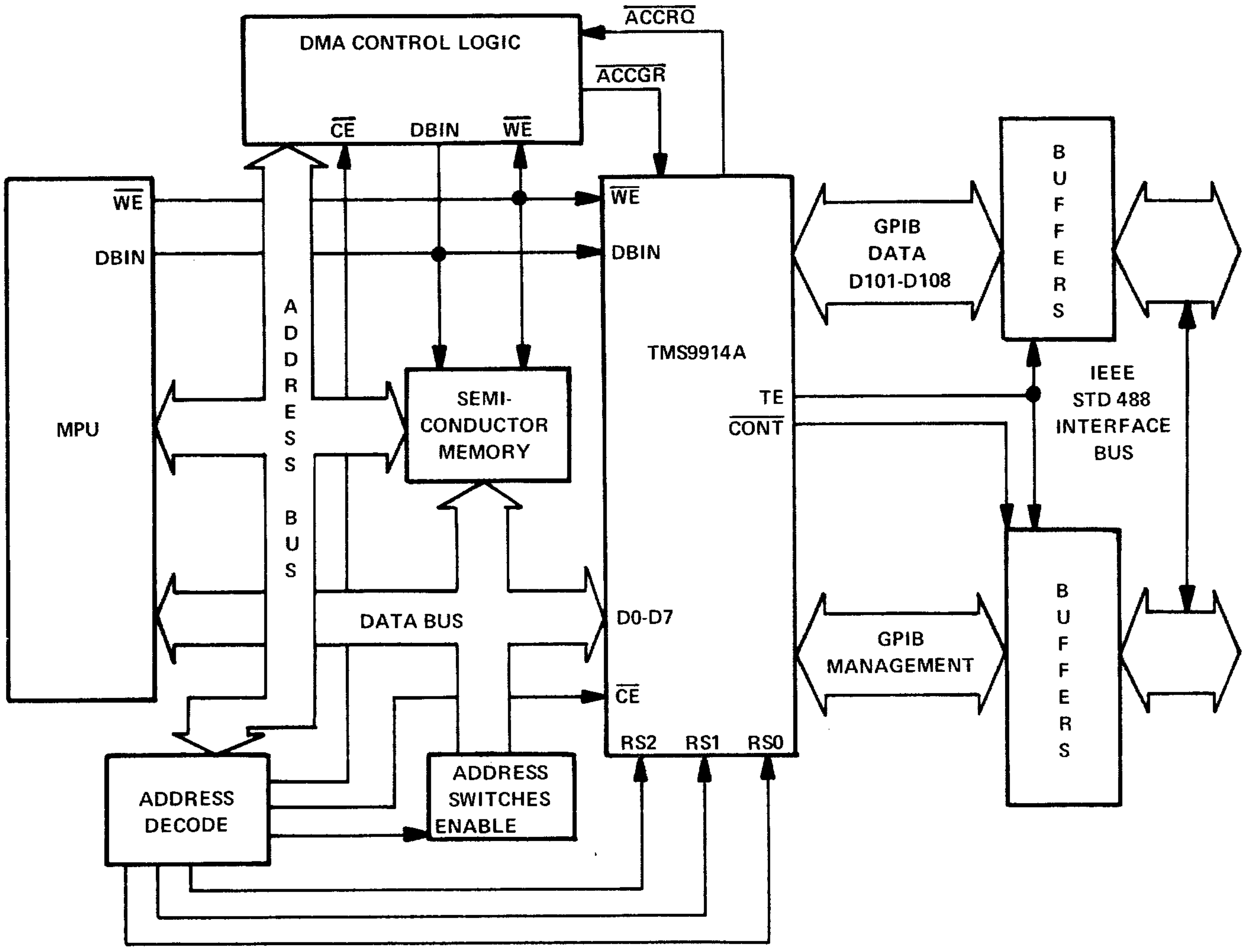
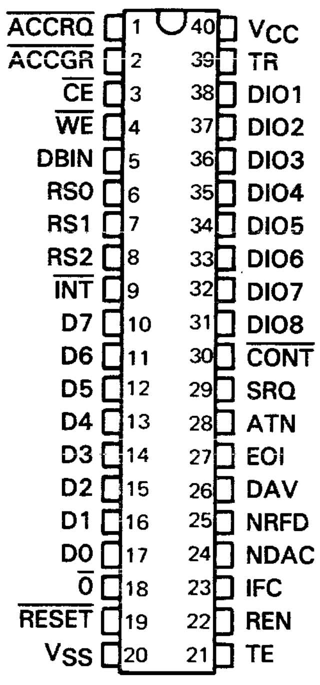
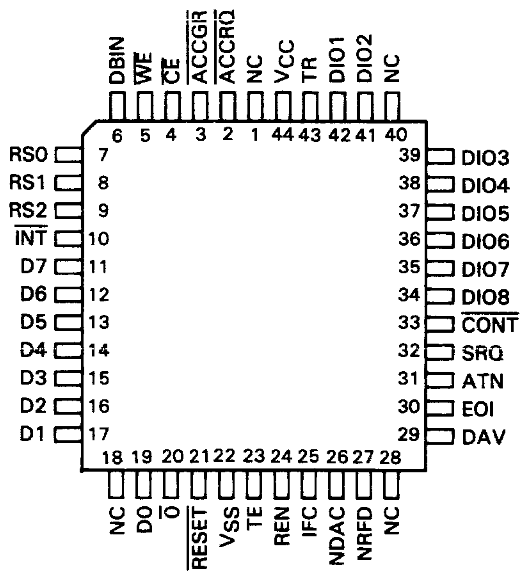


FIGURE 2-2 – DMA CONFIGURATION

TERMINAL ASSIGNMENTS AND FUNCTIONS

SIGNATURE	PIN N/FN	I/O (TYPE)	DESCRIPTION
DIO8	31/34	I/O(p(p)	DIO8 through DIO1 are the data input/output lines on the GPIB side. These pins connect to the IEEE-488 bus via non-inverting transceivers.
DIO7	32/35	I/O(p/p)	N PACKAGE (TOP VIEW)  ACCRQ 1 ACCGR 2 CE 3 WE 4 DBIN 5 RS0 6 RS1 7 RS2 8 INT 9 D7 10 D6 11 D5 12 D4 13 D3 14 D2 15 D1 16 D0 17 0 18 RESET 19 VSS 20 40 VCC 39 TR 38 DIO1 37 DIO2 36 DIO3 35 DIO4 34 DIO5 33 DIO6 32 DIO7 31 DIO8 30 CONT 29 SRQ 28 ATN 27 EOI 26 DAV 25 NRFD 24 NDAC 23 IFC 22 REN 21 TE
DIO6	33/36	I/O(p/p)	
DIO5	34/37	I/O(p/p)	
DIO4	35/38	I/O(p/p)	
DIO3	36/39	I/O(p/p)	
DIO2	37/41	I/O(p/p)	
DIO1	38/42	I/O(p/p)	
DAV	26/29	I/O(p/p)	DATA VALID: handshake line controlled by source to show acceptors when valid data is present to the bus.
NDAC	24/26	I/O(p/p)	NOT DATA ACCEPTED: handshake line. Acceptor sets this false (high) when it has latched the data from the I/O lines.
NRFD	25/27	I/O(p/p)	NOT READY FOR DATA: handshake line. Sent by acceptor to indicate readiness for the next byte.
ATN	28/31	I/O(p/p)	ATTENTION: sent by controller in charge. When true (low), interface commands are being sent over the DIO lines. When false (high), these lines carry data.
REN	22/24	I/O(o/d)	REMOTE ENABLE: sent by system controller to select control either from the front panel or from the IEEE bus.
IFC	23/25	I/O(o/d)	INTERFACE CLEAR: sent by the system controller to set the interface system into a known quiescent state. The system controller becomes the controller in charge.
SRQ	29/32	I/O(p/p)	SERVICE REQUEST: set true (low) by a device to indicate a need for service.
EOI	27/30	I/O(p/p)	END OR IDENTIFY: if ATN is false (high), this indicates the end of a message block. If ATN is true (low), the controller is requesting a parallel poll.
$\overline{\text{CONT}}$	30/33	O(p/p)	Indicates if a device is controller in charge. It is used to control direction of SRQ and ATN in pass control systems. Logically, it is (CIDS + CADS).
TE	21/23	O(p/p)	TALK ENABLE: controls the direction of the transfer of the line transceivers. Logically, it is: (CACS + TACS + EIO·ATN·(CIDS + CADS). $\overline{\text{SWRST}}$)
D0	17/19	I/O(p/p)	Data transfer lines on the MPU side of the device.
D1	16/17	I/O(p/p)	
D2	15/16	I/O(p/p)	FN PACKAGE (TOP VIEW)  DBIN 6 WE 5 CE 4 ACCGR 3 ACCRQ 2 NC 1 VCC 44 TR 43 DIO1 42 DIO2 41 NC 40 RS0 7 RS1 8 RS2 9 INT 10 D7 11 D6 12 D5 13 D4 14 D3 15 D2 16 D1 17 39 DIO3 38 DIO4 37 DIO5 36 DIO6 35 DIO7 34 DIO8 33 CONT 32 SRQ 31 ATN 30 EOI 29 DAV NC 18 D0 19 O 20 RESET 21 VSS 22 TE 23 REN 24 IFC 25 NDAC 26 NRFD 27 NC 28
D3	14/15	I/O(p/p)	
D4	13/14	I/O(p/p)	
D5	12/13	I/O(p/p)	
D6	11/12	I/O(p/p)	
D7	10/11	I/O(p/p)	
RS0	6/7	I	REGISTER SELECT LINES: determine which register is addressed by the MPU during a read or write operation
RS1	7/8	I	
RS2	8/9	I	
$\overline{\text{CE}}$	3/4	I	CHIP ENABLE: $\overline{\text{CE}}$ low allows access of read and write registers. If $\overline{\text{CE}}$ is high, D0-D7 are in high impedance unless $\overline{\text{ACCGR}}$ is low.
$\overline{\text{WE}}$	4/5	I	WRITE ENABLE: when active (low), indicates to the TMS9914A that data is being written to one of its registers.
DBIN	5/6	I	DATA BUS IN: an active (high) state indicates to the TMS9914A that a read is about to be carried out by the MPU.
$\overline{\text{INT}}$	9/10	O(o/d) (no pullup)	INTERRUPT: sent to the MPU to cause a branch to a service routine.

TERMINAL ASSIGNMENTS AND FUNCTIONS (continued)

SIGNATURE	PIN N/FN	I/O (TYPE)	DESCRIPTION
$\overline{\text{ACCRQ}}$	1/2	O(p/p)	ACCESS REQUEST: this pin becomes active (low) to request a direct memory access.
$\overline{\text{ACCGR}}$	2/3	I	ACCESS GRANTED: when received from the direct memory access control logic this enables the byte onto the data bus. ACCGR must be high when not participating in DMA transfer.
$\overline{\text{RESET}}$*	19/21	I	INITIALIZES the TMS9914A at power-on.
TR	39/43	O(p/p)	TRIGGER: activated when the GET command is received over the interface or the fget command is given by the MPU.
$\overline{0}$	18/20	I	CLOCK input: 500 kHz to 5 MHz. Need not be synchronous to system clock.
V_{SS}	20/22		Ground reference voltage.
V_{CC}	40/44		Supply voltage (+5 V nominal).

(p/p) = push/pull output.
(o/d) = open drain output with no internal pull up.
* The hardware RESET pin has the following effect on the TMS9914A:
- Serial and Parallel Poll registers cleared
- All clear/set auxiliary commands cleared except 'swrst'
- 'swrst' auxiliary command set. This holds the TMS9914A in known states, as described in Section 2.1.6.

2.4 TRANSCEIVER CONNECTIONS

There are three linear transceivers designed to work with the TMS9914A: the SN75160, SN75161, and SN75162. Data sheets for these are included as Appendix C. Figure 2-3 shows the possible transceiver connections. Note that there is a corresponding pinout between the TMS9914A and the transceivers. This allows the whole GPIB interface to be laid out in a very small area of printed circuit board.

The SN75160 is a 20 pin device used to buffer the IEEE-488 data lines (DIO(8-1)) in all applications. The direction of the buffers is controlled by the Talk Enable (TE) output of TMS9914A. This active high signal becomes true whenever there is an interface function of the TMS9914A not sending the NUL message on DIO(8-1), that is, when the device is in TACS, CACS, SPAS, or PPAS. The Pull-Up Enable (PE) input of the SN75160 is an active high input which selects whether the 'DIO(8-1)' lines are driven by open collector or push/pull buffers. A push/pull buffer is required if faster data rates are required and the 'stdl' and/or the 'vstdl' features are used. Open collectors must be used if parallel polling is being used in a particular GPIB environment. If only one of these features is desired the PE input may be hardwired otherwise it must be derived from ATN and EOI, as shown in Figure 2-3.

The SN75161 is a 20-pin device used to buffer the IEEE-488 management lines. It may be used for a talker/listener device or for a controller which does not pass control. The direction of the handshake line buffers NRFD, NDAC, DAV are again controlled by the TE signal. However, the SRQ, ATN, REN, and IFC buffers are controlled by the DC input of the SN75161, which connects to the Controller Active ($\overline{\text{CONT}}$) output of the TMS9914A. $\overline{\text{CONT}}$ becomes low whenever the TMS9914A is an active controller, that is, when it is not in CIDS or CADS. The SN75161 also includes the logic necessary to control the direction of the EOI buffer. This is dependent on the TE signal when ATN is false (high) and the DC signal when ATN is true (low).

The SN75162 is a 22-pin device which may be used to buffer the IEEE-488 management lines in all applications including devices which pass control. The SN75162 has a separate pin to control the direction of the REN and IFC buffers, but is otherwise identical to the SN75161 in all other respects. This input is the System Controller input (SC) which may be hardwired or switchable to determine whether or not the instrument in question is a system controller or not. Note that a device which has its buffers configured as a non-system controller should never use the 'sic' and 'sre' auxiliary commands for reasons explained in Section 3.8.3.

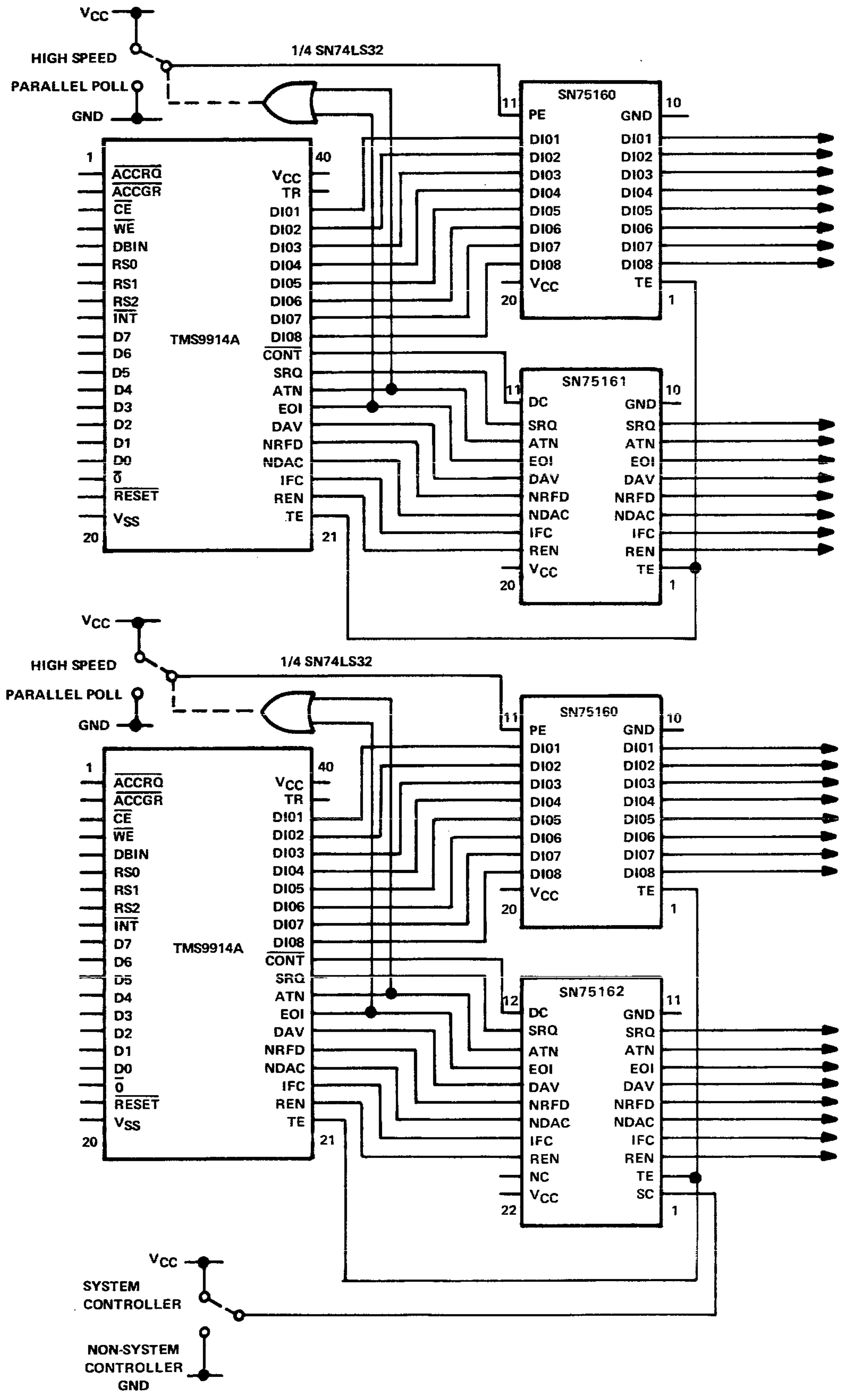


FIGURE 2-3 – TRANSCEIVER CONNECTIONS

3. STATE DIAGRAM IMPLEMENTATION

This section presents the state diagrams for the TMS9914A.

Where equivalent, the names of TMS9914A states are the same as those of IEEE-488. In some cases, IEEE-488 states have been divided, for example, ACDS of the IEEE-488 has been split into ACDS1 and ACDS2. The convention of lower case characters for local messages and upper case for remote messages and interface states is retained.

State diagrams with remote message outputs are supplemented with tables. T is used to represent a true output and F a false output. Parentheses denote a passive output; otherwise, it is active. The outputs shown are the values presented to the bus and assume the use of the SN75160 and SN75161 or SN75162 transceivers or their logical equivalents. The symbol (NUL) associated with DIO(1-8) indicates that each of these lines is sent passive false by the function in question.

NOTE

> An arrow into a state with no state as its origin represents a transition from every other state on the diagram. Note, however, that this does not imply that all exit conditions from the destination state are overridden. If such an entry condition is true and, simultaneously, an exit condition is true then this represents an illegal situation and should be avoided. Such situations will not occur in normal operation of the device.

No maximum timings are discussed. The TMS9914A with its recommended transceivers meets all IEEE-488 maximum timing requirements as may be determined from Section 4.4 and Appendix C. If the TMS9914A is used with other transceivers, then it must be ensured that these requirements are still met.

3.1 AUXILIARY COMMANDS

There are two basic types of commands implemented in the auxiliary command register: immediate execute and clear/set.

The clear/set commands are used to enable and disable the various features of the TMS9914A. The particular feature is selected by the code on f0-f4 (see Section 2.1.6) and it is set or cleared according to the value on the cs bit. For the purposes of the state diagrams, the mnemonic of a clear/set command simply represents its current state.

The immediate execute auxiliary commands remain active for the duration of a strobe signal after the auxiliary command register has been written to. This is represented in the form of a state diagram in Figure 3-1. Note that writes to the auxiliary command register must be spaced by at least five clock cycles. For the purposes of the remaining state diagrams, the immediate execute commands are represented as the mnemonic gated by the auxiliary command strobe state (AXSS).

The clear/set bit of the auxiliary command register is used by several of the immediate execute commands, for example, 'dacr' uses it to differentiate between valid and not valid secondary addresses when releasing a DAC holdoff on a secondary address. The 'lon' and 'ton' auxiliary commands are also considered immediate execute, as described in Section 3.4.

The 'fget' and 'rtl' auxiliary commands are both immediate execute and clear/set. They may be cleared or set in the normal way, but if they are cleared when they are already in the false state, they will pulse true for the duration of AXSS. In the following state diagrams, however, these are simply included in their clear/set form.

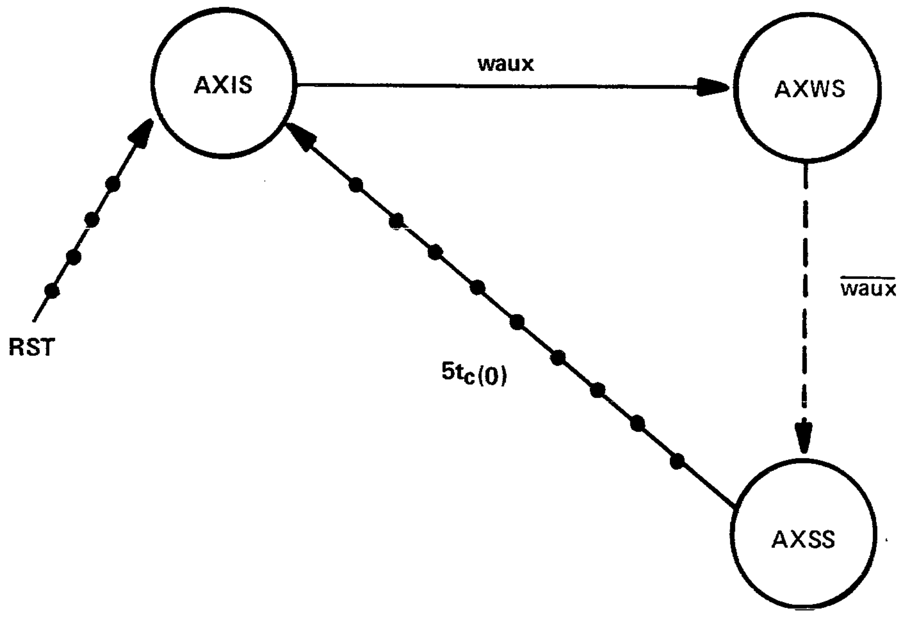


FIGURE 3-1 – TMS9914A AUXILIARY COMMAND STATE DIAGRAM

TABLE 3-1 – AUXILIARY COMMAND STATE DIAGRAM MNEMONICS

MESSAGES			STATES		
waux	=	write to auxiliary command register	AXIS	=	auxiliary command register idle state
$t_c(0)$	=	clock cycle time	AXWS	=	auxiliary command write state
			AXSS	=	auxiliary command strobe state

3.2 ACCEPTOR HANDSHAKE

The TMS9914A acceptor handshake is shown in Figure 3-2. The main variation from IEEE-488 to note is that the device remains in AIDS while the controller function is in CACS. The TMS9914A, therefore, does not monitor the commands which it sends over the bus and this places some restrictions on the user which are outlined in Section 3.8.

The accept data state of IEEE-488 (ACDS) is divided into two states. The first, (ACDS1) is used to strobe data into the Data In Register or to sequence the decoding of commands from the bus. All interrupts generated by the acceptor handshake (GET, MA, MAC, DCAS, APT, UCG, BI, and END) are generated by this state. The second (ACDS2) is used as a holding state where the device will remain in the event of a DAC holdoff.

As discussed in Section 2.1.2, certain of the commands will cause interrupts in ACDS1 and, if the interrupts are unmasked, a DAC holdoff will occur. The interrupts concerned are GET, MA, DCAS, UCG, and APT. This is represented in the state diagram by the signal SAHF which becomes true when one of the above interrupts is set if it is unmasked. It persists for the duration of ACDS1. This event is stored by causing the ADHS to become active which inhibits the transition from ACDS2 to AWNS. ADHS is cleared by 'dacr'. Table 3-15 shows the response of the TMS9914A to the various bus commands.

If a GET command is received in ACDS1, then the TR pin will be set high. This high condition persists throughout ACDS1 and ACDS2, which means that if a DAC holdoff occurs, the TR pin will remain high until the holdoff is released by a 'dacr' auxiliary command.

Two additional state diagrams are included to record the type of data received in ACDS1 when ATN is false. ANHS indicates that a data byte has been received and that an RFD holdoff should be caused before the next data byte is accepted. The holdoff may be released by reading the Data In Register unless the 'hdfa' feature is enabled in which case 'rhdf' must be used. AEHS shows that the last data byte was accepted with the EOI message true and the 'hdfe' feature set. This will cause an RFD holdoff which can only be released by 'rhdf'.

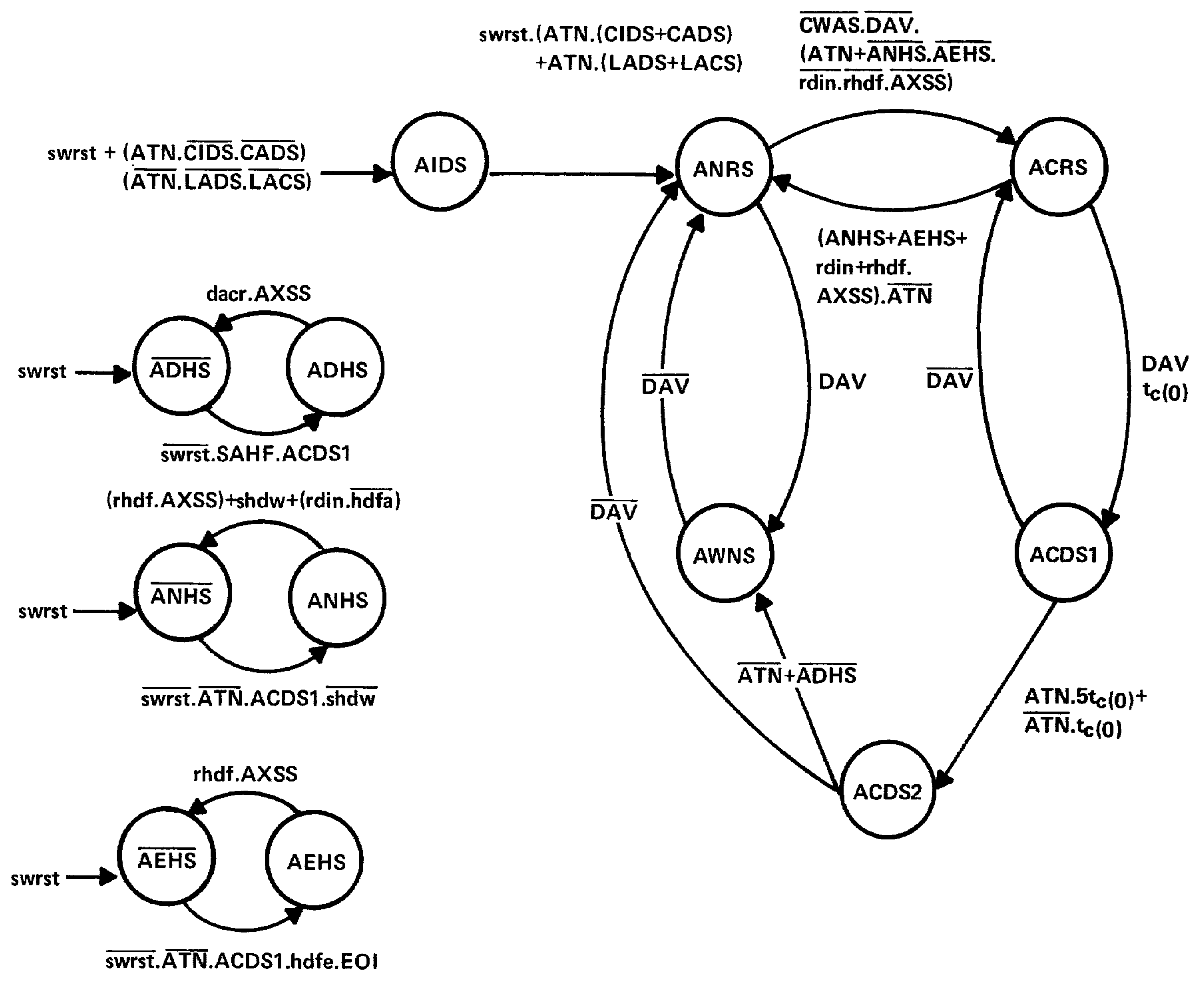


FIGURE 3-2 – TMS9914A ACCEPTOR HANDSHAKE STATE DIAGRAM

TABLE 3-2 – ACCEPTOR HANDSHAKE MNEMONICS

MESSAGES			STATES		
swrst	=	software reset	AIDS	=	acceptor idle state
dacr	=	DAC release	ANRS	=	acceptor not ready state
rhdf	=	release RFD holdoff	ACRS	=	acceptor ready state
shdw	=	shadow handshake	ACDS1	=	accept data state 1
rdin	=	read data in register	ACDS2	=	accept data state 2
hdfe	=	enable RFD holdoff after END messages received	AWNS	=	acceptor wait for new cycle state
hdfa	=	enable RFD holdoff on all data	ADHS	=	accept data holdoff state
ATN	=	attention	ANHS	=	acceptor not ready holdoff state
DAV	=	data valid	AEHS	=	acceptor not ready holdoff after 'END'
EOI	=	end or identify state	CWAS	=	controller wait for ANRS state (controller function)
RFD	=	ready for data	AXSS	=	auxiliary command strobe state (auxiliary command register)
DAC	=	data accepted	LADS	=	listener addressed state (listener function)
SAHF	=	set accept data holdoff state	LACS	=	listener active state (listener function)
$t_{c(0)}$	=	clock cycle time	CIDS	=	controller idle state (controller function)
			CADS	=	controller addressed state (controller function)

TABLE 3-3 – ACCEPTOR HANDSHAKE MESSAGE OUTPUTS

STATE	REMOTE MESSAGES SENT		OTHER ACTIONS	
	RFD	DAC		
AIDS	(T)	(T)		
ANRS	F	F		
ACRS	(T)	F		
ACDS1	F	F	ATN False:	– data entered into Data In Register – BI interrupt generated – end interrupt generated if EOI is true.
			ATN true:	– commands decoded – command related interrupts set – sahf set if command requires a DAC holdoff – TR pin set true if GET message is received – 'pts' feature cleared after UNC interrupt set
ACDS2	F	F	TR	pin set true if GET command was received in ACDS1
AWNS	F	(T)		

3.3 SOURCE HANDSHAKE

The TMS9914A source handshake state diagram is shown in Figure 3-3. IEEE-488 states SIWS and SWNS have been removed. These record the false then true transition of 'nba' (new byte available) as the old data byte is removed and a new data byte is made ready. Instead the TMS9914A uses a separate state (SHFS) to record the availability of a data byte in the Data Out Register. This state is exited when a byte is written to the Data Out Register which enables the transition from SGNS to SDYS and the subsequent transmission of the byte. The SHFS is reentered as the byte is sent in STRS, but if the handshake is interrupted before this, then the fact that the byte has not been sent is recorded until the source handshake again becomes active. If, however, the byte in the data out register is to be disregarded, then 'nbaf' may be used to return the device to SHFS.

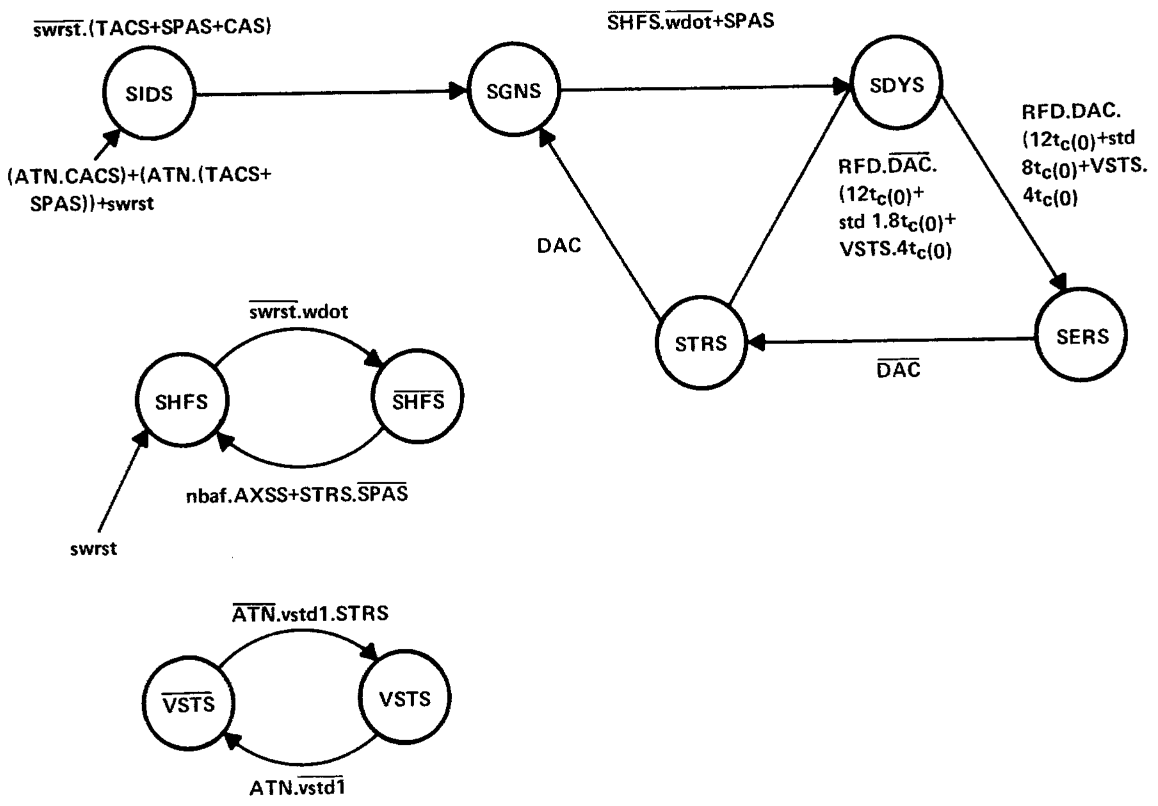
The status byte in the Serial Poll Register is continually available. The transition from SGNS to SDYS is not dependent on SHFS during a serial poll, that is, while SPAS is active. By separately recording the availability of a byte in the Data Out Register, a talker sending data may be interrupted for a serial poll without risk of a byte being lost.

The additional state SERS is included to detect an error condition on the bus. This will be entered when the source handshake tries to send a byte but finds both the NRFD and NDAC lines false at the same time. This condition will normally indicate for a controller that there are no devices powered up on the bus, or for a talker that there are no devices addressed to listen on the bus.

The state VSTS will be entered after the first data byte of a talker has been sent if the 'vstdl' feature is enabled. This enables a very short bus settling time ($4t_{c(0)}$) for all subsequent bytes until ATN next becomes true. The TMS9914A will not use the short bus settling time when it is an active controller.

NOTE

The TMS9914 did not implement the 'VSTS'. The bus settling time was therefore either $8t_{c(0)}$ or $12t_{c(0)}$ for 'stdl' set and not set respectively.



$\overline{\text{swrst}}$.(TACS+SPAS+CAS)
SIDS
SGNS
SDYS
$\overline{\text{SHFS}}.\overline{\text{wdot}}$+SPAS
(ATN.CACS)+(ATN.(TACS+SPAS))+swrst
RFD.DAC.(12$t_{c(0)}$+std 8$t_{c(0)}$+VSTS.4$t_{c(0)}$
RFD.$\overline{\text{DAC}}$.(12$t_{c(0)}$+std 1.8$t_{c(0)}$+VSTS.4$t_{c(0)}$
DAC
STRS
SERS
$\overline{\text{DAC}}$
$\overline{\text{swrst}}$.wdot
SHFS
$\overline{\text{SHFS}}$
nbaf.AXSS+STRS.$\overline{\text{SPAS}}$
swrst
$\overline{\text{ATN}}$.vstd1.STRS
$\overline{\text{VSTS}}$
VSTS
ATN.$\overline{\text{vstd1}}$

FIGURE 3-3 – TMS9914A SOURCE HANDSHAKE STATE DIAGRAM

TABLE 3-4 – SOURCE HANDSHAKE MNEMONICS

MESSAGES			STATES		
swrst	=	software reset	SIDS	=	source idle state
nbaf	=	new byte available false	SGNS	=	source generate state
wdot	=	write to the data out register	SDYS	=	source delay state
stdl	=	enable short bus settling time	SERS	=	source error state
vstdl	=	enable very short bus settling time	STRS	=	source transfer state
ATN	=	attention	SHFS	=	source holdoff state
RFD	=	ready for data	VSTS	=	very short bus settling time state
DAC	=	data accepted	TACS	=	talker active state (talker function)
$t_{c(0)}$	=	clock cycle time	CACS	=	controller active state (controller function)
			SPAS	=	serial poll active state (talker function)
			AXSS	=	auxiliary command strobe state (auxiliary command register)

TABLE 3-5 – SOURCE HANDSHAKE MESSAGE OUTPUTS

STATE	REMOTE MESSAGES SENT DAV	OTHER ACTIONS
SIDS	(F)	BO interrupt and ACCRQ set
SGNS	F	true if SHFS is false and SPAS is not true
SDYS	F	
SERS	F	ERR interrupt set true
STRS	T	

3.4 TALKER AND LISTENER FUNCTIONS

Figures 3-4 and 3-5 show the TMS9914A listener and talker state diagrams, which serve the purpose of the listener and talker or extended listener and extended talker functions of IEEE-488, depending on the state of the APT interrupt mask bit.

The TMS9914A does not recognize secondary addresses on-chip and these must be passed through to the host MPU for verification. Secondary addressing is enabled by unmasking the APT interrupt. A secondary address will cause this interrupt if the last primary command received was a primary address of the device, that is, it is in TPAS or LPAS. A DAC holdoff will also occur. The host MPU must respond to the interrupt by reading the secondary from the Command Pass Through Register and identifying it as being valid or not valid. The holdoff may then be released with a 'dacr' auxiliary command, the sense of the 'cs' bit being used to indicate a valid (cs = 1) or not valid (cs = 0) secondary. If a valid secondary address is indicated then the TMS9914A will enter TADS or LADS depending on whether it is in TPAS or LPAS.

The 'lon' and 'ton' auxiliary commands together with the clear/set bit (cs) have a direct influence on the appropriate state diagrams. Therefore, although they appear as ordinary clear/set auxiliary commands, they can be effectively cleared by other bus events. For example, if a TMS9914A addresses itself as a listener via the 'lon' command it may be returned to LIDS by an UNL command from the bus at a later time.

The 'lon' and 'ton' auxiliary commands are used to implement two features of IEEE-488. First, talk only and listen only are used in situations where there is no active controller on the bus. Note that the 'lon' and 'ton' commands are linked with these features to indicate to the user that these commands are not enabled by CAS as are 'ltn' and 'lun' of IEEE-488.

Second, the 'lon' and 'ton' auxiliary commands are used by an active controller to address itself. IEEE-488 provides for a controller to address itself to listen via the 'ltn' and 'lun' message but there is no corresponding message for the talker. Hence, when a controller addresses itself to talk via 'ton,' it must send its talk address over the bus and similarly, if it sends another talk address over the bus then it must un-address itself by writing 'ton' false.

When the TMS9914A enters SPAS, the contents of the serial poll register are sampled and presented on DIO(8-1). These will remain unchanged until SPAS is exited. The source handshake will, however, send this status byte as many times as the controller will accept it.

The internal IFC signal of the TMS9914A (IFCIN) is suppressed when the device itself is sending IFC in order to simplify implementation of the controller function (see Section 3.8.3). Therefore, the send interface clear (sic) auxiliary command is included with IFCIN to return the talker and listener functions to their idle states and allow a system controller to clear its own interface.

A separate state diagram is included to control the sending of the END message of IEEE-488. If the 'feoi' auxiliary command is written followed by loading a byte into the Data Out Register, the TMS9914A will enter ERAS, and the EOI line will be asserted as 'DIO(8-1)' begin to change. The function will enter ENAS as soon as the source handshake begins to send this byte, and EOI will be released when the Data Out Register is next loaded. If it is desired to send EOI true with the next byte as well, then 'feoi' may be written before the Data Out Register returns the device to ERAS.

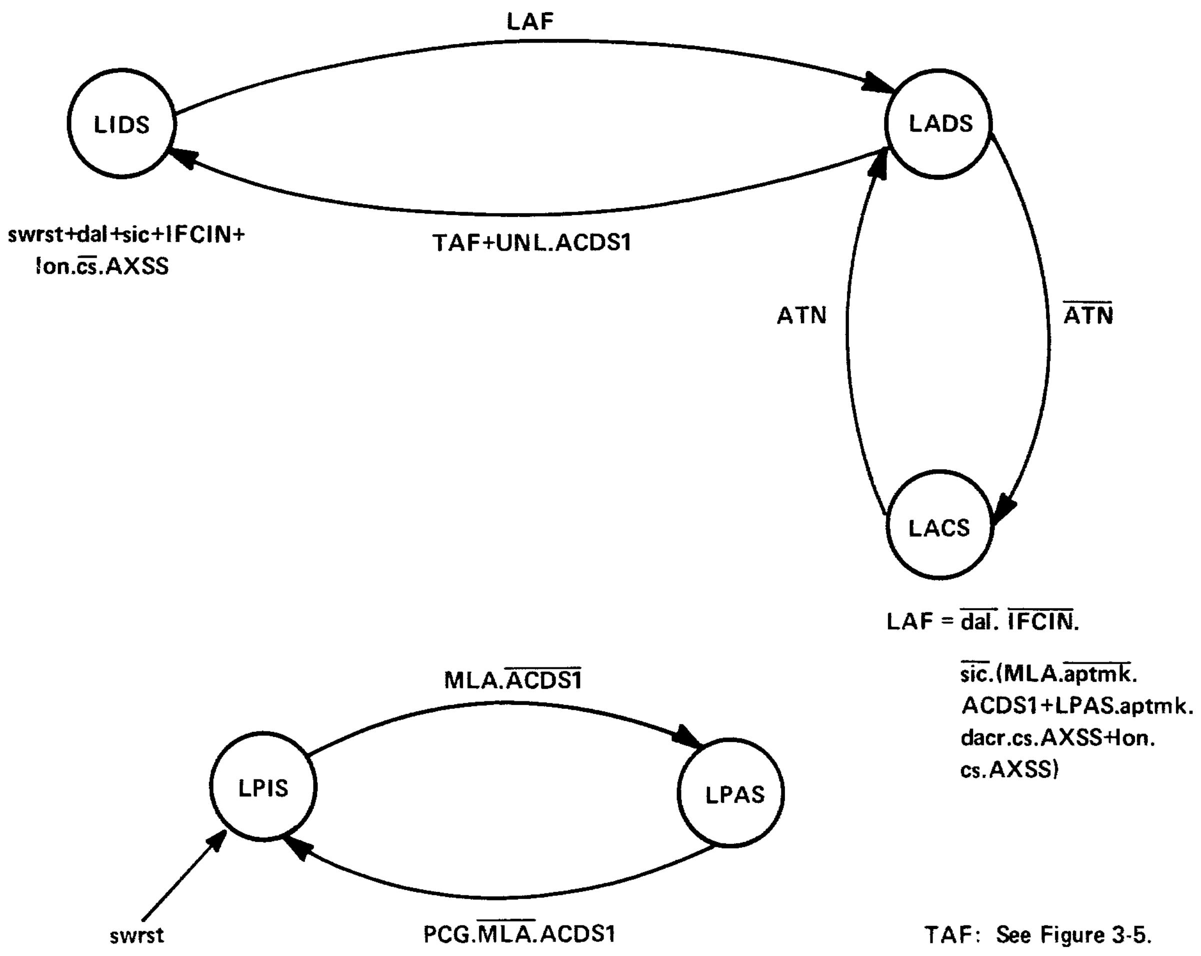
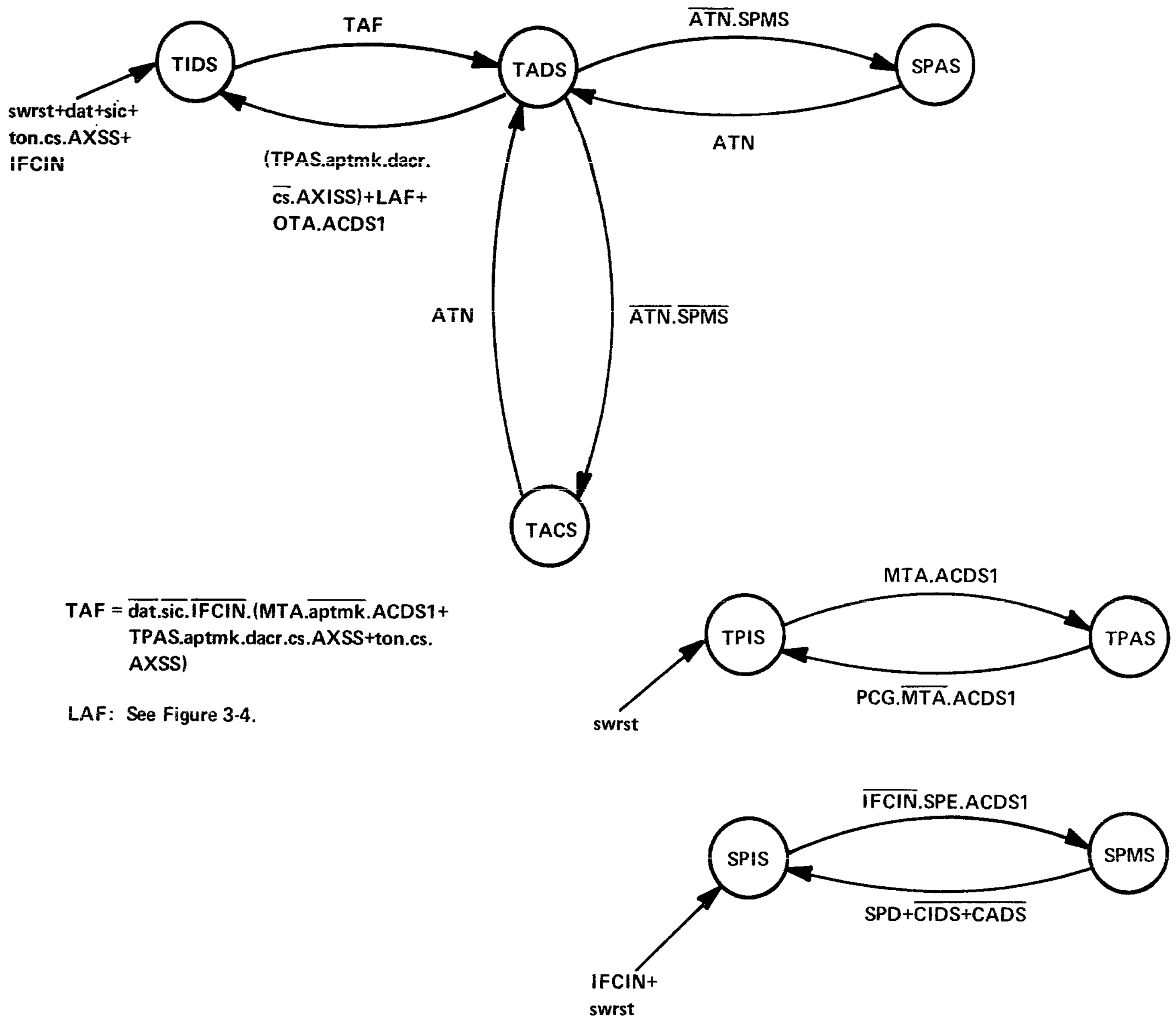

LAF
LIDS
LADS
swrst+dal+sic+IFCIN+
lon.$\overline{\text{cs}}$.AXSS
TAF+UNL.ACDS1
ATN
$\overline{\text{ATN}}$
LACS
LAF = $\overline{\text{dal}}$. $\overline{\text{IFCIN}}$.
$\overline{\text{sic}}$.(MLA.$\overline{\text{aptmk}}$.
ACDS1+LPAS.aptmk.
dacr.cs.AXSS+lon.
cs.AXSS)
MLA.$\overline{\text{ACDS1}}$
LPIS
LPAS
swrst
PCG.$\overline{\text{MLA}}$.ACDS1
TAF: See Figure 3-5.

FIGURE 3-4 – TMS9914A LISTENER STATE DIAGRAM


TIDS
TADS
SPAS
TACS
TPIS
TPAS
SPIS
SPMS
TAF
$\overline{\text{ATN}}$.SPMS
ATN
swrst+dat+sic+ton.cs.AXSS+IFCIN
(TPAS.aptmk.dacr.$\overline{\text{cs}}$.AXISS)+LAF+OTA.ACDS1
ATN
$\overline{\text{ATN}}$.$\overline{\text{SPMS}}$
MTA.ACDS1
PCG.$\overline{\text{MTA}}$.ACDS1
swrst
$\overline{\text{IFCIN}}$.SPE.ACDS1
SPD+$\overline{\text{CIDS}}$+CADS
IFCIN+swrst

TAF = $\overline{\text{dat}}$.$\overline{\text{sic}}$.$\overline{\text{IFCIN}}$.(MTA.$\overline{\text{aptmk}}$.ACDS1+TPAS.aptmk.dacr.cs.AXSS+ton.cs.AXSS)

LAF: See Figure 3-4.

FIGURE 3-5 — TMS9914A TALKER STATE DIAGRAM

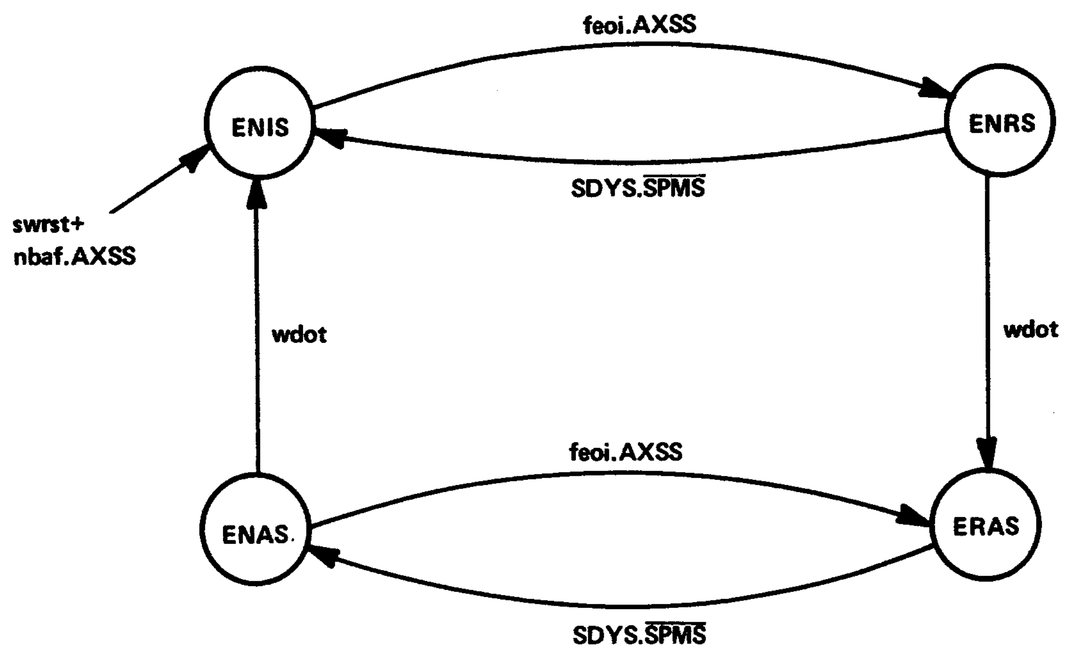


FIGURE 3-5 – TMS9914A TALKER STATE DIAGRAM (Continued)

TABLE 3-6 – TALKER AND LISTENER MNEMONICS

MESSAGES			STATES		
swrst	=	software reset	LIDS	=	listener idle state
dal	=	disable listener	LADS	=	listener addressed state
dat	=	disable talker	LACS	=	listener active state
sic	=	send interface clear	LPIS	=	listener primary idle state
lon	=	listen only	LPAS	=	listener primary addressed state
ton	=	talk only	TIDS	=	talker idle state
cs	=	clear/set bit of the auxiliary command register	TADS	=	talker addressed state
dacr	=	release 'DAC' holdoff	TACS	=	talker active state
aptmk	=	address pass through interrupt mask	SPAS	=	serial poll active state
nbaf	=	new byte available false	SPIS	=	serial poll idle state
feoi	=	force 'EOI'	SPMS	=	serial poll mode state
wdot	=	write to the Data Out Register	TPIS	=	talker primary idle state
ATN	=	attention	TPAS	=	talker primary addressed state
IFCIN	=	internal interface clear message (a debounced signal, suppressed by 'sic')	ENIS	=	end idle state
EOI	=	end or identify	ENRS	=	end ready state
PCG	=	primary command group	ERAS	=	end ready and active state
MLA	=	my listen address	ENAS	=	end active state
MTA	=	my talk address	SDYS	=	source delay state (source handshake)
OTA	=	other talk address	CIDS	=	controller idle state (controller function)
SPE	=	serial poll enable	CADS	=	controller addressed state (controller function)
SPD	=	serial poll disable	ACDS1	=	accept data state 1 (acceptor handshake)
UNL	=	unlisten	AXSS	=	auxiliary command strobe state (auxiliary command register)
PCG	=	primary command group			

TABLE 3-7 – TALKER FUNCTION MESSAGE OUTPUTS

STATE	QUALIFIER	REMOTE MESSAGES SENT		OTHER ACTIONS
		SRQ	EOI	DIO(8-1)
TIDS		(F)	(F)	(NUL)
TADS		(F)	(F)	(NUL)
TACS	ENIS.ENRS	(F)	F	DATA OUT REG
TACS	ENAS.ERAS	(F)	T	DATA OUT REG
SPAS	NPRS.SRQS	F	F	SERIAL POLL REG
SPAS	APRS1.APRS2	T	F	SERIAL POLL REG

3.5 SERVICE REQUEST FUNCTION

Figure 3-6 shows the state diagram for the TMS9914A service request function. The device has two means of implementing the request service (rsv) local message of IEEE-488: the first, 'rsv1', is bit 7 of the Serial Poll Register; the second is the auxiliary command 'rsv2'. These are simply ORed together to provide an input to the service request function, and, in any particular application, only one would normally be used, the other being left in its hardware reset state.

The affirmative poll response state (APRS) of IEEE-488 is split into two states on the TMS9914A for the following reason: Consider the case where a device has requested service, has been serial polled, and then wishes to request service again. The host MPU must clear the 'rsv' message and then set it true again. Now suppose this temporary false condition happens within one occurrence of SPAS. If the service request function has been implemented exactly as per IEEE-488, it will not be recognized, and SRQ will not be asserted a second time. Therefore, 'rsv' may only be cleared when the device is known not to be in SPAS, which can only happen if it is cleared as a consequence of some pre-arranged action of the controller. This action would normally be a part of the service routine executed by the controller as a response to the request for service. For example, if service was requested by an instrument which had some data to send for processing or to a printing device then 'rsv' could be cleared when it is addressed to talk and send its data over the bus.

For many applications, the fact that the device has been serial polled after requesting service is considered sufficient response from the controller. The 'rsv' local message therefore simply becomes a request for the controller to read its serial poll status byte. It is then desirable to be able to clear and reassert 'rsv' at any time after the serial poll status byte has been polled and the SPAS interrupt set. The TMS9914A is able to record a false transition of 'rsv1' or 'rsv2' by moving from APRS1 to APRS2 even if the device is in SPAS. This makes the above approach to serial polling possible.

To further support this approach, the 'rsv2' auxiliary command is automatically cleared when the serial poll status byte is polled, ensuring that "rsv2' is cleared before a second serial poll can occur. If this were not the case, then the same status byte might be polled twice by the controller with the RQS bit true, which may indicate that two reasons for requiring service have arisen.

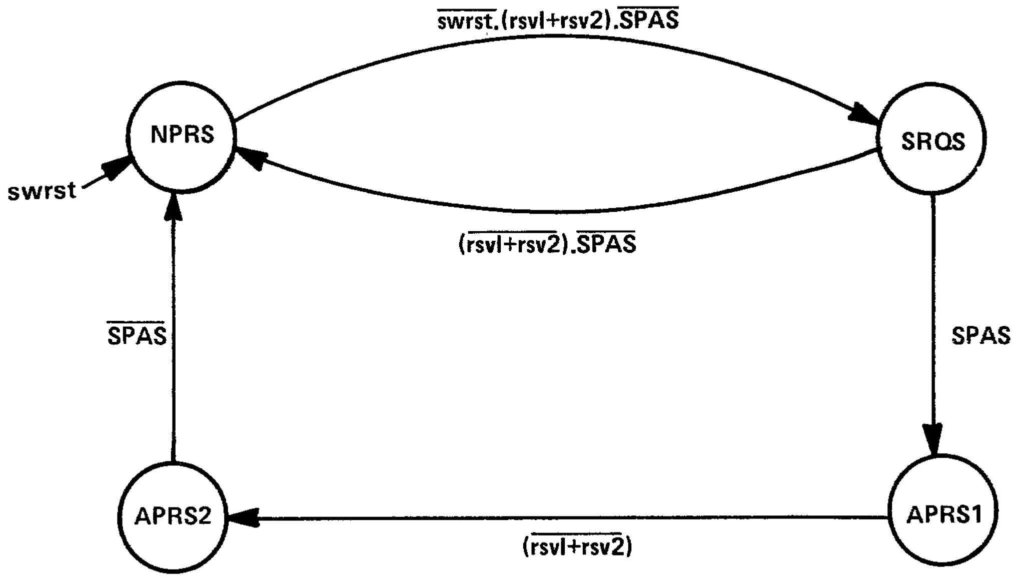

$\overline{\text{swrst}}.(\text{rsvl}+\text{rsv2}).\overline{\text{SPAS}}$
NPRS
SRQS
swrst
$(\overline{\text{rsvl}+\text{rsv2}}).\overline{\text{SPAS}}$
$\overline{\text{SPAS}}$
SPAS
APRS2
APRS1
$(\overline{\text{rsvl}+\text{rsv2}})$

FIGURE 3-6 – SERVICE REQUEST STATE DIAGRAM

NOTE

The TMS9914 service request function was implemented exactly as per IEEE-488. Also, it had only one 'rsv' bit which was equivalent of the TMS9914A's 'rsv1.

The TMS9914A will only send one serial poll status byte during each active period of SPAS., However, it will send this status byte as many times as the controller is prepared to accept it. Therefore, the controller should only read the status byte once per serial poll; otherwise, each time a status byte is sent with the RQS message true, the SPAS interrupt will be generated and 'rsv2' will be cleared.

TABLE 3-8 – SERVICE REQUEST MNEMONICS

MESSAGES			STATES		
swrst	=	software reset	NPRS	=	negative poll response state
srv1	=	request service 1 (bit 7 of serial poll register)	SRQS	=	service request state
rsv2	=	request service 2 (auxiliary command register)	APRS1	=	affirmative poll state 1
			APRS2	=	affirmative poll state 2
			SPAS	=	serial poll active state (talker function)

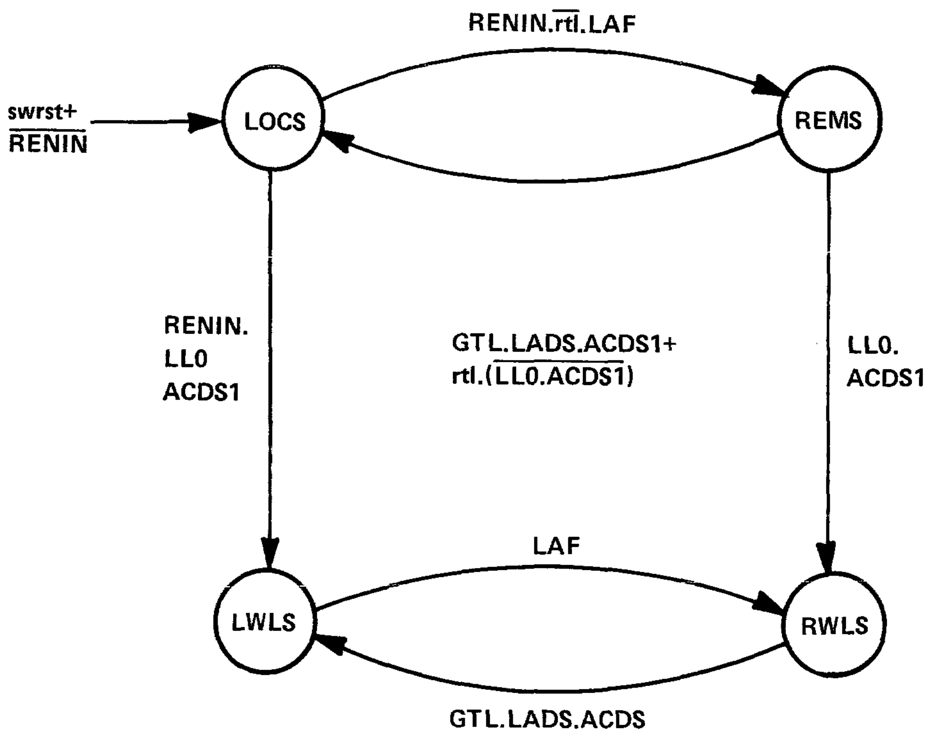
TABLE 3-9 – SERVICE REQUEST MESSAGE OUTPUTS

STATE	REMOTE MESSAGES SENT SRQ	OTHER ACTIONS
NPRS	(F)	
SRQS	T	
APRS1	(F)	– rsv2 cleared if in SPAS and STRS – SPAS interrupt set if in SPAS when STRS is exited
APRS2	(F)	– same as APRS1

3.6 REMOTE/LOCAL FUNCTION

The TMS9914A remote local state diagram is shown in Figure 3-7. It differs little from that of IEEE-488.

The complete listener function (LAF) is used to effect the transition from LOCS to REMS or from LWLS to RWLS. This means that if the APT interrupt is masked, the device will enter one of the remote states in response to its listen address, but if secondary addressing is enabled, then this will not happen until 'dacr' is written with 'cs' true in response to a valid secondary address. In addition, the transition to one of the remote states will occur if 'lon' is used to address the device to listen.


RENIN.$\overline{\text{rtl}}$.LAF
swrst+
$\overline{\text{RENIN}}$
LOCS
REMS
RENIN.
LLO
ACDS1
GTL.LADS.ACDS1+
rtl.($\overline{\text{LLO.ACDS1}}$)
LLO.
ACDS1
LAF
LWLS
RWLS
GTL.LADS.ACDS

LAF: See Figure 3-4

FIGURE 3-7 – TMS9914A REMOTE LOCAL STATE DIAGRAM

TABLE 3-10 – REMOTE/LOCAL MNEMONICS

MESSAGES			STATES		
swrst	=	software reset	LOCS	=	local state
rtl	=	return to local	REMS	=	remote state
RENIN	=	internal remote enable message (debounced)	RWLS	=	remote with lockout state
GTL	=	go to local	LWLS	=	local with lockout state
LLO	=	local lockout	LADS	=	listener addressed state (listener function)
			ACDS1	=	accept data state 1 (acceptor handshake)

3.7 PARALLEL POLL FUNCTION

The parallel poll function of the TMS9914A only nominally supports logically-configured parallel poll. With a suitable software package, remotely-configured parallel poll may also be easily implemented. The state diagram is shown in Figure 3-8.

When the EOI and ATN lines become true simultaneously (the Identify message), the contents of the Parallel Poll Register are output to DIO(8-1). If parallel poll is to be used in a particular bus environment, then the Pull-Up Enable (PE) input of the SN75160 must be held low so that the DIO(8-1) are driven by open collector buffers. Parallel Poll, occurring when the Parallel Poll Register is in the hardware reset condition of all zeros, will result in none of DIO(8-1) being pulled low. This corresponds to the parallel poll idle state (PPIS). If it is desired to participate in a parallel poll, then the bit corresponding to the desired parallel poll response is set true. This implements the parallel poll standby state (PPSS), and, when the Identify message becomes true, the appropriate line of DIO(8-1) is pulled low. This is equivalent to the parallel poll active state (PPAS). Only one bit of the parallel Poll Register should be set true at once.

3.7.1 Remotely Configured Parallel Poll

The parallel poll configure command (PPC) is treated by the TMS9914A as an unrecognized addressed command. It is passed through when the TMS9914A is in LADS. If an instrument is to be remotely configured for parallel poll, then the pass through next secondary (pts) auxiliary command should be written before releasing the DAC holdoff. This will cause the next command received to also set a UNC interrupt if it is a secondary command. The secondary command will be either the parallel poll enable command (PPE) or the parallel poll disable command (PPD) and should be read from the Command Pass Through Register and identified. If it is the PPE command, then the attendant bits (S, P1, P2, P3) should be extracted and stored by the host MPU (see Section 2.9.3 of IEEE-488 1978). The S bit should then be matched against the individual status of the instrument (represented by 'ist'), and if they are the same, the bit corresponding to the parallel poll response, specified by P1, P2, P3, should be set true in the Parallel Poll Register. If this is not the case, then the Parallel Poll Register should be cleared if it is not already clear. After this, each time the individual status of the device changes, the 'ist' should again be matched against the S bit and the Parallel Poll Register updated accordingly until PPD or PPU is received.

If a PPD command is passed through after the 'pts' feature has been written, the Parallel Poll Register should be cleared before the DAC holdoff is released. The PPC command that precedes PPD is an address command; it is a means of eliminating individual members of a parallel poll. The parallel unconfigure command is treated by the TMS9914A as an unrecognized universal command. When it is passed through, the host MPU should clear its Parallel Poll Register before releasing the DAC holdoff. This command will clear all members of a parallel poll.

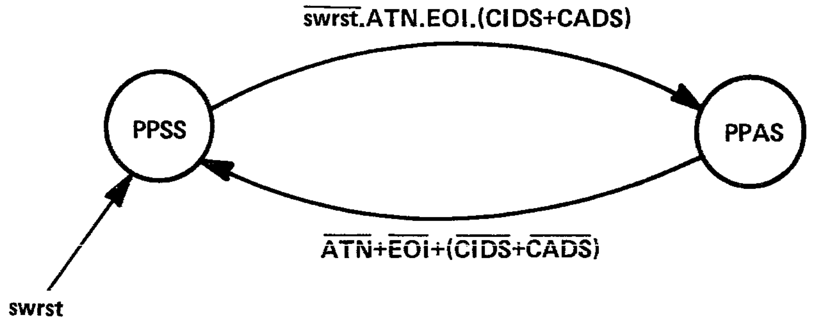

$\overline{\text{swrst}}$.ATN.EOI.(CIDS+CADS)
PPSS
PPAS
$\overline{\text{ATN}}$+$\overline{\text{EOI}}$+($\overline{\text{CIDS}}$+$\overline{\text{CADS}}$)
swrst

FIGURE 3-8 – TMS9914A PARALLEL POLL STATE DIAGRAM

TABLE 3-11 – PARALLEL POLL MNEMONICS

MESSAGES			STATES		
swrst	=	software reset	PPSS	=	parallel poll standby state
ATN	=	attention	PPAS	=	parallel poll active state
EOI	=	end or identify	CIDS	=	controller idle state (controller function)
			CADS	=	controller addressed state (controller function)

TABLE 3-12 – PARALLEL POLL MESSAGE OUTPUTS

STATE	REMOTE MESSAGES SENT	OTHER ACTIONS
	DIO(8-1)	
PPSS	(NUL)	
PPSS	PARALLEL POLL REG*	

* If there is a true bit in the Parallel Poll Register, it must be sent active; any false bit must be sent passive.

3.8 CONTROLLER FUNCTION

The controller function of the TMS9914A is greatly simplified compared with that of IEEE-488. It relies heavily on software support but, with suitable software, it enables all subsets of the controller function to be implemented. With this approach the controller logic is reduced to a small proportion of the chip area which means that the device may be economically used in situations where a talker/listener only is required.

Figure 3-9 shows the controller function state diagram. With suitable software, it will perform the full controller function, as described in the IEEE-488A 1980 supplement to the IEEE-488 1978. It therefore includes the additional state CSHS, which allows time for DAV to be recognized false by all devices on the bus before ATN is asserted. The 'tcs' local message is implemented by an immediate execute auxiliary command. The state CWAS is therefore added to record the occurrence of this command until the acceptor handshake enters ANRS and the device can enter CSHS. The 'tca' auxiliary command also causes entry into CSHS although IEEE-488A 1980 allows it to move directly from CSBS to CSWS. This is done for convenience of implementation and results in the 'tca' auxiliary command taking an extra 1.6 microseconds to assert ATN.

The delay between CSWS and CAWS is slightly less than specified in IEEE-488A 1980 but the total time taken in moving from CSWS to CACS is still greater than the specified minimum.

The Controller Parallel Poll State (CPPS) is not included on the TMS9914A. To conduct a parallel poll, a TMS9914A based controller must set the 'rpp' clear/set auxiliary command true when it is in CACS, moving it to CPWS which sends EOI true. The host MPU must then wait 2 microseconds before reading back the parallel poll responses via the Command Pass Through Register. The 'rpp' auxiliary command can then be cleared, EOI will go false, and the parallel poll is complete. The host MPU will receive a BO interrupt as soon as the TMS9914A reenters CACS and the source handshake becomes active.

3.8.1 Controller Self Addressing

As discussed in Section 3.2, the acceptor handshake does not operate when the controller is active. This means commands being sent are not monitored, and special precautions are required as a consequence of this when addressing devices and when passing control.

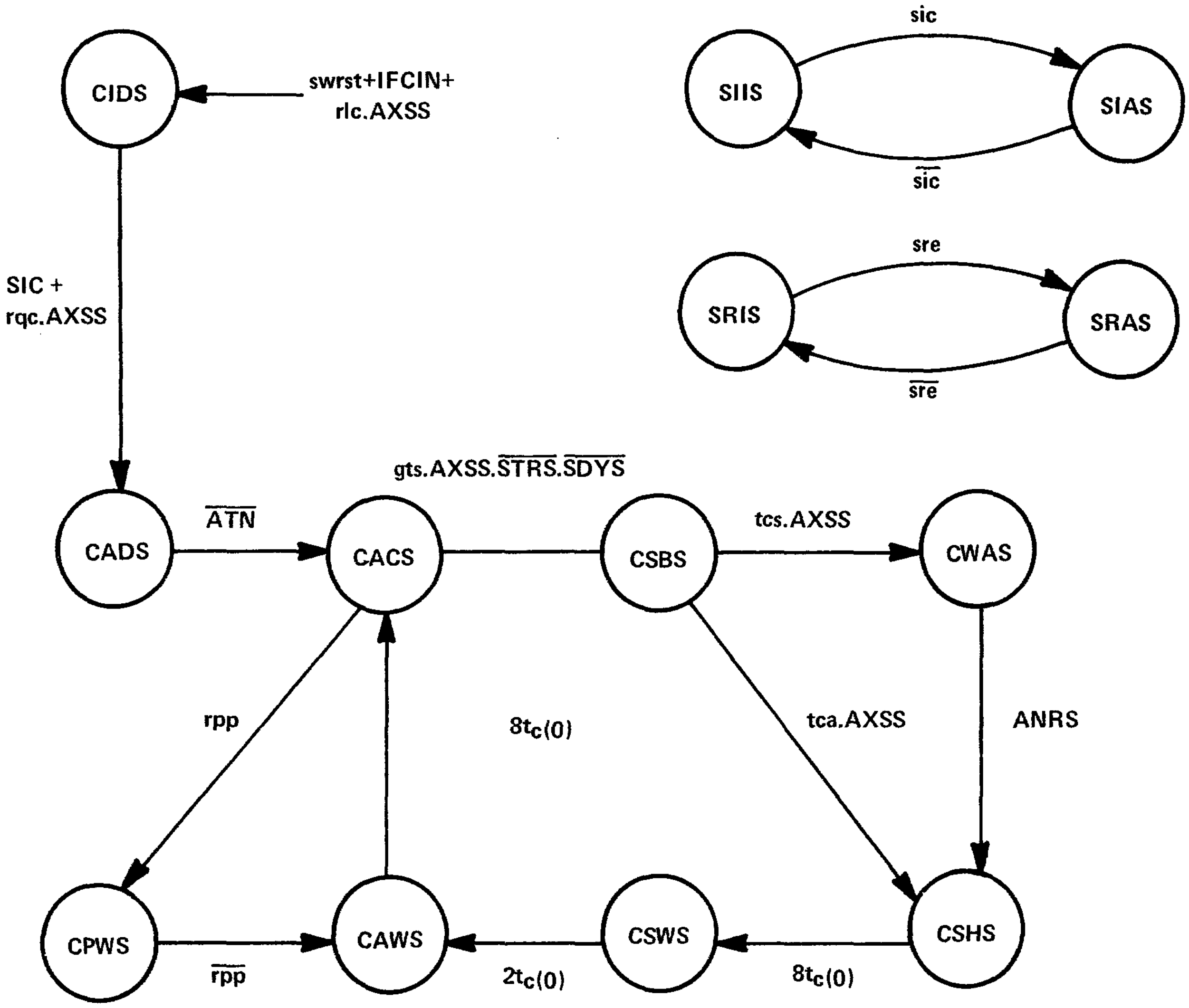


FIGURE 3-9 – TMS9914A CONTROLLER STATE DIAGRAMS

TABLE 3-13 – CONTROLLER FUNCTION MNEMONICS

MESSAGES			STATES		
swrst	=	software reset	CIDS	=	controller idle state
sic	=	send interface clear	CADS	=	controller addressed state
sre	=	send remote enable	CACS	=	controller active state
rqc	=	request control	CSBS	=	controller standby state
rlc	=	release control	CSHS	=	controller standby hold state
gts	=	go to standby	CSWS	=	controller synchronous wait state
tcs	=	take control synchronously	CAWS	=	controller active wait state
tca	=	take control asynchronously	CPWS	=	controller parallel poll wait state
rpp	=	request parallel poll	ANRS	=	acceptor not ready state (acceptor handshake)
IFCIN	=	internal interface clear message (a debounced signal which is suppressed if 'sic' is true)	SDYS	=	source delay state (source handshake)
ATN	=	attention	STRS	=	source transfer state (source handshake)
$t_{c(0)}$	=	clock cycle time	AXSS	=	auxiliary command strobe state (auxiliary command register)
			LWAS	=	controller wait for ANRS state

TABLE 3-14 – CONTROLLER FUNCTION MESSAGE OUTPUTS

STATE	REMOTE MESSAGE SENT			OTHER ACTIONS
	ATN	EOI	DIO(8-1)	
CIDS	(F)	(F)	(NUL)	
CADS	(F)	(F)	(NUL)	
CACS	T	F	DATA OUT REG	Data Out Reg. may contain any of the commands in Table 3-15
CSBS	F	(F)	(NUL)	
CWAS	F	(F)	(NUL)	
CSHS	F	(F)	(NUL)	
CSWS	T	F	(NUL)	
CAWS	T	F	(NUL)	
CPWS	T	T	(NUL)	DIO(8-1) may be read via the Command Pass Through Register

STATE	REMOTE MESSAGES SENT	OTHER ACTIONS
	IFC	
SIIS*	(F)	Internal interface clear message IF-CIN is held false
SIIS	F	
SIAS	T	

STATE	REMOTE MESSAGES SENT	OTHER ACTIONS
	REN	
SRIS*	(F)	
SRIS	F	
SRAS	T	

* Buffers not configured for a system controller; otherwise, buffers are configured for system controller.

When the controller is active, it uses 'ton' or 'lon' to address and unaddress itself. IEEE-488 provides for the controller to locally address itself to listen, but there is no corresponding local message for the talker. The TMS9914A should always accompany a 'ton' auxiliary command with 'cs' true with its own talk address or an UNT command sent over the bus. Similarly, if the TMS9914A sends the talk address of another device over the bus, it should ensure that it is in TIDS by writing the 'ton' auxiliary command false.

Appendix B shows some typical sequences of events when the controller addresses itself, goes to standby, takes control again, etc.

3.8.2 Passing Control

As Figure 3-9 shows, the controller transfer state (CTRS) of IEEE-488 is not present, and all transitions associated with the TCT command have been removed. Instead, two immediate execute auxiliary commands are included. Request control (rqc) will cause a transition from CIDS to CADS, and the release control command (rlc) will return the function to CIDS. The TCT command is treated similarly to an unrecognized addressed command but will cause a UNC interrupt if the device is in TADS.

Figure 3-10 is a representation of the sequence of events involved in passing control from one TMS9914A based device to another. The device passing control must initially ensure that it is not in TADS; then it should send out the talk address of the device to receive control. The receiving device will enter TADS, and after any DAC holdoff has been released, the host MPU of the device passing control will set a BO interrupt indicating that it may then send the TCT command. The TCT command will cause a UNC interrupt to the host MPU of the receiving device, and also a DAC holdoff will occur. The host MPU of the receiving device must examine its Command Pass Through Register, and upon identifying TCT, should write the auxiliary command 'rqc' to put its TMS9914A into CADS. The receiving device may then release DAC with a 'dacr' auxiliary command causing another BO interrupt at the device passing control. This indicates that the 'rlc' auxiliary command may then be used by the host MPU of the device passing control to return its TMS9914A to CIDS and allowing ATN to go false. The receiving device then enters CACS, asserts ATN, and its host MPU gets a BO interrupt as the source handshake becomes active. The passing of control is complete.

3.8.3 System Controller

The TMS9914A has no on-chip means of determining whether or not it is the system controller. Instead, this is determined by the software and by the configuration of the buffers to the IEEE-488 bus.

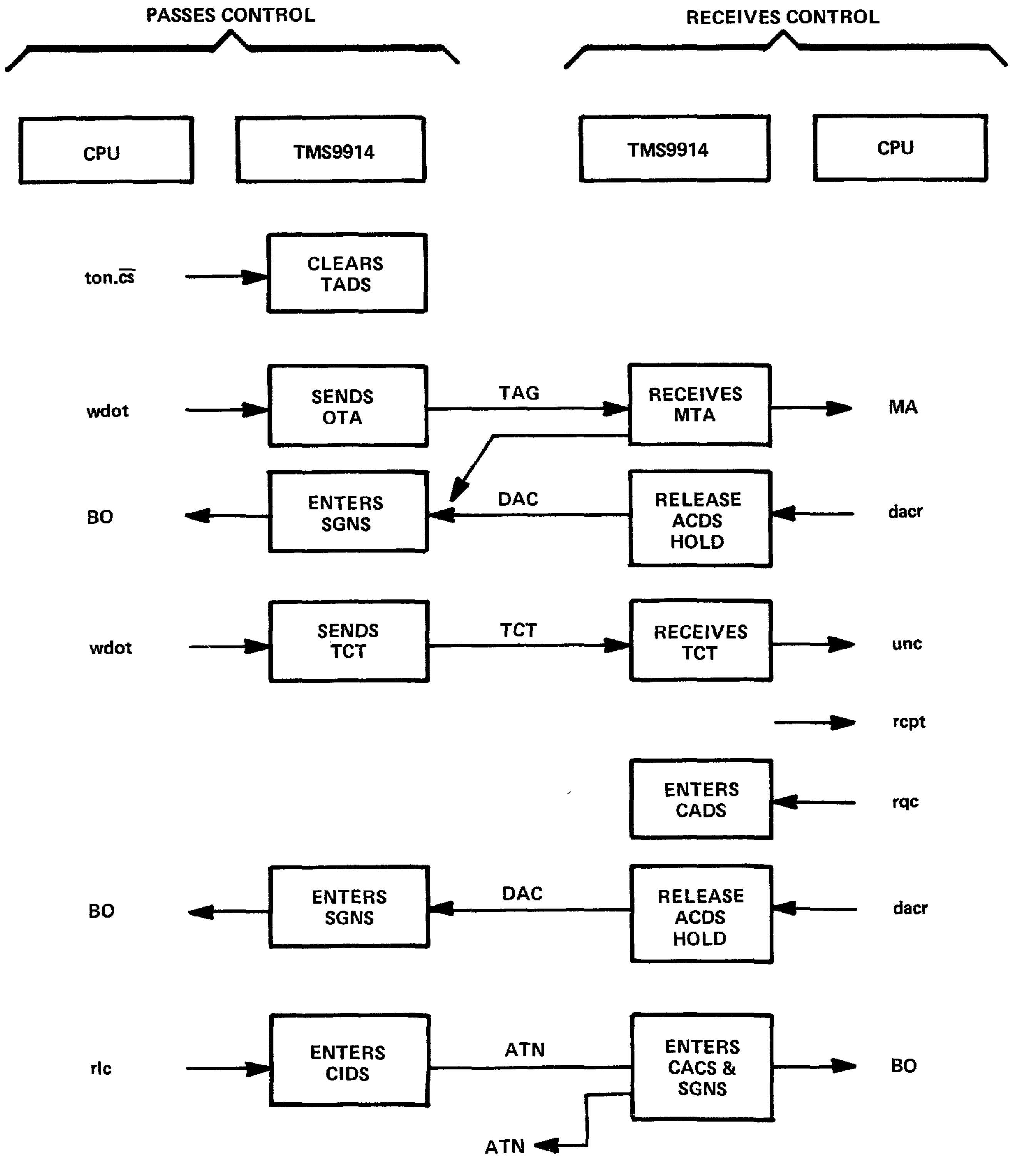
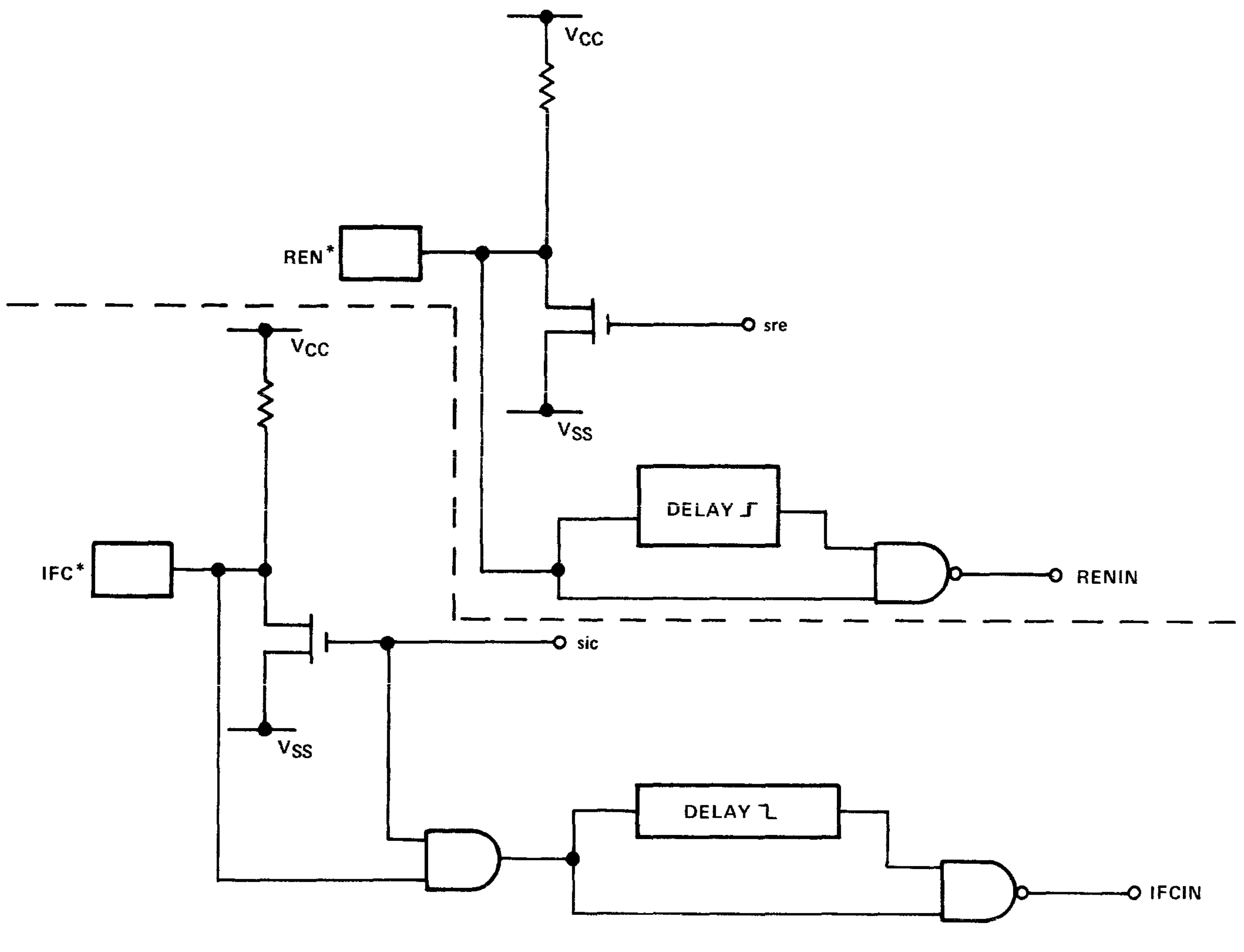


FIGURE 3-10 – PASSING CONTROL BETWEEN TMS9914s

The REN and IFC outputs of the TMS9914A are controlled by the auxiliary commands 'sre' and 'sic'. These should never be used by the host MPU of a device unless it is the system controller. As may be seen from Figure 3-11, the REN and IFC outputs of the TMS9914A are open drains with internal pull-ups. This means that the outputs are capable of driving the inputs of the buffers if the device is a system controller. If not, the buffers will drive into the REN and IFC pins and override the pull-ups. Hence, no direction control is required.

The false transition of REN and the true transition of IFC are both debounced to prevent noise on these lines from causing permanent state changes on the TMS9914A. In addition, the internal interface clear signal (IFCIN) is held false if the TMS9914A is sending IFC. Figure 3-9 shows the reason for this. If the device is not a system controller, then the occurence of IFC will return the controller function to CIDS. If, however, the device is a system controller, when it asserts IFC and is in CIDS, the 'sic' auxiliary command will cause it to enter CADS. As IFCIN is suppressed, it will not be forced back into CIDS, and there will be no conflict.



* The REN and IFC signals are at the pins of the TMS9914A and are therefore negative logic signals. The remaining signals are conventional positive logic signals.

FIGURE 3-11 – IFC AND REN PINS

TABLE 3-15 – MULTILINE INTERFACE MESSAGES

COMMAND	SYMBOL	DIO 8 – 1	CLASS	INTERRUPT (1,2)	DAC (3) HOLDOFF	NOTE
ADDRESSED COMMAND GROUP	ACG	000XXXX	AC	–	–	
DEVICE CLEAR	DCL	X0010100	UC	DCAS	YES	
GROUP EXECUTE TRIGGER	GET	X0001000	AC	GET	YES	
GO TO LOCAL	GTL	X0000001	AC	RLC	NO	14
LISTEN ADDRESS GROUP	LAG	X01XXXXX	AD	–	–	
LOCAL LOCKOUT	LLO	X0010001	UC	NONE	NO	
MY LISTEN ADDRESS	MLA	X01AAAAA	AD	MA,MAC,RLC	MA ONLY	4,14
MY TALK ADDRESS	MTA	X10AAAAA	AD	MA,MAC	MA ONLY	4
MY SECONDARY ADDRESS	MSA	X11SSSSS	SE	APT	YES	5,6
OTHER SECONDARY ADDRESS	OSA	SCG.MSA-	SE	APT	YES	6,7
OTHER TALK ADDRESS	OTA	TAG.MTA-	AD	MAC	NO	
PRIMARY COMMAND GROUP	PCG	ACG+UCG+ LAG+TAG	–	–		
PARALLEL POLL CONFIGURE	PPC	X0000101	AC	UNC	YES	8
PARALLEL POLL ENABLE	PPE	X110SPPP	SE	UNC	YES	9,10
PARALLEL POLL DISABLE	PPD	X111DDDD	SE	UNC	YES	9,11
PARALLEL POLL UNCONFIGURE	PPU	X0010101	UC	UNC	YES	12
SECONDARY COMMAND GROUP	SCG	X11XXXXX	SE	–	–	
SELECTED DEVICE CLEAR	SDC	X0000100	AC	DCAS	YES	
SERIAL POLL DISABLE	SPD	X0011001	UC	NONE	NO	
SERIAL POLL ENABLE	SPE	X0011000	UC	NONE	NO	
TAKE CONTROL	TCT	X0001001	AC	UNC	YES	13
TALK ADDRESS GROUP	TAG	X10XXXXX	AD	–	–	
UNLISTEN	UNL	X0111111	AD	MAC	NO	
UNTALK	UNT	X1011111	AD	–	–	
UNIVERSAL COMMAND GROUP	UCG	X001XXXX	UC	NONE	NO	

Classes:	UC	–	universal command
	AC	–	addressed command
	AD	–	address
	SE	–	secondary command
Symbols:	0	–	logical zero (high level on GPIB)
	1	–	logical one (low level on GPIB)
	x	–	don't care (received message)

NOTES:
1. Interrupts listed are as a direct consequence of the command received. They are set during ACDS1 (see Section 3.2) and will cause the INT pin to be pulled low if unmasked.
2. The addressed commands will only cause their corresponding interrupt if the device is in LADS with the exception of TCT.
3. A DAC holdoff will only be caused if the corresponding interrupt is unmasked.
4. AAAAA represents the primary address of a device.
5. SSSSS represents the secondary address of a device.
6. Secondary addresses are handled via address pass through (APT interrupt). The host MPU should respond by writing the 'dacr' auxiliary command with 'cs' false.
7. If OSA is passed through via the APT interrupt, the host MPU should respond by writing the 'dacr' auxiliary command with 'cs' false.
8. PPC is not recognized by the TMS9914A and is therefore treated as an unrecognized addressed command.
9. **PPE and PPD are secondary commands. These may be passed through to the host MPU using the 'pts' auxiliary command. When the PPC command is received, the 'pts' auxiliary command should be written. PPE or PPD will then cause an UNC interrupt.**
10. SPPP specifies the sense bit, and the desired parallel poll response ia a remotely configured parallel poll (see Section 3.7.1).
11. DDDD specifies don't care bits which must be sent as zeros but need not be decoded by the host MPU of the receiving devices.
12. PPU is not recognized by the TMS9914A and will cause a UNC interrupt.
13. TCT is not recognized directly by the TMS9914A. It will cause a UNC interrupt when the device is in TADS.
14. RLC is set if MLA or GTL causes an appropriate transition in the Remote/Local function.

4. TMS9914A ELECTRICAL SPECIFICATIONS

4.1 ABSOLUTE MAXIMUM RATINGS OVER OPERATING FREE-AIR TEMPERATURE RANGE (UNLESS OTHERWISE NOTED)*

Supply voltage, V_{CC} (see Note 1)	−0.3 V to 20 V
All input and output voltages	−0.3 V to 20 V
Continuous power dissipation	0.8 W
Operating free-air temperature range	0 °C to 70 °C
Storage temperature range	−55 °C to 150 °C

* Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions beyond those indicated in the "Recommended Operating Conditions" section of this specification is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

NOTE 1: Under absolute maximum ratings voltage values are with respect to V_{SS}.

4.2 RECOMMENDED OPERATING CONDITIONS

	MIN	NOM	MAX	UNIT
Supply voltage, V_{CC}	4.75	5	5.25	V
Supply voltage, V_{SS}		0		V
High-level input voltage, V_{IH}	2		$V_{CC}+1$	V
Low-level input voltage, V_{IL}	$V_{SS}-0.3$		0.8	V
Operating free-air temperature, T_A	0		70	°C

4.3 ELECTRICAL CHARACTERISTICS OVER FULL RANGE OF RECOMMENDED OPERATING CONDITIONS

PARAMETER			TEST CONDITIONS‡	MIN	TYP†	MAX	UNIT
V_{OH}	High-level output voltage	Except REN,IFC,INT	I_{OH} = −400 μA	2.4		V_{CC}	V
		REN,IFC only	I_{OH} = −100 μA	2.2		V_{CC}	
V_{OL}	Low-level output voltage		I_{OL} = 2 mA	V_{SS}		0.4	V
I_I	Input current (any input)		V_I = 2 V to V_{CC}			±10	μA
I_{CC}	V_{CC} supply current					150	mA
C_i	Input capacitance (any input)		f = 1 MHz, unmeasured pins at 0 V			15	pF

†All typical values are at T_A = 25°C and nominal voltage.
‡θJ_A = 78°C, θJ_C = 34°C.

4.4 TIMING CHARACTERISTICS AND REQUIREMENTS

Timing characteristics and requirements are given in Section 4.4.1 through Section 4.4.6; relevant timing diagrams are shown in Figure 4-1 through Figure 4-9.

4.4.1 Clock and Host Interface Timing Requirements Over Full Range of Operating Conditions

PARAMETER		MIN	TYP	MAX	UNIT
$t_{c(\phi)}$	Clock cycle time	200		2000	ns
$t_{w(\phi H)}$	Clock high pulse width	100		1955	ns
$t_{w(\phi L)}$	Clock low pulse width	45			ns
$t_{su(AD)}$	Address setup time	0			ns
$t_{su(DBIN)}$	DBIN setup time	0			ns
$t_{su(CE)}$	$\overline{\text{CE}}$ setup time	100			ns
$t_{su(WE)}$	$\overline{\text{WE}}$ setup time	0			ns
$t_{w(WE)}$	$\overline{\text{WE}}$ low pulse width	80			ns
$t_{su(DA)}$	Data setup time	60			ns
$t_{h(DA)}$	Data hold time	0			ns
$t_{h(AD)}$	Address hold time	0			ns
$t_{h(DBIN)}$	DBIN hold time	0			ns
$t_{h(CE)}$	$\overline{\text{CE}}$ hold time	80			ns
$t_{su(GR)}$	$\overline{\text{ACCGR}}$ setup time	100			ns
$t_{h(GR)}$	$\overline{\text{ACCGR}}$ hold time	80			ns

4.4.2 Host Interface Timing Characteristics Over Full Range of Operating Conditions

PARAMETER		MIN	TYP	MAX	UNIT
$t_{a(CE)}$	Access time from $\overline{\text{CE}}$			150	ns
$t_{a(DBIN)}$	Access time from DBIN			150	ns
$t_{su(AD)}$	Address setup time to $\overline{\text{CE}}$	0			ns
$t_{z(DBIN)}$	Hi-Z time from DBIN		50	100	ns
$t_{z(CE)}$	Hi-Z time from $\overline{\text{CE}}$		50	100	ns
$t_{a(GR)}$	Access time from $\overline{\text{ACCGR}}$			150	ns
$t_{z(GR)}$	Hi-Z time from $\overline{\text{ACCGR}}$		50	100	ns
$t_{d(GR/RQ)}$	Delay of $\overline{\text{ACCRQ}}$ high from $\overline{\text{ACCGR}}$			100	ns

4.4.3 Source Handshake Timing Characteristics Over Full Range of Operating Conditions (see Note 1)

PARAMETER		TEST CONDITIONS	MIN	MAX	UNIT
t_{d1}	Delay of DAV true from end of write operation to data out register	Normal T_1 (see Note 2)	$12(\phi)I$	$12(\phi)I + 310$	ns
		Short T_1 (see Note 2)	$8(\phi)I$	$8(\phi)I + 310$	ns
		Very short T_1 (see Note 2)	$4(\phi)I$	$4(\phi)I + 310$	ns
t_{d2}	Delay of valid GPIB data lines from end of write cycle			140	ns
t_{d3}	Delay of BO interrupt from DAC true	BO interrupt unmasked		300	ns
t_{d4}	Delay of $\overline{\text{ACCRQ}}$ DAC true			300	ns
t_{d5}	Delay of DAV false from DAC true			160	ns

NOTES:
1. The timing of the source handshake is the same whether ATN is true or false, i.e., whether the device is in TACS, CACS, or SPAS.
2. A very short bus settling time (T_1) occurs on the second and subsequent data byte sent when ATN is false if the 'vstd1' feature is set. A slightly longer bus settling time takes place if 'std1' is set unless there is a very short bus settling time. In all other instances, a normal bus settling time occurs.

4.4.4 Acceptor Handshake Timing Characteristics Over Full Range of Operating Conditions

PARAMETER		TEST CONDITIONS	MIN	MAX	UNIT
t_{d6}	Delay of BI interrupt from DAV true	BI interrupt unmarked ATN = false device is in LACS	2(ϕ)↑	2(ϕ)↑ + 415	ns
t_{d7}	Delay of $\overline{\text{ACCRQ}}$ from DAV true	ATN = false device is in LACS	2(ϕ)↑	2(ϕ)↑ + 290	ns
t_{d8}	Delay of NDAC false from DAV true	ATN = false device is in LACS	3(ϕ)↑	3(ϕ)↑ + 445	ns
t_{d9}	Delay of NRFD false from end of read operation of Data In register	ATN = false device is in LACS		220	ns
t_{d10}	Delay of interface message interrupt from DAV true (see Note 3)	ATN = true device not in CACS all interface message interrupts (except UNO)	2(ϕ)↑	2(ϕ)↑ + 415	ns
		UNO interrupt only	5(ϕ)↑	5(ϕ)↑ + 415	ns
t_{d11}	Delay of NDAC false from DAV true	ATN = true device not in CACS no DAC holdoff	7(ϕ)↑	7(ϕ)↑ + 415	ns
t_{d12}	Delay of NDAC false from end of write operation			230	ns
t_{d13}	Delay of NRFD false from DAV false	ATN = true device not in CACS		180	ns

NOTE 3: The interrupts generated by interface messages are shown in Table 4-1.

4.4.5 ATN, EOI, and IFC Timing Characteristics Over Full Range of Operating Conditions

PARAMETER		TEST CONDITIONS	MIN	MAX	UNIT
t_{d14}	Delay of NDAC true from ATN true	Device is not in CACS		195	ns
t_{d15}	Delay of TE high from EOI true	Device is not in CACS		125	ns
t_{d16}	Delay of valid data from EOI true	Device is not in CACS		140	ns
t_{d17}	Delay of TE low from EOI false	Device is not in CACS		125	ns
t_{d18}	Delay of NRFD true from ATN false	Device is in LADS/LACS		140	ns
t_{d19}	Response time to IFC		$16t_{c(\phi)}$	$30t_{c(\phi)}$	ns

4.4.6 Controller Timing Characteristics Over Full Range of Operating Conditions

PARAMETER		TEST CONDITIONS	MIN	MAX	UNIT
t_{d20}	Delay of ATN true from end of t_{ca} aux command		$8t_{c(\phi)}$	$10(\phi)\uparrow+220$	ns
t_{d21}	Delay of BO interrupt from end of t_{ca} aux command		$18t_{c(\phi)}$	$22(\phi)\uparrow+415$	ns
t_{d22}	Delay of ATN true from end of t_{cs} aux command	BO unmasked device is in ANRS	$8t_{c(\phi)}$	$10(\phi)\uparrow+220$	ns
t_{d23}	Delay of BO interrupt from end of t_{cs} aux command	BO unmasked device is in ANRS	$18t_{c(\phi)}$	$22(\phi)\uparrow+415$	ns
t_{d24}	Delay of EOI true from r_{pp} aux command set			230	ns
t_{d25}	Delay of EOI false from r_{pp} aux command cleared			230	ns
t_{d26}	Delay of EOI from r_{pp} aux command cleared	BO unmasked	$8t_{c(\phi)}$	$10(\phi)\uparrow+415$	ns
t_{d27}	Delay of ATN false from sts aux command	Device is not in SDYS or STRS		210	ns

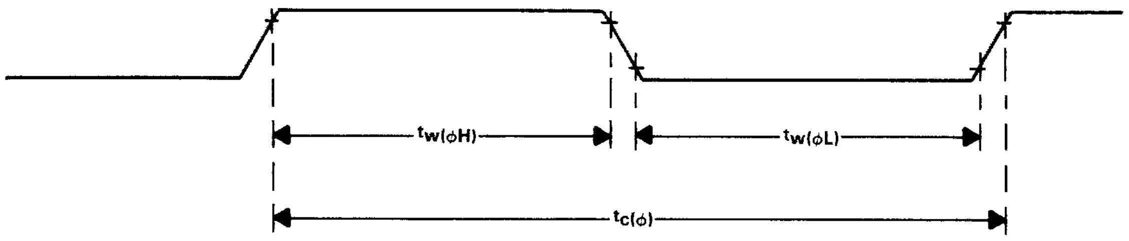


FIGURE 4-1 – TMS9914A CLOCK CYCLE TIMING

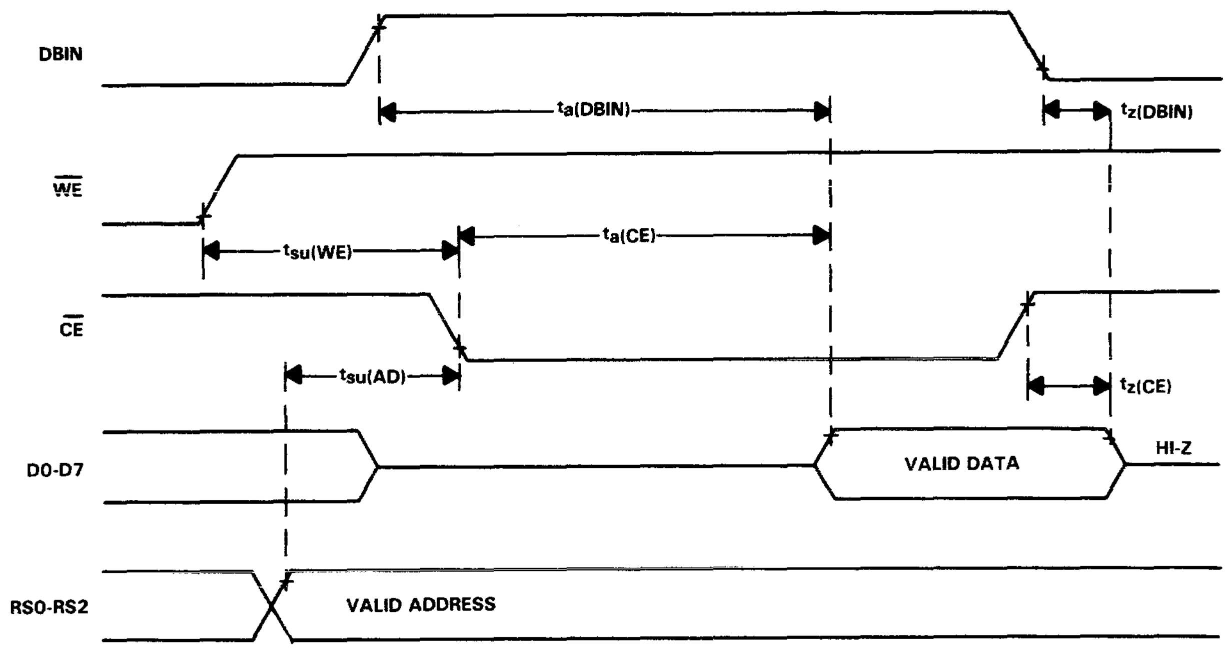
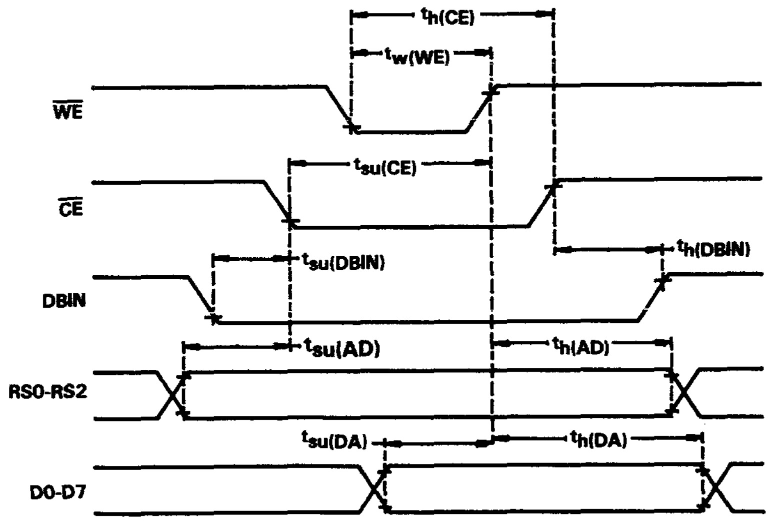
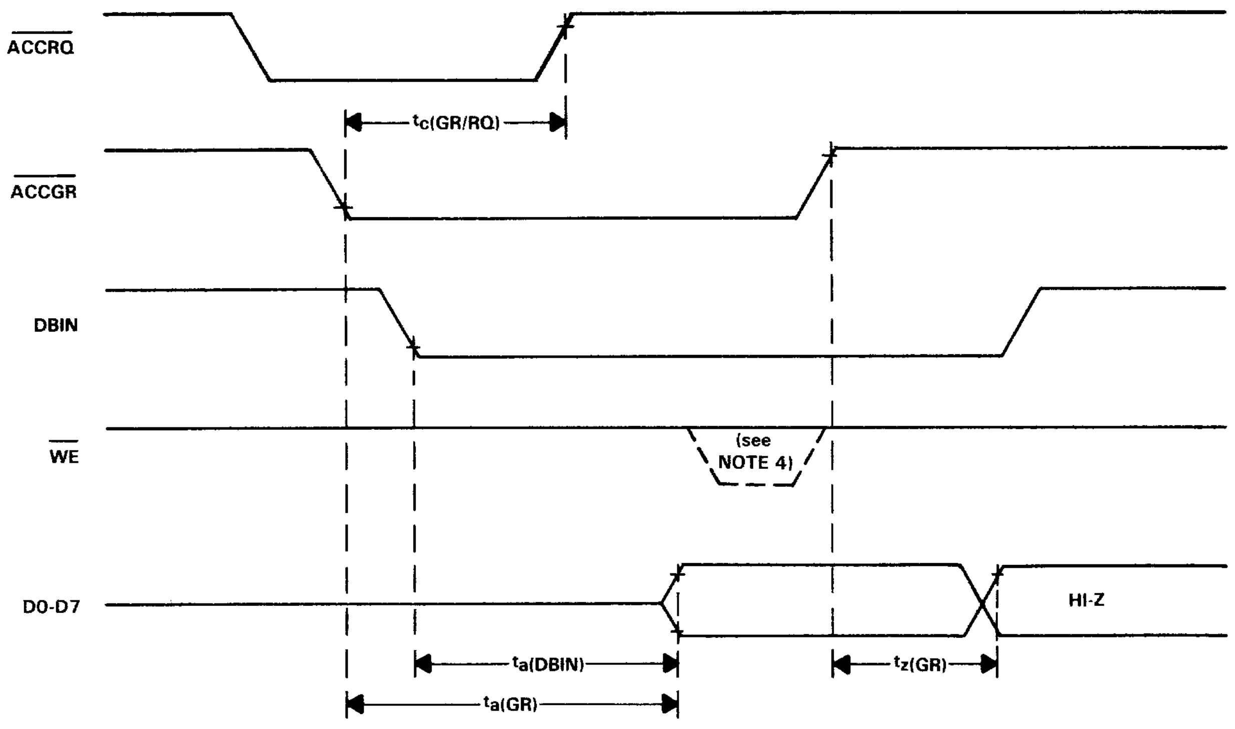


FIGURE 4-2 – TMS9914A READ CYCLE TIMING



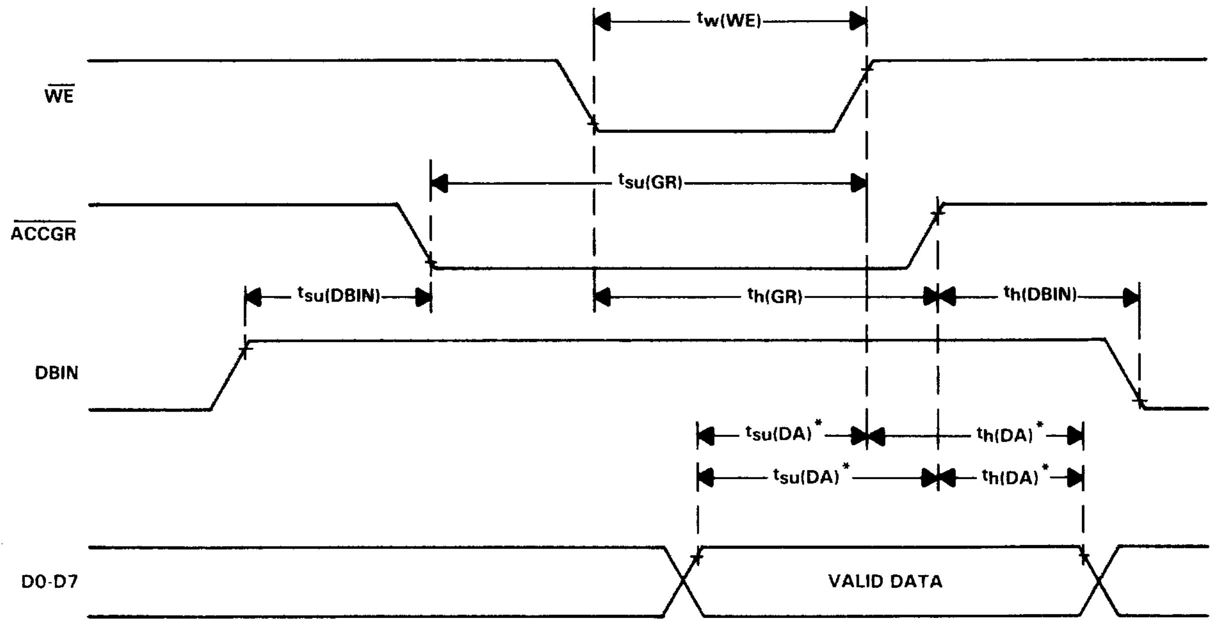
NOTE: $t_{h(AD)}$ and $t_{h(DA)}$ are shown measured from the rising edge of $\overline{WE}$. This is the correct reference point in this figure, since the measurement should be from the rising edge of $\overline{WE}$ or $\overline{CE}$, whichever comes first.

FIGURE 4-3 – TMS9914A WRITE CYCLE TIMING



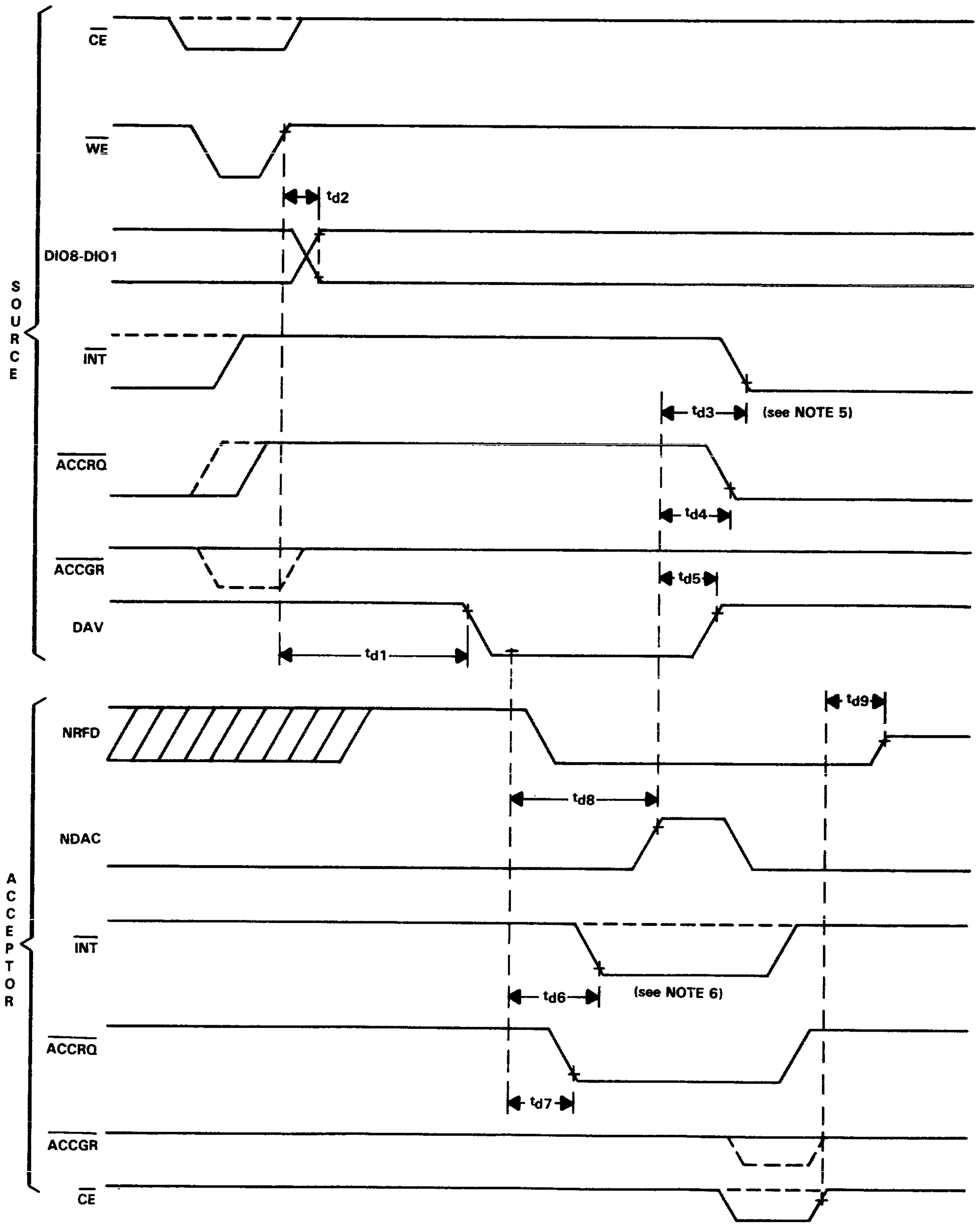
NOTE 4: A write enable pulse may occur in a DMA read operation. A write enable pulse may therefore be provided for system memory and need not be suppressed at the TMS9914A.

FIGURE 4-4 – TMS9914A DMA READ OPERATION



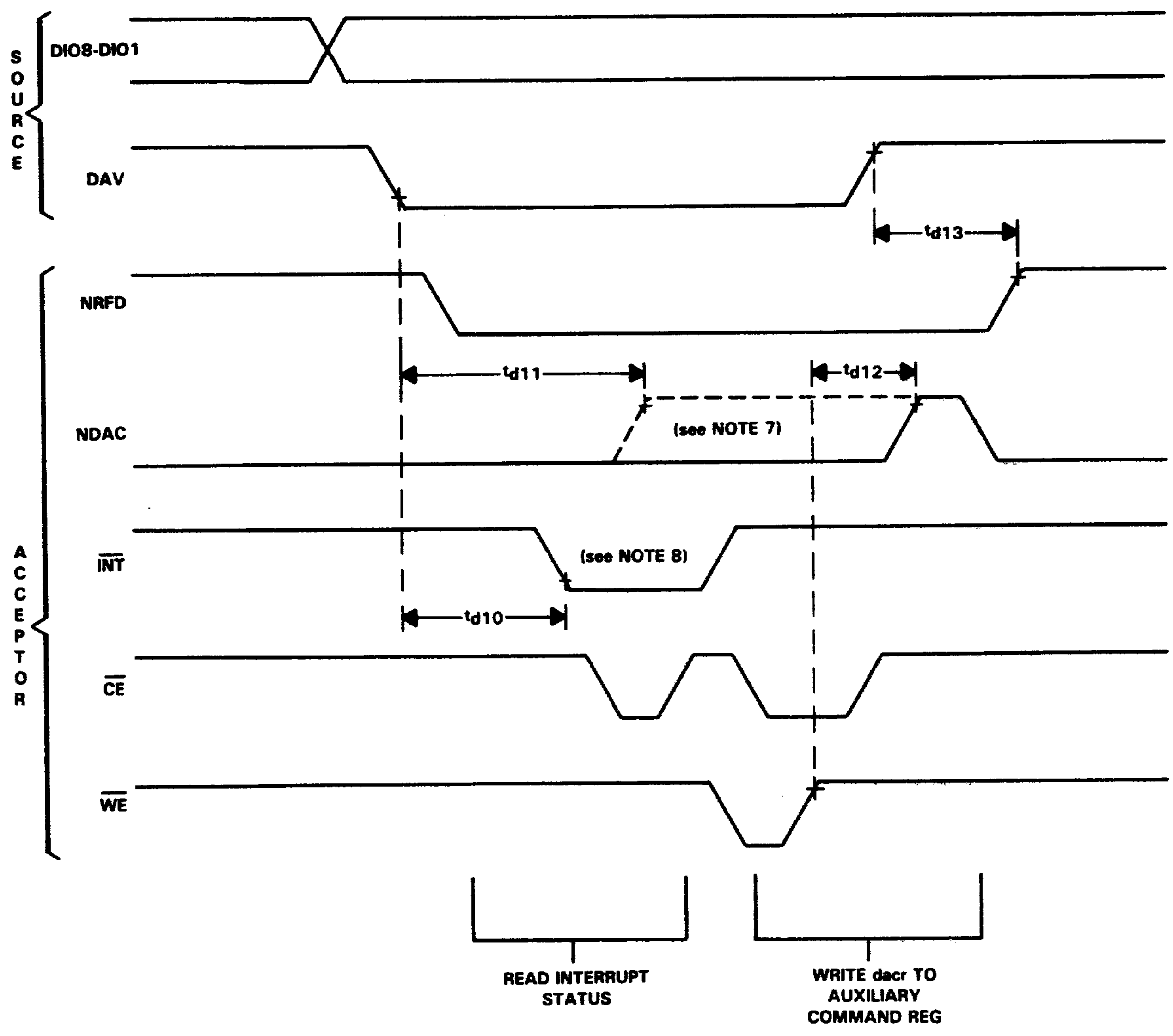
* $t_{su(DA)}$ and $t_{h(DA)}$ are only applicable to the first signal to become inactive, whether it is $\overline{WE}$ or $\overline{ACCGR}$.

FIGURE 4-5 – TMS9914A DMA WRITE OPERATION



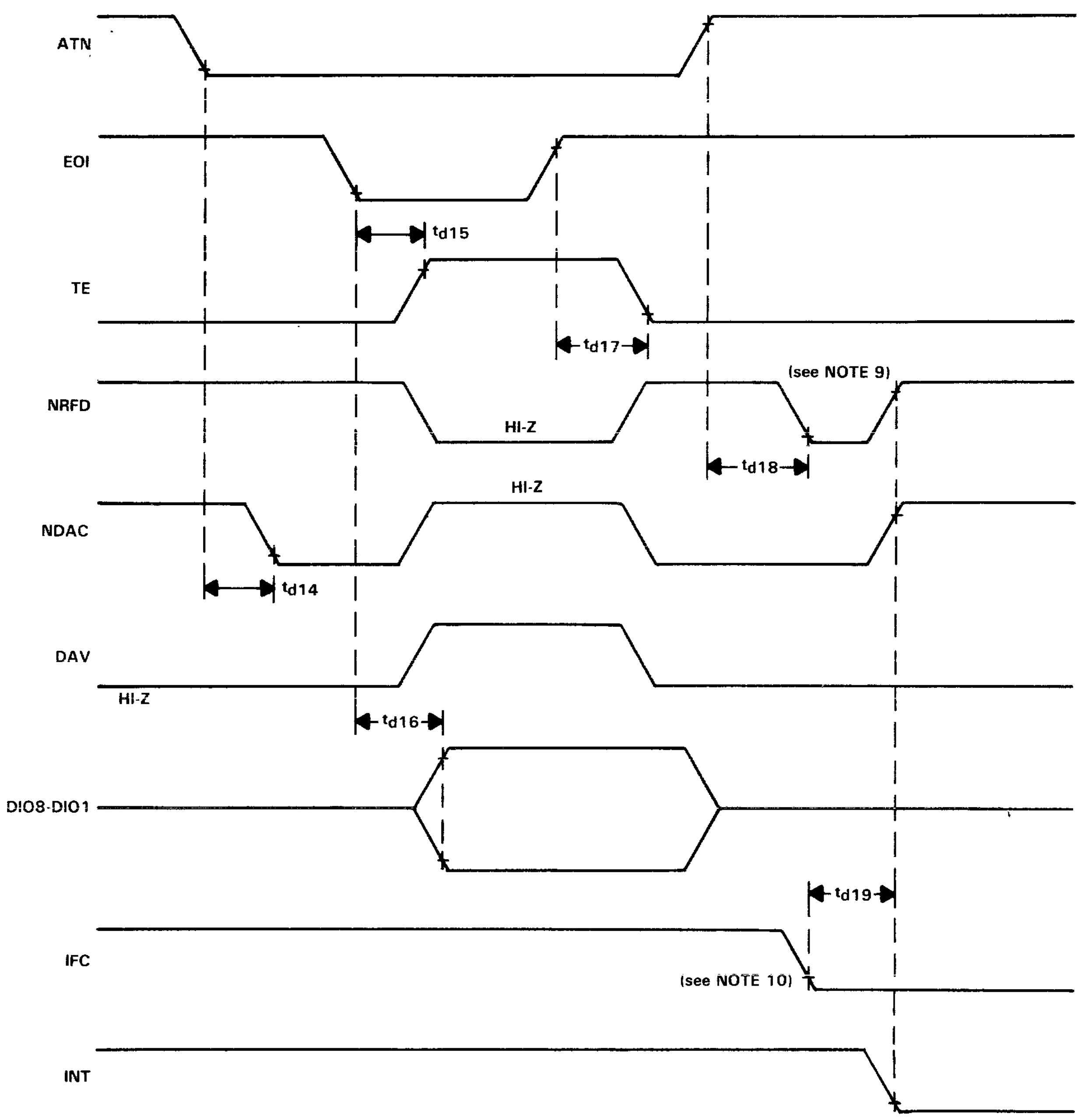
NOTES: 5. The interrupt line is taken low by a BO interrupt.
6. The interrupt line is taken low by a BI interrupt.

FIGURE 4-6 – TMS9914A SOURCE AND ACCEPTOR HANDSHAKE TIMING(S)



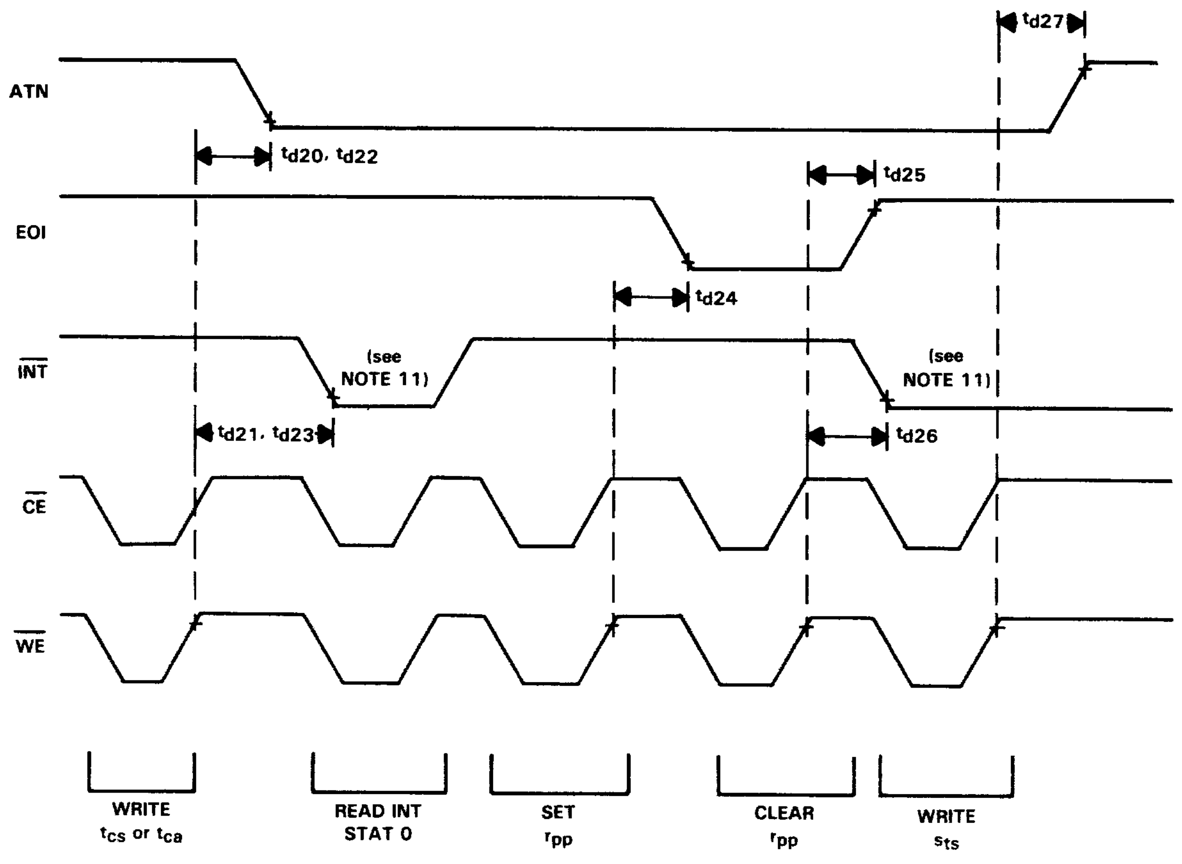
NOTES: 7. The broken line shows the waveform if there is no DAC holdoff. The solid lines assume there is a DAC holdoff.
8. The interrupts generated by interface messages are shown in Table 3-15.

FIGURE 4-7 – TMS9914A ACCEPTOR HANDSHAKE TIMING "ATN" TRUE



NOTES: 9. This assumes that an RFD holdoff occurs.
10. IFC causes the TMS9918A to be unaddressed and an IFC interrupt occurs.

FIGURE 4-8 – TMS9914A RESPONSE TO 'ATN' AND 'EOI'



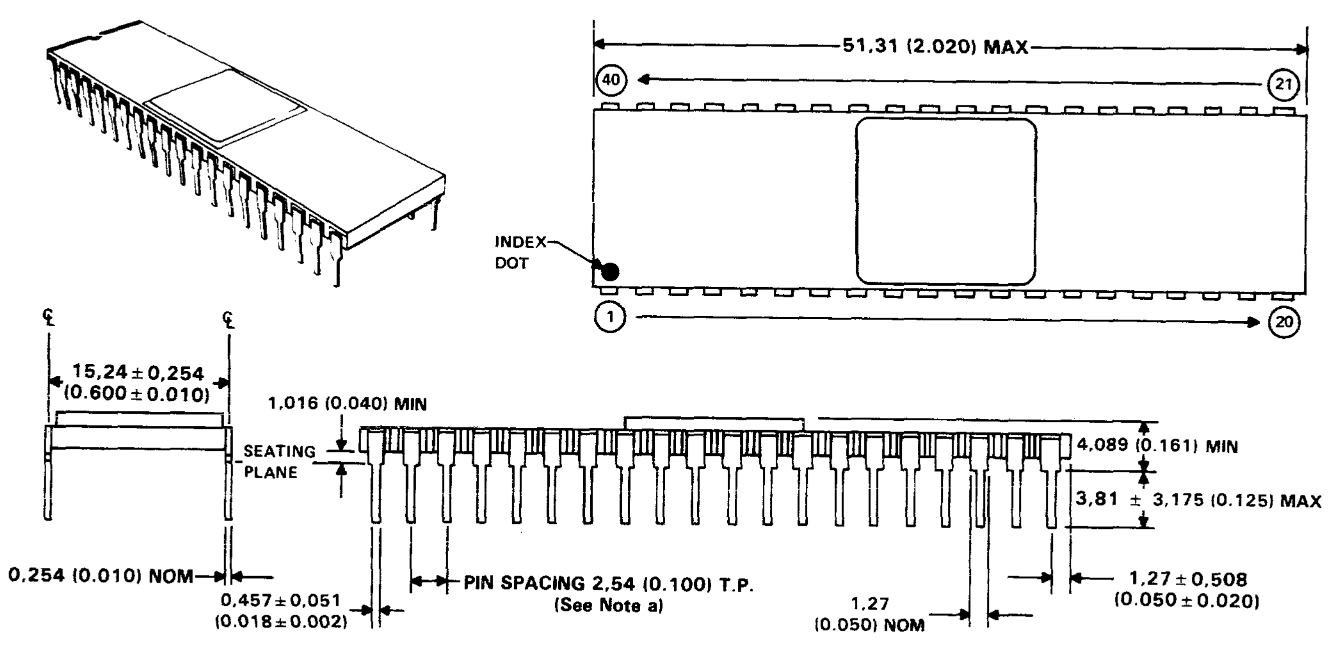
NOTE 11: A B0 interrupt occurs as the TMS9914A enters CACS.

FIGURE 4-9 – TMS9914A CONTROLLER TIMING

5. MECHANICAL SPECIFICATIONS

5.1 TMS9914AJL – 40-PIN CERAMIC PACKAGE

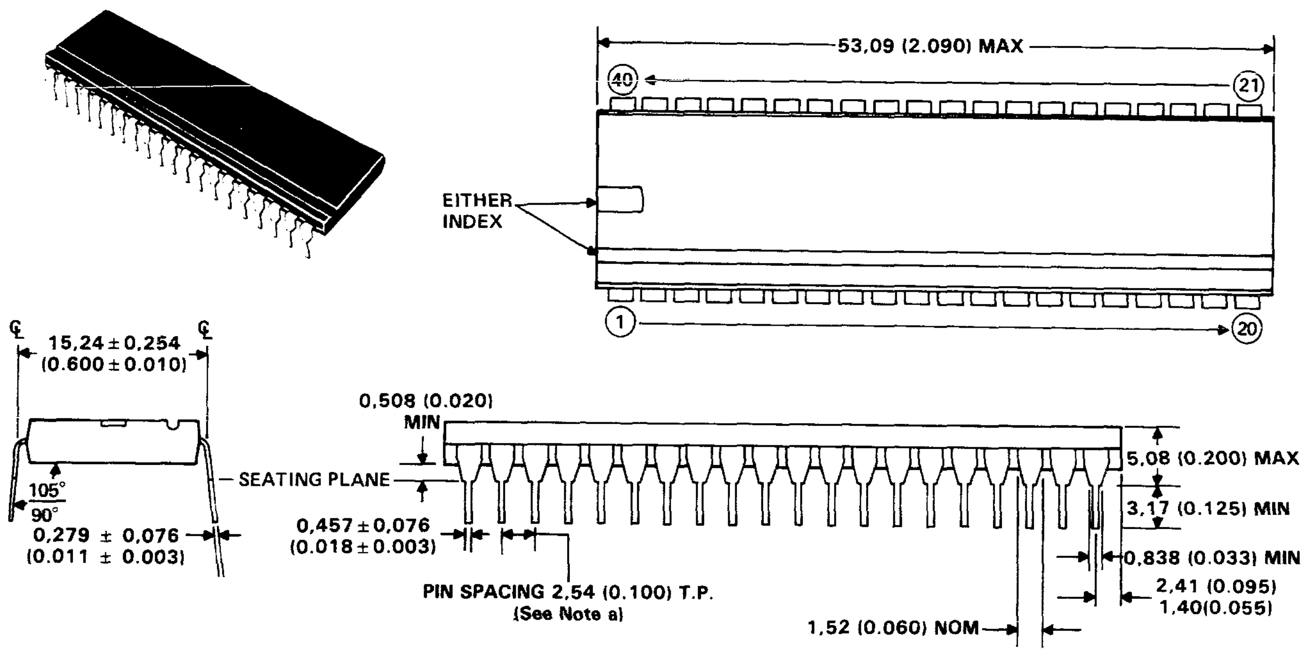
Ceramic packages with side-brazed leads and metal, epoxy, or glass lid seal



NOTES: a. Each pin centerline is located within 0,254 (0.010) of its true longitudinal position.
b. All linear dimensions are in millimeters and parenthetically in inches. Inch dimensions govern.

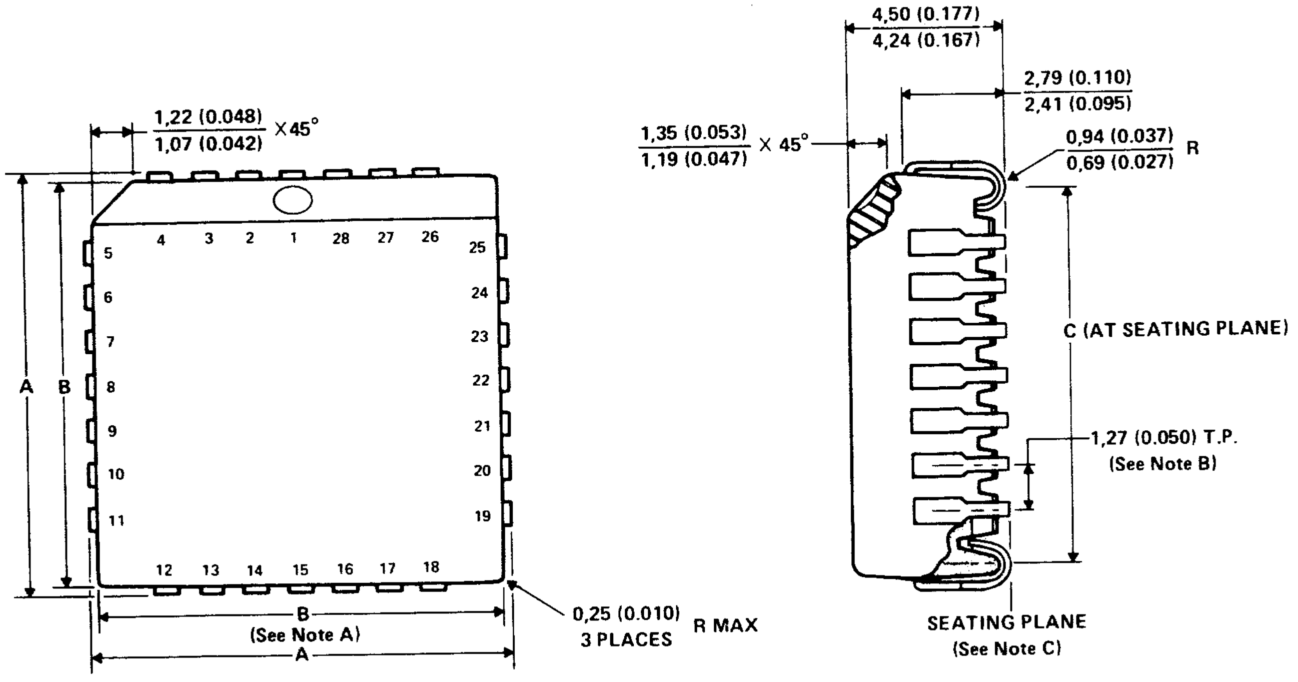
5.2 TMS9914ANL – 40-PIN PLASTIC PACKAGE

Plastic packages



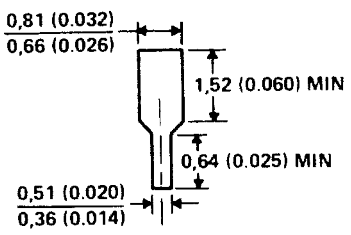
NOTES: a. Each pin centerline is located within 0,254 (0.010) of its true longitudinal position.
b. All linear dimensions are in millimeters and parenthetically in inches. Inch dimensions govern.

5.3 TMS9914AFN—44-Pin PLCC Package



1,22 (0.048) / 1,07 (0.042) X45°
4 3 2 1 28 27 26 25
5 6 7 8 9 10 11
24 23 22 21 20 19
12 13 14 15 16 17 18
A
B
B (See Note A)
A
0,25 (0.010) 3 PLACES R MAX
4,50 (0.177) / 4,24 (0.167)
2,79 (0.110) / 2,41 (0.095)
1,35 (0.053) / 1,19 (0.047) X 45°
0,94 (0.037) / 0,69 (0.027) R
C (AT SEATING PLANE)
1,27 (0.050) T.P. (See Note B)
SEATING PLANE (See Note C)

NO. OF TERMINALS	A		B		C	
	MIN	MAX	MIN	MAX	MIN	MAX
20	9,78 (0.385)	10,03 (0.395)	8,89 (0.350)	9,04 (0.356)	7,87 (0.310)	8,38 (0.330)
28	12,32 (0.485)	12,57 (0.495)	11,43 (0.450)	11,58 (0.456)	10,41 (0.410)	10,92 (0.430)
44	17,40 (0.685)	17,65 (0.695)	16,51 (0.650)	16,66 (0.656)	15,49 (0.610)	16,00 (0.630)
68	25,02 (0.985)	25,27 (0.995)	24,13 (0.950)	24,33 (0.956)	23,11 (0.910)	23,62 (0.930)



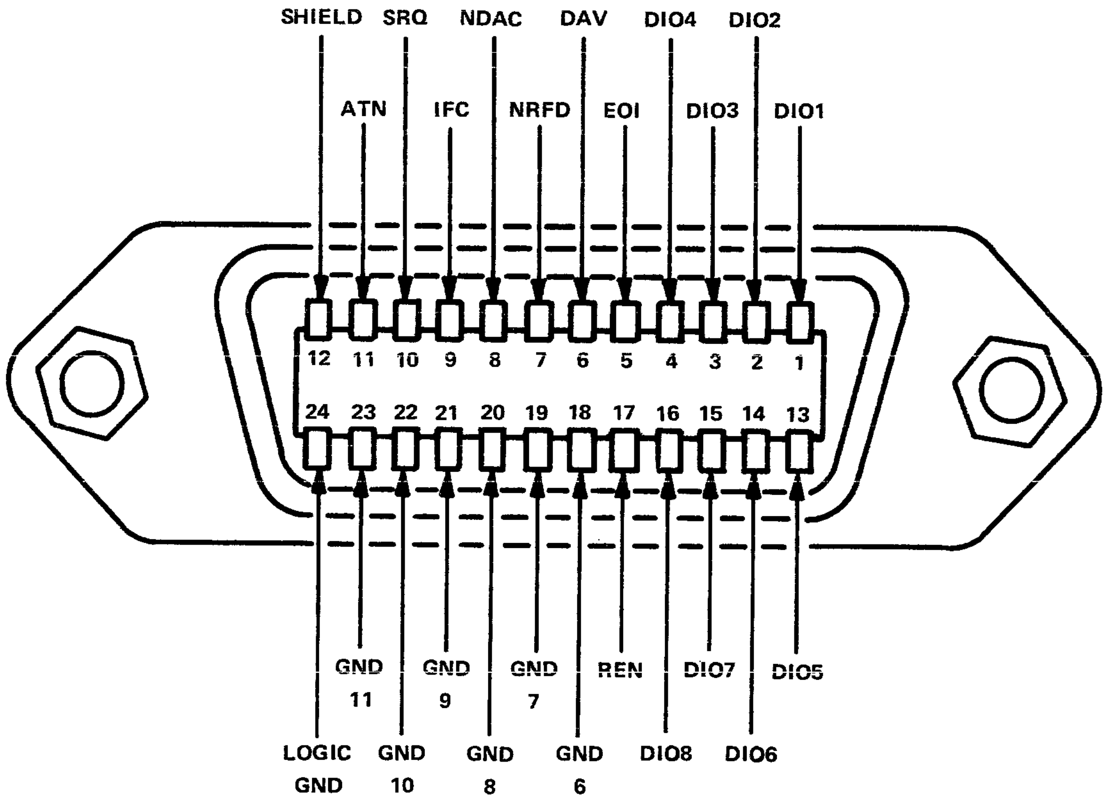
LEAD DETAIL

ALL LINEAR DIMENSIONS ARE IN MILLIMETERS AND PARENTHETICALLY IN INCHES

NOTES: A. Centerline of center pin each side is within 0,10 (0.004) of package centerline as determined by dimension B.
B. Location of each pin is within 0,127 (0.005) of true position with respect to center pin on each side.
C. The lead contact points are planar within 0,10 (0.004).

APPENDIX A

IEEE-488 STANDARD CONNECTOR



APPENDIX B

TYPICAL SEQUENCES OF EVENTS FOR THE CONTROLLER

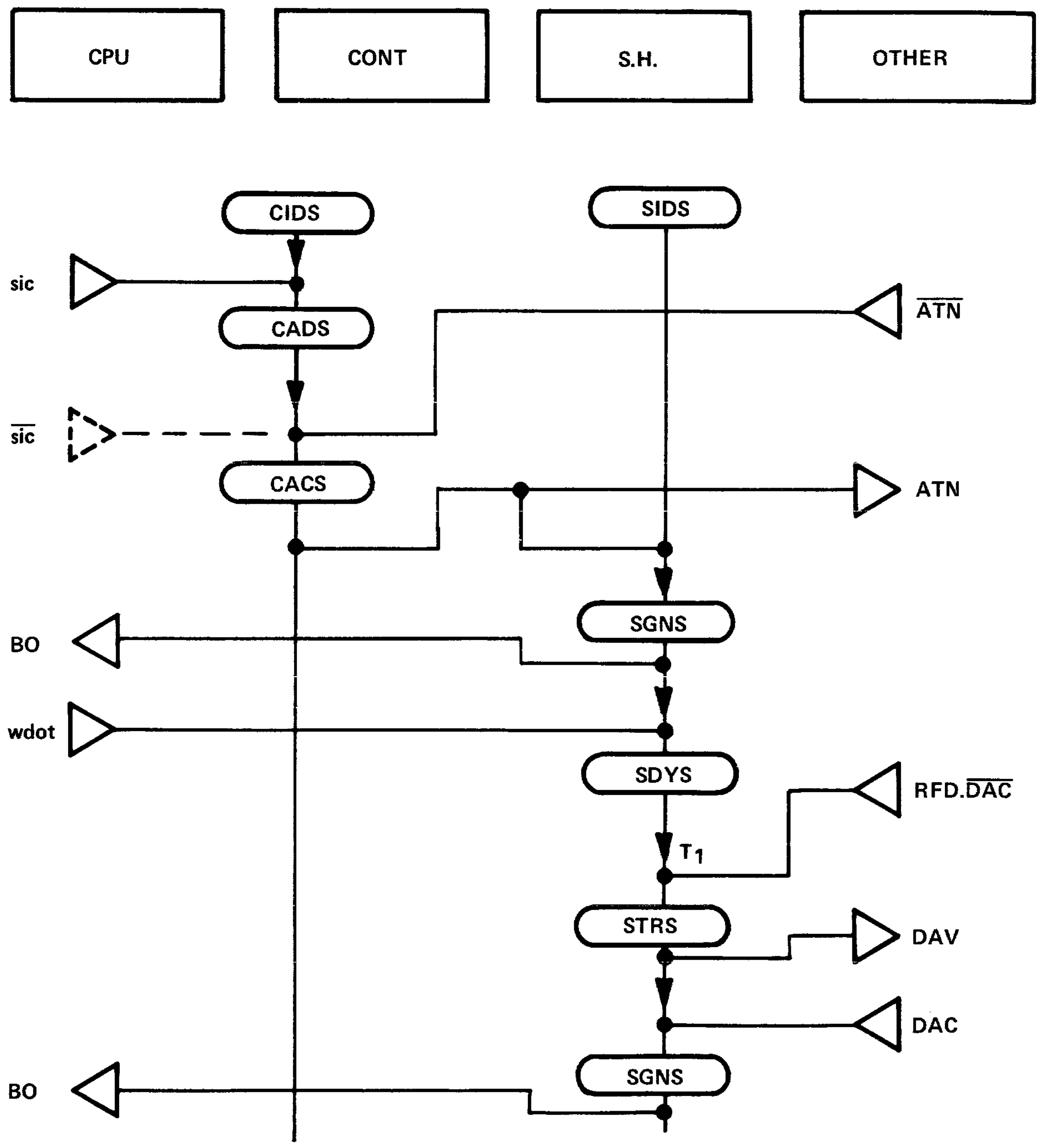
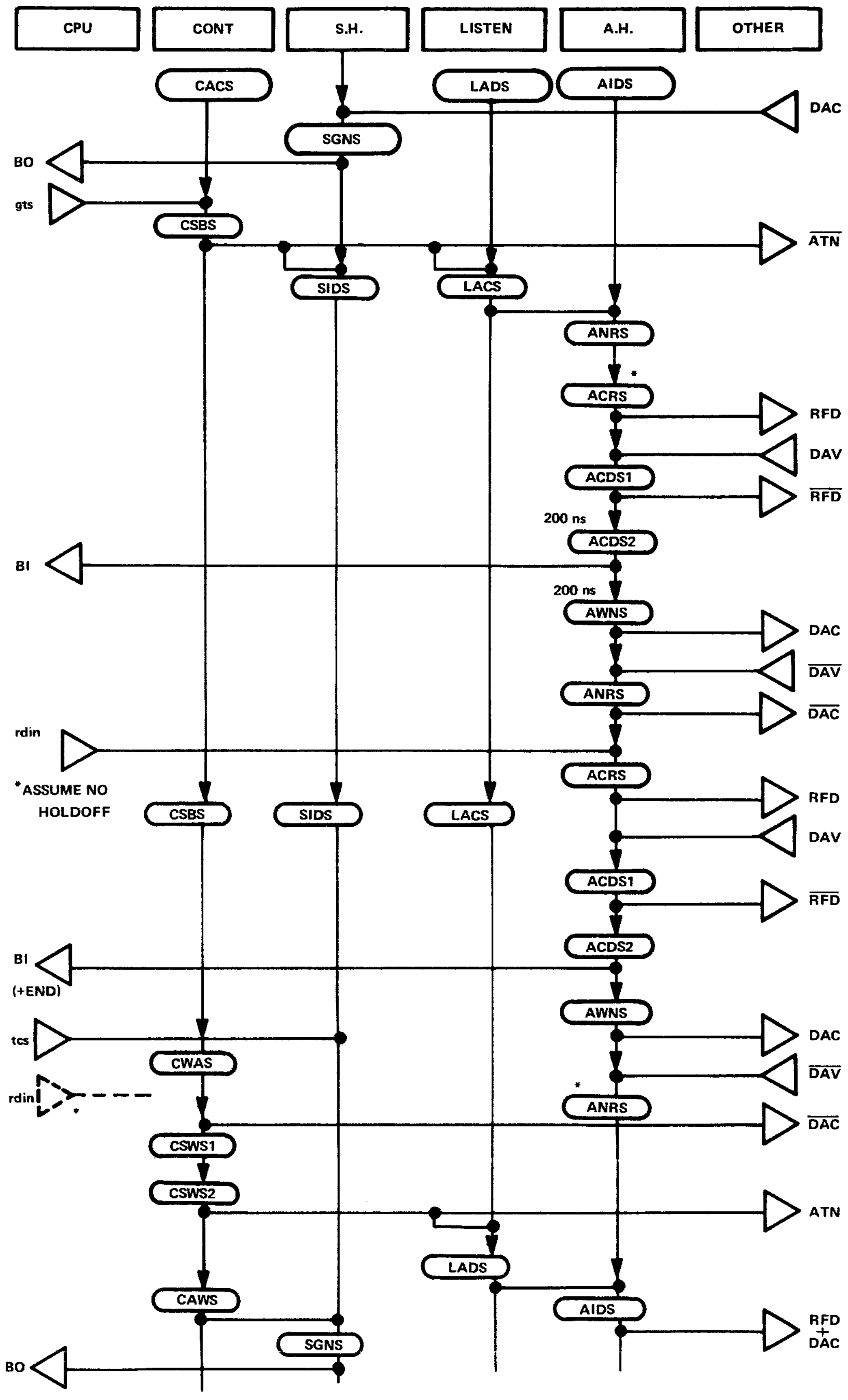
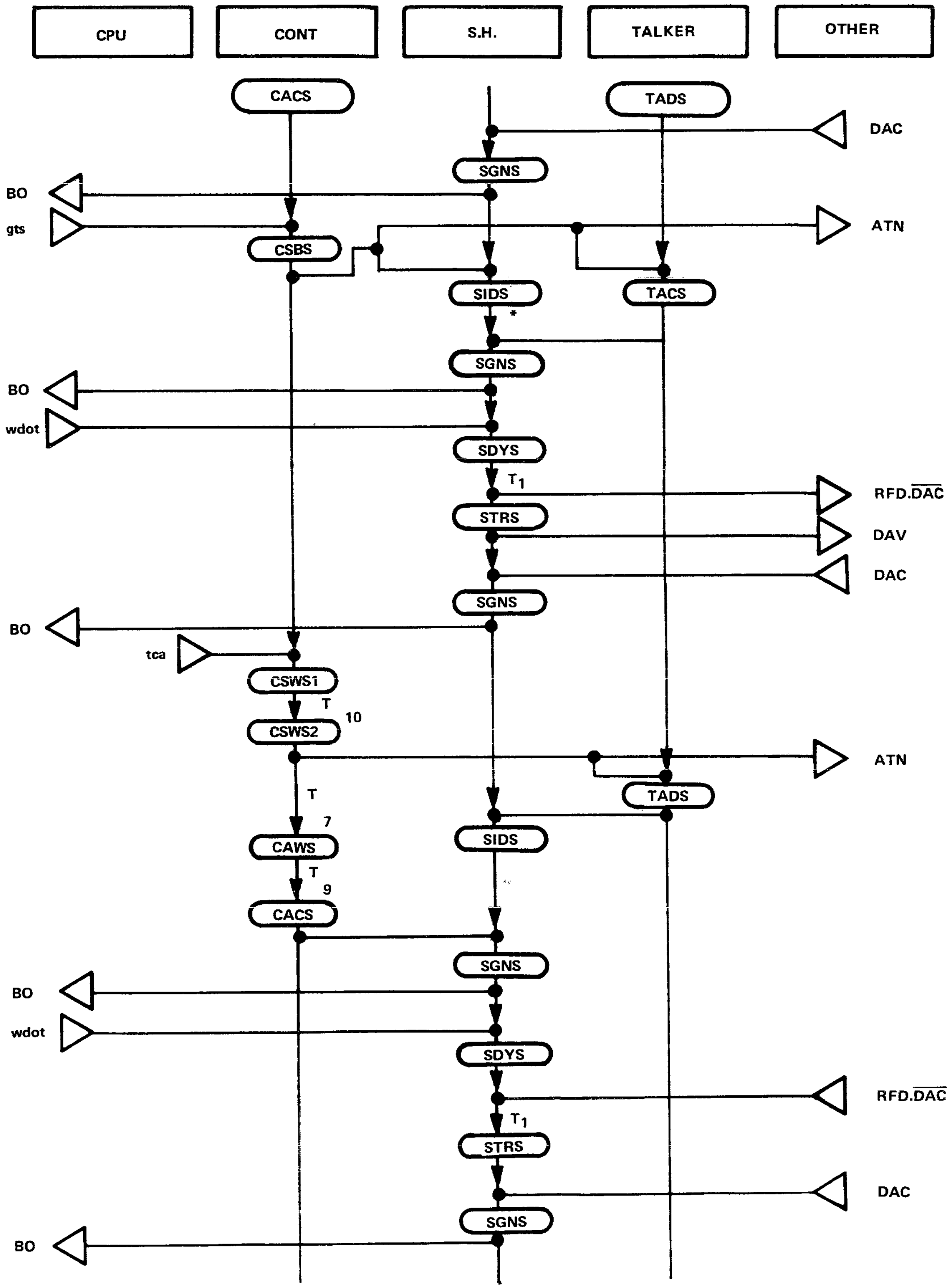


FIGURE B-1 – CONTROLLER TAKING CONTROL



*CWAS inhibits ANRS → ACRS, so rdin can occur before ATN is set.

FIGURE B-2 – CONTROLLER AS A LISTENER (GOING TO STANDBY)



*Momentary transition following BO interrupt may not occur on the TMS9914 but is guaranteed on the TMS9914A.

FIGURE B-3 – CONTROLLER AS A TALKER (GOING TO STANDBY)

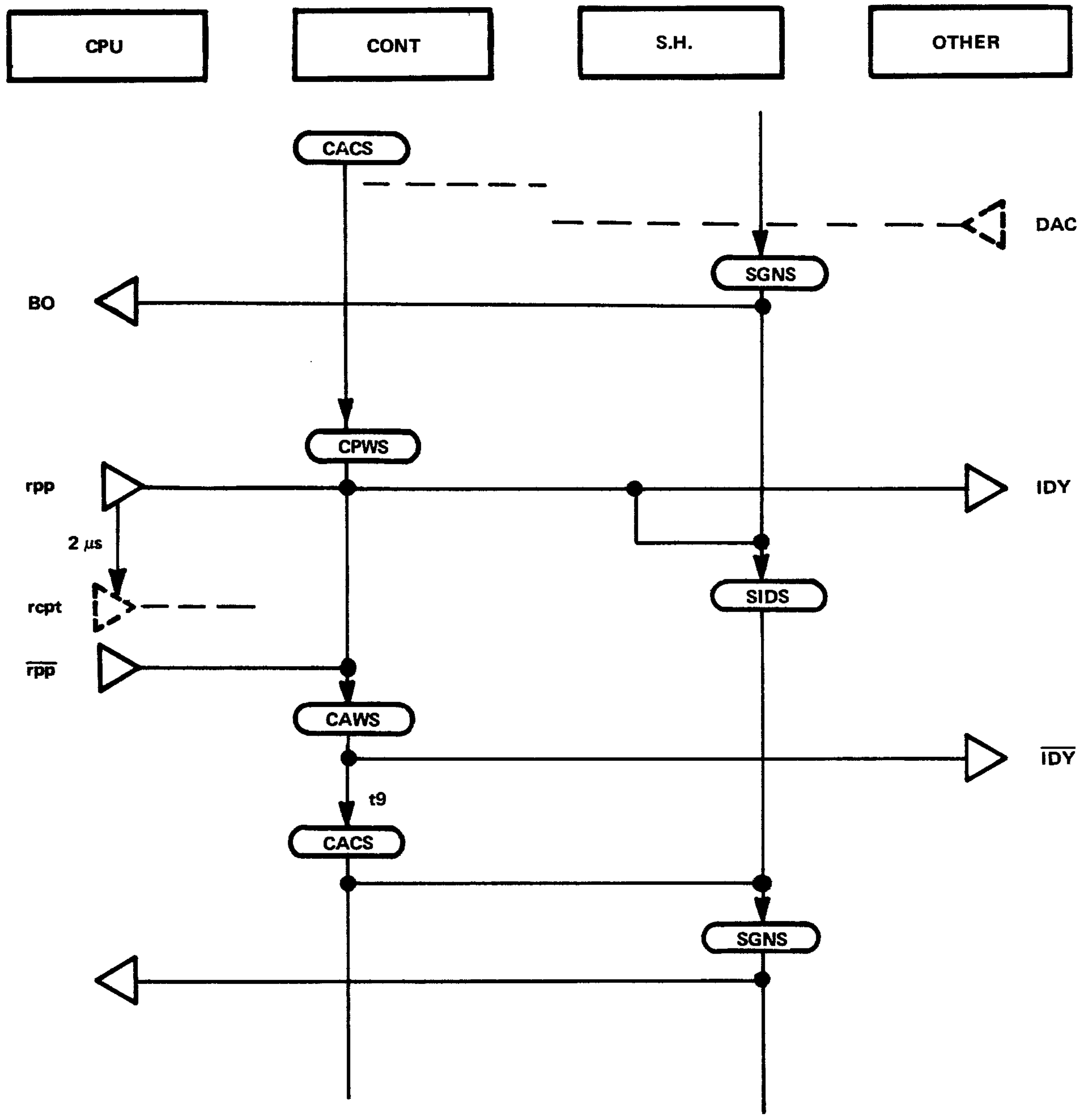


FIGURE B-4 – CONTROLLER PARALLEL POLLING

APPENDIX C

SN75160/161/162 DATA SHEETS

Texas Instruments SN75160 family of bus transceivers are designated to provide the interface between the bus and the bus controller. These transceivers may be used with the TI TMS9914 Bus Controller chip or any of the other GPIB controllers commercially available. They provide the simplest method of interfacing to the bus, because each part is tailored to either the 8-line data bus or 8-line control bus, so they require no extra logic or complicated board layout. With the SN75160 family, it takes only two 20-pin DIP packages to get on the GPIB. The new improved SN75160A series is pin-for-pin compatible with the original SN75160 series but with lower power and faster speeds, as shown in Figure C-1.

All transceivers in the SN75160 family have several features in common. Each driver output has built into it the termination network required by IEEE Standard 488. This termination is designed so that when power is removed from the transceiver, the output presents a high impedance to the bus. Also, each receiver has a minimum of 400 mV hysteresis for additional noise margin.

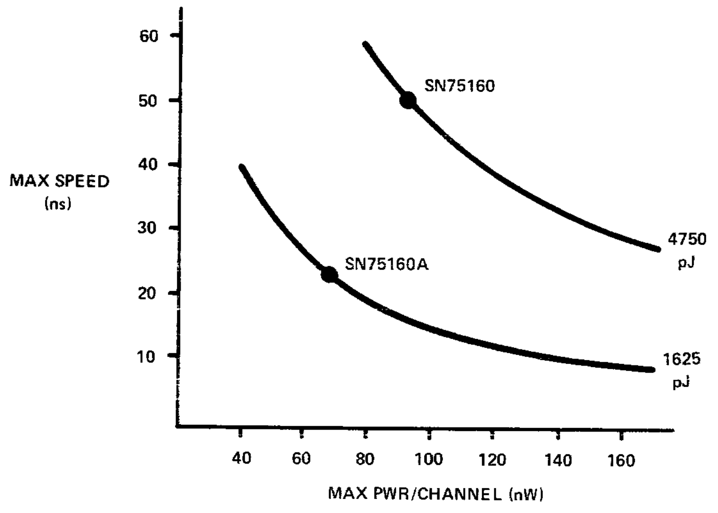


FIGURE C-1

The SN75160A is designed to implement the 8-line data bus. The direction of data flow is controlled by the Talk Enable (TE) input. All eight channels are simultaneously in the receive mode when the TE is low and data is received fron the bus and transferred to the bus controller. When the TE is in the high state, all eight channels go to the transmit mode, and data will be transmitted onto the bus. Each driver features a totem-pole output which can actively drive the bus high or low to give the fastest data rates possible. The SN75160A has a Pull-up Enable (PE) input which, when taken low, disables the upper stage of the driver outputs turning all eight driver outputs into open-collector type outputs. The open-collector output mode does not allow as fast a data rate as with the totem-pole, but it does allow more than one instrument to be transmitting on the bus at the same time. This feature is used in parallel polling where up to eight instruments may be polled simultaneously, each responding on one line of the eight-line data bus, greatly speeding the polling process. They may then be switched back to the totem-pole mode for regular data transmission.

The SN75161A is used to implement the 8-line control bus. Included in it is the necessary logic which, combined with the Talk-Enable (TE) and Direction Control (DC) inputs, insures that each channel is enabled in the correct direction for exchange of bus management and handshaking signals. Three of the channels, NDAC, NRFD, and SRQ, have open-collector driver outputs as required by the IEEE Standard 488. These lines are always used in a wired-OR configuration. The other five channels have totem-pole outputs. The SN75162A offers an alternate method of implementing the control bus. The SN75162A is identical to the SN75161A except that the direction of the REN and IFC channels is controlled by a separate input called the System Controller (SC). With this additional flexibility, control of the entire Bus System may be transferred from one instrument to another (multiple controller systems). Because of this extra input, the SN75162A package has 22 pins.

SN75160A, SN75161A AND SN75162A OCTAL IEEE-488 GPIB BUS TRANSCEIVERS

features

- 8-channel bidirectional transceivers
- Meet IEEE standard 488 1978
- Low power dissipation (65 mW max per channel)
- High-impedance PNP inputs
- Receiver hysteresis (500 mV typ)
- Open-collector driver output option (SN75160A)
- Bus-terminating resistors provided on driver outputs
- No loading of bus when device is powered down (V_{CC} = 0 V)
- SN75161A for single-controller systems; SN75162A for multi-controller systems

NOTE: Integrated Schottky-Barrier diode-clamped transistor is patented by Texas Instruments. U.S. Patent Number 3,463,975.

description

These octal bus transceivers are designed to provide communication on the general-purpose interface bus (GPIB) between operating units of the instrumentation system. The sixteen signal lines are normally required by the SN75162A in systems with more than one controller. An active turn-off feature has been incorporated into the bus-terminating resistors so that the devices exhibit a high impedance to the bus when V_{CC} = 0 V.

When PE is low, the bus outputs of the SN75160A have the characteristics of open-collector outputs. They act as three-state ports when PE is high. Taking TE low places those ports in the free-state, wherein they can be driven by the bus lines, and enables the D outputs.

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)

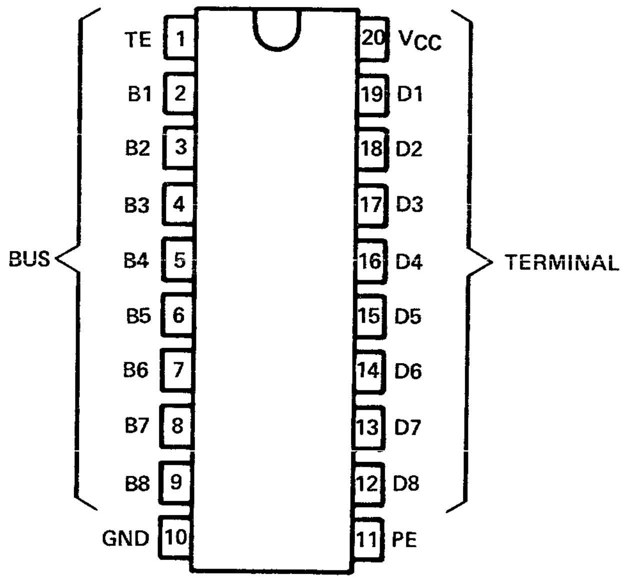
Supply voltage, V_{CC} (see Note 1)	7 V
Input voltage	5.5 V
Low-level driver output current	100 mA
Continuous total dissipation at (or below) 25 °C free-air temperature (see Note 2)	1150 mW
Operating free-air temperature range	0 °C to 70 °C
Storage temperature range	−65 °C to 150 °C
Lead temperature 1/16 inch (1.6 mm) from case for 10 seconds	260 °C

NOTES: 1. All voltage values are with respect to network ground terminal.
2. For operation above 25 °C free-air temperature, derate linearly at the rate of 9.3 mW/ C to 740 mW at 70 °C.

table of abbreviations

CLASS	NAME	IDENTITY
CONTROL INPUTS	DC	Direction Control
	PE	Pull-up Enable
	TE	Talk Enable
SN75161A I/O PORTS	B	Bus side of device
	D	Terminal side of device
SN75161A/162A SIGNAL MNEMONICS	ATN	Attention
	DAV	Data Valid
	EOI	End or Identify
	IFC	Interface Clear
	NDAC	Not Data Accepted
	NRFD	Not Ready for Data
	REN	Remote Enable
	SRQ	Service Request
	SC	System Controller

SN75160A N DUAL-IN-LINE PACKAGE (TOP VIEW)

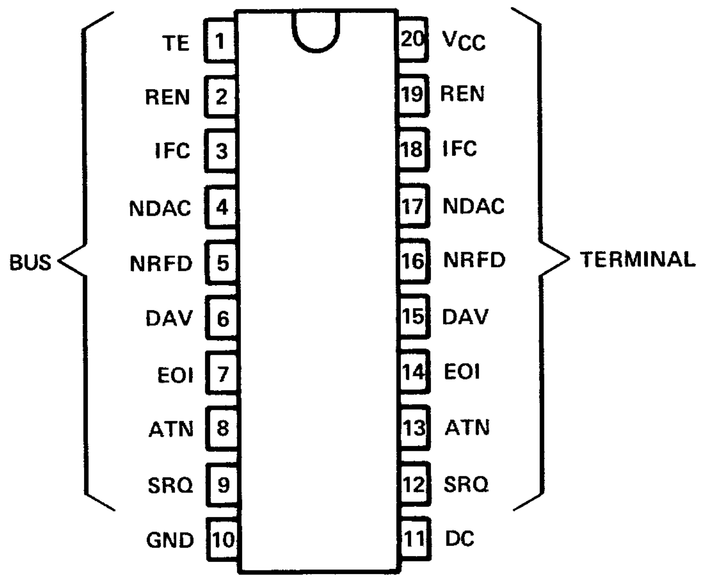


SN75160A function table

DRIVERS				RECEIVERS			
INPUTS			OUTPUTS	INPUTS			OUTPUTS
D	TE	PE	B	B	TE	PE	D
H	H	H	H	L	L	X	L
L	H	H	L	H	L	X	H
H	X	L	F	X	H	X	Z
L	H	L	L				
X	L	X	F				

F = free state*, H = high level, L = low level, X = irrelevant, Z = high-impedance state.
*This is the high-impedance state of a normal 3-state output modified by the internal resistors to V_{CC} and ground.
modified by the internal resistors to V_{CC} and ground.

SN75161A N DUAL-IN-LINE PACKAGE (TOP VIEW)



SN75161A function table

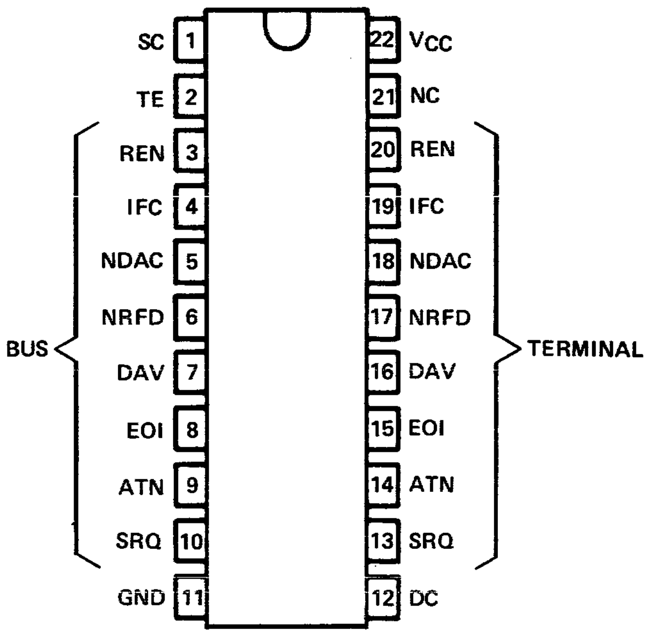
CONTROLS[†]				DIRECTION OF DATA[‡]						
TE	DC	ATN LEVEL	ATN DIRECTION	EOI	REN	IFC	SRQ	NRFD	NDAC	DAV
H	H	H	R	T	R	R	T	R	R	T
H	H	L	R	R	R	R	T	R	R	T
H	L	X	T	T	T	T	R	R	R	T
L	H	X	R	R	R	R	T	T	T	R
L	L	H	T	R	T	T	R	T	T	R
L	L	L	T	T	T	T	R	T	T	R

H = high level, L = low level, R = receive, T = transmit, X = irrelevant

[†] ATN is a normal transceiver channel that functions additionally as an internal direction control or talk enable for EOI whenever the TE and DC inputs are in the same state. When TE and DC are in opposite states, the ATN channel functions as an independent transceiver only.

[‡] Direction of data transmission is from the terminal side to the bus side, and the direction of data receiving is from the bus side to the terminal side. Data transfer is noninverting in both directions.

SN75162A N DUAL-IN-LINE PACKAGE (TOP VIEW)



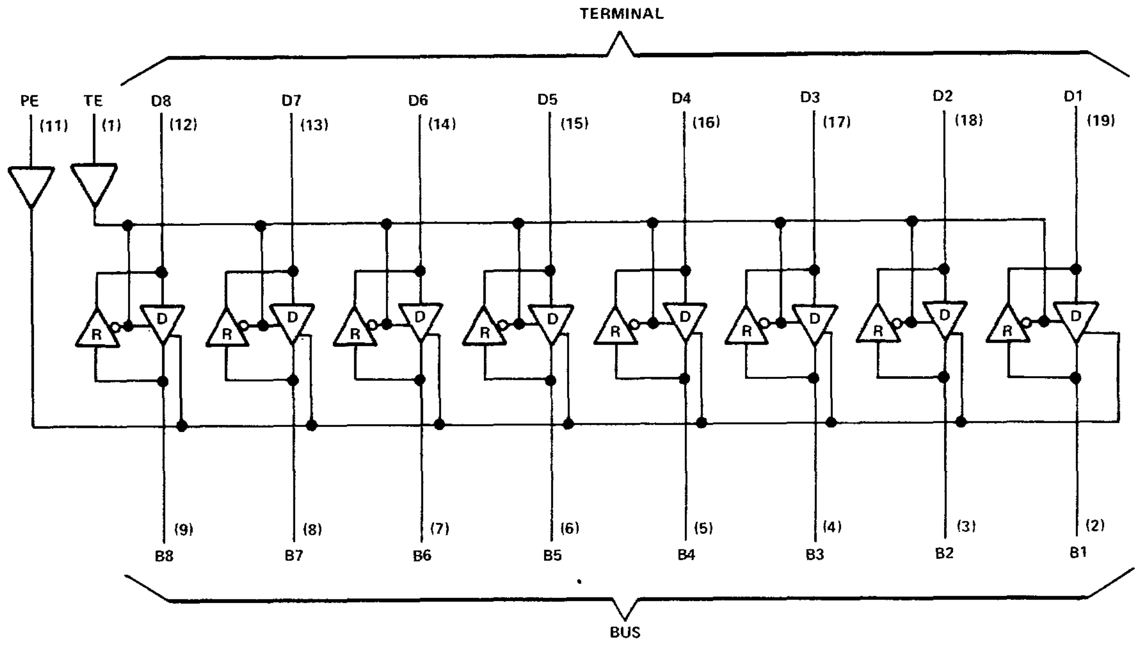
SN75162A function table

CONTROLS					DIRECTION OF DATA						
TE	DE	SC	ATN LEVEL	ATN DIRECTION	EOI	REN	IFC	SRQ	NRFD	NFAC	DAV
H	H	L	H	R	T	R	R	T	R	R	T
H	H	L	L	R	R	R	R	T	R	R	T
H	L	L	X	T	T	R	R	R	R	R	T
L	H	L	X	R	R	R	R	T	T	T	R
L	L	L	H	T	R	R	R	R	T	T	R
L	L	L	L	T	T	R	R	R	T	T	R
H	H	H	H	R	T	T	T	T	R	R	T
H	H	H	L	R	R	T	T	T	R	R	T
H	L	H	X	T	T	T	T	R	R	R	T
L	H	H	X	R	R	T	T	T	T	T	R
L	L	H	H	T	R	T	T	R	T	T	R
L	L	H	L	T	T	T	T	R	T	T	R

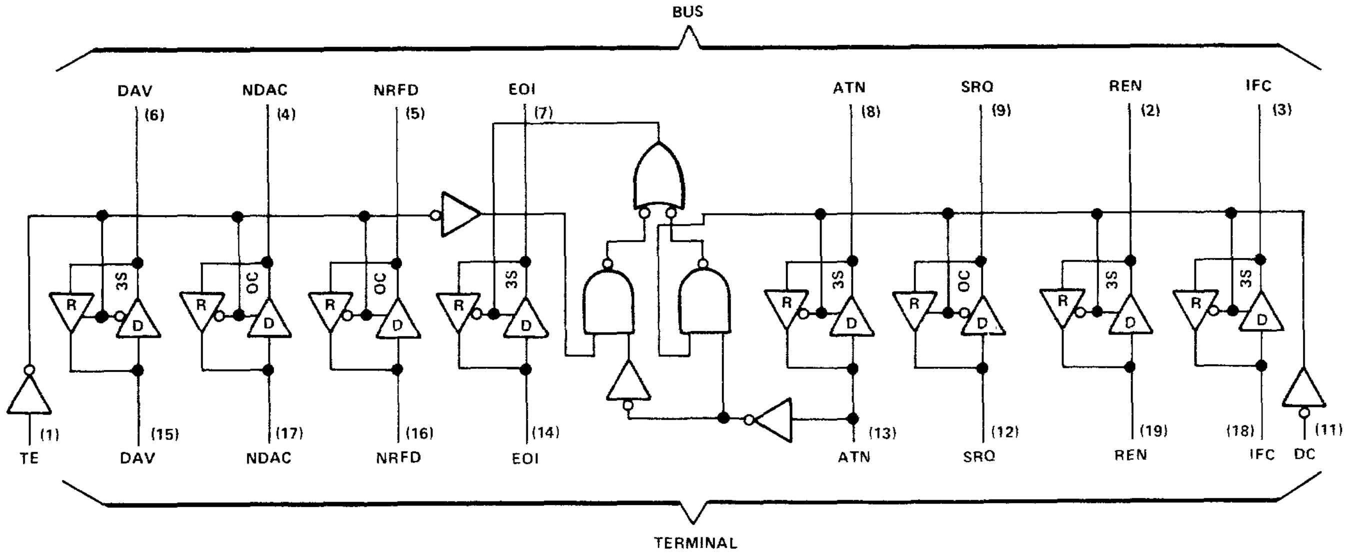
H = high, L = low, R = receive, T = transmit, X = irrelevant.

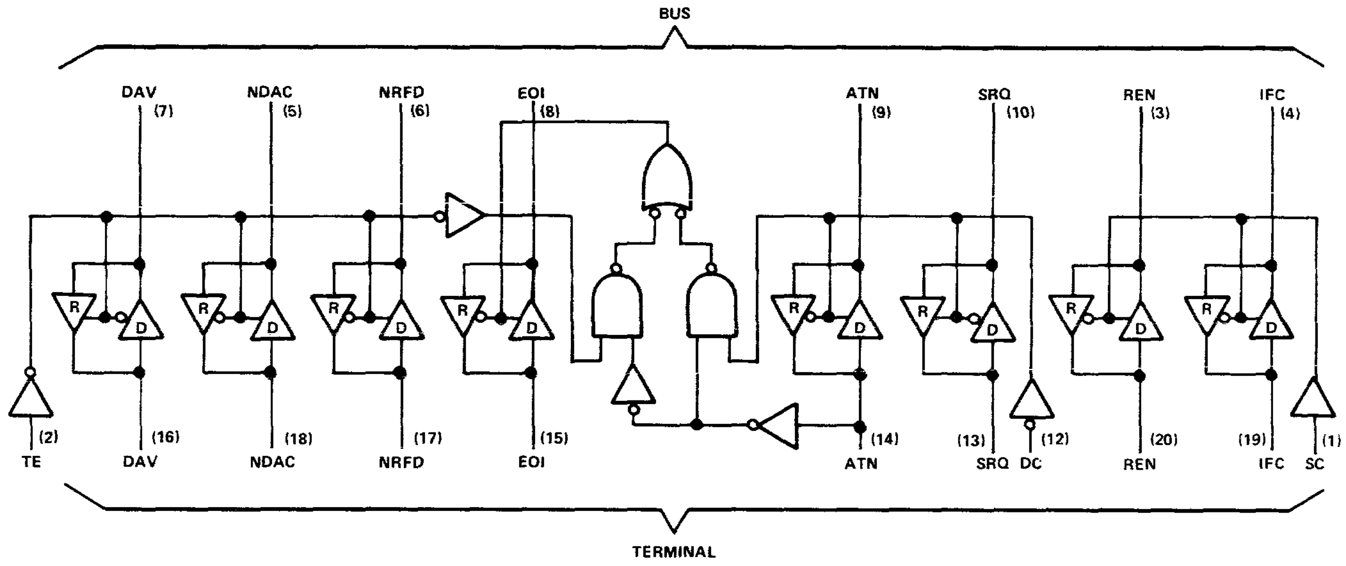
functional block diagrams

SN75160A



SN75161A





switching characteristics, V_{CC} = 5 V, C_L = 15 pF, T_A = 25°C (unless otherwise noted)

PARAMETER		FROM	TO	TEST CONDITIONS	SN75160A MIN	SN75160A TYP	SN75160A MAX	SN75161A MIN	SN75161A TYP	SN75161A MAX	SN75162A MIN	SN75162A TYP	SN75162A MAX	UNIT
t_{PLH}	Propagation delay time, low-to-high-level output	Terminal	Bus	$C_L = 30$ pF, $R_L = 38.3\ \Omega$ to 2.3 V		14	20		17	25		17	25	ns
t_{PHL}	Propagation delay time, high-to-low-level output					14	20		17	25		17	25	
t_{PLH}	Propagation delay time, low-to-high-level output	Bus	Terminal	$C_L = 30$ pF, $R_L = 240\ \Omega$ to 5 V		12	20		16	25		16	25	ns
t_{PHL}	Propagation delay time, high-to-low-level output					15	22		16	25		16	25	
t_{PZH}	Output enable time to high level	$\overline{TE}$ or DC	Bus	$R_L = 480\ \Omega$ to 0 V		25								ns
t_{PHZ}	Output disable time from high level					12								
t_{PZL}	Output enable time to low level			$R_L = 38.3\ \Omega$ to 2.3 V		22								
t_{PLZ}	Output disable time from low level					21								
t_{PZH}	Output enable time to high level	TE or DC	Terminal	$R_L = 3\text{k}\Omega$ to 0 V		20								ns
t_{PHZ}	Output disable time from high level					13								
t_{PZL}	Output enable time to low level			$R_L = 280\ \Omega$ to 0 V		23								
t_{PLZ}	Output disable time from low level					19								
	Output pull-up enable time	PE	Terminal	$R_L = 480\ \Omega$ to 0 V		15								ns
	Output pull-up disable time					13								

APPENDIX D

EXAMPLE SOFTWARE

DESIGN EXAMPLE

DESIGN OBJECTIVES

- Illustrate the procedures and protocol of the remote and local messages used to configure a controller.
- Illustrate the procedures used to configure a remote instrument to acquire data and then transmit the data over the GPIB bus to some other device.
- Demonstrate the use of the TMS9914 as a controller, talker, and listener.
- Show the software necessary to drive the TMS9914 when it is interfaced to TM990 products.
- Show the elements of hardware design which must be dealt with when interfacing the TMS9914 to an MPU.
- Show the hardware necessary to interface the TMS9914 to the GPIB bus.

STATEMENT OF THE EXAMPLE DESIGN PROBLEM

- TM990/101-based system
- Use the TMS9914 to communicate with the GPIB.
- Configure an HP3455A digital voltmeter to take two-wire resistance measurements.
- Trigger the meter to take a measurement and send the data over the GPIB to the TM990 system.
- Decode the ASCII data into speech and provide a vocal annunciation of the resistance measurement.

SYSTEM COMPONENTS CHOSEN

- Hewlett Packard model 3438A digital voltmeter
  - IEEE-488 compatible device
  - Range and function selection locally selected and indicated in measurement data
  - Remote measurement trigger capability
- TI TM990/101M microcomputer CPU module
  - TMS9900 16-bit CPU
  - 4K bytes of RAM mappable to either the bottom or top of the memory address space
  - Up to 8K bytes of EPROM at the bottom of memory space
  - TMS9901 programmable system interface providing up to 16 prioritized maskable interrupts and interval timer
  - Two serial I/O ports (RS232C compatible)
- TI TM990/306 speech module
  - TM990 series bus compatible
  - 180-word vocabulary
  - 2.5 W amplifier for direct speaker drive
  - Programmed through TMS9900 CRU interface
  - Serviced on interrupt or polled basis
  - Additional edge connector for interfacing to non-TM990 devices
- TI TMS9914-based interface board
  - Designed to illustrate the interfacing of the TMS9914 to a CPU and to the IEEE-488 bus
  - Interfaces directly to the components described above

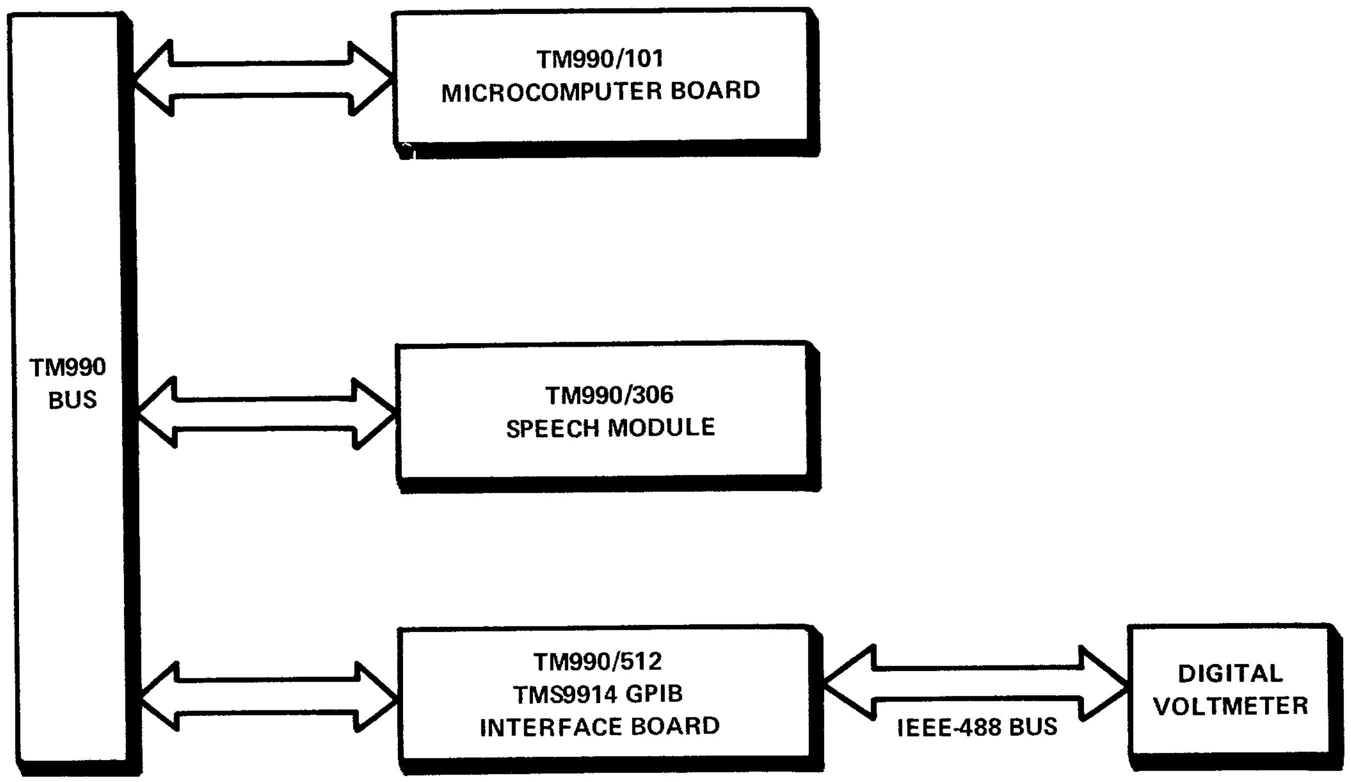

TM990 BUS
TM990/101 MICROCOMPUTER BOARD
TM990/306 SPEECH MODULE
TM990/512 TMS9914 GPIB INTERFACE BOARD
IEEE-488 BUS
DIGITAL VOLTMETER

FIGURE D-1 – APPLICATIONS HARDWARE BLOCK DIAGRAM

PROGRAMMER'S MODEL OF THE TM990/306 SPEECH BOARD

- 14-bit address selects word to speak
  - Bits 16 to 29 on the CRU interface
- 1 bit to enable EPROM speech data
  - Set to 1 during initialization
- 1 bit to start/stop speaking
  - 0 starts/enables speech
  - 1 stops/disables speech
- 1 bit to indicate the busy status of speech board
  - 0 means speech board busy
  - 1 means speech board not busy

METER DATA OUTPUT CHARACTERISTICS

- Data is output as a string of ASCII characters.
  - Format of the string is +D.DDDE+D,R OD)A
- R indicates the range setting of the meter.
  '1' = DC volts
  '2' = AC volts
  '3' = DC amps
  '4' = AC amps
  '5' = ohms
- EOI message is sent with LINE FEED character.

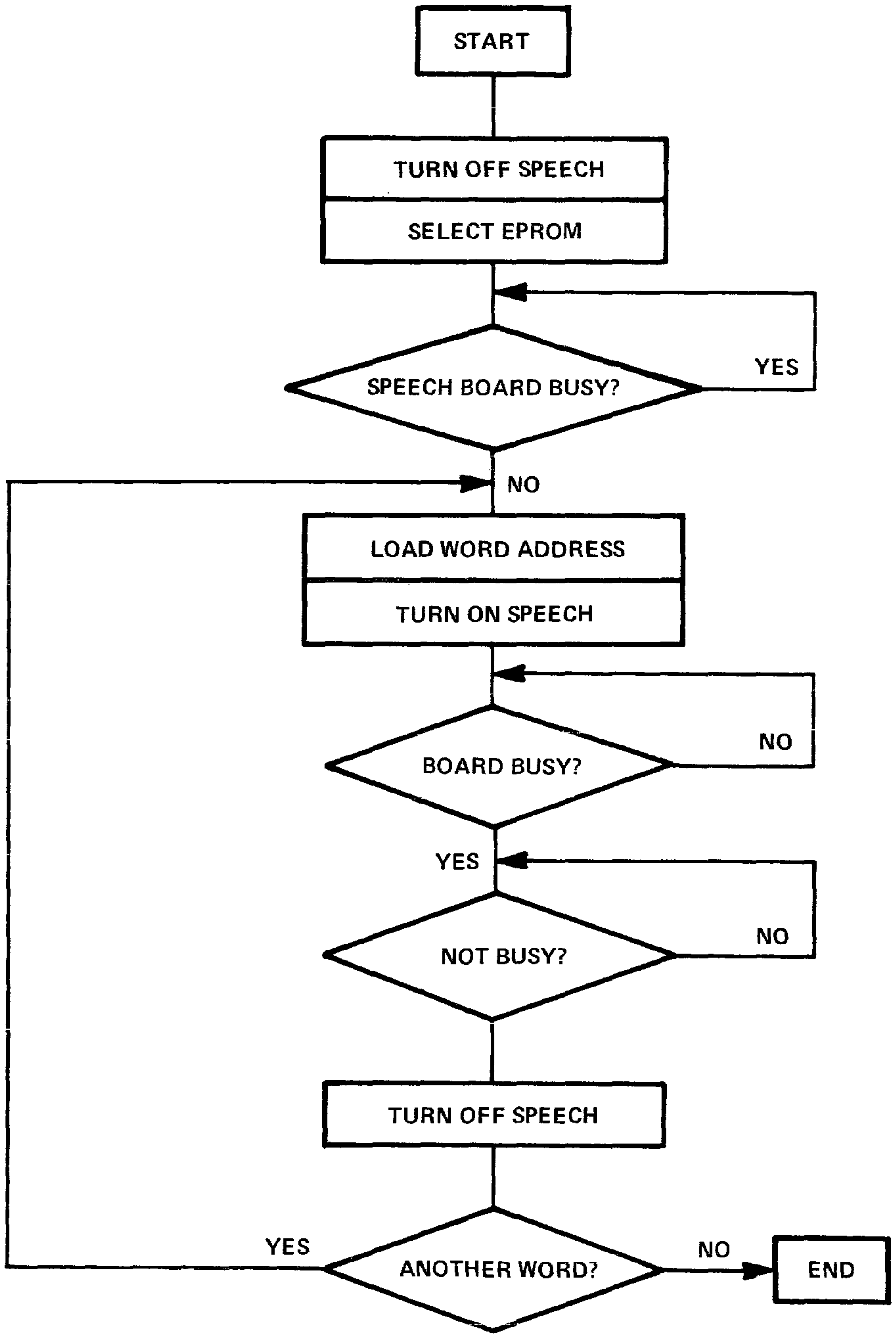


FIGURE D-2 – TYPICAL SPEECH BOARD USE

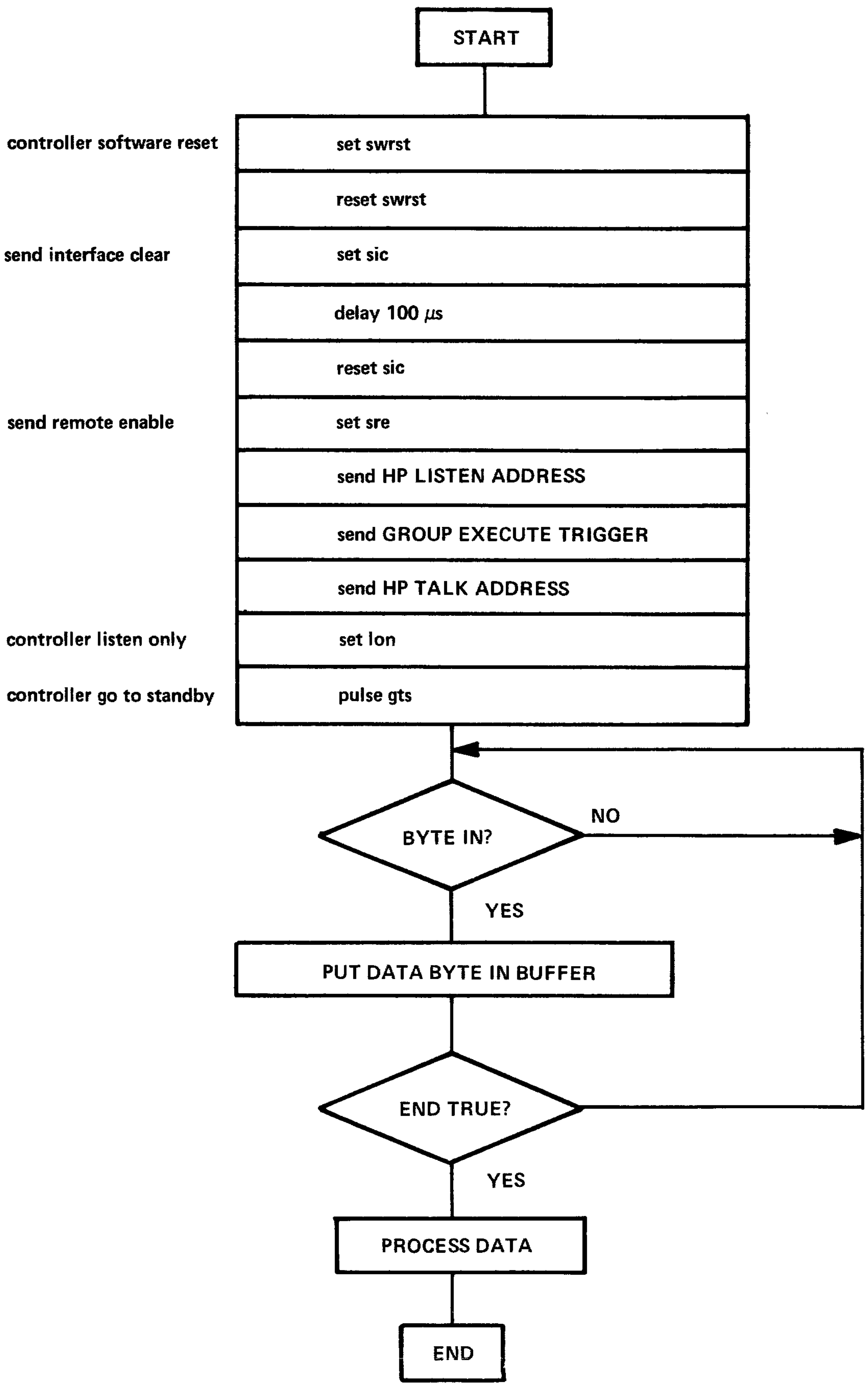


FIGURE D-3 – TYPICAL SOFTWARE TO CONTROL DIGITAL VOLTMETER

```
        IDT     'DEM3438A'
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*                                                             *
*       GPIB DEMONSTRATION      THIS VERSION FOR HP 3438A     *
*                                                             *
*       "TALKING CONTROLLER"    REVISION 4/14-NOV-80/PNK      *
*                                                             *
*             TMS9914 BASE ADDRESS:     >5540                 *
*             HP 3438A DEVICE ADDRESS: >17                    *
*             TM990/306 CRU ADDRESS:    >1FE0                 *
*                                                             *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*
*       TMS 9914 REGISTER EQUATES
*
BASE    EQU     >5540           BASE ADDRESS OF TMS 9914
IMASK0  EQU     BASE+0          INTERRUPT MASK 0
ISTAT0  EQU     BASE+0          INTERRUPT STATUS 0
IMASK1  EQU     BASE+2          INTERRUPT MASK 1
ISTAT1  EQU     BASE+2          INTERRUPT STATUS 1
ADSTAT  EQU     BASE+4          ADDRESS STATUS REGISTER
BUSTAT  EQU     BASE+6          BUS STATUS REGISTER
AUXCMD  EQU     BASE+6          AUXILARY COMMAND REGISTER
ADRSWI  EQU     BASE+8          ADDRESS SWITCH "REGISTER"
ADDRES  EQU     BASE+8          ADDRESS REGISTER
SERPOL  EQU     BASE+10         SERIAL POLL REGISTER
CMDPAS  EQU     BASE+12         COMMAND PASS THROUGH REGISTER
PARPOL  EQU     BASE+12         PARALLEL POLL REGISTER
DATIN   EQU     BASE+14         DATA FROM BUS REGISTER
DATOUT  EQU     BASE+14         DATA TO BUS REGISTER
*
*       TMS 9914 INTERRUPT AND POLLING MASKS
*
INT0    EQU     >8000           INTERRUPT GROUP 0
INT1    EQU     >4000           INTERRUPT GROUP 1
BI      EQU     >2000           BYTE READY FOR INPUT
BO      EQU     >1000           BYTE READY FOR OUTPUT
ENDB    EQU     >0800           BYTE IS LAST ONE
SRQ     EQU     >0200           SERVICE REQUEST
MASK0   EQU     BI+BO+ENDB      ENABLE INTERRUPTS GIVEN
```

```
*
*         SELECTED TMS 9914 AUXILARY COMMANDS
*
SWRST     EQU       >8000               SET SOFTWARE RESET
SWRSTC    EQU       >0000               CLEAR SOFTWARE RESET
FEOI      EQU       >0800               FORCE END OR IDENTIFY
LON       EQU       >8900               SET LISTEN ONLY
LONC      EQU       >0900               CLEAR LISTEN ONLY
TON       EQU       >8A00               SET TALK ONLY
TONC      EQU       >0A00               CLEAR TALK ONLY
GTS       EQU       >0B00               GO TO STANDBY
TCA       EQU       >0C00               TAKE CONTROL ASYNCHRONOUSLY
SIC       EQU       >8F00               SET SEND INTERFACE CLEAR
SICC      EQU       >0F00               CLEAR SEND INTERFACE CLEAR
SRE       EQU       >9000               SET SEND REMOTE ENABLE
SREC      EQU       >1000               CLEAR SEND REMOTE ENABLE
*
*         SELECTED MULTILINE I/F MESSAGES
*
GET       EQU       >0800               GROUP EXECUTE TRIGGER (ADDRESSED)
LLO       EQU       >1100               LOCAL LOCKOUT (UNIVERSAL)
SDC       EQU       >0400               DEVICE CLEAR (ADDRESSED)
DCL       EQU       >1400               DEVICE CLEAR (UNIVERSAL)
SPE       EQU       >1800               SERIAL POLL ENABLE (UNIVERSAL)
SPD       EQU       >1900               SERIAL POLL DISABLE (UNIVERSAL)
UNL       EQU       >3F00               UNLISTEN COMMAND (UNIVERSAL)
UNT       EQU       >5F00               UNTALK COMMAND (UNIVERSAL)
*
*         HP3455A DVM ADDRESSES
*
HPLA      EQU       >3700               LISTEN ADDRESS
HPTA      EQU       >5700               TALK ADDRESS
*
          AORG      >0000               STARTING ADDRESS
*
*         INTERRUPT VECTOR(S)
*
RESET     DATA      MAINWS              INITIAL WP
          DATA      START               INITIAL PC
*
```

```
        AORG     >0100                 PROGRAM ADDRESS
*
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*                                                                 *
*       GPIB SUBROUTINE DEFINITIONS                               *
*                                                                 *
*             BYTOUT - SENDS BYTE IN R0 OVER GPIB                 *
*             BYTIN  - RECEIVES BYTE FROM GPIB INTO R0            *
*             STROUT - SENDS STRING POINTED TO BY R0 OVER GPIB    *
*                      SENDING EOI WITH LAST BYTE (INDICATED BY   *
*                      FOLLOWING BYTE WITH BYTE = 00)             *
*             STRIN  - RECEIVES STRING FROM GPIB INTO BUFFER      *
*                      POINTED TO BY R0                           *
*             DOAUX  - PERFORMS 9914 AUX CMD SPECIFIED IN BYTE    *
*                      AFTER SUBROUTINE CALL                      *
*             DELAY  - DELAYS NUMBER OF MS INDICATED BY THE       *
*                      CONTENTS OF R0                             *
*                                                                 *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*
BYTOUT  MOV      @ISTAT0,R1            CHECK IF BO FLAG SET
        ANDI     R1,BO
        JEQ      BYTOUT                WAIT UNTIL IT IS
        MOV      R0,@DATOUT            SEND BYTE IN R0 OVER GPIB
        RT
*
BYTIN   MOV      @ISTAT0,R0            CHECK IF BI FLAG SET
        MOV      R0,R1                 COPY TO R1
        ANDI     R0,BI
        JEQ      BYTIN                 WAIT UNTIL IT IS
        MOV      @DATIN,R0             PUT BYTE FROM GPIB INTO R0
        RT
*
STROUT  MOV      R11,R10               SAVE RETURN ADDRESS
        MOV      R0,R2                 SAVE POINTER
        MOVB     *R2+,R0               GET FIRST BYTE TO SEND
STROU1  MOVB     *R2+,R4               GET NEXT BYTE TO SEND
        JNE      STROU2                SKIP IF NEXT IS NOT STOP FLAG
        BL       @DOAUX                PERFORM AUXILARY COMMAND
        DATA     FEOI                  FORCE END WITH THIS BYTE
STROU2  BL       @BYTOUT               ELSE SEND THE BYTE
        MOVB     R4,R0                 MOVE NEXT BYTE INTO R0
        JNE      STROU1                KEEP SENDING IF NOT STOP FLAG
        B        *R10
*
STRIN   MOV      R11,R10               SAVE RETURN ADDRESS
        MOV      R0,R2                 COPY BUFFER ADDRESS
STRIN1  BL       @BYTIN                GET BYTE FROM GPIB
        MOVB     R0,*R2+               COPY BYTE INTO BUFFER
        ANDI     R1,ENDB               CHECK IF LAST BYTE
        JEQ      STRIN1                KEEP GETTING BYTES IF NOT
        SB       *R2,*R2               CLEAR BYTE AFTER LAST ONE
        B        *R10
```

```
*
DOAUX    MOV      *R11,@AUXCMD      SEND AUX CMD IN WORD AFTER CALL
         INCT     R11               BUMP RETURN ADDRESS
         RT
*
DELAY    LI       R1,150            DELAY NUMBER OF MS IN R0
DEL01    DEC      R1                DECREMENT MS TIMER
         JNE      DEL01             COUNT DOWN 1 MS
         DEC      R0                DECREMENT MAIN TIMER
         JGT      DELAY             LOOP UNTIL TIME IS UP
         RT
*
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*                                                                 *
*        TALKING SUBROUTINE DEFINITIONS (TM990/306 BOARD)         *
*                                                                 *
*             TLKSET - INITIALIZES TALKING ROUTINE WORKSPACE      *
*             TLKWRD - SPEAKS WORD AT SPEECH ADDRESS INDICATED    *
*                      BY (R0) IF (R0) >= 0 OR DELAYS NUMBER OF   *
*                      MS INDICATED BY -(R0) IF (R0) < 0          *
*             TLKSEN - SPEAKS SENTENCE POINTED TO BY R1, HALTING  *
*                      WHEN WORD TO SPEAK IS >FFFF                *
*             DIGCVT - CONVERTS ASCII CODE FOR DECIMAL DIGITS     *
*                      INTO SPEECH ADDRESSES                      *
*             REDTLK - SPEAKS VALUE OF HP3455A FORMATTED OUTPUT   *
*                      DATA POINTED TO BY R2                      *
*                                                                 *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*
TLKCRU   EQU      >1FE0             TM990/306 BOARD HI CRU BASE ADDRESS
TLKBSY   EQU      -15               BUSY STATUS BIT
TLKNTK   EQU      14                TALK/NOTALK BIT
TLKEPM   EQU      15                TALK EPROM ENABLE BIT
*
TLKSET   LI       R12,TLKCRU        INITIALIZE TALK CRU
         SBO      TLKEPM            SET EPROM ENABLE BIT
         SBO      TLKNTK            AND TURN OFF TALKING
         RTWP
*
TLKGO    DATA     TALKWS            BLWP XFER VECTOR
         DATA     TLKSET
*
TLKWRD   MOV      *R13,R0           GET WORD FROM OLD WS
         ABS      R0                CHECK IF WORD OR DELAY
         JLT      TLKW03            SKIP IF DELAY
         LDCR     R0,14             SELECT WORD TO SPEAK
         SBZ      TLKNTK            TURN ON TALKING
TLKW01   TB       TLKBSY            WAIT FOR 306 TO START
         JNE      TLKW01
TLKW02   TB       TLKBSY            WAIT FOR 306 TO STOP
         JEQ      TLKW02
         SBO      TLKNTK            TURN OFF TALKING
         RTWP
TLKW03   BL       @DELAY            PERFORM DELAY
         RTWP
```

```
*
TALK     DATA     TALKWS             BLWP XFER VECTOR
         DATA     TLKWRD
*
TLKSEN   MOV      *R1+,R0            COPY WORD TO R0
         CI       R0,-1              QUIT IF STOP FLAG
         JEQ      TLKS01
         BLWP     @TALK              SAY ONE WORD OR DELAY
         JMP      TLKSEN             GET NEXT WORD
TLKS01   RT
*
DIGCVT   CI       R1,>30             CHECK IF ASCII DIGIT
         JLT      DIGC01
         CI       R1,>3A
         JLT      DIGC02
DIGC01   LI       R0,-1              RETURN DELAY IF NOT
         RT
DIGC02   ANDI     R1,>000F           ISOLATE BCD VALUE
         SLA      R1,1               MULTIPLY BY 2
         MOV      @DIGTLK(R1),R0     GET SPEECH VALUE
         RT
DIGTLK   DATA     >2E94              ZERO
         DATA     >06FC              ONE
         DATA     >0900              TWO
         DATA     >249A              THREE
         DATA     >0372              FOUR
         DATA     >03C0              FIVE
         DATA     >355E              SIX
         DATA     >23AE              SEVEN
         DATA     >02AC              EIGHT
         DATA     >31C2              NINE
*
REDTLK   MOV      R11,R10            SAVE RETURN ADDRESS
REDT01   MOVB     *R2+,R1            GET BYTE TO SPEAK
         JEQ      REDT07             QUIT IF STOP BYTE
         SRL      R1,8               ISOLATE BYTE IN LS
         CI       R1,>2C             CHECK FOR ","
         JEQ      REDT06             SKIP TO END IF SO
         CI       R1,>2D             COMPARE TO "-"
         JNE      REDT02             SKIP IF NOT
         LI       R0,>0F02           INDICATE "MINUS"
         JMP      REDT05             SKIP AHEAD
REDT02   CI       R1,>2E             COMPARE TO "."
         JNE      REDT03             SKIP IF NOT
         LI       R0,>2146           INDICATE "POINT"
         JMP      REDT05             SKIP AHEAD
REDT03   CI       R1,>45             COMPARE TO "E"
         JNE      REDT04             SKIP AHEAD IF NOT
         LI       R1,ETOTHE          SAY "10 TO THE"
         BL       @TLKSEN
         JMP      REDT01
REDT04   BL       @DIGCVT            CONVERT DIGIT
```

```
REDT05  BLWP    @TALK           SAY THE WORD
        JMP     REDT01          GET NEXT BYTE
REDT06  MOVB    *R2+,R1         GET FORMAT BYTE
        ANDI    R1,>0F00        ISOLATE FORMAT CODE
        SRL     R1,7            SHIFT AND CREATE INDEX
        DECT    R1              MAKE INDEX 0 -> 4
        MOV     @UNITS(R1),R1   GET UNITS MESSAGE PTR
        BL      @TLKSEN         AND SAY IT
REDT07  B       *R10
ETOTHE  DATA    -200            DELAY 200MS
        DATA    >2D06           "TEN"
        DATA    >0900           "TO"
        DATA    >087A           "THE"
        DATA    -1
UNITS   DATA    DCV             DC VOLTS
        DATA    ACV             AC VOLTS
        DATA    DCI             DC AMPS
        DATA    ACI             AC AMPS
        DATA    OHMS            OHMS
DCV     EQU     $
        DATA    -200
        DATA    >36FA           "VOLTS"
        DATA    >1740           "D"
        DATA    >15A0           "C"
        DATA    -1
ACV     EQU     $
        DATA    -200
        DATA    >36FA           "VOLTS"
        DATA    >09CC           "A"
        DATA    >15A0           "C"
        DATA    -1
DCI     EQU     $
        DATA    -200
        DATA    >142E           "AMPS"
        DATA    >1740           "D"
        DATA    >15A0           "C"
        DATA    -1
ACI     EQU     $
        DATA    -200
        DATA    >142E           "AMPS"
        DATA    >09CC           "A"
        DATA    >15A0           "C"
        DATA    -1
OHMS    EQU     $
        DATA    -200
        DATA    >2078           OHMS
        DATA    -1
*
```

```
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*                                                               *
*       MAIN PROGRAM                                            *
*                                                               *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*
START   BLWP    @TLKGO          INITIALIZE 306 BOARD
        LI      R1,SPINTR       GIVEN INTIAL MESSAGE
        BL      @TLKSEN
        BL      @DOAUX          SEND SWRST
        DATA    SWRST
        LI      R0,MASK0        SET UP INTERRUPTS
        MOV     R0,@IMASK0
        MOV     @ADRSWI,R0      READ ADDRESS SWITCH
        ANDI    R0,>1F00        TRIM TO 5 BITS
        MOV     R0,@ADDRES      INITIALIZE 9914 ADDRESS
        BL      @DOAUX          SEND IFC
        DATA    SIC
        DATA    SWRSTC
        BL      @DOAUX          RESET SWRST
        LI      R0,1            DELAY 1 MS
        BL      @DELAY
        BL      @DOAUX          CLEAR IFC/ENTER CACS
        DATA    SICC
        BL      @DOAUX          SEND REMOTE ENABLE
        DATA    SRE
        LI      R0,HPLA         SEND HP3438A LISTEN ADDRESS
        BL      @BYTOUT
        LI      R0,GET          SEND TRIGGER COMMAND
        BL      @BYTOUT
        LI      R0,UNL          SEND UNLISTEN COMMAND
        BL      @BYTOUT
        BL      @DOAUX          TURN ON 9914 LISTENER
        DATA    LON
        LI      R0,HPTA         SEND HP3438A TALK ADDRESS
        BL      @BYTOUT
        BL      @DOAUX          GO TO STANDBY/ATN = 0
        DATA    GTS
        LI      R0,DATBUF       GET READING FROM GPIB
        BL      @STRIN
        BL      @DOAUX          TURN OFF LISTENER
        DATA    LONC
        LI      R2,DATBUF       POINT TO READING
        BL      @REDTLK         AND SAY IT
        BL      @DOAUX          CLEAR THE INTERFACE
        DATA    SIC
        LI      R0,1            DELAY 1 MS
        BL      @DELAY
        BL      @DOAUX          RELEASE INTERFACE CLEAR
        DATA    SICC
        BL      @DOAUX          DISABLE REMOTE OPERATION
        DATA    SREC
        BL      @DOAUX          RELEASE CONTROL
        DATA    GTS
        LI      R1,SPPMPT       GIVE NEW START MESSAGE
START2  BL      @TLKSEN
        LI      R0,30000        WAIT 30 SECONDS
        BL      @DELAY
        LI      R1,SPPMPT       POINT TO PROMPT
        JMP     START2          AND REPEAT
*
*       SPEECH LISTS
*
SPPMPT  DATA    >2188           PRESS
        DATA    >087A           THE
        DATA    >229C           RED
        DATA    >082C           SWITCH
        DATA    >0900           TO
        DATA    >0EA0           MEASURE
        DATA    -1
*
SPINTR  DATA    >33D6           READY
        DATA    >0900           TO
        DATA    >0D5C           GO
        DATA    -500
        DATA    -1
*
        DORG    >FF00
*
MAINWS  BSS     32              TEMPORARY LOCATIONS
TALKWS  BSS     32
DATBUF  BSS     20              DATA BUFFER
*
        END     START
```

TEXAS INSTRUMENTS

Printed in U.S.A.
June 1989

SPOD001