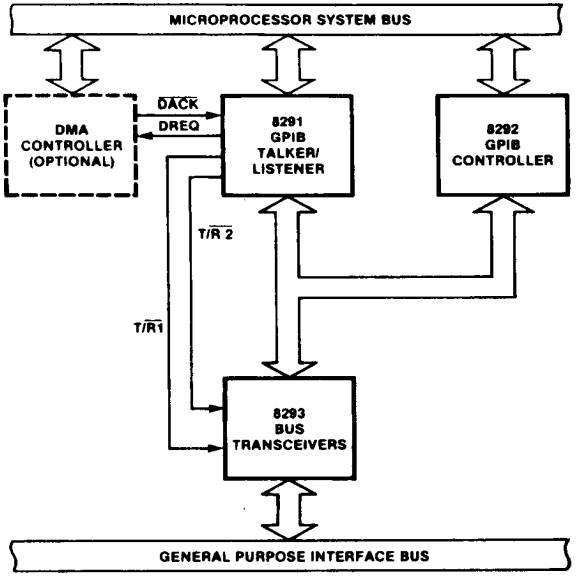
8292
GPIB CONTROLLER

- **Complete IEEE Standard 488 Controller Function**
- **Interface Clear (IFC) Sending Capability Allows Seizure of Bus Control and/or Initialization of the Bus**
- **Responds to Service Requests (SRQ)**
- **Sends Remote Enable (REN), Allowing Instruments to Switch to Remote Control**
- **Complete Implementation of Transfer Control Protocol**
- **Synchronous Control Seizure Prevents the Destruction of Any Data Transmission in Progress**
- **Connects with the 8291 to Form a Complete IEEE Standard 488 Interface Talker/Listener/Controller**

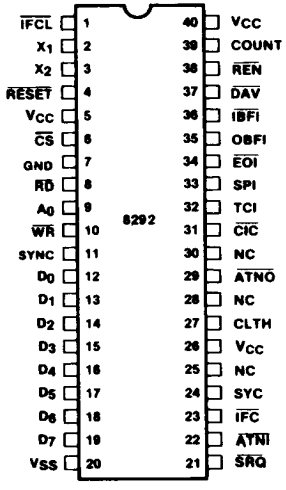
The 8292 GPIB Controller is a microprocessor-controlled chip designed to function with the 8291 GPIB Talker/Listener to implement the full IEEE Standard 488 controller function, including transfer control protocol. The 8292 is a preprogrammed Intel® 8041A.

3



205250-1

Figure 1. 8291, 8292 Block Diagram



205250-2

Figure 2. Pin Configuration

December 1991
Order Number: 205250-002

Table 1. Pin Description

Symbol	Pin Number	Type	Name and Function
$\overline{IFCL}$	1	I	**IFC RECEIVED (LATCHED):** The 8292 monitors the IFC Line (when not system controller) through this pin.
X_1, X_2	2, 3	I	**CRYSTAL INPUTS:** Inputs for a crystal, LC or an external timing signal to determine the internal oscillator frequency.
$\overline{RESET}$	4	I	**RESET:** Used to initialize the chip to a known state during power on.
$\overline{CS}$	6	I	**CHIP SELECT INPUT:** Used to select the 8292 from other devices on the common data bus.
$\overline{RD}$	8	I	**READ ENABLE:** Allows the master CPU to read from the 8292.
A_0	9	I	**ADDRESS LINE:** Used to select between the data bus and the status register during read operations and to distinguish between data and commands written into the 8292 during write operations.
$\overline{WR}$	10	I	**WRITE ENABLE:** Allows the master CPU to write to the 8292.
SYNC	11	O	**SYNC:** 8041A instruction cycle synchronization signal; it is an output clock with a frequency of XTAL ÷ 15.
D_0-D_7	12–19	I/O	**DATA:** 8 bidirectional lines used for communication between the central processor and the 8292's data bus buffers and status register.
V_{SS}	7, 20	P.S.	**GROUND:** Circuit ground potential.
$\overline{SRQ}$	21	I	**SERVICE REQUEST:** One of the IEEE control lines. Sampled by the 8292 when it is controller in charge. If true, SPI interrupt to the master will be generated.
$\overline{ATNI}$	22	I	**ATTENTION IN:** Used by the 8292 to monitor the GPIBATN control line. If is used during the transfer control procedure.
$\overline{IFC}$	23	I/O	**INTERFACE CLEAR:** One of the GPIB management lines, as defined by IEEE Std. 488-1978, places all devices in a known quiescent state.
SYC	24	I	**SYSTEM CONTROLLER:** Monitors the system controller switch.
CLTH	27	O	**CLEAR LATCH:** Used to clear the $\overline{IFCR}$ latch after being recognized by the 8292. Usually low (except after hardware Reset), it will be pulsed high when $\overline{IFCR}$ is recognized by the 8292.
$\overline{ATNO}$	29	O	**ATTENTION OUT:** Controls the ATN control line of the bus through external logic for tcs and tca procedures. (ATN is a GPIB control line, as defined by IEEE Std. 488-1978.)
V_{CC}	5, 26, 40	P.S.	**VOLTAGE:** +5V supply input ±10%.
COUNT	39	I	**EVENT COUNT:** When enabled by the proper command the internal counter will count external events through this pin. High to low transition will increment the internal counter by one. The pin is sampled once per three internal instruction cycles (7.5 μsec sample period when using 5 MHz XTAL). It can be used for byte counting when connected to NDAC, or for block counting when connected to the EOI.
$\overline{REN}$	38	O	**REMOTE ENABLE:** The Remote Enable bus signal selects remote or local control of the device on the bus. A GPIB bus signal selects remote or local control of the device on the bus. A GPIB bus management line, as defined by IEEE Std. 488-1978.

Table 1. Pin Description (Continued)

Symbol	Pin No.	Type	Name and Function
$\overline{DAV}$	37	I/O	**DATA VALID:** Used during parallel poll to force the 8291 to accept the parallel poll status bit. It is also used during the tcs procedure.
$\overline{IBFI}$	36	O	**INPUT BUFFER NOT FULL:** Used to interrupt the central processor while the input buffer of the 8292 is empty. This feature is enabled and disabled by the interrupt mask register.
OBFI	36	O	**OUTPUT BUFFER FULL:** Used as an interrupt to the central processor while the output buffer of the 8292 is full. The feature can be enabled and disabled by the interrupt mask register.
$\overline{EO12}$	34	I/O	**END OR IDENTIFY:** One of the GPIB management lines, as defined by IEEE Std. 488-1978. Used with ATN as Identify Message during parallel poll.
SPI	33	O	**SPECIAL INTERRUPT:** Used as an interrupt on events not initiated by the central processor.
TCI	32	O	**TASK COMPLETE INTERRUPT:** Interrupt to the control processor used to indicate that the task requested was completed by the 8292 and the information requested is ready in the data bus buffer.
CIC	31	O	**CONTROLLER IN CHARGE:** Controls the S/R input of the SRQ bus transceiver. It can also be used to indicate that the 8292 is in charge of the GPIB bus.

3

FUNCTIONAL DESCRIPTION

The 8292 is an Intel 8041A which has been programmed as a GPIB Controller Interface element. It is used with the 8291 GPIB Talker/Listener and two 8293 GPIB Transceivers to form a complete IEEE-488 Bus Interface for a microprocessor. The electrical interface is performed by the transceivers, data transfer is done by the talker/listener, and control of the bus is done by the 8292. Figure 3 is a typical controller interface using Intel's GPIB peripherals.

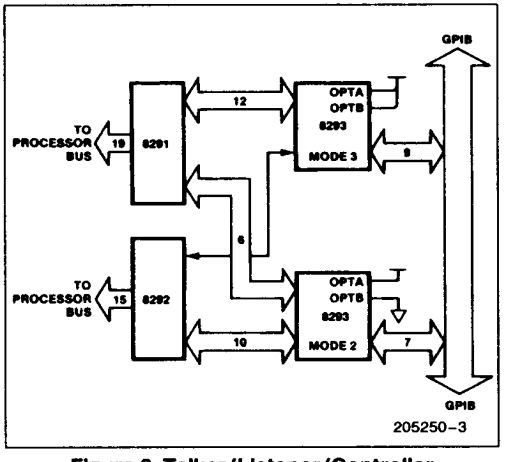


Figure 3. Talker/Listener/Controller Configuration

The internal RAM in the 8041A is used as a special purpose register bank for the 8292. Most of these registers (except for the interrupt flag) can be accessed through commands to the 8292. Table 2 identifies the registers used by the 8292 and how they are accessed.

Interrupt Status Register

SYC	ERR	SRQ	EV	X	IFCR	IBF	OBF
D_7							D_0

The 8292 can be configured to interrupt the microprocessor on one of several conditions. Upon receipt of the interrupt the microprocessor must read the 8292 interrupt status register to determine which event caused the interrupt, and then the appropriate subroutine can be performed. The interrupt status register is read with A_0 high. With the exception of OBF and IBF, these interrupts are enabled or disabled by the SPI interrupt mask. OBF and IBF have their own bits in the interrupt mask (OBFI and $\overline{IBFI}$).

OBF Output Buffer Full. A byte is waiting to be read by the microprocessor. This flag is cleared when the output data bus buffer is read.

IBF Input Buffer Full. The byte previously written by the microprocessor has not been read yet by the 8292. If another byte is written to the 8292 before this flag clears, data will be lost. IBF is cleared when the 8292 reads the data byte.

IFCR Interface Clear Received. The GPIB system controller has set IFC. The 8292 has become idle and is no longer is charge of the bus. The flag is cleared when the IACK command is issued.

EV Event Counter Interrupt. The requested number of blocks of data byte has been transferred. The EV interrupt flag is cleared by the IACK command.

SRQ Service Request. Notified the 8292 that a service request (SRQ) message has been received. It is cleared by the IACK command.

ERR Error occurred. The type of error can be determined by reading the error status register. This interrupt flag is cleared by the IACK command.

SYC System Controller Switch Change. Notifies the processor that the state of the system controller switch has changed. The actual state is contained in the GPIB Status Register. This flag is cleared by the IACK command.

Interrupt Mask Register

1	SPI	TCI	SYC	OBFI	$\overline{IBFI}$	0	SRQ
D_7							D_0

The Interrupt Mask Register is used to enable features and to mask the SPI and TCI interrupts. The flags in the Interrupt Status Register will be active even when masked out. The Interrupt Mask Register is written when A_0 is low and reset by the RINM command. When the register is read, D_1 and D_7 are undefined. An interrupt is enabled by setting the corresponding register bit.

SRQ Enable interrupts on SRQ received.

$\overline{IBFI}$ Enable interrupts on input buffer empty.

OBFI Enable interrupts on output buffer full.

Table 2. 8292 Registers

READ FROM 8292

	D_7						D_0	A_0
INTERRUPT STATUS	SYC	ERR	SRQ	EV	X	IFCR	IBF	OBF	1
ERROR FLAG	X	X	USER	X	X	$TOUT_3$	$TOUT_2$	$TOUT_1$	0*
CONTROLLER STATUS	CSBS	CA	X	X	SYCS	IFC	REN	SRQ	0*
GPIB (BUS) STATUS	REN	DAV	EOI	X	SYC	IFC	ANTI	SRQ	0*
EVENT COUNTER STATUS	D	D	D	D	D	D	D	D	0*
TIME OUT STATUS	D	D	D	D	D	D	D	D	0*

WRITE TO 8292

	D_7							D_0	A_0
INTERRUPT MASK	1	SPI	TCI	SYC	OBFI	$\overline{IBFI}$	0	SRQ	0
ERROR MASK	0	0	USER	0	0	$TOUT_3$	$TOUT_2$	$TOUT_1$	0
COMMAND FIELD	1	1	1	OP	C	C	C	C	1
EVENT COUNTER	D	D	D	D	D	D	D	D	0*
TIME OUT	D	D	D	D	D	D	D	D	0*

NOTE: These registers are accessed by a special utility command, see page 7.

SYC Enable interrupts on a change in the system controller switch.

TCI Enable interrupts on the task completed.

SPI Enable interrupts on special events.

NOTE:
The event counter is enabled by the GSEC command, the error interrupt is enabled by the error mask register, and IFC cannot be masked (it will always cause an interrupt).

Controller Status Register

CSBS	CA	X	X	SYCS	IFC	REN	SRQ
D_7							D_0

The Controller Status Register is used to determine the status of the controller function. This register is accessed by the RCST command.

SRQ Service Request line active (CSRS).

REN Sending Remote Enable.

IFC Sending or receiving interface clear.

SYCS System Controller Switch Status (SACS).

CA Controller Active (CACS + CAWS + CSWS).

CSBS Controller Stand-by State (CSBS, CA) = (0,0)—Controller Idle.

GPIB Bus Status Register

REN	DAV	EOI	X	SYC	IFC	ATNI	SRQ
D_7							D_0

This register contains GPIB bus status information. It can be used by the microprocessor to monitor and manage the bus. The GPIB Bus Register can be read using the RBST command.

Each of these status bits reflect the current status of the corresponding pin on the 8292.

SRQ Service Request

ATNI Attention In

IFC Interface Clear

SYC System Controller Switch

EOI End or Identify

DAV Data Valid

REN Remote Enable

Event Counter Register

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0

The Event Counter Register contains the initial value for the event counter. The counter can count pulses on pin 39 of the 8292 (COUNT). It can be connected to EOI or NDAC to count blocks or bytes respectively during standby state. A count of zero equals 256. This register cannot be read, and is written using the WEVC command.

Event Counter Status Register

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0

This register contains the current value in the event counter. The event counter counts back from the initial value stored in the Event Counter Register to zero and then generates an Event Counter Interrupt. This register cannot be written and can be read using a REVC command.

Time Out Register

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0

The Time Out Register is used to store the time used for the time out error function. See the individual timeouts (TOUT1, 2, 3) to determine the units of this counter. This Time Out Register cannot be read, and it is written with the WTOUT command.

Time Out Status Register

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0

This register contains the current value in the time out counter. The time out counter decrements from the original value stored in the Time Out Register. When zero is reached, the appropriate error interrupt is generated. If the register is read while none of the time out functions are active, the register will contain the last value reached the last time a function was active. The Time Out Status Register cannot be written, and it is read with RTOUT command.

Error Flag Register

X	X	USER	X	X	$TOUT_3$	$TOUT_2$	$TOUT_1$

Four errors are flagged by the 8292 with a bit in the Error Flag Register. Each of these errors can be masked by the Error Mask Register. The Error Flag Register cannot be written, and it is read by the IACK command when the error flag in the Interrupt Status Register is set.

TOUT1 Time Out Error 1 occurs when the current controller has not stopped sending ATN after receiving the TCT message for the time period specified by the Time Out Register. Each count in the Time Out Register is at least 1800 t_{CY}. After flagging the error, the 8292 will remain in a loop trying to take control until the current controller stops send-

ing ATN or a new command is written by the microprocessor. If a new command is written, the 8292 will return to the loop after executing it.

TOUT2 Time Out Error 2 occurs when the transmission between the addressed talker and listener has not started for the time period specified by the Time Out Register. Each count in the Time Out Register is at least 45 t_{CY}. This feature is only enabled when the controller is in the CSBS state.

TOUT3 Time Out Error 3 occurs when the handshake signals are stuck and the 8292 is not succeeding in taking control synchronously for the time period specified by the Time Out Register. Each count in the Time Out Register is at least 1800 t_{CY}. The 8292 will continue checking $\overline{ATNI}$ until it becomes true or a new commnand is received. After performing the new command, the 8292 will return to the $\overline{ATNI}$ checking loop.

USER User error occurs when request to assert IFC or REN was received and the 8292 was not the system controller.

Error Mask Register

0	0	USER	0	0	$TOUT_3$	$TOUT_2$	$TOUT_1$

D_7 D_0

The Error Mask Register is used to mask the interrupt from a particular type of error. Each type of error interrupt is enabled by setting the corresponding bit in the Error Mask Register. This register can be read with the RERM command and written with A_0 low.

Command Register

1	1	1	OP	C	C	C	C

D_7 D_0

Commands are performed by the 8292 whenever a byte is written with A_0 high. There are two categories of commands distinguished by the OP bit (bit 4). The first category is the operation command (OP = 1). These commands initiate some action on the interface bus. The second category is the utility command (OP = 0). These commands are used to aid the communication between the processor and the 8292.

OPERATION COMMANDS

Operation commands initiate some action on the GPIB interface bus. It is using these commands that the control functions such as polling, taking and passing control, and system controller functions are performed.

F0—SPCNI—Stop Counter Interrupts

This command disables the internal counter interrupt so that the 8292 will stop interrupting the master on event counter underflows. However, the counter will continue counting and its contents can still be used.

F1—GIDL—Go To Idle

This command is used during the transfer of control procedure while transferring control to another controller. The 8292 will respond to this command only if it is in the active state. $\overline{ATNO}$ will go high, and $\overline{CIC}$ will be high so that this 8292 will no longer be driving the ATN line on the GPIB interface bus. TCI will be set upon completion.

F2—RST—Reset

This command has the same effect as asserting the external reset on the 8292. For details, refer to the reset procedure described later.

F3—RSTI—Reset Interrupts

This command resets any pending interrupts and clears the error flags. The 8292 will not return to any loop it was in (such as from the time out interrupts).

F4—GSEC—Go To Standby, Enable Counting

The function causes $\overline{ATNO}$ to go high and the counter will be enabled. If the 8292 was not the active controller, this command will exit immediately. If the 8292 is the active controller, the counter will be loaded with the value stored in the Event Counter Register, and the internal interrupt will be enabled so that when the counter reaches zero, the SPI interrupt will be generated. SPI will be generated every 256 counts thereafter until the controller exits the standby state or the SPCNI command is written. An initial count of 256 (zero in the Event Counter Register) will be used if the WEVC command is not executed. If the data transmission does not start, a TOUT2 error will be generated.

F5—EXPP—Execute Parallel Poll

This command initiates a parallel poll by asserting EOI when ATN is already active. TCI will be set at the end of the command. The 8291 should be previously configured as a listener. Upon detection of DAV true, the 8291 enters ACDS and latches the parallel poll response (PPR) byte into its data in register. The master will be interrupted by the 8291 BI interrupt when the PPR byte is available. No interrupts except the $\overline{IBFI}$ will be generated by the 8292. The 8292 will respond to this command only when it is the active controller.

F6—GTSB—Go To Standby

If the 8292 is the active controller, $\overline{ATNO}$ will go high then TCI will be generated. If the data transmission does not start, a TOUT2 error will be generated.

F7—SLOC—Set Local Mode

If the 8292 is the system controller, then REN will be asserted false and TCI will be set true. If it is not the system controller, the User Error bit will be set in the Error Flag Register.

F8—SREM—Set Interface To Remote Control

This command will set REN true and TCI true if this 8292 is the system controller. If not, the User Error bit will be set in the Error Flag Register.

F9—ABORT—Abort All Operation, Clear Interface

This command will cause IFC to be asserted true for at least 100 μsec if this 8292 is the system controller. If it is in CIDS, it will take control over the bus (see the TCNTR command).

FA—TCNTR—Take Control

The transfer of control procedure is coordinated by the master with the 8291 and 8292. When the master receives a TCT message from the 8291, it should issue the TCNTR command to the 8292. The following events occur to take control:

1) The 8292 checks to see if it is in CIDS, and if not, it exits.
2) Then $\overline{ATNI}$ is checked until it becomes high. If the current controller does not release ATN for the time specified by the Time Out Register, then a TOUT1 error is generated. The 8292 will return to this loop after an error or any command except the RST and RSTI commands.
3) After the current controller releases ATN, the 8292 will assert $\overline{ATNO}$ and $\overline{CIC}$ low.
4) Finally, the TCI interrupt is generated to inform the master that it is in control of the bus.

FC—TCASY—Take Control Asynchronously

TCAS transfers the 8292 from CSBS to CACS independent of the handshake lines. If a bus hangup is detected (by an error flag), this command will force the 8292 to take control (asserting ATN) even if the AH function is not in ANRS (Acceptor Not Ready State). This command should be used very carefully since it may cause the loss of a data byte. Normally, control should be taken synchronously. After checking the controller function for being in the CSBS (else it will exit immediately), $\overline{ATNO}$ will go low, and a TCI interrupt will be generated.

FD—TCSY—Take Control Synchronously

There are two different procedures used to transfer the 8292 from CSBS to CACS depending on the state of the 8291 in the system. If the 8291 is in "continuous AH cycling" mode (Aux. Reg. A0 = A1 = 1), then the following procedures should be followed:

1) The master microprocessor stops the continuous AH cycling mode in the 8291;
2) The master reads the 8291 Interrupt Status 1 Register;
3) If the END bit is set, the master sends the TCSY command to the 8292;
4) If the END bit was not set, the master reads the 8291 Data In Register and then waits for another BI interrupt from the 8291. When it occurs, the master sends the 8292 the TCSY command.

If the 8291 is not in AH cycling mode, then the master just waits for a BI interrupt and then sends the TCSY command. After the TCSY command has been issued, the 8292 checks for $\overline{CSBS}$. If $\overline{CSBS}$, then it exits the routine. Otherwise, it then checks the DAV bit in the GPIB status. When DAV becomes false, the 8292 will wait for at least 1.5 μsec. (T10) and then $\overline{ATNO}$ will go low. If DAV does not go low, a TOUT3 error will be generated. If the 8292 successfully takes control, it sets TCI true.

FE—STCNI—Start Counter Interrupts

This command enables the internal counter interrupt. The counter is enabled by the GSEC command.

UTILITY COMMANDS

All these commands are either Read or Write to registers in the 8292. Note that writing to the Error Mask Register and the Interrupt Mask Register are done directly.

E1—WTOUT—Write To Time Out Register

The byte written to the data bus buffer (with $A_0 = 0$) following this command will determine the time used for the time out function. Since this function is implemented in software, this will not be an accurate time measurement. This feature is enable or disable by the Error Mask Register. No interrupts except for the $\overline{IBFI}$ will be generated upon completion.

E2—WEVC—Write To Event Counter

The byte written to the data bus buffer (with $A_0 = 0$) following this command will be loaded into the Event Counter Register and the Event Counter Status for byte counting of EOI counting. Only $\overline{IBFI}$ will indicate completion of this command.

E3—REVC—Read Event Counter Status

This command transfers the contents of the Event Counter into the data bus buffer. A TCI is generated when the data is available in the data bus buffer.

E4—RERF—Read Error Flag Register

This command transfers the contents of the Error Flag Register into the data bus buffer. A TCI is generated when the data is available.

E5—RINM—Read Interrupt Mask Register

This command transfers the contents of the Interrupt Mask Register into the data bus buffer. This register is available to the processor so that it does not need to store this information elsewhere. A TCI is generated when the data is available in the data bus buffer.

E6—RCST—Read Controller Status Register

This command transfers the contents of the Controller Status Register into the data bus buffer and a TCI interrupt is generated.

E7—RBST—Read GPIB Bus Status Register

This command transfers the contents of the GPIB Bus Status Register into the data bus buffer, and a TCI interrupt is generated when the data is available.

E9—RTOUT—Read Time Out Status Register

This command transfers the contents of the Time Out Status Register into the data bus buffer, and a TCI interrupt is generated when the data is available.

EA—RERM—Read Error Mask Register

This command transfers the contents of the Error Mask Register to the data bus buffer so that the processor does not need to store this information elsewhere. A TCI interrupt is generated when the data is available.

Interrupt Acknowledge

SYC	ERR	SRQ	EV	1	IFCR	1	1
D_7							D_0

Each named bit in an Interrupt Acknowledge (IACK) corresponds to a flag in the Interrupt Status Register. When the 8292 receives this command, it will clear the SPI and the corresponding bits in the Interrupt Status Register. If not all the bits were cleared, then the SPI will be set true again. If the error flag is not acknowledged by the IACK command, then the Error Flag Register will be transferred to the data bus buffer, and a TCI will be generated.

NOTE:
XXXX1X11 is an undefined operation or utility command, so no conflict exists between the IACK operation and utility commands.

SYSTEM OPERATION

8292 To Master Processor Interface

Communication between the 8292 and the Master Processor can be either interrupt based communication or based upon polling the interrupt status register in predetermined intervals.

Interrupt Based Communication

Four different interrupts are available from the 8292:

OBFI Output Buffer Full Interrupt

$\overline{IBFI}$ Input Buffer Not Full Interrupt

TCI Task Completed Interrupt

SPI Special Interrupt

Each of the interrupts is enabled or disabled by a bit in the interrupt mask register. Since OBFI and $\overline{IBFI}$ are directly connected to the OBF and IBF flags, the master can write a new command to the input data bus buffer as soon as the previous command has been read.

The TCI interrupt is useful when the master is sending commands to the 8292. The pending TCI will be cleared with each new command written to the 8292. Commands sent to the 8292 can be divided into two major groups:

1) Commands that require response back from the 8292 to the master, e.g., reading register.
2) Commands that initiate some action or enable features but do not require response back from the 8292, e.g., enable data bus buffer interrupts.

With the first group, the TCI interrupt will be used to indicate that the required response is ready in the data bus buffer and the master may continue and read it. With the second group, the interrupt will be used to indicate completion of the required task, so that the master may send new commands.

The SPI should be used when immediate information or special events is required (see the Interrupt Status Register).

"Polling Status" Based Communication

When interrupt based communication is not desired, all interrupts can be masked by the interrupt mask register. The communication with the 8292 is based upon sequential poll of the interrupt status register. By testing the OBF and IBF flags, the data bus buffer status is determined while special events are determined by testing the other bits.

Receiving IFC

The IFC pulse defined by the IEEE-488 standard is at least 100 μsec. In this time, all operation on the bus should be aborted. Most important, the current controller (the one that is in charge at that time) should stop sending ATN or EOI. Thus, IFC must externally gate $\overline{CIC}$ (controller in charge) and $\overline{ATNO}$ to ensure that this occurs.

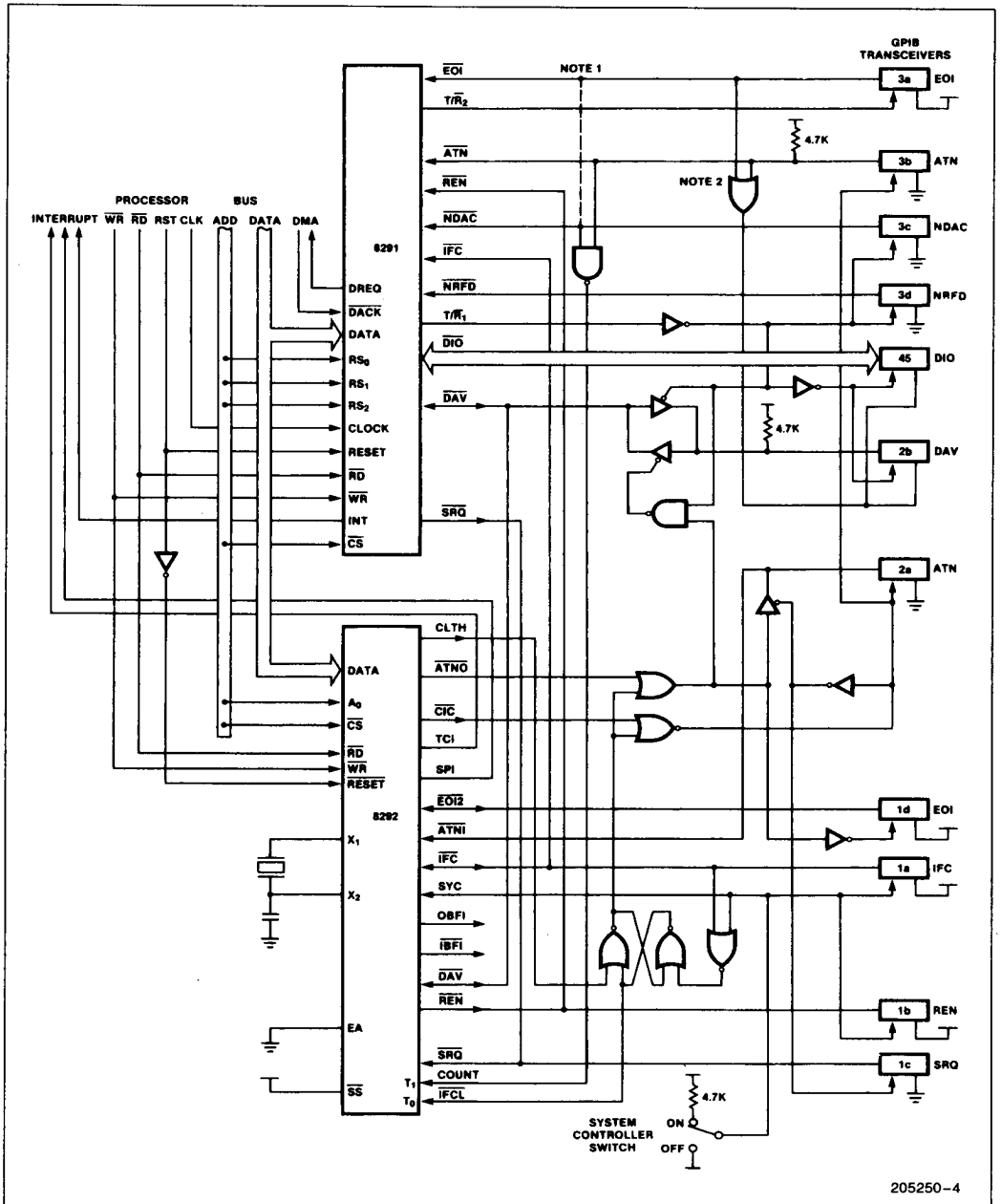
Reset and Power Up Procedure

After the 8292 has been reset either by the external reset pin, the device being powered on, or a RST command, the following sequential events will take place:

1) All outputs to the GPIB interface will go high ($\overline{SRQ}$, $\overline{ATNI}$, $\overline{IFC}$, SYC, CLTH, $\overline{ATNO}$, $\overline{CIC}$, TCI, SPI, $\overline{EOI}$, OBFI, $\overline{IBFI}$, $\overline{DAV}$, $\overline{REV}$).
2) The four interrupt outputs (TCI, SPI, OBFI, $\overline{IBFI}$) and CLTH output will go low.
3) The following registers will be cleared:
   Interrrupt Status
   Interrupt Mask
   Error Flag
   Erorr Mask
   Time Out
   Event Counter (= 256), counter is disabled.
4) If the 8292 is the system controller, and ABORT command will be executed, the 8292 will become the controller in charge, and it will enter the CACS state.
   If it is not the system controller, it will remain in CIDS.

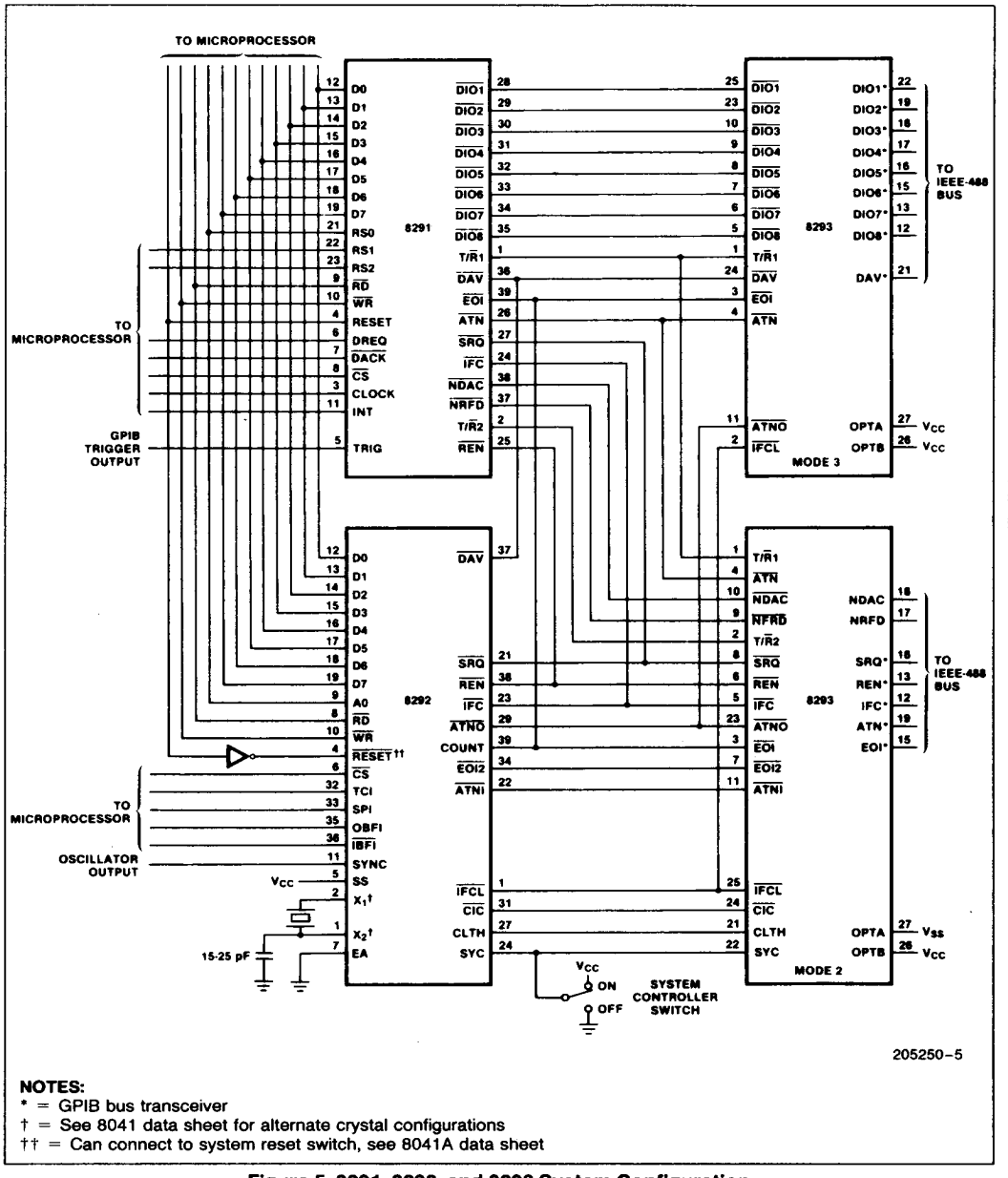
System Configuration

The 8291 and 8292 must be interfaced to an IEEE-488 bus meeting a variety of specifications including drive capability and loading characteristics. To interface the 8291 and the 8292 without the 8293's, several external gates are required, using a configuration similar to that used in Figure 5.



NOTES:
1. Connect to NDAC for byte count or to EOI for block count.
2. Gate ensures open collector operation during parallel poll.

Figure 4. 8291 and 8292 System Configuration



NOTES:
* = GPIB bus transceiver
† = See 8041 data sheet for alternate crystal configurations
†† = Can connect to system reset switch, see 8041A data sheet

Figure 5. 8291, 8292, and 8293 System Configuration

ABSOLUTE MAXIMUM RATINGS*

Ambient Temperature Under Bias .....0°C to 70°C
Storage Temperature .....−65°C to +150°C
Voltage to Any Pin with Respect to Ground.....0.5V to +7V
Power Dissipation.....1.5 Watt

NOTICE: This is a production data sheet. The specifications are subject to change without notice.

**WARNING: Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.*

D.C. CHARACTERISTICS T_A = 0°C to 70°C, V_{SS} = 0V: 8292, V_{CC} = ±5V ±10%

Symbol	Parameter	Min	Max	Unit	Test Conditions
V_{IL1}	Input Low Voltage (All Except X_1, X_2, $\overline{RESET}$)	−0.5	0.8	V	
V_{IL2}	Input Low Voltage (X_1, X_2, $\overline{RESET}$)	−0.5	06	V	
V_{IH1}	Input High Voltage (All Except X_1, X_2, $\overline{RESET}$)	2.2	V_{CC}	V	
V_{IH2}	Input High Voltage (X_1, X_2, $\overline{RESET}$)	3.8	V_{CC}	V	
V_{OL1}	Output Low Voltage (D_0–D_7)		0.45	V	I_{OL} = 2.0 mA
V_{OL2}	Output Low Voltage (All Other Outputs)		0.45	V	I_{OL} = 1.6 mA
V_{OH1}	Output High Voltage (D_0–D_7)	2.4		V	I_{OH} = −400 μA
V_{OH2}	Output High Voltage (All Other Outputs)	2.4		V	I_{OH} = −50 μA
I_{IL}	Input Leakage Current (COUNT, $\overline{IFCL}$, $\overline{RD}$, $\overline{WR}$, $\overline{CS}$, A_0		±10	μA	$V_{SS} \le V_{IN} \le V_{CC}$
I_{OZ}	Output Leakage Current (D_0–D_7, High Z State)		±10	μA	V_{SS} +0.45 ≤ $V_{IN} \le V_{CC}$
I_{LI1}	Low Input Load Current (Pins 21–24, 27–38)		0.5	mA	V_{IL} = 0.8V
I_{LI2}	Low Input Load Current ($\overline{RESET}$)		0.2	mA	V_{IL} = 0.8V
I_{CC}	Total Supply Current		125	mA	Typical = 65 mA
I_{IH}	Input High Leakage Current (Pins 21–24, 27–38)		100	μA	$V_{IN} = V_{CC}$
C_{IN}	Input Capacitance		10	pF	
$C_{I/O}$	I/O Capacitance		20	pF	

A.C. CHARACTERISTICS T_A = 0°C to 70°C, V_{SS} = 0V: 8292, V_{CC} = ±5V ±10%

DBB READ

Symbol	Parameter	Min	Max	Unit	Test Conditions
t_{AR}	$\overline{CS}$, A_0 Setup to $\overline{RD}$ ↓	0		ns	
t_{RA}	$\overline{CS}$, A_0 Hold to $\overline{RD}$ ↑	0		ns	
t_{RR}	$\overline{RD}$ Pulse Width	250		ns	
t_{AD}	$\overline{CS}$, A_0 to Data Out Delay		225	ns	C_L = 150 pF
t_{RD}	$\overline{RD}$ ↓ to Data Out Delay		225	ns	C_L = 150 pF
t_{DF}	$\overline{RD}$ ↑ to Data Float Delay		100	ns	
t_{CY}	Cycle Time	2.5	15	μs	

DBB WRITE

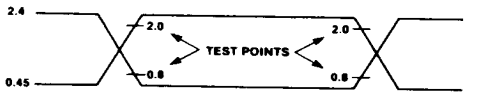
Symbol	Parameter	Min	Max	Unit	Test Conditions
t_{AW}	CS, A_0 Setup to WR ↓	0		ns	
t_{WA}	CS, A_0 Hold after WR ↑	0		ns	
t_{WW}	WR Pulse Width	250		ns	
t_{DW}	Data Setup to WR ↑	150		ns	
t_{WD}	Data Hold after WR ↓	0		ns	

COMMAND TIMINGS(1, 3)

Code	Name	Execution Time	$\overline{IBFI}$ ↑	TCI(2)	SPI	$\overline{ATNO}$	$\overline{CIC}$	$\overline{IFC}$	$\overline{REN}$	$\overline{EOI}$	$\overline{DAV}$	Comments
E1	WTOUT	63	24									
E2	WEVC	63	24									
E3	REVC	71	24	51								
E4	RERF	67	24	47								
E5	RINM	69	24	49								
E6	RCST	97	24	77								
E7	RBST	92	24	72								
E8												
E9	RTOUT	69	24	49								
EA	RERM	69	24	49								
F0	SPCNI	53	24									Count Stops after 39
F1	GIOL	88	24	70		↑61	↑61					
F2	RST	94	24		↓52							Not System Controller
F2	RST	214	24	192	↓52	↓179	↓174	↓101				System Controller
F3	RSTI	61	24									
F4	GSEC	125	24	107		↑98						
F5	EXPP	75	24						↓53 ↑59	↓55 ↑57		
F6	GTSB	118	24	100		↑91						
F7	SLOC	73	24	55				↑46				
F8	SREM	91	24	73				↓64				
F9	ABORT	155	24	133		↓120	↓115	↓42				
FA	TCNTR	108	24	86		↓71	↓68					
FC	TCAS	92	24	67		↓55						
FD	TCSY	115	24	91		↓80						
FE	STCNI	59	24									Starts Count after 43
PIN	RESET	29	—	↓7	↓7							Not System Controller
X	IACK	116	—		↓73 ↑98							If Interrupt Pending

NOTES:
1. All times are multiples of t_{CY} from the 8041A command interrupt.
2. TCI clears after 7 t_{CY} on all commands.
3. ↑ indicates a level transition from low to high, ↓ indicates a high to low transition.

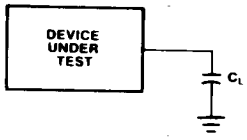
A.C. TESTING INPUT, OUTPUT WAVEFORM



205250-6

A.C. Testing: Inputs are driven at 2.4V for a Logic "1" and 0.45V for a Logic "0". Timing measurements are made at 2.0V for a Logic "1" and 0.8V for a Logic "0".

A.C. TESTING LOAD CIRCUIT

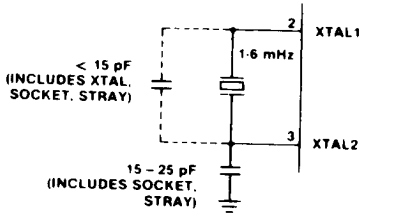


205250-7

C_L Include Jig Capacitance.

CLOCK DRIVER CIRCUITS

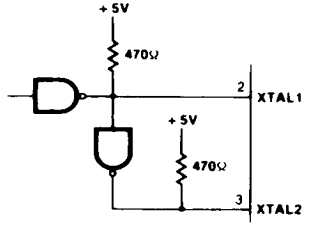
CRYSTAL OSCILLATOR MODE



205250-8

NOTE:
Crystal series resistance should be <75Ω at 6 MHz; <180Ω at 3.6 MHz.

DRIVING FROM EXTERNAL SOURCE

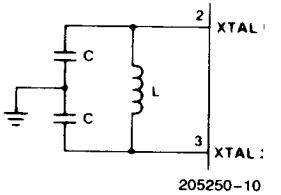


205250-9

NOTE:
Both XTAL1 and XTAL2 should be driven. Resistors to V_{CC} are needed to ensure V_{IH} = 2.8V if TTL circuitry is used.

LC OSCILLATOR MODE

L	C	NOMINAL f
45 μH	20 pF	5.2 MHz
120 μH	20 pF	3.2 MHz



205250-10

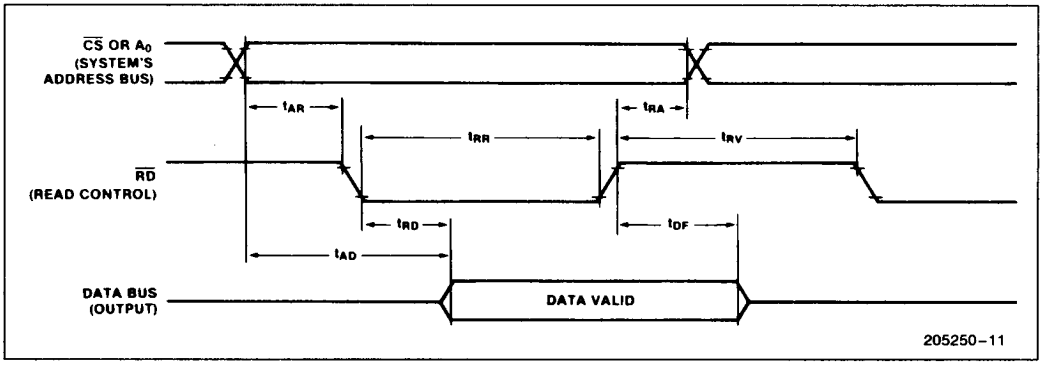
$$f = \frac{1}{2\pi\sqrt{LC'}}$$

$$C' = \frac{C + 3Cpp}{2}$$

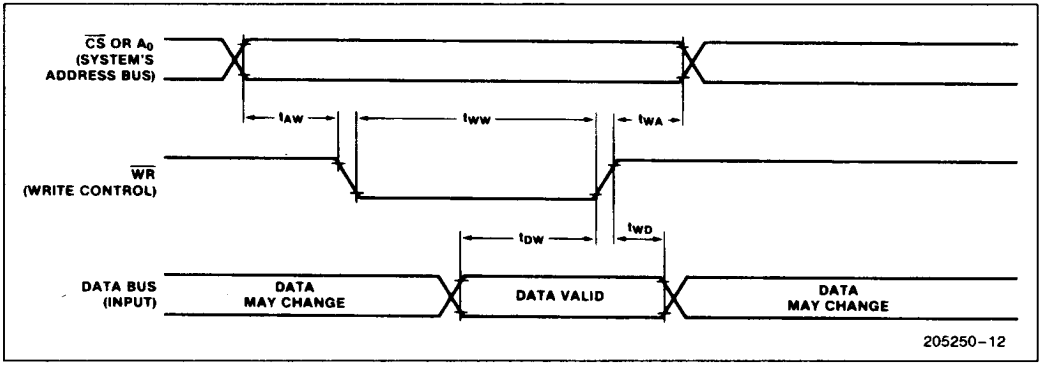
NOTES:
1. Cpp ≈ 5-10 pF pin-to-pin capacitance
2. Each C should be approximately 20 pF, including stray capacitance.

WAVEFORMS

READ OPERATION—DATA BUS BUFFER REGISTER



WRITE OPERATION—DATA BUS BUFFER REGISTER



APPENDIX A

The following tables and state diagrams were taken from the IEEE Standard Digital Interface for Programmable Instrumentation, IEEE Std. 488-1978. This document is the official standard for the GPIB bus and can be purchased from IEEE, 345 East 47th St., New York, NY 10017.

C MNEMONICS

Messages	Interface States
pon = power on	CIDS = controller idle state
rsc = request system control	CADS = controller addressed state
rpp = request parallel poll	CTRS = controller transfer state
gts = go to standby	CACS = controller active state
tca = take control asynchronously	CPWS = controller parallel poll wait state
tcs = take control synchronously	CPPS = controller parallel poll state
sic = send interface clear	CSBS = controller standby state
sre = send remote enable	CSHS = controller standby hold state
IFC = interface clear	CAWS = controller active wait state
ATN = attention	CSWS = controller synchronous wait state
TCT = take control	CSRS = controller service requested state
	CSNS = controller service not requested state
	SNAS = system control not active state
	SACS = system control active state
	SRIS = system control remote enable idle state
	SRNS = system control remote enable not active state
	SRAS = system control remote enable active state
	SIIS = system control interface clear idle state
	SINS = system control interface clear not active state
	SIAS = system control interface clear active state
	(ACDS) = accept data state (AH function)
	(ANRS) = acceptor not ready state (AH function)
	(SDYS) = source delay state (SH function)
	(STRS) = source transfer state (SH function)
	(TADS) = talker addressed state (T function)

205250-14

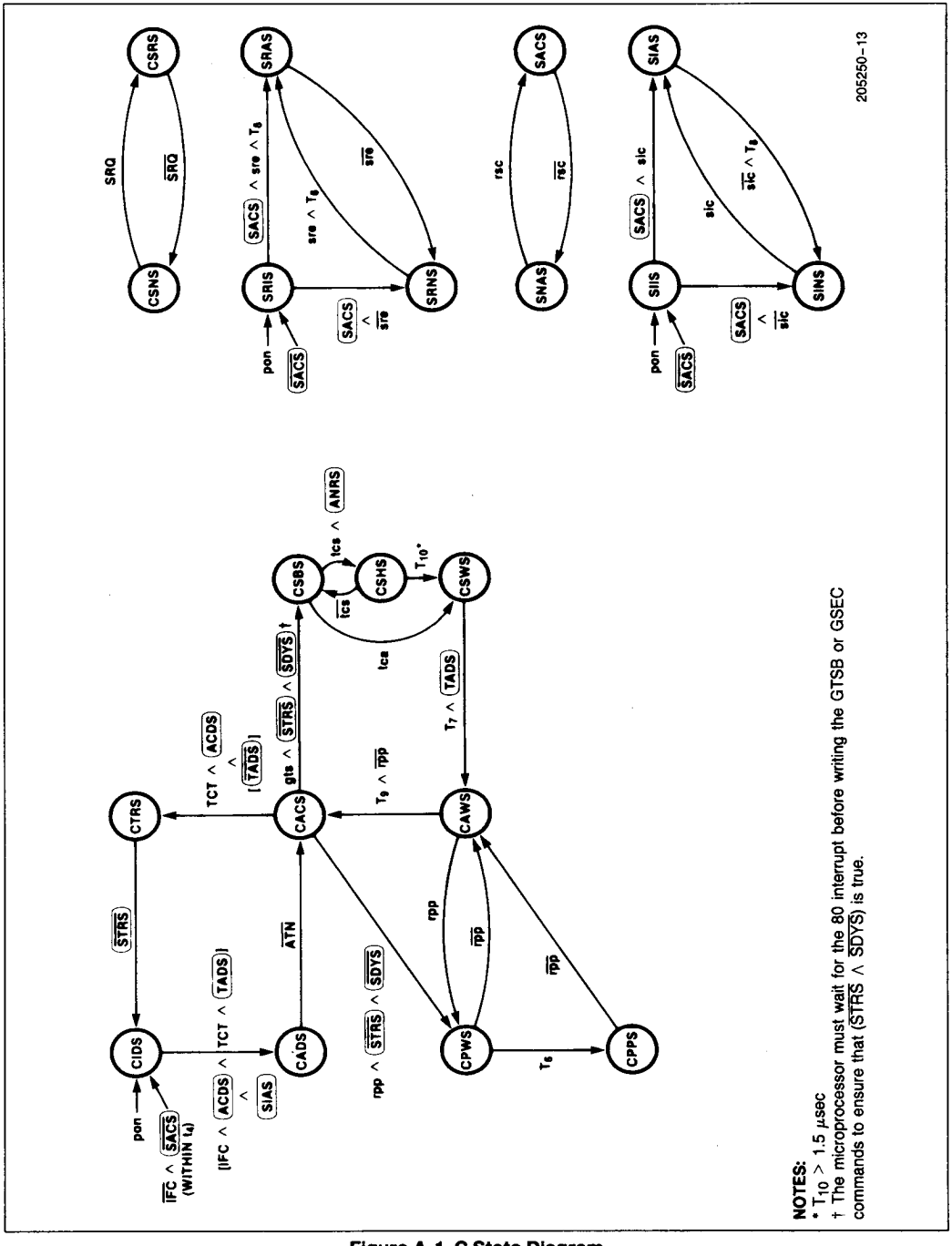


Figure A-1. C State Diagram

REMOTE MESSAGE CODING

Mnemonic	Message Name		TYPE	CLASS	DIO8	7	6	5	4	3	2	DIO1	NDAV	NRFD	NDAC	ATN	EOI	SRQ	IFC	REN
			Bus Signal Line(s) and Coding That Asserts the True Value of the Message																	
ACG	Addressed Command Group		M	AC	Y	0	0	0	X	X	X	X	X	X	X	1	X	X	X	X
ATN	Attention		U	UC	X	X	X	X	X	X	X	X	X	X	X	1	X	X	X	X
DAB	Data Byte	(Notes 1, 9)	M	DD	D8	D7	D6	D5	D4	D3	D2	D1	X	X	X	0	X	X	X	X
DAC	Data Accepted		U	HS	X	X	X	X	X	X	X	X	X	X	0	X	X	X	X	X
DAV	Data Valid		U	HS	X	X	X	X	X	X	X	X	1	X	X	X	X	X	X	X
DCL	Device Clear		M	UC	Y	0	0	1	0	1	0	0	X	X	X	1	X	X	X	X
END	End		U	ST	X	X	X	X	X	X	X	X	X	X	X	0	1	X	X	X
EOS	End of String	(Notes 2, 9)	M	DD	E8	E7	E6	E5	E4	E3	E2	E1	X	X	X	0	X	X	X	X
GET	Group Execute Trigger		M	AC	Y	0	0	0	1	0	0	0	X	X	X	1	X	X	X	X
GTL	Go to Local		M	AC	Y	0	0	0	0	0	0	1	X	X	X	1	X	X	X	X
IDY	Identify		U	UC	X	X	X	X	X	X	X	X	X	X	X	X	1	X	X	X
IFC	Interface Clear		U	UC	X	X	X	X	X	X	X	X	X	X	X	X	X	X	1	X
LAG	Listen Address Group		M	AD	Y	0	1	X	X	X	X	X	X	X	X	1	X	X	X	X
LLO	Local Lock Out		M	UC	Y	0	0	1	0	0	0	1	X	X	X	1	X	X	X	X
MLA	My Listen Address	(Note 3)	M	AD	Y	0	1	L5	L4	L3	L2	L1	X	X	X	1	X	X	X	X
MTA	My Talk Address	(Note 4)	M	AD	Y	1	0	T4	T3	T2	T1	T5	X	X	X	1	X	X	X	X
MSA	My Secondary Address	(Note 5)	M	SE	Y	1	1	S5	S4	S3	S2	S1	X	X	X	1	X	X	X	X
NUL	Null Byte		M	DD	0	0	0	0	0	0	0	0	X	X	X	X	X	X	X	X
OSA	Other Secondary Address		M	SE	(OSA = SCG ∧ $\overline{\text{MSA}}$)															
OTA	Other Talk Address		M	AD	(OTA = TAG ∧ $\overline{\text{MTA}}$)															
PCG	Primary Command Group		M	—	(PCG = ACG ∨ UCG ∨ LAG ∨ TAG)															
PPC	Parallel Poll Configure		M	AC	Y	0	0	0	0	1	0	1	X	X	X	1	X	X	X	X
PPE	Parallel Poll Enable	(Note 6)	M	SE	Y	1	1	0	S	P3	P2	P1	X	X	X	1	X	X	X	X
PPD	Parallel Poll Disable	(Note 7)	M	SE	Y	1	1	1	D4	D3	D2	D1	X	X	X	1	X	X	X	X
PPR1	Parallel Poll Response 1	(Note 10)	U	ST	X	X	X	X	X	X	X	1	X	X	X	1	1	X	X	X
PPR2	Parallel Poll Response 2		U	ST	X	X	X	X	X	X	1	X	X	X	X	1	1	X	X	X
PPR3	Parallel Poll Response 3		U	ST	X	X	X	X	X	1	X	X	X	X	X	1	1	X	X	X
PPR4	Parallel Poll Response 4		U	ST	X	X	X	X	1	X	X	X	X	X	X	1	1	X	X	X
PPR5	Parallel Poll Response 5		U	ST	X	X	X	1	X	X	X	X	X	X	X	1	1	X	X	X
PPR6	Parallel Poll Response 6		U	ST	X	X	1	X	X	X	X	X	X	X	X	1	1	X	X	X
PPR7	Parallel Poll Response 7		U	ST	X	1	X	X	X	X	X	X	X	X	X	1	1	X	X	X
PPR8	Parallel Poll Response 8		U	ST	1	X	X	X	X	X	X	X	X	X	X	1	1	X	X	X
PPU	Parallel Poll Unconfigure		M	UC	Y	0	0	1	0	1	0	1	X	X	X	1	X	X	X	X
REN	Remote Enable		U	UC	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	1
RFD	Ready for Data		U	HS	X	X	X	X	X	X	X	X	X	0	X	X	X	X	X	X
RQS	Request Service	(Note 9)	U	ST	X	1	X	X	X	X	X	X	X	X	X	0	X	X	X	X
SCG	Secondary Command Group		M	SE	Y	1	1	X	X	X	X	X	X	X	X	1	X	X	X	X
SDC	Selected Device Clear		M	AC	Y	0	0	0	0	1	0	0	X	X	X	1	X	X	X	X
SPD	Serial Poll Disable		M	UC	Y	0	0	1	1	0	0	1	X	X	X	1	X	X	X	X

REMOTE MESSAGE CODING (Continued)

Mnemonic	Message Name		Bus Signal Line(s) and Coding That Asserts the True Value of the Message																			
			T	C L	D							D		N	N							
			Y	A	I							I	D	R	D	A	E	S	I	R
			P	S	O							O	A	F	A	T	O	R	F	E
			E	S	8	7	6	5	4	3	2	1	V	D	C	N	I	Q	C	N
SPE	Serial Poll Enable		M	UC	Y	0	0	1	1	0	0	0	X	X	X	1	X	X	X	X
SRQ	Service Request		U	ST	X	X	X	X	X	X	X	X	X	X	X	X	X	1	X	X
STB	Status Byte	(Notes 8, 9)	M	ST	S	X	S	S	S	S	S	S	X	X	X	0	X	X	X	X
					8		6	5	4	3	2	1								
TCT	Take Control		M	AC	Y	0	0	0	1	0	0	1	X	X	X	1	X	X	X	X
TAG	Talk Address Group		M	AD	Y	1	0	X	X	X	X	X	X	X	X	1	X	X	X	X
UCG	Universal Command Group		M	UC	Y	0	0	1	X	X	X	X	X	X	X	1	X	X	X	X
UNL	Unlisten		M	1D	Y	0	1	1	1	1	1	1	X	X	X	1	X	X	X	X
UNT	Untalk	(Note 11)	M	1D	Y	1	0	1	1	1	1	1	X	X	X	1	X	X	X	X

The 1/0 coding on ATN when sent concurrent with multiline messages has been added to this revision for interpretive convenience.

3

NOTES:
1. D1–D8 specify the device dependent data bits.
2. E1–E8 specify the device dependent code used to indicate the EOS message.
3. L1–L5 specify the device dependent bits of the device's listen address.
4. T1–T5 specify the device dependent bits of the device's talk address.
5. S1–S5 specify the device dependent bits of the device's secondary address.
6. S specifies the sense of the PPR.
   Response = $\overline{S \oplus ist}$

P1–P3 specify the PPR message to be sent when a parallel poll is executed.

P3	P2	P1	PPR Message
0	0	0	PPR1
.	.	.	.
.	.	.	.
.	.	.	.
1	1	1	PPR8

7. D1–D4 specify don't-care bits that shall not be decoded by the receiving device. It is recommended that all zeroes be sent.
8. S1–S6, S8 specify the device dependent status (DIO7 is used for the RQS message.)
9. The source of the message on the ATN line is always the C function, whereas the messages on the DIO and EOI lines are enabled by the T function.
10. The source of the messages on the ATN and EOI lines is always the C function, whereas the source of the messages on the DIO lines is always the PP function.
11. This code is provided for system use.