μPB8289
BUS ARBITER

Description

The μPB8289 bus arbiter is used with the μPB8288 bus controller to interface 8086 and 8088 microprocessors to a multimaster system bus. The μPB8289 controls the μPB8288 bus controller and the bus transceivers and address latches, preventing them from accessing the system bus if the processor does not have use of the bus.

An external command sequence will cause the associated microprocessor to enter a wait state until the bus is ready. The processor remains in the wait state until the bus arbiter acquires use of the multimaster system bus. Then, the arbiter allows the bus controller, data transceivers, and address latches to access the system.

Once use of the bus has been acquired and data has been transferred, transfer acknowledge (XACK) is returned to the processor to indicate that the accessed slave device is ready. The processor may then complete its transfer cycle.

Features

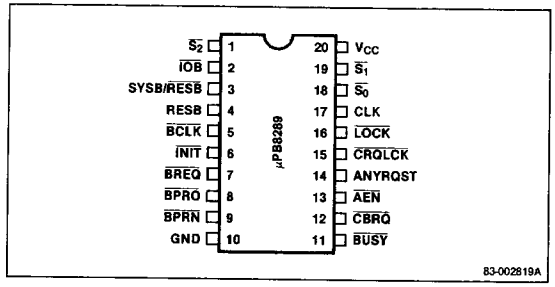
- ☐ Multimaster system bus protocol
- ☐ 8086 and 8088 processor synchronization with multimaster bus
- ☐ Simple interface with the 8288 bus controller and 8283/8282 address latches to a system bus
- ☐ Four operating modes for flexible system configuration
- ☐ Simplified interface to Multibus® systems
- ☐ Parallel, serial, and rotating priority resolution
- ☐ Bipolar buffering and drive capability

Ordering Information

Part Number	Package Type	Max Frequency of Operation
μPB8289D	20-Pin Cerdip	8 MHz

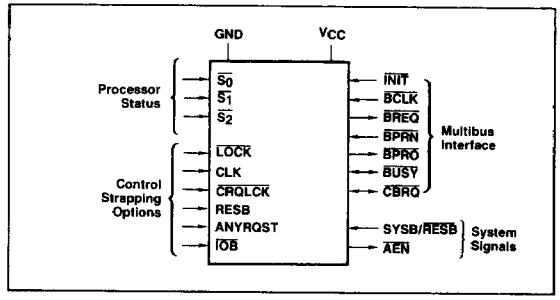
Multibus is a registered trademark of Intel Corporation.

Pin Configuration



83-002819A

Functional Configuration



Pin Identification

No.	Symbol	Function
1, 18, 19	$\overline{S_0}$-$\overline{S_2}$	Status inputs
2	IOB	I/O bus input
3	SYSB/$\overline{RESB}$	System bus/resident bus inputs
4	$\overline{RESB}$	Resident bus input
5	$\overline{BCLK}$	System bus clock input
6	$\overline{INIT}$	Initialize input
7	$\overline{BREQ}$	Bus request output
8	$\overline{BPRO}$	Bus priority output
9	$\overline{BPRN}$	Bus priority input
10	GND	Ground
11	$\overline{BUSY}$	Bus interface signal I/O
12	$\overline{CBRQ}$	Common bus request I/O
13	$\overline{AEN}$	Address enable output
14	ANYRQST	Any request input
15	$\overline{CRQLCK}$	Common request lock input
16	$\overline{LOCK}$	Lock input
17	CLK	Clock input
20	V_{CC}	+5 V power supply

8

Pin Functions

$\overline{S}_0$-$\overline{S}_2$ (Status Inputs)

The μPB8289 decodes these status inputs from the 8086 or 8088 processor to begin bus requests and surrenders.

$\overline{IOB}$ (I/O Bus)

This input signal tells the μPB8289 that there is an I/O peripheral bus and a multimaster system bus.

SYSB/$\overline{RESB}$ (System Bus/Resident Bus)

This input determines when bus requests and surrenders are permitted in SR mode.

$\overline{RESB}$ (Resident Bus Input)

$\overline{RESB}$ tells the μPB8289 that there is a multimaster and resident bus. When the signal is high, the SYSB/$\overline{RESB}$ pin handles bus arbitration.

$\overline{BLCK}$ (System Bus Clock)

This clock input synchronize all system bus interface signals.

$\overline{INIT}$ (Initialize)

This active low-input resets all bus arbiters on the multimaster bus. No arbiters have use of the bus following INIT.

$\overline{BREQ}$ (Bus Request)

An arbiter uses this output to request use of the multimaster system bus.

$\overline{BPRO}$ (Bus Priority Output)

In serial priority resolving schemes, this output daisy-chains to $\overline{BPRN}$ of the next lower priority arbiter.

$\overline{BPRN}$ (Bus Priority Input)

This input tells the arbiter it may acquire the bus on the next falling edge at $\overline{BCLK}$.

$\overline{BUSY}$ (Bus Interface Signal)

When the bus is available, this I/O signal notifies all arbiters on the bus. The highest requesting arbiter seizes the bus and pulls BUSY low to keep other arbiters off the bus.

$\overline{CBRQ}$ (Common Bus Request)

This signal is an input from a lower priority arbiter requesting the bus. It is an output from arbiters that surrender the multimaster bus upon request.

$\overline{AEN}$ (Address Enable)

This output tells the 8288 bus controller, 8284 clock driver, and the processor's address latches when to tri-state their output drivers.

ANYRQST (Any Request)

This signal allows the multimaster bus to be surrendered to a lower priority arbiter.

$\overline{CRQLCK}$ (Common Request Lock)

This input prevents the μPB8289 from surrendering the bus in response to a request on the $\overline{CBRQ}$ input.

$\overline{LOCK}$ (Lock)

This input prevents the arbiter from surrendering the multimaster system bus to any other bus arbiter, regardless of its priority.

CLK (Clock)

This is the clock signal from the 8284 clock generator.

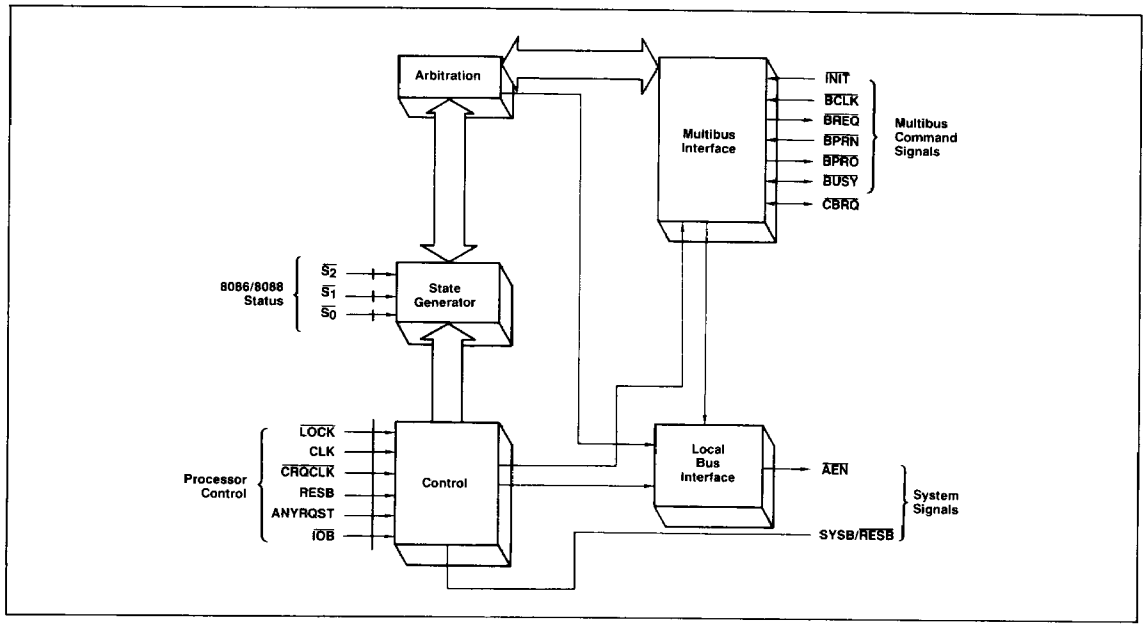
GND (Ground)

This is the ground.

V_{CC} (Power Supply)

This is the +5 V power supply.

Block Diagram



Functional Description

Bus Master Arbitration

Higher priority masters generally acquire use of the bus when a lower priority master completes its present transfer cycle. Lower priority masters acquire the bus when no higher priority master is accessing the system bus. The ANYRQST strapping option allows the arbiter to surrender the bus to a lower priority master as if it were a higher priority master. The arbiter maintains the bus as long as no other bus masters are requesting the bus and its processor has not entered the halt state. The arbiter does not voluntarily surrender the bus and must be forced off by a request from another bus master, unless the arbiter's processor has entered the halt state. Additional strapping options allow for other sets of conditions.

Priority Resolving Techniques

The μPB8289 provides several techniques for resolving priority between the many possible bus masters of a multimaster system bus. All of these techniques assume that one bus master will have priority over all others at any given time. You may use parallel, serial, or rotating priority resolving.

Parallel Priority Resolving

This technique (figures 1, 2) uses a bus request line ($\overline{BREQ}$) for each arbiter on the multimaster system bus. Each $\overline{BREQ}$ line goes to a priority encoder that generates the address of the highest priority active $\overline{BREQ}$ line. This binary address is decoded to select the bus priority in line ($\overline{BPRN}$) that is returned to the highest priority arbiter. The arbiter that receives priority ($\overline{BPRN}$ true) allows its bus master onto the multimaster system bus as soon as the bus becomes available. An arbiter that gets priority over another arbiter cannot immediately seize the bus, but must wait until the current bus transaction is complete. When the transaction is complete, the current occupant of the bus surrenders the bus by releasing $\overline{BUSY}$. $\overline{BUSY}$ is an active low OR tied line which goes to every arbiter on the system bus. When $\overline{BUSY}$ goes high (inactive), the priority arbiter seizes the bus and brings $\overline{BUSY}$ low to keep other arbiters off the bus. Note that all multimaster system bus transactions are synchronized to the bus clock ($\overline{BCLK}$).

8

Figure 1. Parallel Priority Resolving

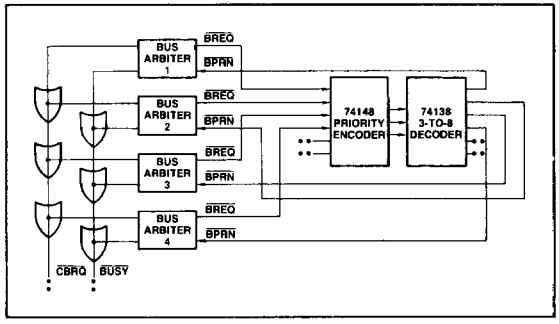
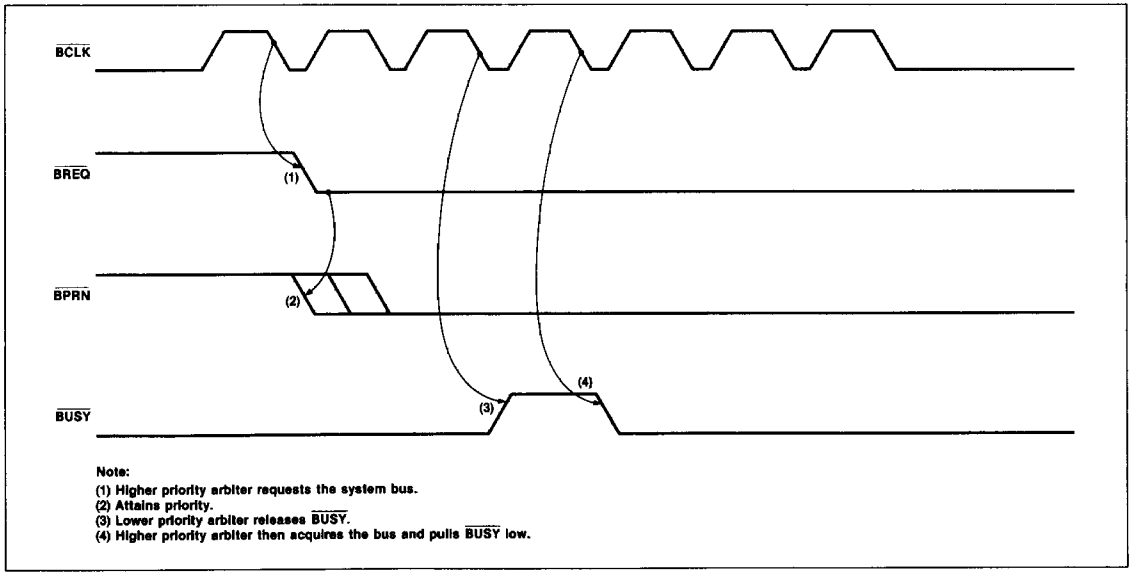


Figure 2. Higher Priority Arbiter Obtaining the Bus from a Lower Priority Arbiter

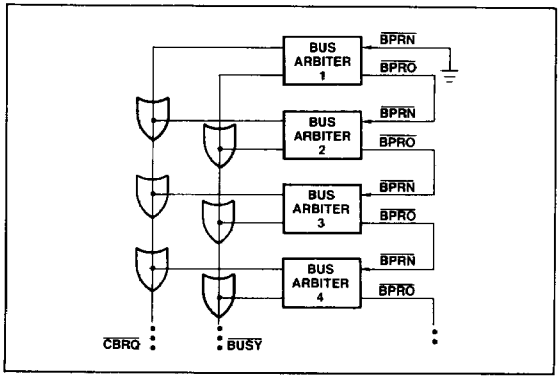


Note:
(1) Higher priority arbiter requests the system bus.
(2) Attains priority.
(3) Lower priority arbiter releases $\overline{\text{BUSY}}$.
(4) Higher priority arbiter then acquires the bus and pulls $\overline{\text{BUSY}}$ low.

Serial Priority Resolving

The serial priority resolving technique (figure 3) daisy-chains the bus arbiters together by connecting the higher priority arbiter's $\overline{BPRQ}$ output to the $\overline{BPRN}$ of the next lowest priority arbiter. This eliminates the need for the priority encoder-decoder arrangement. The number of arbiters that may be daisy-chained together is a function of $\overline{BCLK}$ and the propagation delay from arbiter to arbiter. At 10 MHz, only 3 arbiters may be daisy-chained.

Figure 3. Serial Priority Resolving



Rotating Priority Resolving

This technique resembles the parallel priority resolving technique except that priority is dynamically reassigned. The priority encoder is replaced by a circuit that rotates priority between arbiters to allow each arbiter an equal chance to use the system bus.

Modes of Operation

The μPB8289 has two basic operating modes: I/O peripheral bus mode ($\overline{IOB}$ mode), and resident bus mode (RESB mode). The $\overline{IOB}$ strapping option configures the μPB8289 into $\overline{IOB}$ mode and the RESB strapping option configures it to RESB mode. If both options are strapped false, the arbiter interfaces the processor to a multimaster system bus only. If both options are strapped true, the arbiter interfaces the processor to a multimaster system bus, a resident bus, and an I/O bus. Figure 4 shows the μPB8289 in a typical CPU system.

$\overline{IOB}$ Mode

$\overline{IOB}$ mode allows the processor to access both an I/O peripheral bus and a multimaster system bus. On an I/O peripheral bus, all devices on the bus, including memory, are treated as I/O devices and addressed by I/O commands. All memory commands are directed to the multimaster system bus. In $\overline{IOB}$ mode, the processor communicates with and controls peripherals over the peripheral bus and communicates with system memory over the system memory bus. Figure 5 shows the μPB8289 in this mode.

RESB Mode

RESB mode allows the processor to communicate over both a resident bus and a multimaster system bus. A resident bus can issue memory and I/O commands, but it is separate from the multimaster system bus. The resident bus has one master and is dedicated to only that master. The 8086 and 8088 can communicate with a resident bus and a multimaster system bus. The processor can access the memory and peripherals of both buses. Memory mapping selects which bus is accessed. The SYSB/$\overline{RESB}$ input on the arbiter instructs the arbiter on which bus to access. The signal connected to SYSB/$\overline{RESB}$ also enables and disables commands from one of the bus controllers. Figure 6 shows the μPB8289 in this mode.

Mode Summary

	Status Lines From 8086 or 8088 or 8089			IOB Mode Only	RESB (Mode) Only IOB = High RESB = High		IOB Mode RESB Mode IOB = Low RESB = High		Single Bus Mode IOB = High RESB = Low
	$\overline{S2}$	$\overline{S1}$	$\overline{S0}$	IOB = Low	SYSB/RESB = High	SYSB/RESB = Low	SYSB/RESB = High	SYSB/RESB = Low	
	0	0	0	x	✓	x	x	x	✓
I/O commands	0	0	1	x	✓	x	x	x	✓
	0	1	0	x	✓	x	x	x	✓
Halt	0	1	1	x	x	x	x	x	x
Memory commands	1	0	0	✓	✓	x	✓	x	✓
	1	0	1	✓	✓	x	✓	x	✓
	1	1	0	✓	✓	x	✓	x	✓
Idle	1	1	1	x	x	x	x	x	x

Notes:

(1) x = Multimaster system bus is allowed to be surrendered.

(2) ✓ = Multimaster system bus is requested.

Multimaster System Bus

Mode	Pin Strapping	Requested (1)	Surrendered (2)
Single bus multimaster mode	$\overline{IOB}$ = high RESB = low	When the processor's status lines go active	HLT + TI • HPBRQ†
RESB mode only	$\overline{IOB}$ = high RESB = high	SYSB/$\overline{RESB}$ = High 2 active	(SYSB/$\overline{RESB}$ = low + TI) CBRQ + HLT + HPBRQ
IOB mode only	$\overline{IOB}$ = low RESB = low	Memory commands	(I/O status + TI) • CBRQ + HLT + HPBRQ
IOB mode • RESB mode	$\overline{IOB}$ = low RESB = high	(Memory command) • (SYSB/$\overline{RESB}$ = high)	(I/O status commands) + (TI) (SYSB/$\overline{RESB}$ = low) • CBRQ + HPBRQ† + HLT)

Note:

(1) Except for HALT and idle status.

(2) $\overline{LOCK}$ prevents surrender of bus to any other arbiter. $\overline{CRQLCK}$ prevents surrender of bus to a lower priority arbiter.

(3) HLT = processor halt; $\overline{S}_2\overline{S}_1\overline{S}_0$ = 011.

(4) TI = processor idle; $\overline{S}_2\overline{S}_1\overline{S}_0$ = 111.

(5) + means OR.

(6) • means AND.

† HPBRQ = higher priority bus request or $\overline{BPRN}$ = 1.

Absolute Maximum Ratings

T_A = 25°C

Operating temperature	0°C to 70°C
Storage temperature	−65°C to +150°C
Voltage on any pin	−0.5 V to +7 V
All input voltages	−1.0 V to +5.5 V
Power dissipation	1.5 W

Comment: Exposing the device to stresses above those listed in Absolute Maximum Ratings could cause permanent damage. The device is not meant to be operated under conditions outside the limits described in the operational sections of this specification. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

DC Characteristics

T_A = 0°C to +70°C; V_{CC} = 5 V ±10%

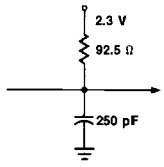
Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
Input low voltage	V_{IL}			0.8	V	
Input high voltage	V_{IH}	2.0			V	
Input clamp voltage	V_C			−1.0	V	V_{CC} = 4.50 V, I_C = −5 mA
Input forward current	I_F			−0.5	mA	V_{CC} = 5.50 V, V_F = 0.45 V
Reverse input leakage current	I_R			60	μA	V_{CC} = 5.50 V V_R = 5.50 V
Output low voltage	V_{OL}					
$\overline{BUSY}$, $\overline{CBRQ}$				0.45	V	I_{OL} = 20 mA
$\overline{AEN}$				0.45	V	I_{OL} = 16 mA
$\overline{BPRO}$, $\overline{BREQ}$				0.45	V	I_{OL} = 10 mA
Output high voltage	V_{OH}	Open collector				
$\overline{BUSY}$, $\overline{CBRQ}$		2.4			V	I_{OH} = 400 μA
All other outputs						
Power supply current	I_{CC}			165	mA	

Capacitance

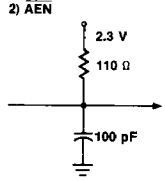
Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
Input capacitance	C_{IN} status			25	pF	
Input capacitance	C_{IN} (others)			12	pF	

Note:

1) $\overline{BUSY}$, $\overline{CBRQ}$



2) $\overline{AEN}$



3) $\overline{BPRO}$, $\overline{BREQ}$

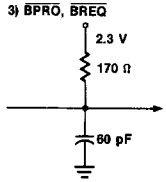
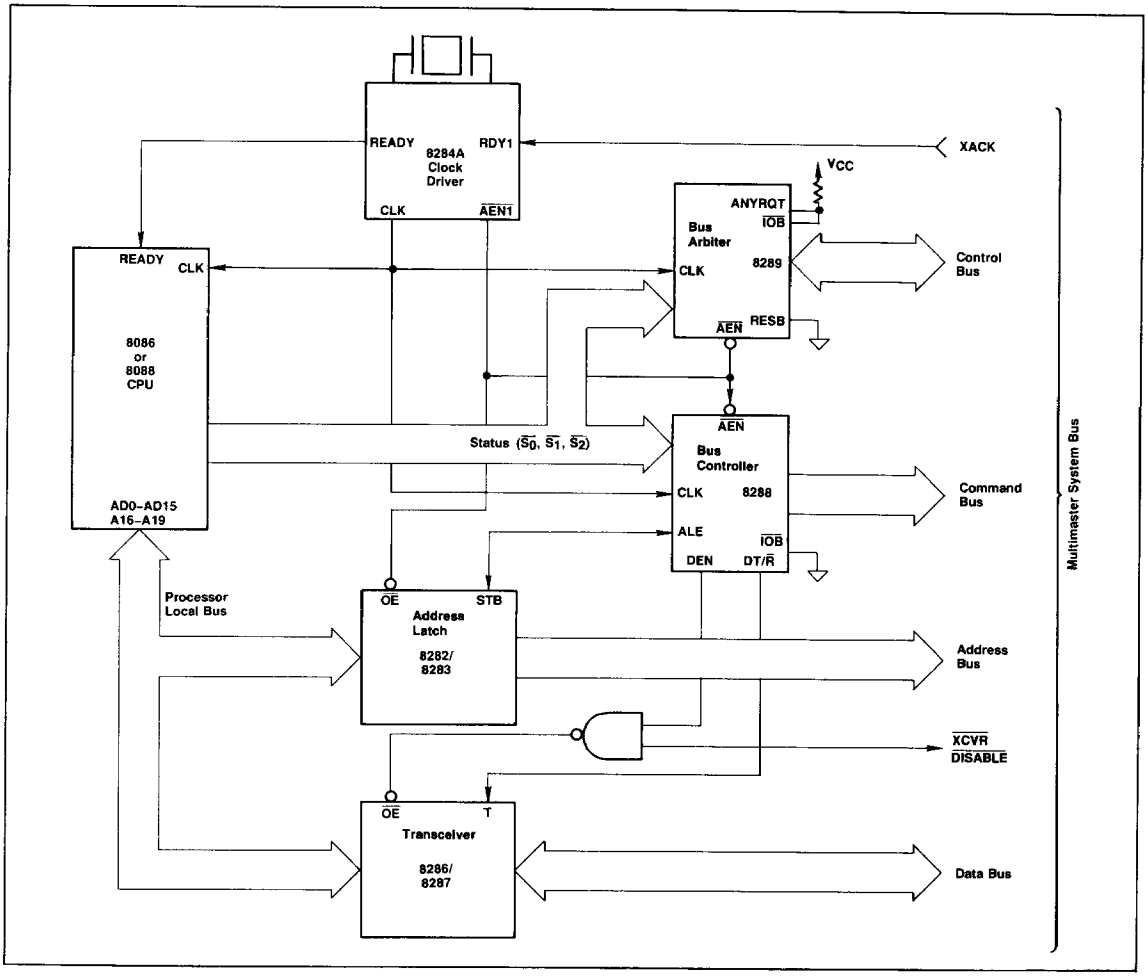


Figure 4. Typical CPU System Using the μPD8289 Bus Arbiter



8

Figure 5. Typical Medium-Complexity IOB System

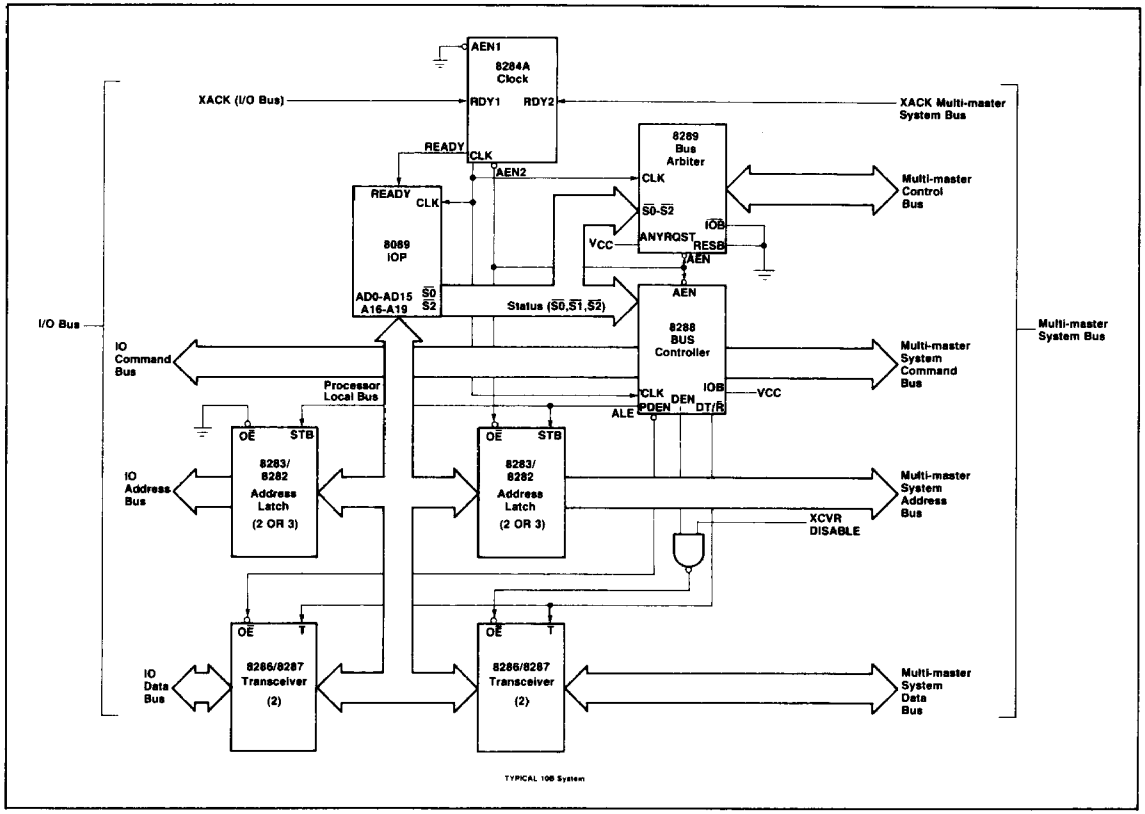
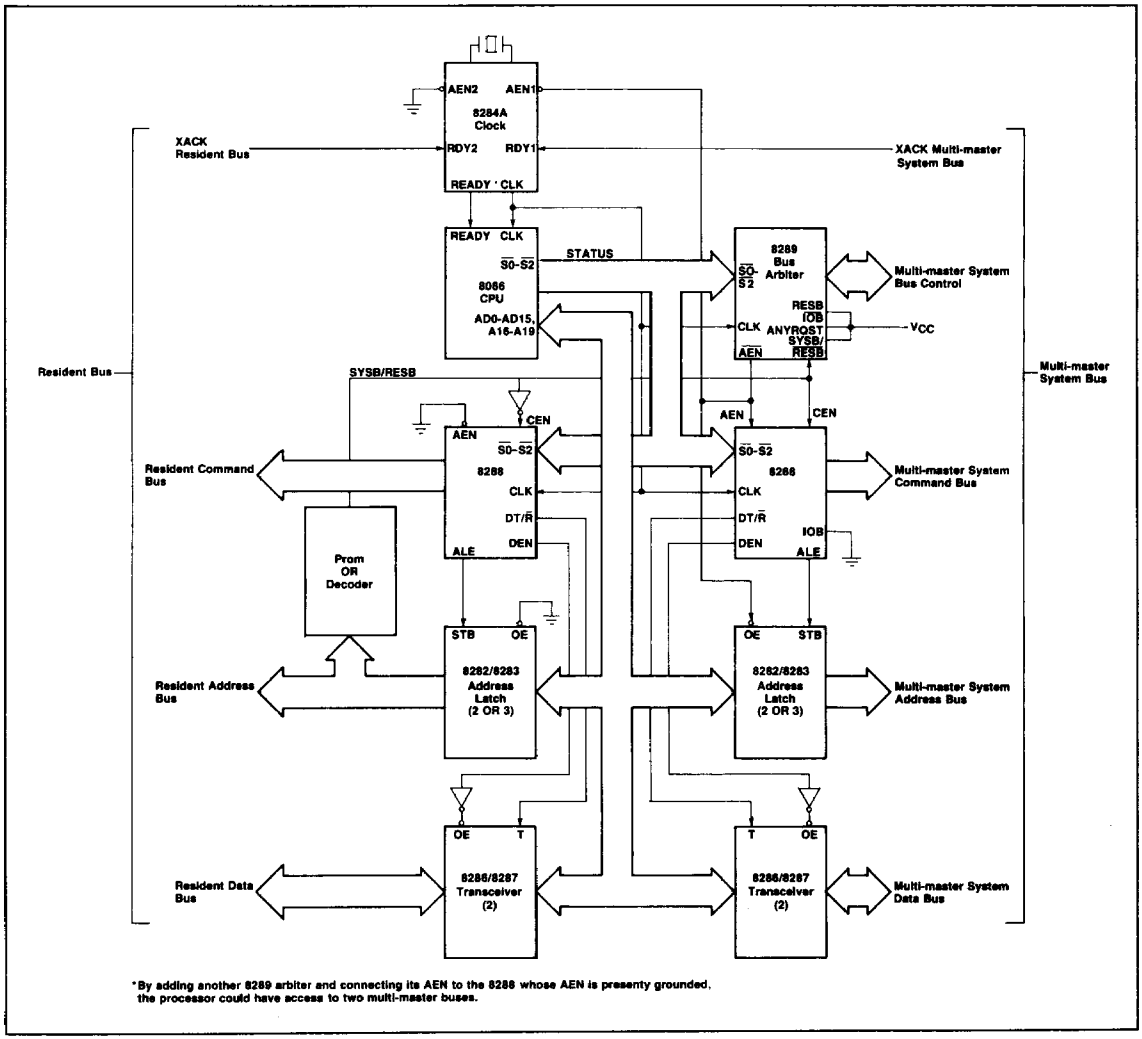


Figure 6. Typical System, Resident Bus Configuration



*By adding another 8289 arbiter and connecting its AEN to the 8288 whose AEN is presently grounded, the processor could have access to two multi-master buses.

8

AC Characteristics

T_A = 0°C to +70°C; V_{CC} = 5V ±10%

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
CLK cycle period	t_{CLCL}	125			ns	
CLK low time	t_{CLCH}	65			ns	
CLK high time	t_{CHCL}	35			ns	
Status active setup	t_{SVCH}	65		$t_{CLCL}-10$	ns	
Status inactive setup	t_{SHCL}	50		$t_{CLCL}-10$	ns	
Status active hold	t_{HVCH}	10			ns	
Status inactive hold	t_{HVCL}	10			ns	
$\overline{BUSY}$↑↓ setup to $\overline{BCLK}$↓	t_{BYSBL}	20			ns	
$\overline{CBRQ}$↑↓ setup to $\overline{BCLK}$↓	t_{CBSBL}	20			ns	
$\overline{BCLK}$ cycle time	t_{BLBL}	100			ns	
$\overline{BCLK}$ high time	t_{BHCL}	30		0.65 (t_{BLBL})	ns	
LOCK inactive hold	t_{CLLL1}	10			ns	
LOCK active setup	t_{CLLL2}	40			ns	
$\overline{BPRN}$↓↑ to $\overline{BCLK}$ setup time	t_{PNBL}	15			ns	
SYSB/$\overline{RESB}$ setup	t_{CLSR1}	0			ns	
SYSB/$\overline{RESB}$ hold	t_{CLSR2}	20			ns	
Initialization pulse width	t_{IVIH}	3 t_{BLBL} + 3 t_{CLCL}			ns	
Input rise time	t_{ILIH}			20	ns	From 0.8 V to 2.0 V
Input fall time	t_{IHIL}			12	ns	From 2.0 V to 0.8 V

Timing Response

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
BCLK to BREQ delay (1)	t_{BLBRL}			35	ns	
BCLK to BPRO (1)(2)	t_{BLPOH}			40	ns	
$\overline{BPRN}$↓↑ to BPRO↓↑ delay (1)(2)	t_{PNPO}			25	ns	
BCLK to BUSY low	t_{BLBYL}			60	ns	
BCLK to BUSY float (3)	t_{BLBYH}			35	ns	
CLK to AEN high	t_{CLAEH}			65	ns	
BCLK to AEN low	t_{BLAEL}			40	ns	
BCLK to CBRQ low	t_{BLCBL}			60	ns	
BCLK to CBRQ float (3)	t_{BLCRH}			35	ns	
Output rise time	t_{OLOH}			20	ns	From 0.8 V to 2.0 V
Output fall time	t_{OHOL}			12	ns	From 2.0 V to 0.8 V

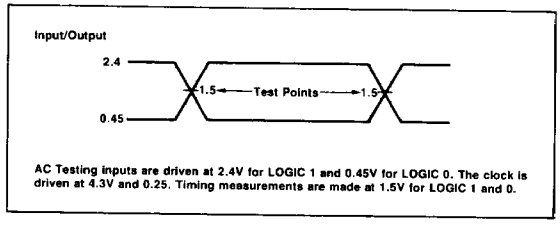
Note:

(1) Denotes that the spec applies to both transitions of the signal.

(2) BCLK generates the first BPRO. Subsequent changes of BPRO are generated through BPRON.

(3) Measured at 0.5 V above GND.

AC Test Condition



AC Testing inputs are driven at 2.4V for LOGIC 1 and 0.45V for LOGIC 0. The clock is driven at 4.3V and 0.25. Timing measurements are made at 1.5V for LOGIC 1 and 0.

Timing Waveforms

The signals related to CLK are typical processor signals and do not relate to the depicted sequence of events of the signals referenced to $\overline{BCLK}$. The signals shown related to the $\overline{BCLK}$ represent a hypothetical sequence of events for illustration. Assume three bus arbiters of priorities 1, 2, and 3 configured in the serial priority resolving scheme.

Assume arbiter 1 has the bus and is holding $\overline{BUSY}$ low. Arbiter 2 detects its processor wants the bus and pulls $\overline{BREQ}$ #2 low. If $\overline{BPRN}$ #2 is high (as shown), arbiter 2 pulls $\overline{CBRQ}$ low. $\overline{CBRQ}$ signals to higher-priority arbiter 1 that a lower-priority arbiter wants the bus. A higher-priority arbiter would be given $\overline{BPRN}$ when it makes the bus request rather than having to wait for another arbiter to release the bus through $\overline{CBRQ}$.

Arbiter 1 relinquishes the multimaster system bus when it enters a state of not requiring it, by lowering its $\overline{BPRO}$ #1 (tied to $\overline{BPRN}$ #2) and releasing $\overline{BUSY}$. Arbiter 2 now sees that it has priority from $\overline{BPRN}$ #2 being low and releases $\overline{CBRQ}$. As soon as $\overline{BUSY}$ signifies the bus is available (high), arbiter 2 pulls $\overline{BUSY}$ low on the next falling edge of $\overline{BCLK}$.

Note that if arbiter 2 didn't want the bus at the time it received priority, it would pass priority to the next lower priority by lowering its $\overline{BPRO}$ #2 (TPNPO). Note also that even a higher-priority arbiter which is aquiring the bus through $\overline{BPRN}$ will momentarily drop $\overline{CBRQ}$ until it has acquired the bus.

8

Timing Waveforms