WEITEK

ABACUS 3170 FLOATING-POINT COPROCESSOR FOR SPARC

PRELIMINARY DATA

May 1989

The Abacus 3170 is a single-chip floating-point coprocessor for the Fujitsu S-20/S-25 and LSI Logic L64801 implementations of the SPARC architecture. It incorporates a floating-point datapath and a floating-point controller. The Abacus 3170 provides direct interface to the integer unit and memory. It is available in speed grades of 20 and 25 MHz.

Related product: The Abacus 3171 single-chip floating-point coprocessor for Cypress 7C601 implementation of SPARC architecture.

Contents

USING THIS DATA SHEET

In the writing of this data sheet, it was assumed that the user is familiar with the SPARC architecture, as well as the hardware details of its implementation. This data sheet does not cover details that are explained in the following and other related literature:

The SPARC Architecture Manual, by Sun Microsystems

SPARC MB86901 (S-25) High Performance 32-Bit RISC Processor, by Fujitsu Microelectronics, Inc.

L64801 High Performance Open Architecture RISC Microprocessor, by LSI Logic Corporation.

WEITEK Abacus 3170 Floating-Point Coprocessor for SPARC
May, 1989

1060 East Arques Avenue
Sunnyvale, California 94086
Telephone (408) 738-8400

Printed in the United States of America
90 89 6 5 4 3 2 1

DOC 89XX

ABACUS 3170 FLOATING-POINT COPROCESSOR FOR SPARC

PRELIMINARY DATA
May 1989

Features

SINGLE-CHIP 64-BIT FLOATING-POINT DATA PATH AND CONTROLLER

64-bit multiplier and divide/square root unit
64-bit ALU
16×64 or 32×32 three-port register file with an independent load/store port

DIRECT INTERFACE TO FUJITSU S-20/S-25 AND LSI LOGIC L64801 SPARC PROCESSORS

DIRECT INTERFACE TO MEMORY

20 AND 25 MHz OPERATION

FULL COMPLIANCE WITH ANSI/IEEE-754 STANDARD FOR BINARY FLOATING-POINT ARITHMETIC

143-PIN PGA PACKAGE

LOW-POWER CMOS

Description

The Abacus 3170 is a high-performance, single-chip floating-point coprocessor for the Fujitsu S-20 and S-25/LSI Logic L64801 implementation of the SPARC architecture. It incorporates a floating-point datapath and a floating-point controller. The Abacus 3170 provides direct interface to the integer unit and memory. It is available in speed grades of 20 and 25 MHz.

The floating-point datapath circuitry contains a 64-bit multiplier, a 64-bit ALU, a 64-bit divide/square root unit, and a 16-word by 64-bit (or 32-word by 32-bit) three-port register file.

The floating-point controller circuitry handles IEEE exceptions and the interface between the floating-point datapath and the integer unit, as well as between the datapath and memory.

CONFORMANCE TO SPARC ARCHITECTURE

The Abacus 3170 processes instructions within the specifications of the SPARC architecture as described in the *SPARC Architecture Manual*, by Sun Microsystems.

DATA TYPES

The SPARC architecture specifies four data types that can be used in conjunction with the floating-point unit (FPU):

32-bit two's complement integer
single-precision floating-point
double-precision floating-point
extended-precision floating-point

The Abacus 3170 supports all of these data types except extended-precision. Any operation specifying extended-precision data types will be trapped to system software, with unimplemented instruction trap type.

INSTRUCTION PROCESSING

When the integer unit (IU) decodes a floating-point operate (FPop) or a floating-point load/store (FPLd/St) instruction, it sends the instruction to the FPU over the F bus during the Execute stage of the IU pipeline.

During the Write stage of the IU pipeline, the IU sends the FPop address over the F bus to the FPU so that it will be available for floating-point exception handling. Also during this cycle, the FPU will assert FHOLD– if a dependency exists. FHOLD– will remain asserted until the dependency has been resolved.

CONFORMANCE TO ANSI/IEEE-754 SPECIFICATION FOR BINARY FLOATING-POINT ARITHMETIC

The Abacus 3170 conforms to the requirements of the ANSI/IEEE-754 specification.

FLOATING-POINT STATE REGISTER (FSR)

The *SPARC Architecture Manual* contains detailed information about the Floating-Point State Register (FSR). Bits 19:17 of the FSR comprise the version field. The version field specifies the particular floating-point unit/controller implementation. In the case of the 3170, FSR (19:17) = 011_2.

Description, continued

IMPLEMENTED INSTRUCTIONS

Operations involving NaNs and denormalized numbers require system software assistance or intervention. They terminate with trap type unfinished.

Mnemonic(s)		Operation
ldf		Load floating-point register
lddf		Load double floating-point register
ldfsr		Load floating-point status register
stf		Store floating-point register
stdf		Store double floating-point register
stfsr		Store floating-point status register
stdfq		Store double floating-point queue
fitos	fitod	convert integer to floating-point (rounded as per *fsr.rd*) (single/double)
fstoi	fdtoi	convert floating-point to integer (rounded toward zero) (single/double)
fstod	fdtos	convert single to double/double to single floating-point
fmovs		register to register move
fnegs		register to register move with sign bit inverted
fabss		register to register move with sign bit set to 0
fsqrts	fsqrtd	floating-point square root (single/double)
fadds	faddd	floating-point add (single/double)
fsubs	fsubd	floating-point subtract (single/double)
fmuls	fmuld	floating-point multiply (single/double)
fdivs	fdivd	floating-point divide (single/double)
fcmps	fcmpd	floating-point compare (single/double)
fcmpes	fcmped	floating-point compare and exception if unordered (single/double)

Figure 1. Implemented instructions

UNIMPLEMENTED INSTRUCTIONS

Mnemonic(s)		Operation
fitox		convert integer to extended floating-point (rounded as per *fsr.rd*)
fxtoi		convert extended floating-point to integer (rounded toward zero)
fxtos	fxtod	convert extended floating-point to single/double floating-point
fstox	fdtox	convert single/double floating-point to extended floating-point
fsqrtx		floating-point square root (extended-precision)
faddx		floating-point add (extended-precision)
fsubx		floating-point subtract (extended-precision)
fmulx		floating-point multiply (extended-precision)
fdivx		floating-point divide (extended-precision)
fcmpx		floating-point compare (extended-precision)
fcmpex		floating-point compare and exception if unordered (extended-precision)
fsmuld		single product to double
fdmulx		double product to extended

Figure 2. Unimplemented instructions

Description, continued

DEVICE DESCRIPTION

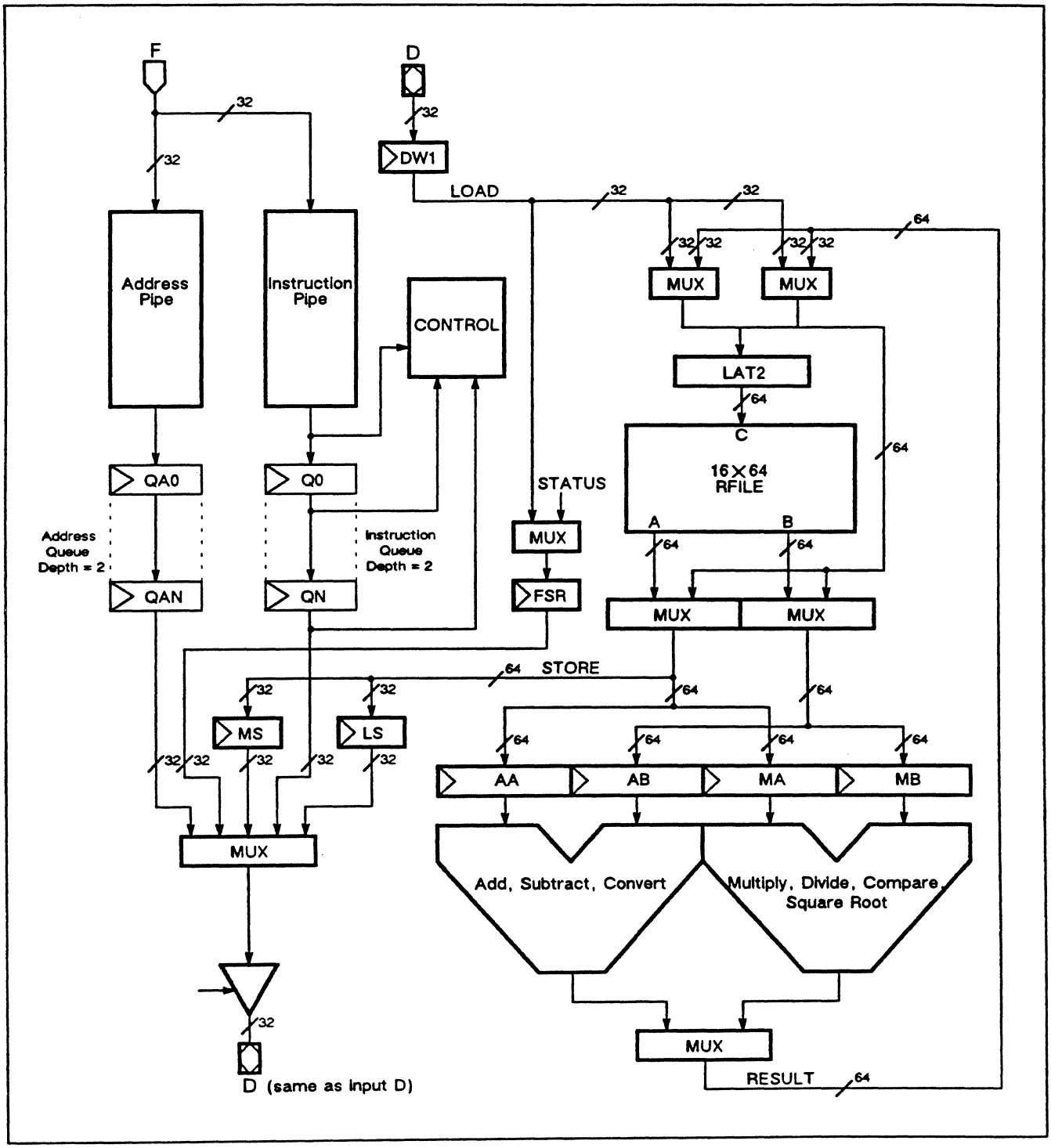


Figure 3. Conceptual block diagram

Description, continued

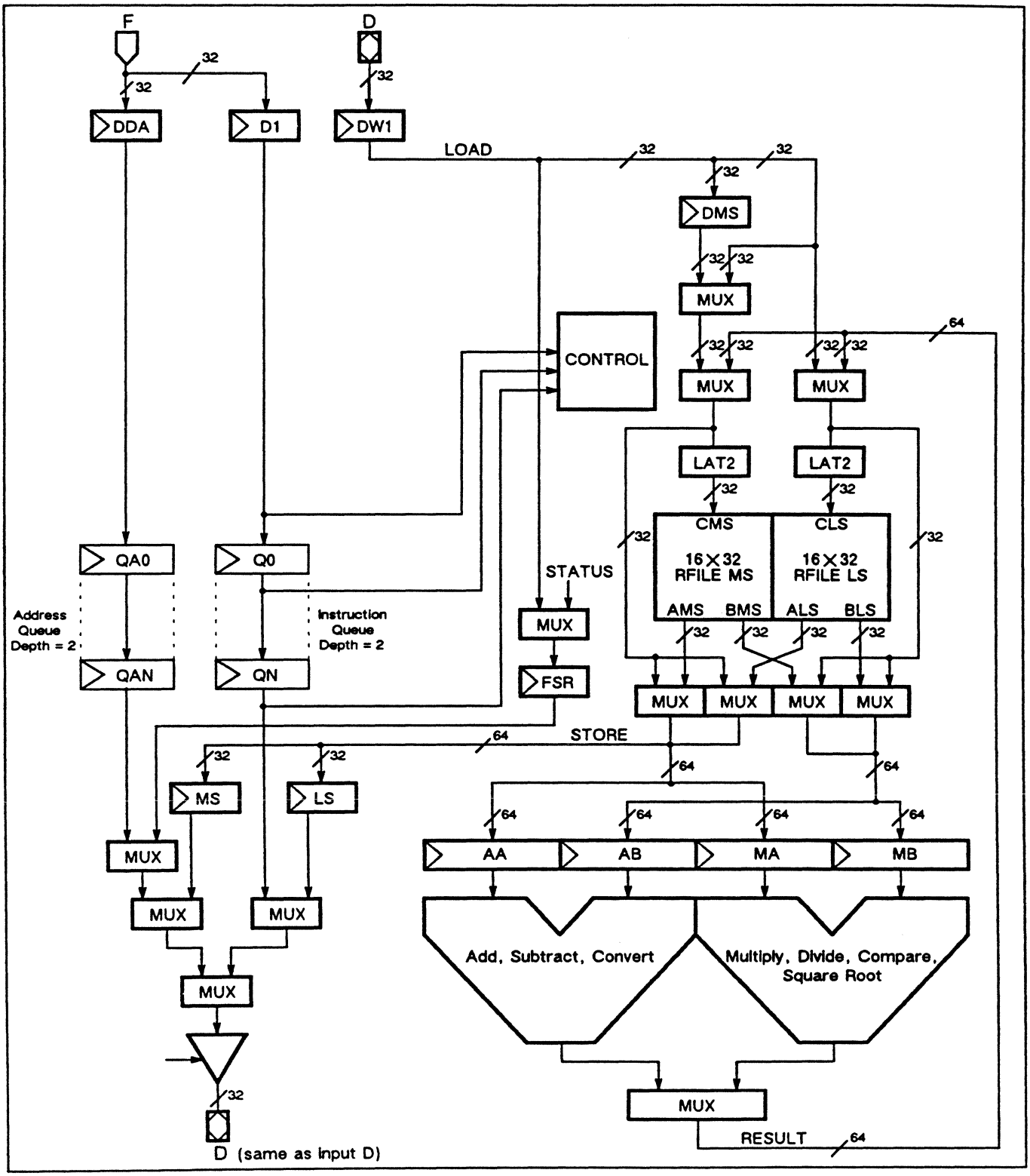


Figure 4. Simplified block diagram

Description, continued

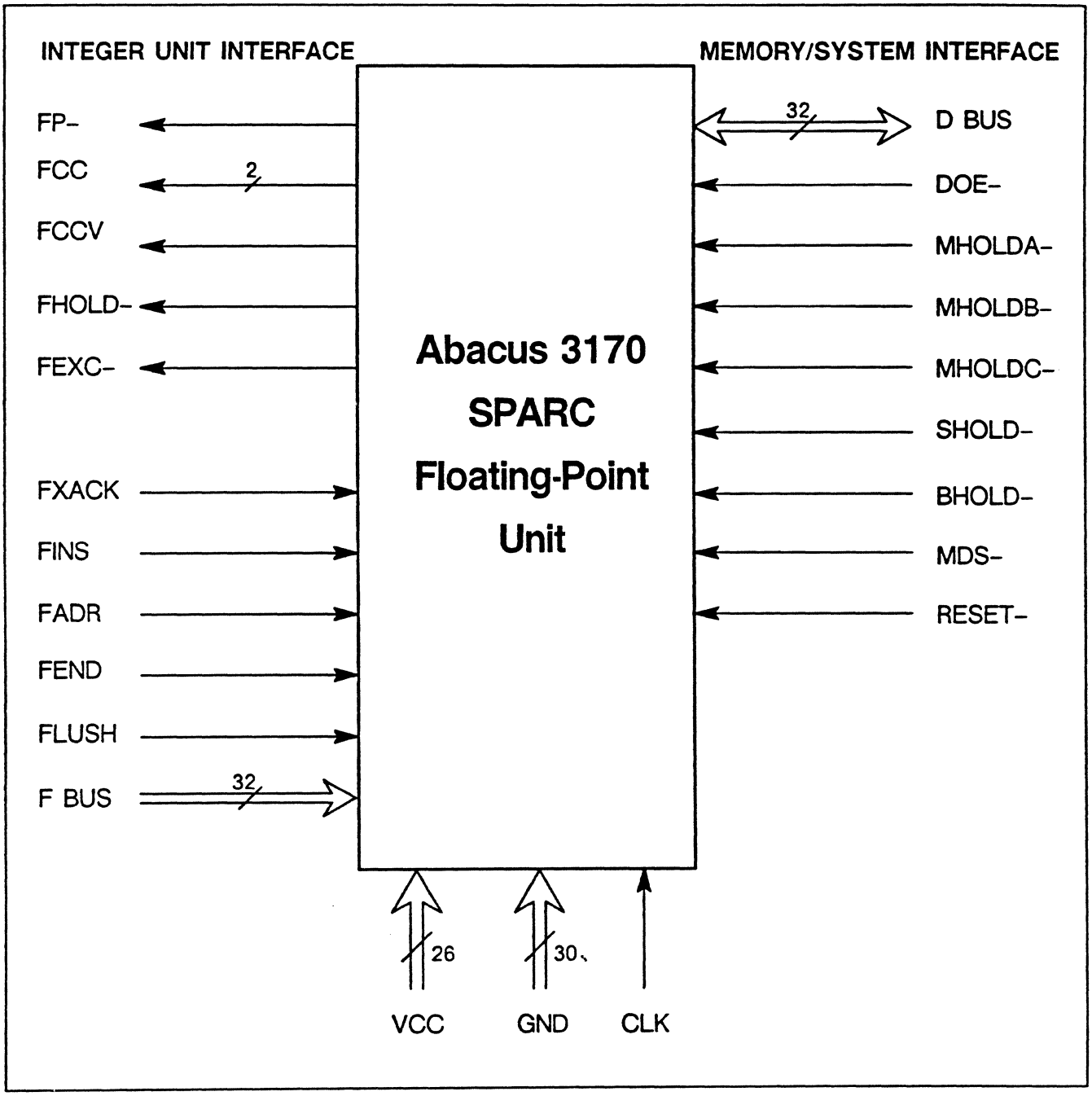


Figure 5. Abacus 3170 signals

Description, continued

SIGNAL DESCRIPTION

Signals marked with a minus sign (–) after their names are active low; all other signals are active high.

INTEGER UNIT INTERFACE SIGNALS

FP– OUTPUT

Floating-point unit present. The FP– signal indicates whether a floating-point unit FPU is present in the system. In the absence of an FPU the FP– signal is pulled up to VCC by a resistor. When an FPU is present the FP– signal is grounded.

FCC OUTPUT

Floating-point condition code. The $FCC_{1..0}$ bits represent the current condition code of the FPU. They are valid only if FCCV is asserted.

FBfcc instructions use these bits during the execute cycle if they are valid, and delay the execute cycle if they are not valid. The condition codes are shown below.

FCC (1)	FCC (0)	CONDITION
0	0	Equal
0	1	Op1 < Op2
1	0	Op1 > Op2
1	1	Unordered

Figure 6.

FCCV OUTPUT

Floating-point condition code valid. The FPU asserts the FCCV signal when FCC bits represent a valid condition. The FPU deasserts FCCV if pending floating-point compare instructions exist in the floating-point queue. FCCV is reasserted when the compare instruction is completed and FCC bits are valid.

FHOLD– OUTPUT

Floating-point hold. The FHOLD– signal is asserted by the FPU if it cannot continue execution due to a resource or operand dependency. The FPU checks for all dependencies in the write stage and, if necessary, asserts FHOLD– in the same cycle. The FHOLD– signal is used by the IU to freeze its pipeline in the next cycle. The FPU must eventually deassert FHOLD– to release the IU's pipeline.

FEXC– OUTPUT

Floating-point exception. The FEXC– signal is asserted if a floating-point exception has occured. It remains asserted until the IU acknowledges that it has taken a trap by asserting FXACK. Floating-point exceptions are taken only during the execution of a floating-point instruction, FBfcc instruction, or floating-point load or store instructions. When the FPU receives an asserted level of the FXACK signal it deasserts FEXC–.

FXACK INPUT

Floating-point exception acknowledge. The FXACK signal is asserted by the IU to acknowledge to the FPU that the current FEXC– trap is taken.

FINS INPUT

Floating-point instruction. The IU asserts FINS during the cycle in which $F_{31..0}$ carries a valid floating-point instruction. The FPU uses this signal to latch the instruction into its instruction register.

FADR INPUT

Floating-point address. The IU asserts FADR during the cycle in which $F_{31..0}$ carries a valid floating-point instruction address. The FPU uses this signal to latch the instruction into its address register.

FEND INPUT

End floating-point instruction. The IU asserts FEND during the last cycle of a floating-point instruction in the IU pipeline. The FPU uses FEND to synchronize the instruction/address in it execution pipeline with the IU pipeline.

FLUSH INPUT

Floating-point instruction flush. The FLUSH signal is asserted by the IU to signal to the FPU to flush the instructions in its instruction registers. This may happen when a trap is taken by the IU. The IU will restart the flushed instructions after returning from the trap. FLUSH has no effect on instructions in the floating-point queue.

F BUS INPUT

Floating-point bus. $F_{31..0}$ is a dedicated 32-bit bus that receives floating-point instructions and addresses from the IU. Each floating-point instruction must use this bus for two cycles. The first cycle carries the instruction and the second its address.

Description, continued

SYSTEM/MEMORY INTERFACE SIGNALS

D BUS INPUT/OUTPUT

Data bus. The $D_{31..0}$ bus is driven by the FPU only during the execution of floating-point store instructions. The alignment for load and store instructions is done in the FPU. A double word is aligned on an 8-byte boundary, a word is aligned on a 4-byte boundary.

DOE- INPUT

Data output enable. The DOE- signal is connected directly to the data output drivers and must be asserted during normal operation. Deassertion of this signal tri-states all output drivers on the data bus. This signal should be deasserted only when the bus is granted to another bus master, i.e., when either BHOLD-, MHOLDA-, or MHOLDB-, MHOLDC- or SHOLD- is asserted.

MHOLDA-, MHOLDB-, MHOLDC-, SHOLD- INPUTS

Memory hold. Asserting either MHOLDA-, MHOLDB-, MHOLDC-, or SHOLD- freezes the FPU pipeline.

BHOLD- INPUT

Bus hold. The BHOLD- signal is asserted by the system's I/O controller when an external bus master requests the data bus. Assertion of this signal will freeze the FPU pipeline.

MDS- INPUT

Memory data strobe. The MDS- signal is used to load data into the FPU when the internal FPU clock is stopped while on hold.

RESET- INPUT

Reset. Asserting the RESET- signal resets the pipeline and sets the writable fields of the floating-point status register (FSR) to zero. The RESET- signal must remain asserted for a minimum of eight cycles. After a reset, the IU will start fetching from address 0.

CLK INPUT

Clock. CLK is used for clocking the FPU. It is high during the first half of the processor cycle and low during the second half. The rising edge of CLK defines the beginning of each pipeline stage in the FPU.

VCC

Power supply. All VCC pins must be connected to 5.0 volt power supply.

GND

System ground. All GND pins must be connected to system ground.

NC

No connection. All no-connect pins must remain unconnected.

SYSTEM CONSIDERATIONS

LINPACK BENCHMARK ESTIMATE

The code shown below represents the inner loop of the SAXPY subroutine of the LINPACK benchmark. This loop requires 60 cycles on the Abacus 3170. At 25 MHz, this translates into a peak performance of 3.33 MFLOPS.

```
loop_top:
     ldd       [dx+0],dx0
     fmuld     dx0,da,dx0
     ldd       [dy+0],dy0
     ldd       [dx+8],dx1
     addcc     n,-4,n
     faddd     dx0,dy0,dy0
     fmuld     dx1,da,dx1
     ldd       [dy+8],dy1
     ldd       [dy+16],dx2
     add       dx,32,dx
     faddd     dx1,dy1,dy1
     fmuld     dx2,da,dx2
     ldd       [dy+16],dy2
     ldd       [dx-8],dx3
     add       dy,32,dy
     faddd     dx2,dy2,dy2
     fmuld     dx3,da,dx3
     ldd       [dy-8],dy3
     std       dy0,[dy-32]
     std       dy1,[dy-24]
     faddd     dx3,dy3,dy3
     std       dy2,[dy-16]
     bg        loop_top
     std       dy3,[dy-8]
```

Figure 7. LINPACK benchmark code

System Considerations

INTERFACE TO IU AND MEMORY

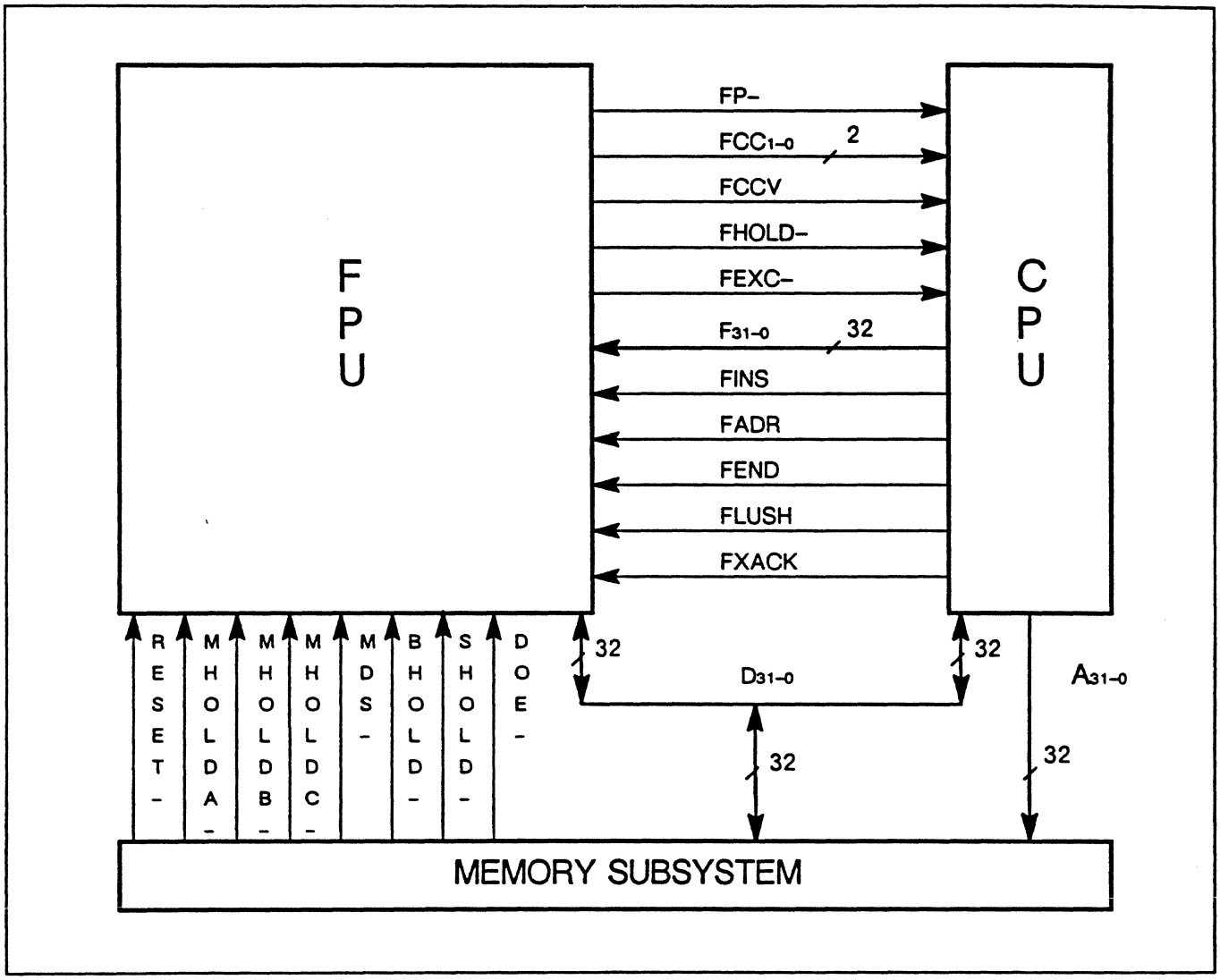


Figure 8. Interface to integer unit and memory

System Considerations, continued

INSTRUCTION OPERATION

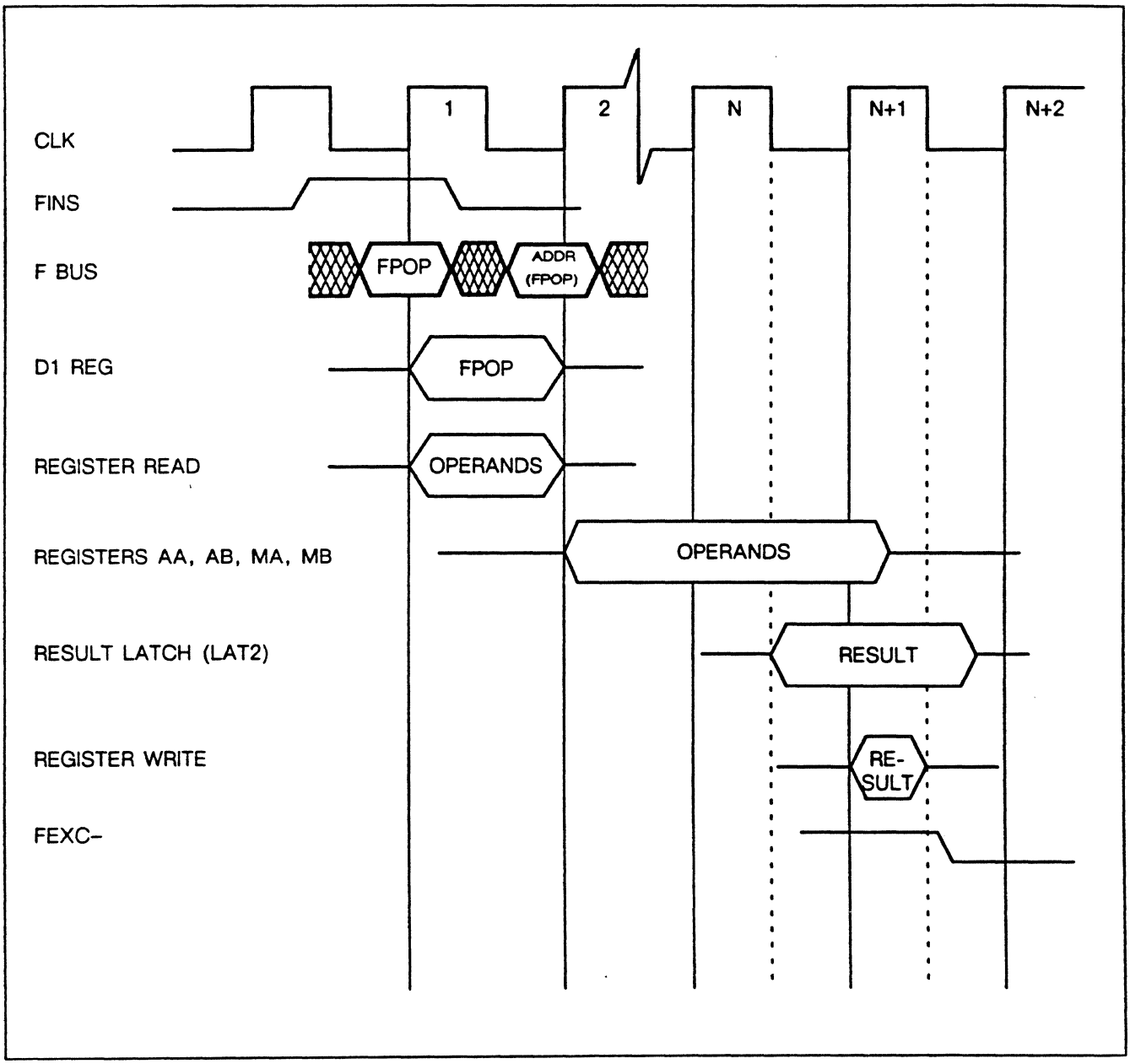


Figure 9. Instruction operation

Specifications

ABSOLUTE MAXIMUM RATINGS

Supply voltage	-0.5 to 7.0 V
Input voltage	-0.5V to VCC
Output voltage	-0.5V to VCC
Operating temperature range (T_{CASE})	0° to 85° C
Storage temperature range	-65° C to 150° C
Lead temperature (10 seconds)	300° C
Junction temperature	155° C

Figure 10.

OPERATING CONDITIONS

PARAMETER		MIN	MAX	UNIT
V_{CC}	Supply voltage	4.75	5.25	V
I_{OH}	High-level output current		-1.0	mA
I_{OL}	Low-level output current		4.0	mA
T_{CASE}	Operating case temperature	0	85	°C

Figure 11.

DC SPECIFICATIONS

PARAMETER		TEST CONDITIONS	MIN	MAX	UNIT
V_{IH}	High-level input voltage	V_{CC} = MIN	2.1		V
V_{IL}	Low-level input voltage	V_{CC} = MIN		0.8	V
V_{IHC}	High-level input voltage	V_{CC} = MIN	2.4		V
V_{ILC}	Low-level input voltage	V_{CC} = MIN		0.8	V
V_{OH}	High-level output voltage	V_{CC} = MIN, I_{OH} = MAX	2.4		V
V_{OL}	Low-level output voltage	V_{CC} = MIN, I_{OL} = MAX		0.4	V
I_{LI}	Input leakage current	V_{CC} = MAX, V_{IN} = 0 to V_{CC}		±10	µA
I_{LO}	Output leakage current (output disabled)	V_{CC} = MAX, V_{IN} = 0 or V_{CC}		±10	µA
C_{IN}	Input capacitance*	V_{CC} = MAX, V_{IN} = 0 to Vcc		15	pF
C_{OUT}	Output capacitance*	V_{CC} = MAX, V_{OUT} = 0 to Vcc		20	pF
C_{CLK}	Clock Input capacitance*	Vcc = MAX, V_{IN} = 0 to Vcc		25	pF
C_{DOE-}	DOE- Input capacitance*	Vcc = MAX, V_{IN} = 0 to Vcc		30	pF
I_{CC}	Supply current	V_{CC} = MAX, T_{CY} = MIN; TTL inputs			mA
* Guaranteed, but not tested					

Figure 12. DC specifications

Specifications, continued

AC SPECIFICATIONS AND TIMING DIAGRAMS

SYMBOL	DESCRIPTION	Min/Max	Reference	20 MHz	25 MHz
TCY	Clock Cycle Time	MIN		50	40
TCH	Clock High time	MIN		15	12
TCL	Clock Low Time	MIN		15	12
TR	CLK Rise time	MIN		3	2
TF	CLK Fall time	MIN		3	2
T1	FINS Setup Time	MIN	CLK+	16	12
T2	FINS Hold Time	MIN	CLK+	4	3
T3	F bus (Abus) Instruction Setup Time	MIN	CLK+	6	5
T4	F bus (Abus) Instruction Hold Time	MIN	CLK+	6	5
T5	FADR Setup Time	MIN	CLK+	16	12
T6	FADR Hold Time	MIN	CLK+	4	3
T7	D bus Data Load Setup Time	MIN	CLK+	5	4
T8	D bus Data Load Hold Time	MIN	CLK+	5	5
T9	FEND Setup Time	MIN	CLK+	16	12
T10	FEND Hold Time	MIN	CLK+	4	3
T11	D bus Data Store Output Delay Time	MAX	CLK+	33	27
T12	D bus Data Store Output Valid Time	MIN	CLK+	6	5
T13	MHOLDA- Setup Time*	MIN	CLK-/+	6/25	6/20
T14	MHOLDA- Hold Time*	MIN	CLK-	6	6
T15	FHOLD- Output Delay Time	MAX	CLK+	44	35
T16	FHOLD- Output Valid Time	MIN	CLK+	8	7
T17	MDS- Setup Time	MIN	CLK-/+	6/25	6/20
T18	MDS- Hold Time	MIN	CLK-	6	6
T19	FCCV Output Delay Time	MAX	CLK+	44	34
T20	FCCV Output Valid Time	MIN	CLK+	8	7
T21	$FCC_{1..0}$ Output Delay Time	MAX	CLK+	44	34
T22	$FCC_{1..0}$ Output Valid Time	MIN	CLK+	8	7
T23	FLUSH Setup Time	MIN	CLK+	22	16
T24	FLUSH Hold Time	MIN	CLK+	4	3
T25	FXACK Setup Time	MIN	CLK+	16	12
T26	FXACK Hold Time	MIN	CLK+	4	3
T27	FEXC- Output Delay Time	MAX	CLK+	30	24
T28	FEXC- Output Valid Time	MIN	CLK+	8	7
T29	RESET- Setup Time	MIN	CLK+	12	9
T30	RESET- Hold Time	MIN	CLK+	5	4
T31 **	D Bus Turn-off Time	MIN/MAX	DOE-	6/33	5/26
T32 **	D Bus Turn-on Time	MIN/MAX	DOE-	6/33	5/26

* Specifications for MHOLDB-, MHOLDC-, SHOLD-, and BHOLD- are the same.
** Guaranteed, but not tested

Figure 13. AC specifications

Specifications, continued

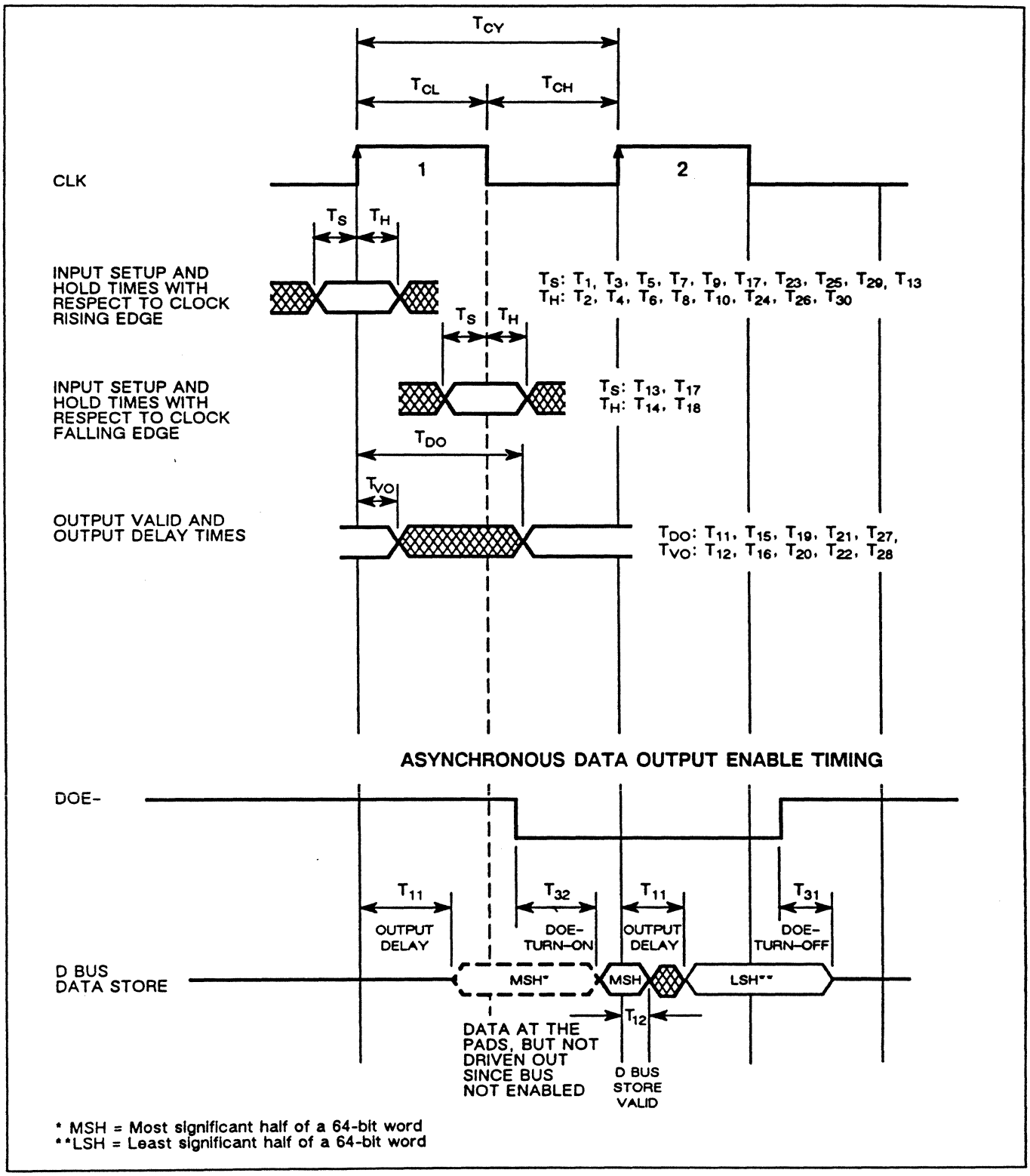


Figure 14. Timing diagrams

Specifications, continued

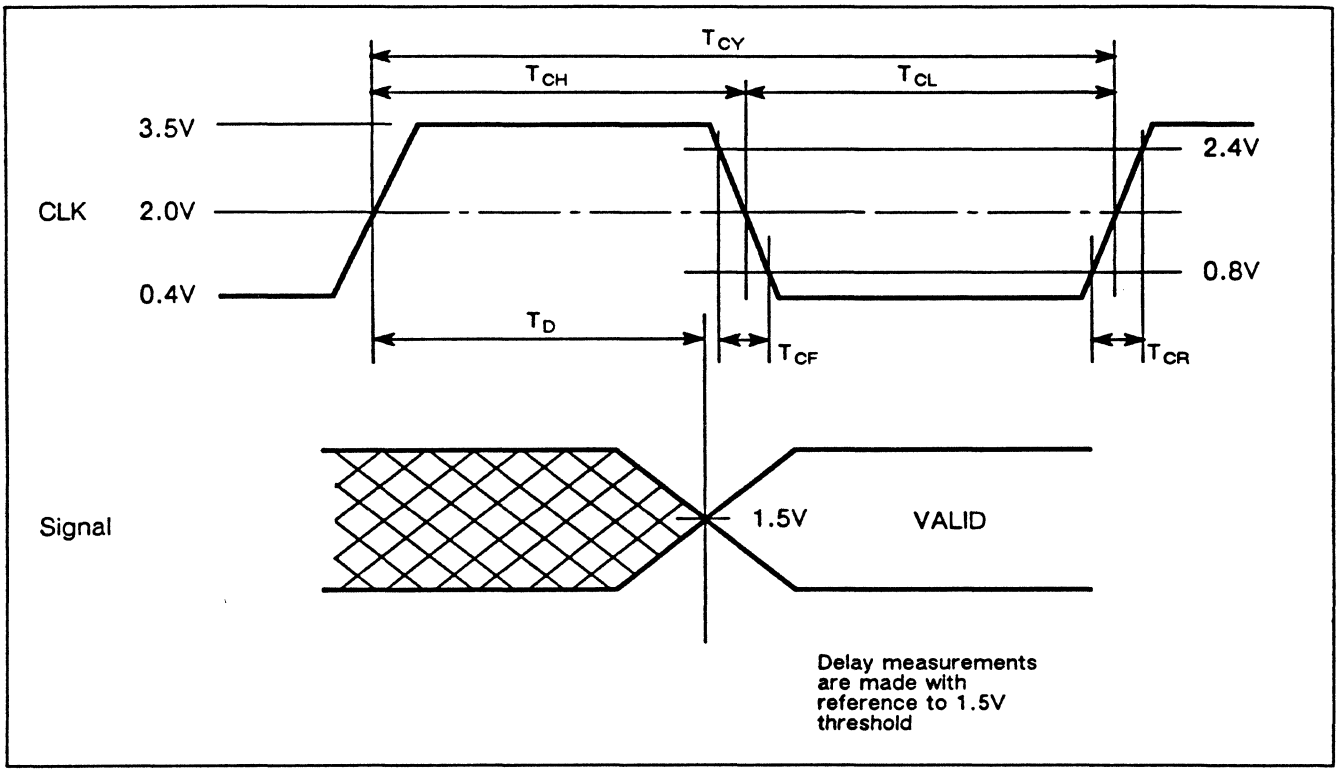


Figure 15. Reference levels in delay measurements

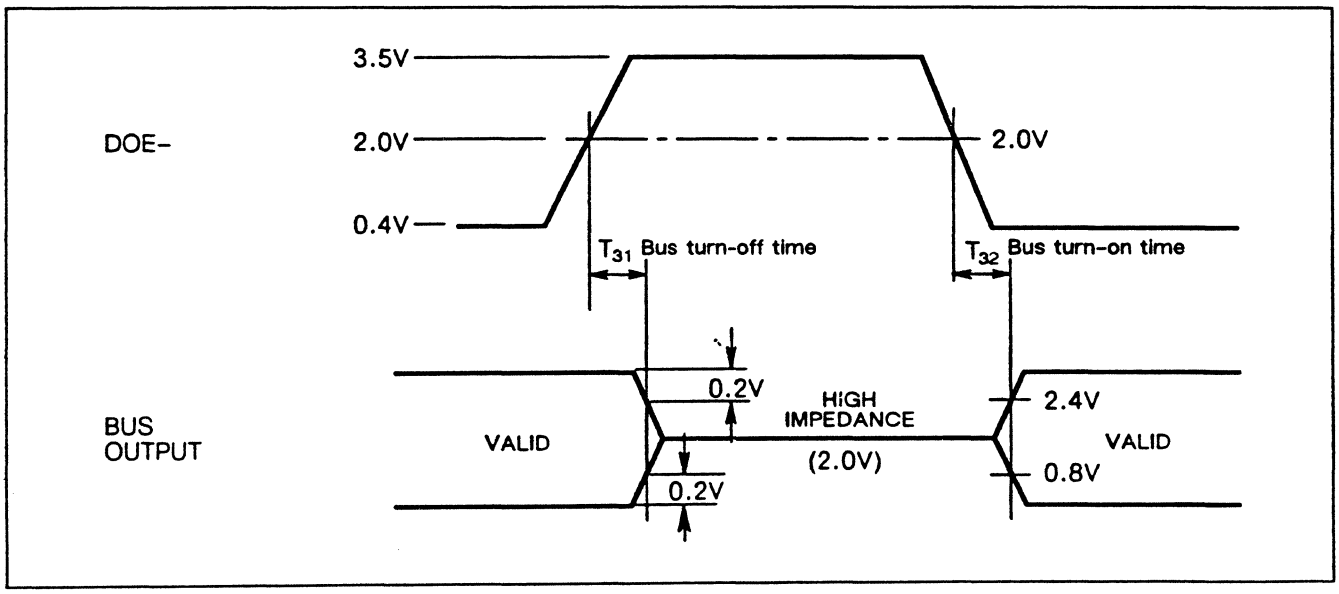


Figure 16. Tri-state timing

Specifications, continued

I/O CHARACTERISTICS

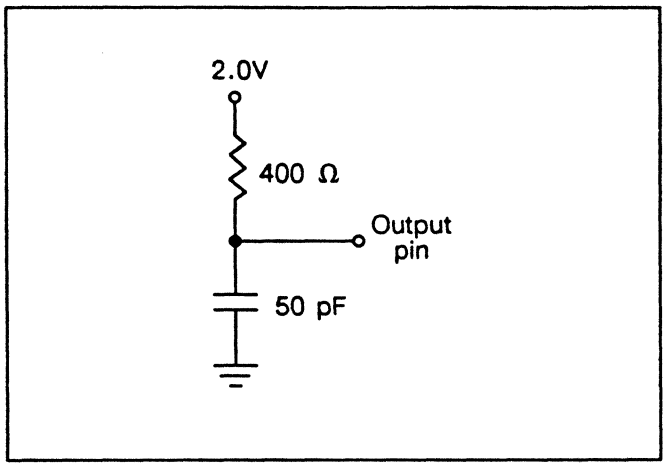


Figure 17. AC test load

Pin Configuration

Pin A1 Identifier	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
A		D22	F22	D24	F24	F25	D26	F26	F27	F28	F29	F30	F31	D31	NC	A
B	D21	VCC	VCC	F23	D23	VCC	D25	VCC	D27	D28	D29	D30	VCC	VCC	VCC	B
C	D20	F21	GND	GND	VCC	GND	GND	VCC	GND	GND	GND	GND	GND	VCC	FCCV	C
D	D19	VCC	GND										GND	GND	FCC1	D
E	F18	F19	F20										VCC	FCC0	FXACK	E
F	F16	D17	D18										RESET-	GND	FEXC-	F
G	D16	F17	GND										CLK	GND	NC	G
H	F0	F1	D0										GND	VCC	FHOLD-	H
J	D1	DOE-	GND										VCC	MHLDA-	BHOLD-	J
K	D2	VCC	GND										GND	MDS-	MHLDB-	K
L	F2	D3	GND										FLUSH	MHLDC-	SHOLD-	L
M	F3	VCC	D5										GND	FADR	FINS	M
N	D4	VCC	GND	GND	GND	D8	GND	D10	GND	GND	D14	GND	GND	VCC	FEND	N
P	F4	VCC	GND	A6	VCC	A8	VCC	A11	D12	VCC	VCC	VCC	D15	VCC	NC	P
R	F5	VCC	D6	F7	D7	F9	D9	F10	D11	F12	F13	D13	F14	F15	FP-	R
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	

15×15 143-PIN PGA

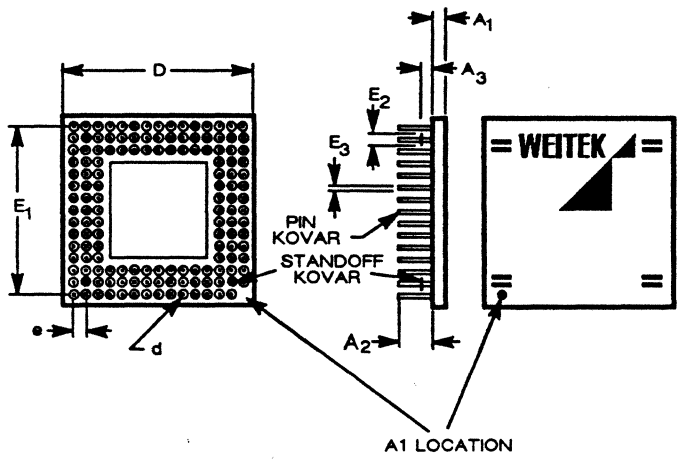
TOP VIEW
CAVITY DOWN

Note: NC = not connected; pins so marked must be left unconnected.
There is no pin at A1. A1 is a locator hole.

Figure 18.

Physical Dimensions

143-PIN GRID ARRAY



Symbol	DIMENSIONS	
	INCHES	MM
A1	0.100 ± 0.010	2.54 ± .20
A2	0.180 typ.	4.57 typ.
A3	0.050 typ.	1.27 typ.
D	1.575 sq. ± 0.016	40.0 ± 0.41
E1	1.400 sq. ± 0.012	35.56 ± 0.30
E2	0.050 dia. typ.	1.27 dia. typ.
E3	0.018 ± 0.002	.46 ± 0.05
d	0.065 dia. typ.	1.65 dia. typ.
e	0.100 typ.	2.54 typ.

Ordering Information

Package Type	Frequency	Case Temperature Range	Order Number
143-pin PGA	20 MHz	0–85°C	3170-020-GCD
143-pin PGA	25 MHz	0–85°C	3170-025-GCD

WEITEK

Headquarters

WEITEK Corporation
1060 East Arques
Sunnyvale, CA 94086
TEL (408) 738-8400
TWX 910-339-9545
WEITEK SVL
FAX (408) 738-1185

WEITEK U.S.A.

WEITEK Corporation
1060 East Arques
Sunnyvale, CA 94086
TEL (408) 738-8400
TWX 910-339-9545
WEITEK SVL
FAX (408) 738-1185

Corporate Place IV
111 South Bedford St.
Suite 200
Burlington, MA 01803
TEL (617) 229-8080
FAX (617) 229-4902

WEITEK Europe

Greyhound House
23/24 George St.
Richmond, Surrey
England TW9 1JY
TEL (011) 441-948-8608
TELEX 928940 RICHBI G
FAX (011) 441-940-6208

WEITEK Japan

4-8-1 Tsuchihashi
Miyamae-Ku
Kawasaki, Kanagawa-Pre
213 Japan
TEL 044-852-1135
FAX 044-877-4268