AMD

Am486®DE2

8-Kbyte Write-Through Embedded Microprocessor

DISTINCTIVE CHARACTERISTICS

- **High-Performance Design**
  - 66-MHz operating frequency
  - Frequent instructions execute in one clock
  - 105.6-million bytes/second burst bus at 33 MHz
  - Flexible write-through address control
  - Dynamic bus sizing for 8-, 16-, and 32-bit buses
  - Soft reset capability

- **High On-Chip Integration**
  - 8-Kbyte unified code and data cache
  - Floating-point unit
  - Paged, virtual memory management

- **Enhanced System and Power Management**
  - Stop clock control for reduced power consumption
  - Industry-standard, two-pin System Management Interrupt (SMI) for power management independent of processor operating mode and operating system
  - Static design with Auto Halt Power-Down support
  - Wide range of chipsets supporting SMM available to allow product differentiation

- **Complete 32-Bit Architecture**
  - Address and data buses
  - All registers
  - 8-, 16-, and 32-bit data types

- **Standard Features**
  - 3-V core with 5-V-tolerant I/O
  - Binary compatible with all Am486® DX and Am486DX2 microprocessors
  - Wide range of support available through the AMD® FusionE86SM Program

- **IEEE 1149.1 JTAG Boundary-Scan Compatibility**

- **Supports Environmental Protection Agency's Energy Star program**
  - 3-V operation reduces power consumption up to 40%
  - Energy management capability provides an excellent base for energy-efficient design
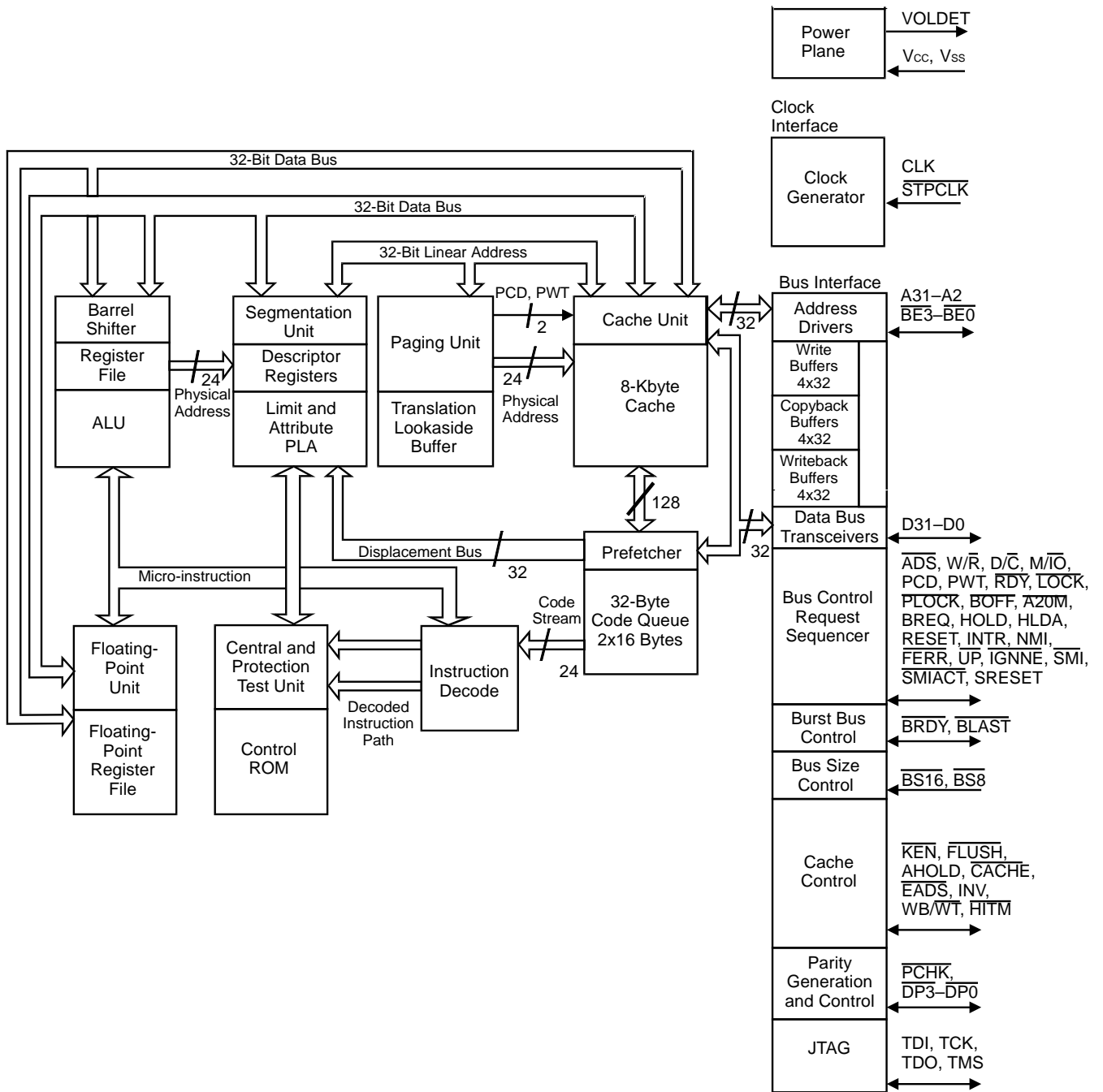 - Works with a variety of energy-efficient, power-managed devices

- **208-Lead SQFP or 168-Pin PGA Package**

GENERAL DESCRIPTION

The Am486DE2 microprocessor is an addition to the AMD Am486 microprocessor family. The Am486DE2 enhances system performance by incorporating flexible clock control and enhanced SMM.

The Am486DE2 CPU clock control feature permits the CPU to be stopped under controlled conditions, allowing reduced power consumption during system inactivity. The SMM function is implemented with an industry-standard, two-pin interface.
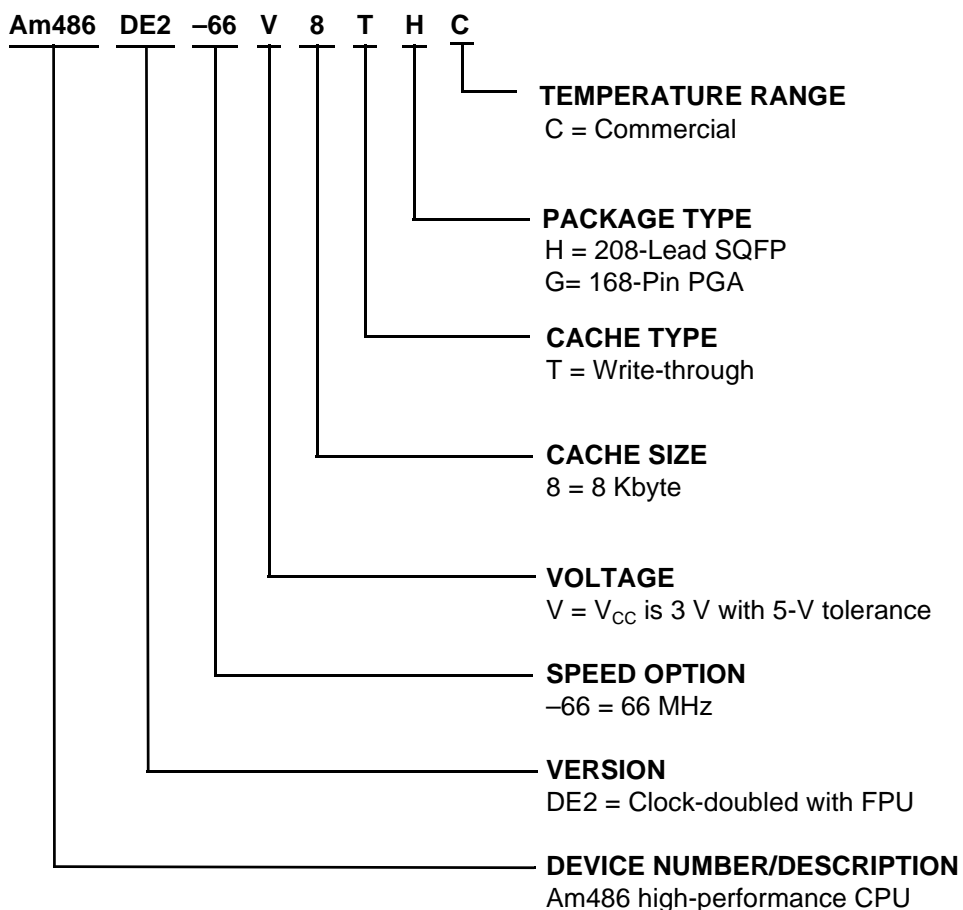
AMD

BLOCK DIAGRAM

Power Plane — VOLDET, V_CC, V_SS

Clock Interface
Clock Generator — CLK, $\overline{\text{STPCLK}}$

32-Bit Data Bus

32-Bit Data Bus

32-Bit Linear Address

Barrel Shifter
Register File
ALU

Segmentation Unit
Descriptor Registers
Limit and Attribute PLA

Paging Unit
Translation Lookaside Buffer

PCD, PWT / 2

24 Physical Address

24 Physical Address

Cache Unit
8-Kbyte Cache

Bus Interface

Address Drivers — A31–A2, $\overline{\text{BE3}}$–$\overline{\text{BE0}}$

Write Buffers 4x32

Copyback Buffers 4x32

Writeback Buffers 4x32

Data Bus Transceivers — D31–D0

32

128

Displacement Bus / 32

Micro-instruction

Prefetcher
32-Byte Code Queue 2x16 Bytes

Code Stream / 24

Floating-Point Unit
Floating-Point Register File

Central and Protection Test Unit
Control ROM

Instruction Decode

Decoded Instruction Path

Bus Control Request Sequencer — $\overline{\text{ADS}}$, W/$\overline{\text{R}}$, D/$\overline{\text{C}}$, M/$\overline{\text{IO}}$, PCD, PWT, $\overline{\text{RDY}}$, $\overline{\text{LOCK}}$, $\overline{\text{PLOCK}}$, $\overline{\text{BOFF}}$, $\overline{\text{A20M}}$, BREQ, HOLD, HLDA, RESET, INTR, NMI, $\overline{\text{FERR}}$, UP, $\overline{\text{IGNNE}}$, $\overline{\text{SMI}}$, $\overline{\text{SMIACT}}$, SRESET

Burst Bus Control — $\overline{\text{BRDY}}$, $\overline{\text{BLAST}}$

Bus Size Control — $\overline{\text{BS16}}$, $\overline{\text{BS8}}$

Cache Control — $\overline{\text{KEN}}$, $\overline{\text{FLUSH}}$, AHOLD, $\overline{\text{CACHE}}$, $\overline{\text{EADS}}$, INV, WB/$\overline{\text{WT}}$, $\overline{\text{HITM}}$

Parity Generation and Control — $\overline{\text{PCHK}}$, $\overline{\text{DP3}}$–$\overline{\text{DP0}}$

JTAG — TDI, TCK, TDO, TMS

ORDERING INFORMATION

Standard Product

AMD standard products are available in several packages and operating ranges. Valid order numbers are formed by a combination of the elements below.

Am486   DE2   –66   V   8   T   H   C

TEMPERATURE RANGE
C = Commercial

PACKAGE TYPE
H = 208-Lead SQFP
G= 168-Pin PGA

CACHE TYPE
T = Write-through

CACHE SIZE
8 = 8 Kbyte

VOLTAGE
V = V_{CC} is 3 V with 5-V tolerance

SPEED OPTION
–66 = 66 MHz

VERSION
DE2 = Clock-doubled with FPU

DEVICE NUMBER/DESCRIPTION
Am486 high-performance CPU

Valid Combination	Comment
Am486DE2-66V8THC	SQFP package
Am486DE2-66V8TGC	PGA package

Valid Combinations

Valid combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations and to check on newly released combinations.

AMD冗

Table of Contents

AMD ◢

FIGURES

TABLES

CONNECTION DIAGRAMS AND PIN DESIGNATIONS

168-Pin Grid Array (PGA) Package

	A	B	C	D	E	F	G	H	J	K	L	M	N	P	Q	R	S	
1	D20	D19	D11	D9	Vss	DP1	Vss	Vss	INC	Vss	Vss	Vss	D2	D0	A31	A28	A27	1
2	D22	D21	D18	D13	Vcc	D8	Vcc	D3	D5	Vcc	D6	Vcc	D1	A29	Vss	A25	A26	2
3	TCK	Vss	CLK	D17	D10	D15	D12	DP2	D16	D14	D7	D4	DP0	A30	A17	Vcc	A23	3
4	D23	Vss	Vcc												A19	Vss	VOLDET	4
5	DP3	Vss	Vcc												A21	A18	A14	5
6	D24	D25	D27												A24	Vcc	Vss	6
7	Vss	Vcc	D26												A22	A15	A12	7
8	D29	D31	D28					Pin Side View							A20	Vcc	Vss	8
9	Vss	Vcc	D30												A16	Vcc	Vss	9
10	INV	$\overline{\text{SMI}}$	SRESET												A13	Vcc	Vss	10
11	Vss	Vcc	$\overline{\text{UP}}$												A9	Vcc	Vss	11
12	$\overline{\text{HITM}}$	$\overline{\text{CACHE}}$	$\overline{\text{SMIACT}}$												A5	A11	Vss	12
13	INC	WB/$\overline{\text{WT}}$	INC												A7	A8	A10	13
14	TDI	TMS	$\overline{\text{FERR}}$												A2	Vcc	Vss	14
15	$\overline{\text{IGNNE}}$	NMI	$\overline{\text{FLUSH}}$	$\overline{\text{A20M}}$	HOLD	$\overline{\text{KEN}}$	$\overline{\text{STPCLK}}$	$\overline{\text{BRDY}}$	$\overline{\text{BE2}}$	$\overline{\text{BE0}}$	PWT	D/$\overline{\text{C}}$	$\overline{\text{LOCK}}$	HLDA	BREQ	A3	A6	15
16	INTR	TDO	RESET	$\overline{\text{BS8}}$	Vcc	$\overline{\text{RDY}}$	Vcc	Vcc	$\overline{\text{BE1}}$	Vcc	Vcc	Vcc	M/$\overline{\text{IO}}$	Vcc	$\overline{\text{PLOCK}}$	$\overline{\text{BLAST}}$	A4	16
17	AHOLD	$\overline{\text{EADS}}$	$\overline{\text{BS16}}$	$\overline{\text{BOFF}}$	Vss	$\overline{\text{BE3}}$	Vss	Vss	PCD	Vss	Vss	Vss	W/$\overline{\text{R}}$	Vss	$\overline{\text{PCHK}}$	INC	L $\overline{\text{ADS}}$	17
	A	B	C	D	E	F	G	H	J	K	L	M	N	P	Q	R	S	

AMD

168-Pin PGA Designations (Functional Grouping)

Address		Data		Control		Test		INC	V_cc	V_ss
Pin Name	Pin No.	Pin Name	Pin No.	Pin Name	Pin No.	Pin Name	Pin No.	Pin No.	Pin No.	Pin No.
A2	Q-14	D0	P-1	$\overline{A20M}$	D-15	TCK	A-3	A-13	B-7	A-7
A3	R-15	D1	N-2	$\overline{ADS}$	S-17	TDI	A-14	C-13	B-9	A-9
A4	S-16	D2	N-1	AHOLD	A-17	TDO	B-16	J-1	B-11	A-11
A5	Q-12	D3	H-2	$\overline{BE0}$	K-15	TMS	B-14	R-17	C-4	B-3
A6	S-15	D4	M-3	$\overline{BE1}$	J-16				C-5	B-4
A7	Q-13	D5	J-2	$\overline{BE2}$	J-15				E-2	B-5
A8	R-13	D6	L-2	$\overline{BE3}$	F-17				E-16	E-1
A9	Q-11	D7	L-3	$\overline{BLAST}$	R-16				G-2	E-17
A10	S-13	D8	F-2	$\overline{BOFF}$	D-17				G-16	G-1
A11	R-12	D9	D-1	$\overline{BRDY}$	H-15				H-16	G-17
A12	S-7	D10	E-3	BREQ	Q-15				K-2	H-1
A13	Q-10	D11	C-1	$\overline{BS8}$	D-16				K-16	H-17
A14	S-5	D12	G-3	$\overline{BS16}$	C-17				L-16	K-1
A15	R-7	D13	D-2	$\overline{CACHE}$	B-12				M-2	K-17
A16	Q-9	D14	K-3	CLK	C-3				M-16	L-1
A17	Q-3	D15	F-3	D/$\overline{C}$	M-15				P-16	L-17
A18	R-5	D16	J-3	DP0	N-3				R-3	M-1
A19	Q-4	D17	D-3	DP1	F-1				R-6	M-17
A20	Q-8	D18	C-2	DP2	H-3				R-8	P-17
A21	Q-5	D19	B-1	DP3	A-5				R-9	Q-2
A22	Q-7	D20	A-1	$\overline{EADS}$	B-17				R-10	R-4
A23	S-3	D21	B-2	$\overline{FERR}$	C-14				R-11	S-6
A24	Q-6	D22	A-2	$\overline{FLUSH}$	C-15				R-14	S-8
A25	R-2	D23	A-4	$\overline{HITM}$	A-12					S-9
A26	S-2	D24	A-6	HLDA	P-15					S-10
A27	S-1	D25	B-6	HOLD	E-15					S-11
A28	R-1	D26	C-7	$\overline{IGNNE}$	A-15					S-12
A29	P-2	D27	C-6	INTR	A-16					S-14
A30	P-3	D28	C-8	INV	A-10					
A31	Q-1	D29	A-8	$\overline{KEN}$	F-15					
		D30	C-9	$\overline{LOCK}$	N-15					
		D31	B-8	M/$\overline{IO}$	N-16					
				NMI	B-15					
				PCD	J-17					
				$\overline{PCHK}$	Q-17					
				$\overline{PLOCK}$	Q-16					
				PWT	L-15					
				$\overline{RDY}$	F-16					
				RESET	C-16					
				$\overline{SMI}$	B-10					
				$\overline{SMIACT}$	C-12					
				SRESET	C-10					
				$\overline{STPCLK}$	G-15					
				$\overline{UP}$	C-11					
				VOLDET	S-4					
				WB/$\overline{WT}$	B-13					
				W/$\overline{R}$	N-17					

Notes:

VOLDET is connected internally to V_{SS}.

INC = Internal No Connect

208-Lead Shrink Quad Flat Pack (SQFP) Package

Top View

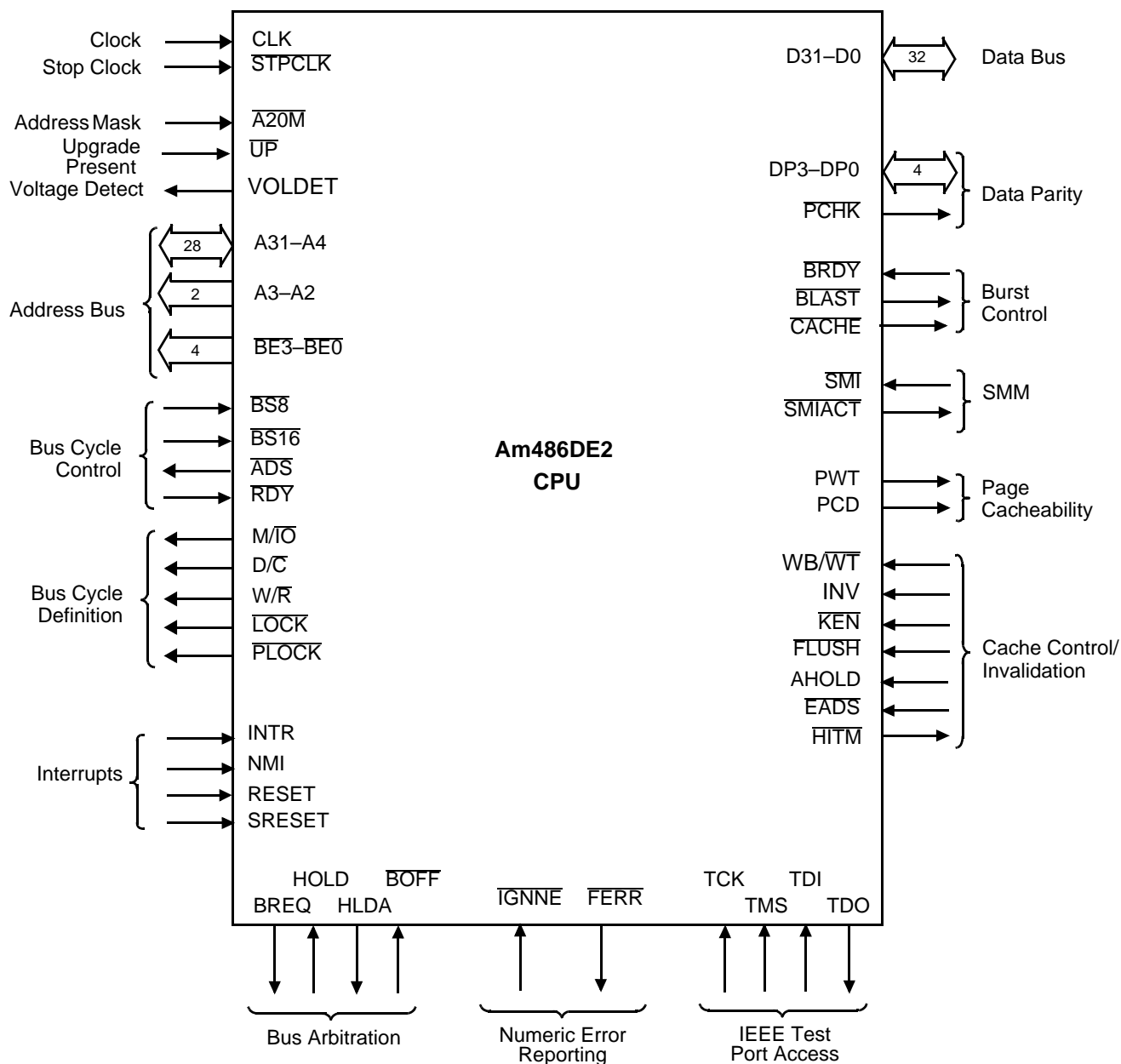Left side (pins 1–52):

Pin	Signal
1	V_{SS}
2	V_{CC}
3	INC
4	$\overline{PCHK}$
5	$\overline{BRDY}$
6	$\overline{BOFF}$
7	$\overline{BS16}$
8	$\overline{BS8}$
9	V_{CC}
10	V_{SS}
11	INC
12	$\overline{RDY}$
13	$\overline{KEN}$
14	V_{CC}
15	V_{SS}
16	HOLD
17	AHOLD
18	TCK
19	V_{CC}
20	V_{CC}
21	V_{SS}
22	V_{CC}
23	V_{CC}
24	CLK
25	V_{CC}
26	HLDA
27	$W/\overline{R}$
28	V_{SS}
29	V_{CC}
30	BREQ
31	$\overline{BE0}$
32	$\overline{BE1}$
33	$\overline{BE2}$
34	$\overline{BE3}$
35	V_{CC}
36	V_{SS}
37	$M/\overline{IO}$
38	V_{CC}
39	$D/\overline{C}$
40	PWT
41	PCD
42	V_{CC}
43	V_{SS}
44	V_{CC}
45	V_{CC}
46	$\overline{EADS}$
47	$\overline{A20M}$
48	RESET
49	$\overline{FLUSH}$
50	INTR
51	NMI
52	V_{SS}

Top side (pins 208–157):

Pin	Signal
208	V_{SS}
207	$\overline{LOCK}$
206	$\overline{PLOCK}$
205	V_{CC}
204	$\overline{BLAST}$
203	$\overline{ADS}$
202	A2
201	V_{SS}
200	V_{CC}
199	V_{SS}
198	V_{CC}
197	A3
196	A4
195	A5
194	$\overline{UP}$
193	A6
192	A7
191	V_{CC}
190	A8
189	V_{SS}
188	V_{CC}
187	A9
186	A10
185	V_{CC}
184	V_{SS}
183	V_{CC}
182	A11
181	V_{SS}
180	A12
179	V_{CC}
178	A13
177	A14
176	V_{CC}
175	V_{SS}
174	A15
173	A16
172	V_{CC}
171	A17
170	V_{CC}
169	V_{SS}
168	TDI
167	TMS
166	A18
165	A19
164	A20
163	V_{CC}
162	V_{SS}
161	A21
160	A22
159	A23
158	A24
157	V_{SS}

Right side (pins 156–105):

Pin	Signal
156	V_{SS}
155	V_{CC}
154	A25
153	A26
152	A27
151	A28
150	V_{CC}
149	A29
148	A30
147	A31
146	V_{SS}
145	DP0
144	D0
143	D1
142	D2
141	D3
140	D4
139	V_{CC}
138	V_{SS}
137	V_{CC}
136	V_{CC}
135	V_{SS}
134	V_{CC}
133	V_{CC}
132	V_{SS}
131	V_{CC}
130	D5
129	D6
128	V_{CC}
127	INC
126	D7
125	DP1
124	D8
123	D9
122	V_{SS}
121	V_{CC}
120	V_{SS}
119	D10
118	D11
117	D12
116	D13
115	V_{SS}
114	V_{CC}
113	D14
112	D15
111	V_{CC}
110	V_{SS}
109	DP2
108	D16
107	V_{SS}
106	V_{CC}
105	V_{SS}

Bottom side (pins 53–104):

Pin	Signal
53	V_{SS}
54	V_{CC}
55	V_{SS}
56	V_{CC}
57	V_{SS}
58	SRESET
59	$\overline{SMIACT}$
60	V_{CC}
61	V_{SS}
62	V_{CC}
63	$\overline{HITM}$
64	WB/WT
65	$\overline{SMI}$
66	$\overline{FERR}$
67	INC
68	TDO
69	V_{CC}
70	$\overline{CACHE}$
71	INV
72	$\overline{IGNNE}$
73	$\overline{STPCLK}$
74	D31
75	D30
76	V_{SS}
77	V_{CC}
78	D29
79	D28
80	V_{CC}
81	V_{SS}
82	V_{CC}
83	D27
84	D26
85	D25
86	V_{CC}
87	D24
88	V_{SS}
89	V_{CC}
90	DP3
91	D23
92	D22
93	D21
94	V_{SS}
95	V_{CC}
96	INC
97	V_{SS}
98	V_{CC}
99	D20
100	D19
101	D18
102	V_{CC}
103	D17
104	V_{SS}

AMD

208-Lead SQFP Designations (Functional Grouping)

Address		Data		Control		Test		INC	V_cc	V_ss
Pin Name	No.	Pin Name	No.	Pin Name	No.	Pin Name	No.	Pin No.	Pin No.	Pin No.
A2	202	D0	144	$\overline{A20M}$	47	TCK	18	3	2	1
A3	197	D1	143	$\overline{ADS}$	203	TDI	168	11	9	10
A4	196	D2	142	AHOLD	17	TDO	68	67	14	15
A5	195	D3	141	$\overline{BE0}$	31	TMS	167	96	19	21
A6	193	D4	140	$\overline{BE1}$	32			127	20	28
A7	192	D5	130	$\overline{BE2}$	33				22	36
A8	190	D6	129	$\overline{BE3}$	34				23	43
A9	187	D7	126	$\overline{BLAST}$	204				25	52
A10	186	D8	124	$\overline{BOFF}$	6				29	53
A11	182	D9	123	$\overline{BRDY}$	5				35	55
A12	180	D10	119	BREQ	30				38	57
A13	178	D11	118	$\overline{BS8}$	8				42	61
A14	177	D12	117	$\overline{BS16}$	7				44	76
A15	174	D13	116	$\overline{CACHE}$	70				45	81
A16	173	D14	113	CLK	24				54	88
A17	171	D15	112	$D/\overline{C}$	39				56	94
A18	166	D16	108	DP0	145				60	97
A19	165	D17	103	DP1	125				62	104
A20	164	D18	101	DP2	109				69	105
A21	161	D19	100	DP3	90				77	107
A22	160	D20	99	$\overline{EADS}$	46				80	110
A23	159	D21	93	$\overline{FERR}$	66				82	115
A24	158	D22	92	$\overline{FLUSH}$	49				86	120
A25	154	D23	91	$\overline{HITM}$	63				89	122
A26	153	D24	87	HLDA	26				95	132
A27	152	D25	85	HOLD	16				98	135
A28	151	D26	84	$\overline{IGNNE}$	72				102	138
A29	149	D27	83	INTR	50				106	146
A30	148	D28	79	INV	71				111	156
A31	147	D29	78	$\overline{KEN}$	13				114	157
		D30	75	$\overline{LOCK}$	207				121	170
		D31	74	$M/\overline{IO}$	37				128	175
				NMI	51				131	181
				PCD	41				133	184
				$\overline{PCHK}$	4				134	189
				$\overline{PLOCK}$	206				136	199
				PWT	40				137	201
				$\overline{RDY}$	12				139	208
				RESET	48				150	
				$\overline{SMI}$	65				155	
				$\overline{SMIACT}$	59				162	
				SRESET	58				163	
				$\overline{STPCLK}$	73				169	
				$\overline{UP}$	194				172	
				$WB/\overline{WT}$	64				176	
				$W/\overline{R}$	27				179	
									183	
									185	
									188	
									191	
									198	
									200	
									205	

Note: INC = Internal No Connect

LOGIC SYMBOL

| | Am486DE2 CPU | |

Clock → CLK
Stop Clock → $\overline{STPCLK}$

Address Mask → $\overline{A20M}$
Upgrade Present → $\overline{UP}$
Voltage Detect ← VOLDET

Address Bus
- 28 A31–A4
- 2 A3–A2
- 4 $\overline{BE3}$–$\overline{BE0}$

Bus Cycle Control
- $\overline{BS8}$
- $\overline{BS16}$
- $\overline{ADS}$
- $\overline{RDY}$

Bus Cycle Definition
- M/$\overline{IO}$
- D/$\overline{C}$
- W/$\overline{R}$
- $\overline{LOCK}$
- $\overline{PLOCK}$

Interrupts
- INTR
- NMI
- RESET
- SRESET

Bus Arbitration
- BREQ
- HOLD
- HLDA
- $\overline{BOFF}$

Numeric Error Reporting
- $\overline{IGNNE}$
- $\overline{FERR}$

IEEE Test Port Access
- TCK
- TMS
- TDI
- TDO

D31–D0 32 → Data Bus

DP3–DP0 4 → Data Parity
$\overline{PCHK}$ →

Burst Control
- $\overline{BRDY}$
- $\overline{BLAST}$
- $\overline{CACHE}$

SMM
- $\overline{SMI}$
- $\overline{SMIACT}$

Page Cacheability
- PWT
- PCD

Cache Control/ Invalidation
- WB/$\overline{WT}$
- INV
- $\overline{KEN}$
- $\overline{FLUSH}$
- AHOLD
- $\overline{EADS}$
- $\overline{HITM}$

AMD

PIN DESCRIPTIONS

The Am486DE2 microprocessor is a new member of the AMD Am486 family, which also includes the Enhanced Am486 and the Am486DX microprocessors.

Like the AMD Enhanced Am486 family, the Am486DE2 adds new signals to those used by the Am486DX processors. These added signals support new processor features and are indicated as *new* in the pin description titles.

Although the Am486DE2 processor is based on and compatible with the Enhanced Am486 microprocessors, it has no support for write-back cache. Because of this, some Am486DE2 signals are supported differently than the signals in either the Enhanced Am486 or the Am486DX microprocessors. These signals are indicated as *modified* in the pin descriptions below.

All other processor signals provide the same functionality as the standard Am486DX processor.

$\overline{\text{A20M}}$
Address Bit 20 Mask (Active-Low Input)

A Low signal on the $\overline{\text{A20M}}$ pin causes the microprocessor to mask address line A20 before performing a lookup to the internal cache, or driving a memory cycle on the bus. Asserting $\overline{\text{A20M}}$ causes the processor to wrap the address at 1 Mbyte, emulating Real mode operation. The signal is asynchronous, but must meet setup and hold times t_{20} and t_{21} for recognition during a specific clock. During normal operation, $\overline{\text{A20M}}$ should be sampled High at the falling edge of RESET.

A31–A2
Address Lines A31-A4 (Inputs/Outputs)
Address Lines A3-A2 (Outputs)

Pins A31–A2 define a physical area in memory or indicate an input/output (I/O) device. Address lines A31–A4 drive addresses into the microprocessor to perform cache line invalidations. Input signals must meet setup and hold times t_{22} and t_{23}. A31–A2 are not driven during bus or address hold.

$\overline{\text{ADS}}$
Address Status (Active-Low Output)

A Low output from this pin indicates that a valid bus cycle definition and address are available on the cycle definition lines and address bus. $\overline{\text{ADS}}$ is driven active by the same clock as the addresses. $\overline{\text{ADS}}$ is active Low and is not driven during bus hold.

AHOLD (Modified)
Address Hold (Input)

The external system may assert AHOLD to perform a cache snoop. In response to the assertion of AHOLD, the microprocessor stops driving the address bus A31–A2 in the next clock. The data bus remains active and data can be transferred for previously issued read or write bus cycles during address hold. AHOLD is recognized even during RESET and $\overline{\text{LOCK}}$. The earliest that AHOLD can be deasserted is two clock cycles after $\overline{\text{EADS}}$ is asserted to start a cache snoop.

$\overline{\text{BE3}}$–$\overline{\text{BE0}}$
Byte Enable (Active-Low Outputs)

The byte enable pins indicate which bytes are enabled and active during read or write cycles. During the first cache fill cycle, however, an external system should ignore these signals and assume that all bytes are active.

- $\overline{\text{BE3}}$ for D31–D24
- $\overline{\text{BE2}}$ for D23–D16
- $\overline{\text{BE1}}$ for D15–D8
- $\overline{\text{BE0}}$ for D7–D0

$\overline{\text{BE3}}$–$\overline{\text{BE0}}$ are active Low and are not driven during bus hold.

$\overline{\text{BLAST}}$ (Modified)
Burst Last (Active-Low Output)

Burst Last goes Low to tell the CPU that the next $\overline{\text{BRDY}}$ signal completes the burst bus cycle. $\overline{\text{BLAST}}$ is active for both burst and non-burst cycles. $\overline{\text{BLAST}}$ is active Low and is not driven during a bus hold.

$\overline{\text{BOFF}}$
Back Off (Active-Low Input)

This input signal forces the microprocessor to float all pins normally floated during hold, but HLDA is not asserted in response to $\overline{\text{BOFF}}$. $\overline{\text{BOFF}}$ has higher priority than $\overline{\text{RDY}}$ or $\overline{\text{BRDY}}$; if both are returned in the same clock, $\overline{\text{BOFF}}$ takes effect. The microprocessor remains in bus hold until $\overline{\text{BOFF}}$ goes High. If a bus cycle is in progress when BOFF is asserted, the cycle restarts. $\overline{\text{BOFF}}$ must meet setup and hold times t_{18} and t_{19} for proper operation. $\overline{\text{BOFF}}$ has an internal weak pull-up.

$\overline{\text{BRDY}}$
Burst Ready Input (Active-Low Input)

The $\overline{\text{BRDY}}$ signal performs the same function during a burst cycle that $\overline{\text{RDY}}$ performs during a non-burst cycle. $\overline{\text{BRDY}}$ indicates that the external system has presented valid data in response to a read, or that the external system has accepted data in response to write. $\overline{\text{BRDY}}$

is ignored when the bus is idle and at the end of the first clock in a bus cycle. $\overline{BRDY}$ is sampled in the second and subsequent clocks of a burst cycle. The data presented on the data bus is strobed into the microprocessor when $\overline{BRDY}$ is sampled active. If $\overline{RDY}$ is returned simultaneously with $\overline{BRDY}$, $\overline{BRDY}$ is ignored and the cycle is converted to a non-burst cycle. $\overline{BRDY}$ is active Low and has a small pull-up resistor, and must satisfy the setup and hold times t_{16} and t_{17}.

BREQ
Internal Cycle Pending (Output)
BREQ indicates that the microprocessor has generated a bus request internally, whether or not the microprocessor is driving the bus. BREQ is active High and is floated only during Three-state Test mode. (See $\overline{FLUSH}$.)

$\overline{BS}8/\overline{BS}16$
Bus Size 8 (Active-Low Input)
Bus Size 16 (Active-Low Input)
The $\overline{BS8}$ and $\overline{BS16}$ signals allow the processor to operate with 8-bit and 16-bit I/O devices by running multiple bus cycles to respond to data requests: four for 8-bit devices, and two for 16-bit devices. The bus sizing pins are sampled every clock. The microprocessor samples the pins every clock before $\overline{RDY}$ to determine the appropriate bus size for the requesting device. The signals are active Low input with internal pull-up resistors, and must satisfy setup and hold times t_{14} and t_{15} for correct operation. Bus sizing is not permitted during copy-back or write-back operation. $\overline{BS8}$ and $\overline{BS16}$ are ignored during copy-back or write-back cycles.

$\overline{CACHE}$ (New)
Internal Cacheability (Active-Low Output)
In Write-through mode, this signal always floats.

CLK (Modified)
Clock (Input)
The CLK input provides the basic microprocessor timing signal. All external timing parameters are specified with respect to the rising edge of CLK. The clock signal passes through an internal Phase-Lock Loop (PLL). The CLK input is multiplied by two by an internal phase lock loop (PLL) to generate the internal operating frequency.

D31–D0
Data Lines (Inputs/Outputs)
Lines D31–D0 define the data bus. The signals must meet setup and hold times t_{22} and t_{23} for proper read operations. These pins are driven during the second and subsequent clocks of write cycles.

D/$\overline{C}$
Data/Control (Output)
This bus cycle definition pin distinguishes memory and I/O data cycles from control cycles. The control cycles are:

■ Interrupt Acknowledge

■ Halt/Special Cycle

■ Code Read (instruction fetching)

DP3–DP0
Data Parity (Inputs/Outputs)
Data parity is generated on all write data cycles with the same timing as the data driven by the microprocessor. Even parity information must be driven back into the microprocessor on the data parity pins with the same timing as read information to ensure that the processor uses the correct parity check. The signals read on these pins do not affect program execution. Input signals must meet setup and hold times t_{22} and t_{23}. DP3–DP0 should be connected to V_{CC} through a pull-up resistor in systems not using parity. DP3–DP0 are active High and are driven during the second and subsequent clocks of write cycles.

$\overline{EADS}$ (Modified)
External Address Strobe (Active-Low Input)
This signal indicates that a valid external address has been driven on the address pins A31–A4 of the microprocessor to be used for a cache snoop. This signal is recognized while the processor is in hold (HLDA is driven active), while forced off the bus with the $\overline{BOFF}$ input, or while AHOLD is asserted. The microprocessor ignores $\overline{EADS}$ at all other times. $\overline{EADS}$ is not recognized during the clock after $\overline{ADS}$, nor during the clock after a valid assertion of $\overline{EADS}$. Snoops to the on-chip cache must be completed before another snoop cycle is initiated. Table 1 describes $\overline{EADS}$ when first sampled. $\overline{EADS}$ can be asserted every other clock cycle as long as the hold remains active and $\overline{HITM}$ remains inactive. INV is sampled in the same clock period that $\overline{EADS}$ is asserted. $\overline{EADS}$ has an internal weak pull-up.

Table 1. $\overline{EADS}$ Sample Time

Trigger	$\overline{EADS}$ First Sampled
AHOLD	Second clock after AHOLD asserted
HOLD	First clock after HLDA asserted
$\overline{BOFF}$	Second clock after $\overline{BOFF}$ asserted

Note: The triggering signal (AHOLD, HOLD, or $\overline{BOFF}$) must remain active for at least 1 clock after $\overline{EADS}$ to ensure proper operation.

AMD❚

This signal is used in cache snooping. The Am486DE2 processor does not support write-back cache. $\overline{\text{EADS}}$ has a weak internal pull-up, which disables this pin.

$\overline{\text{FERR}}$
Floating-Point Error (Active-Low Output)

Driven active when a floating-point error occurs, $\overline{\text{FERR}}$ is similar to the $\overline{\text{ERROR}}$ pin on a 387 math coprocessor. $\overline{\text{FERR}}$ is included for compatibility with systems using DOS-type floating-point error reporting. $\overline{\text{FERR}}$ is active Low and is not floated during bus hold, except during Three-state Test mode (see $\overline{\text{FLUSH}}$).

$\overline{\text{FLUSH}}$ (Modified)
Cache Flush (Active-Low Input)

In Write-through mode, $\overline{\text{FLUSH}}$ invalidates the cache without issuing a special bus cycle. $\overline{\text{FLUSH}}$ is an active Low input that needs to be asserted only for one clock. $\overline{\text{FLUSH}}$ is asynchronous, but setup and hold times t_{20} and t_{21} must be met for recognition in any specific clock. Sampling $\overline{\text{FLUSH}}$ Low in the clock before the falling edge of RESET causes the microprocessor to enter Three-state Test mode.

$\overline{\text{HITM}}$ (New)
Hit Modified Line (Active-Low Output)

In Write-through mode, $\overline{\text{HITM}}$ floats at all times.

HLDA
Hold Acknowledge (Output)

The HLDA signal is activated in response to a hold request presented on the HOLD pin. HLDA indicates that the microprocessor has given the bus to another local bus master. HLDA is driven active in the same clock in which the microprocessor floats its bus. HLDA is driven inactive when leaving bus hold. HLDA is active High and remains driven during bus hold. HLDA is floated only during Three-state Test mode. (See $\overline{\text{FLUSH}}$.)

HOLD
Bus Hold Request (Input)

HOLD gives control of the microprocessor bus to another bus master. In response to HOLD going active, the microprocessor floats most of its output and input/output pins. HLDA is asserted after completing the current bus cycle, burst cycle, or sequence of locked cycles. The microprocessor remains in this state until HOLD is deasserted. HOLD is active High and does not have an internal pull-down resistor. HOLD must satisfy setup and hold times t_{18} and t_{19} for proper operation.

$\overline{\text{IGNNE}}$
Ignore Numeric Error (Active-Low Input)

When this pin is asserted, the Am486DE2 microprocessor will ignore a numeric error and continue executing non-control floating-point instructions. When $\overline{\text{IGNNE}}$ is deasserted, the Am486DE2 microprocessor will freeze on a non-control floating-point instruction if a previous floating-point instruction caused an error. $\overline{\text{IGNNE}}$ has no effect when the NE bit in Control Register 0 is set. $\overline{\text{IGNNE}}$ is active Low and is provided with a small internal pull-up resistor. $\overline{\text{IGNNE}}$ is asynchronous but must meet setup and hold times t_{20} and t_{21} to ensure recognition in any specific clock.

INTR
Maskable Interrupt (Input)

When asserted, this signal indicates that an external interrupt has been generated. If the internal interrupt flag is set in EFLAGS, active interrupt processing is initiated. The microprocessor generates two locked interrupt acknowledge bus cycles in response to the INTR pin going active. INTR must remain active until the interrupt acknowledges have been performed to ensure that the interrupt is recognized. INTR is active High and is not provided with an internal pull-down resistor. INTR is asynchronous, but must meet setup and hold times t_{20} and t_{21} for recognition in any specific clock.

INV (New)
Invalidate (Input)

The external system asserts INV to invalidate the cache-line state when an external bus master proposes a write. It is sampled together with A31–A4 during the clock in which $\overline{\text{EADS}}$ is active. INV has an internal weak pull-up. INV is ignored in Write-through mode.

$\overline{\text{KEN}}$
Cache Enable (Active-Low Input)

$\overline{\text{KEN}}$ determines whether the current cycle is cacheable. When the microprocessor generates a cacheable cycle and $\overline{\text{KEN}}$ is active one clock before $\overline{\text{RDY}}$ or $\overline{\text{BRDY}}$ during the first transfer of the cycle, the cycle becomes a cache-line-fill cycle. Returning $\overline{\text{KEN}}$ active one clock before $\overline{\text{RDY}}$ during the last read in the cache line fill causes the line to be placed in the on-chip cache. $\overline{\text{KEN}}$ is active Low and is provided with a small internal pull-up resistor. $\overline{\text{KEN}}$ must satisfy setup and hold times t_{14} and t_{15} for proper operation.

$\overline{\text{LOCK}}$
Bus Lock (Active-Low Output)

A Low output on this pin indicates that the current bus cycle is locked. The microprocessor ignores HOLD when $\overline{\text{LOCK}}$ is asserted (although it does acknowledge AHOLD and $\overline{\text{BOFF}}$). $\overline{\text{LOCK}}$ goes active in the first clock of the first locked bus cycle and goes inactive after the

last clock of the last locked bus cycle. The last locked cycle ends when $\overline{RDY}$ is returned. $\overline{LOCK}$ is active Low and is not driven during bus hold. Locked read cycles are not transformed into cache fill cycles if $\overline{KEN}$ is active.

M/$\overline{IO}$
Memory/IO (Output)
A High output indicates a memory cycle. A Low output indicates an I/O cycle.

NMI
Non-Maskable Interrupt (Input)
A High NMI input signal indicates that an external non-maskable interrupt has occurred. NMI is rising-edge sensitive. NMI must be held Low for at least four CLK periods before this rising edge. The NMI input does not have an internal pull-down resistor. The NMI input is asynchronous, but must meet setup and hold times t_{20} and t_{21} for recognition in any specific clock.

PCD
Page Cache Disable (Output)
This pin reflects the state of the PCD bit in the page table entry or page directory entry (programmable through the PCD bit in CR3). If paging is disabled, the CPU ignores the PCD bit and drives the PCD output Low. PCD has the same timing as the cycle definition pins (M/$\overline{IO}$, D/$\overline{C}$, and W/$\overline{R}$). PCD is active High and is not driven during bus hold. PCD is masked by the Cache Disable Bit (CD) in Control Register 0 (CR0).

$\overline{PCHK}$
Parity Status (Active-Low Output)
Parity status is driven on the $\overline{PCHK}$ pin the clock after $\overline{RDY}$ for read operations. The parity status reflects data sampled at the end of the previous clock. A Low $\overline{PCHK}$ indicates a parity error. Parity status is checked only for enabled bytes as is indicated by the byte enable and bus size signals. $\overline{PCHK}$ is valid only in the clock immediately after read data is returned to the microprocessor; at all other times $\overline{PCHK}$ is inactive High. $\overline{PCHK}$ is floated only during Three-state Test mode. (See $\overline{FLUSH}$.)

$\overline{PLOCK}$ (Modified)
Pseudo-Lock (Active-Low Output)
When $\overline{PLOCK}$ is asserted in Write-through mode, it indicates that the current bus transaction requires more than one bus cycle. Examples of such operations are segment table descriptor reads (8 bytes) and cache line fills (16 bytes). The microprocessor drives $\overline{PLOCK}$ active until the addresses for the last bus cycle of the transaction have been driven, whether or not $\overline{RDY}$ or $\overline{BRDY}$ is returned. $\overline{PLOCK}$ is a function of the $\overline{BS8}$, $\overline{BS16}$, and $\overline{KEN}$ inputs. $\overline{PLOCK}$ should be sampled on the clock when $\overline{RDY}$ is returned. $\overline{PLOCK}$ is active Low and is not driven during bus hold.

PWT
Page Write-Through (Output)
This pin reflects the state of the PWT bit in the page table entry or page directory entry (programmable through the PWT bit in CR3). If paging is disabled, the CPU ignores the PWT bit and drives the PWT output Low. PWT has the same timing as the cycle definition pins (M/$\overline{IO}$, D/$\overline{C}$, and W/$\overline{R}$). PWT is active High and is not driven during bus hold.

$\overline{RDY}$
Non-Burst Ready (Active-Low Input)
A Low input on this pin indicates that the current bus cycle is complete, that is, either the external system has presented valid data on the data pins in response to a read, or the external system has accepted data from the microprocessor in response to a write. $\overline{RDY}$ is ignored when the bus is idle and at the end of the bus cycle's first clock. $\overline{RDY}$ is active during address hold. Data can be returned to the processor while AHOLD is active. $\overline{RDY}$ is active Low and does not have an internal pull-up resistor. $\overline{RDY}$ must satisfy setup and hold times t_{16} and t_{17} for proper chip operation.

RESET
Reset (Input)
RESET forces the microprocessor to initialize. The microprocessor cannot begin instruction execution of instructions until at least 1 ms after V_{CC} and CLK have reached their proper DC and AC specifications. To ensure proper microprocessor operation, the RESET pin should remain active during this time. RESET is active High. RESET is asynchronous, but must meet setup and hold times t_{20} and t_{21} to ensure recognition on any specific clock.

$\overline{SMI}$ (New)
SMM Interrupt (Active-Low Input)
A Low signal on the $\overline{SMI}$ pin signals the processor to enter System Management Mode (SMM). $\overline{SMI}$ is the highest-level processor interrupt. The $\overline{SMI}$ signal is recognized on an instruction boundary, similar to the NMI and INTR signals. $\overline{SMI}$ is sampled on every rising clock edge. $\overline{SMI}$ is a falling-edge sensitive input. Recognition of $\overline{SMI}$ is guaranteed in a specific clock if it is asserted synchronously and meets the setup and hold times. If $\overline{SMI}$ is asserted asynchronously, it must go High for a minimum of two clocks before going Low, and it must remain Low for at least two clocks to guarantee recognition. When the CPU recognizes $\overline{SMI}$, it enters SMM before executing the next instruction and saves internal registers in SMM space.

AMD

$\overline{\text{SMIACT}}$ (New)
SMM Interrupt Active (Active-Low Output)

$\overline{\text{SMIACT}}$ goes Low in response to $\overline{\text{SMI}}$. It indicates that the processor is operating under SMM control. $\overline{\text{SMIACT}}$ remains Low until the processor receives a RESET signal or executes the Resume instruction (RSM) to leave SMM. This signal is always driven. It does not float during bus HOLD or $\overline{\text{BOFF}}$.

Note: Do not use SRESET to exit from SMM. The system should block SRESET during SMM.

SRESET (New)
Soft Reset (Input)

The CPU samples SRESET on every rising clock edge. If SRESET is sampled active, the SRESET sequence begins on the next instruction boundary. SRESET resets the processor, but, unlike RESET, does not cause it to sample $\overline{\text{UP}}$ or WB/$\overline{\text{WT}}$, or affect the FPU, cache, CD and NW bits in CR0, and SMBASE. SRESET is asynchronous and must meet the same timing as RESET.

$\overline{\text{STPCLK}}$ (New)
Stop Clock (Active-Low Input)

A Low input signal indicates a request has been made to turn off the CLK input. When the CPU recognizes a $\overline{\text{STPCLK}}$, the processor:

■ stops execution on the next instruction boundary (unless superseded by a higher priority interrupt)

■ empties all internal pipelines and write buffers

■ generates a Stop Grant acknowledge bus cycle

$\overline{\text{STPCLK}}$ is active Low and has an internal pull-up resistor. $\overline{\text{STPCLK}}$ is asynchronous, but it must meet setup and hold times t_{20} and t_{21} to ensure recognition in any specific clock. $\overline{\text{STPCLK}}$ must remain active until the Stop Clock special bus cycle is issued and the system returns either $\overline{\text{RDY}}$ or $\overline{\text{BRDY}}$.

TCK
Test Clock (Input)

Test Clock provides the clocking function for the JTAG boundary scan feature. TCK clocks state information and data into the component on the rising edge of TCK on TMS and TDI, respectively. Data is clocked out of the component on the falling edge of TCK on TDO.

TDI
Test Data Input (Input)

TDI is the serial input that shifts JTAG instructions and data into the tested component. TDI is sampled on the rising edge of TCK during the SHIFT-IR and the SHIFT-DR TAP (Test Access Port) controller states. During all other TAP controller states, TDI is ignored. TDI uses an internal weak pull-up.

TDO
Test Data Output (Output)

TDO is the serial output that shifts JTAG instructions and data out of the component. TDO is driven on the falling edge of TCK during the SHIFT-IR and SHIFT-DR TAP controller states. Otherwise, TDO is three-stated.

TMS
Test Mode Select (Input)

TMS is decoded by the JTAG TAP to select the operation of the test logic. TMS is sampled on the rising edge of TCK. To guarantee deterministic behavior of the TAP controller, the TMS pin has an internal pull-up resistor.

$\overline{\text{UP}}$
Upgrade Present (Input)

The processor samples the Upgrade Present ($\overline{\text{UP}}$) pin in the clock before the falling edge of RESET. If it is Low, the processor three-states its outputs immediately. $\overline{\text{UP}}$ must remain asserted to keep the processor inactive. The pin uses an internal pull-up resistor.

VOLDET (New, 168-Pin PGA Package only)
Voltage Detect (Output)

VOLDET provides an external signal to allow the system to determine the CPU input power level (3 V or 5 V). For the Am486DE2, the pin ties internally to V_{SS}.

WB/$\overline{\text{WT}}$ (New)
Write-Back/Write-Through (Input)

WB/$\overline{\text{WT}}$ is sampled Low at RESET, and all cache-line fills are write-through. WB/$\overline{\text{WT}}$ has an internal weak pull-down. This pin should be tied Low for the Am486DE2 microprocessor.

W/$\overline{\text{R}}$
Write/Read (Output)

A High output indicates a write cycle. A Low output indicates a read cycle.

Note: The Am486DE2 microprocessor does not use the V_{CC5} pin used by some 3-V, clock-tripled, 486-based processors. The corresponding pin on the Am486DE2 microprocessor is an Internal No Connect (INC).

FUNCTIONAL DESCRIPTION

Note: This Am486DE microprocessor does not support Write-back mode. If you are designing in a shared-memory system or using cache coherency (including snooping and locked accesses), use one of the Am486DE products that supports write back.

Overview

The Am486DE2 microprocessor uses a 32-bit architecture with on-chip memory management and cache memory units. The instruction set includes the complete 486 microprocessor instruction set, along with extensions to serve the new extended applications. All software written for the 486 microprocessor and previous members of the x86 architectural family can run on the Am486DE2 microprocessor without modification.

The on-chip Memory Management Unit (MMU) is completely compatible with the 486 MMU. The MMU includes a segmentation unit and a paging unit. Segmentation allows management of the logical address space by providing easy data and code relocatability and efficient sharing of global resources. The paging mechanism operates beneath segmentation and is transparent to the segmentation process. Paging is optional and can be disabled by system software. Each segment can be divided into one or more 4-Kbyte segments. To implement a virtual memory system, the Am486DE2 microprocessor supports full restartability for all page and segment faults.

Memory

Memory is organized into one or more variable length segments, each up to 4 Gbyte (2^{32} bytes). A segment can have attributes associated with it, including its location, size, type (e.g., stack, code, or data), and protection characteristics. Each task on a microprocessor can have a maximum of 16,381 segments, each up to 4 Gbyte. Thus, each task has a maximum of 64 Tbyte of virtual memory.

The segmentation unit provides four levels of protection for isolating and protecting applications and the operating system from each other. The hardware-enforced protection allows high-integrity system designs.

Modes of Operation

The Am486DE2 microprocessor has four modes of operation: Real Address mode (Real mode), Virtual 8086 Address mode (Virtual mode), Protected Address mode (Protected mode), and System Management mode (SMM).

Real Mode

In Real mode, the Am486DE2 microprocessor operates as a fast 8086. Real mode is required primarily to set up the processor for Protected mode operation.

Virtual Mode

In Virtual mode, the processor appears to be in Real mode, but can use the extended memory accessing of Protected mode.

Protected Mode

Protected mode provides access to the sophisticated memory management paging and privilege capabilities of the processor.

System Management Mode

SMM is a special operating mode described in detail in "System Management Mode" on page 22.

Write-Through Cache Architecture

The Am486DE2 microprocessor supports the standard 486DX-type write-through cache architecture, which is characterized by the following:

■ External read accesses are placed in the cache if they meet proper caching requirements.

■ Subsequent reads to the data in the cache are made if the address is stored in the cache tag array.

■ Write operations to a valid address in the cache are updated in the cache *and* to external memory. This data writing technique is called *write-through*.

The write-through cache implementation forces all writes to flow through to the external bus and back to main memory. Consequently, the write-through cache generates a large amount of bus traffic on the external data bus.

Cache Replacement Description

The cache-line-replacement algorithm uses the standard Am486 CPU pseudo LRU (least recently used) strategy. When a line must be placed in the internal cache, the microprocessor first checks to see if there is an invalid line available in the set. If no invalid line is available, the LRU algorithm replaces the least-recently used cache line in the four-way set with the new cache line. If the cache line for replacement is modified, the modified cache line is placed into the copy-back buffer for copying back to external memory, and the new cache line is placed into the cache. This copy-back ensures that the external memory is updated with the modified data upon replacement.

Memory Configuration

In computer systems, memory regions require specific caching and memory write methods. For example, some memory regions are noncacheable while others are cacheable but are write-through. To allow maximum memory configuration, the microprocessor supports

AMD

specific memory region requirements. All bus masters, such as DMA controllers, must reflect all data transfers on the microprocessor local bus so that the microprocessor can respond appropriately.

Cacheability

The Am486DE2 processor caches data based on the state of the CD and NW bits in CR0, in conjunction with the $\overline{KEN}$ signal, at the time of a burst read access from memory. When the WB/$\overline{WT}$ signal is Low during the first $\overline{BRDY}$, $\overline{KEN}$ meets the standard setup and hold requirements, and the four 32-bit doublewords are placed in the cache. However, all cacheable accesses in this mode are considered write-through.

Note: The CD bit in CR0 enables (0) or disables (1) the internal cache. The NW bit in CR0 enables (0) or disables (1) write-through and snooping cycles. RESET sets CD and NW to 1. Unlike RESET, however, SRESET does not invalidate the cache nor does it modify the values of CD and NW in CR0.

Write-Through

When the WB/$\overline{WT}$ signal is Low during the first $\overline{BRDY}$ of the cache line read access, the cache line is considered a write-through access. Therefore, all writes to this location in the cache are reflected on the external bus, even if the cache line is write protected.

CLOCK CONTROL

Clock Generation

The Am486DE2 CPU is driven by a 1X clock that relies on phased-lock loop (PLL) to generate the two internal clock phases: phase one and phase two. The rising edge of CLK corresponds to the start of phase one (ph1). All external timing parameters are specified relative to the rising edge of CLK.

Stop Clock

The Am486DE2 CPU also provides an interrupt mechanism, $\overline{STPCLK}$, that allows system hardware to control the power consumption of the CPU by stopping the internal clock to the CPU core in a sequenced manner. The first low-power state is called the Stop Grant state. If the CLK input is completely stopped, the CPU enters into the Stop Clock state (the lowest power state). When the CPU recognizes a $\overline{STPCLK}$ interrupt, the processor:

■ stops execution on the next instruction boundary (unless superseded by a higher priority interrupt)

■ waits for completion of cache flush

■ stops the pre-fetch unit

■ empties all internal pipelines and write buffers

■ generates a Stop Grant bus cycle

■ stops the internal clock

At this point the CPU is in the Stop Grant state

The CPU cannot respond to a $\overline{STPCLK}$ request from an HLDA state because it cannot empty the write buffers and, therefore, cannot generate a Stop Grant cycle. The rising edge of $\overline{STPCLK}$ signals the CPU to return to pro-

gram execution at the instruction following the interrupted instruction. Unlike the normal interrupts (INTR and NMI), $\overline{STPCLK}$ does not initiate interrupt acknowledge cycles or interrupt table reads.

External Interrupts in Order of Priority

In Write-through mode, the priority order of external interrupts is:

1. RESET/SRESET
2. FLUSH
3. SMI
4. NMI
5. INTR
6. STPCLK

$\overline{STPCLK}$ is active Low and has an internal pull-up resistor. $\overline{STPCLK}$ is asynchronous, but setup and hold times must be met to ensure recognition in any specific clock. $\overline{STPCLK}$ must remain active until the Stop Grant special bus cycle is asserted and the system responds with either $\overline{RDY}$ or $\overline{BRDY}$. When the CPU enters the Stop Grant state, the internal pull-up resistor is disabled, reducing the CPU power consumption. The $\overline{STPCLK}$ input must be driven High (not floated) to exit the Stop Grant state. $\overline{STPCLK}$ must be deasserted for a minimum of five clocks after $\overline{RDY}$ or $\overline{BRDY}$ is returned active for the Stop Grant bus cycle before being asserted again. There are two regions for the Low-Power-mode supply current:

1. *Low Power:* Stop Grant state (fast wake-up, frequency- and voltage-dependent)
2. *Lowest Power:* Stop Clock state (slow wake-up, voltage-dependent)

Stop Grant Bus Cycle

The processor drives a special Stop Grant bus cycle to the bus after recognizing the $\overline{\text{STPCLK}}$ interrupt. This bus cycle is the same as the HALT cycle used by a standard Am486 microprocessor, with the exception that the Stop Grant bus cycle drives the value 0000 0010h on the address pins.

■ $M/\overline{IO} = 0$

■ $D/\overline{C} = 0$

■ $W/\overline{R} = 1$

■ Address Bus = 0000 0010h ($A_4 = 1$)

■ $\overline{\text{BE}}3–\overline{\text{BE}}0 = 1011$

■ Data bus = undefined

The system hardware must acknowledge this cycle by returning $\overline{\text{RDY}}$ or $\overline{\text{BRDY}}$, or the processor will not enter the Stop Grant state (see Figure 1). The latency between a $\overline{\text{STPCLK}}$ request and the Stop Grant bus cycle depends on the current instruction, the amount of data in the CPU write buffers, and the system memory performance.

Pin State During Stop Grant

Table 2 shows the pin states during Stop Grant bus states. During the Stop Grant state, most output and input/output signals of the microprocessor maintain the level they held when entering the Stop Grant state. The data and data parity signals are three-stated. In response to HOLD being driven active during the Stop Grant state (when the CLK input is running), the CPU generates HLDA and three-states all output and input/output signals that are three-stated during the HOLD/HLDA state. After HOLD is deasserted, all signals return to the same state they were before the HOLD/HLDA sequence.

To achieve the lowest possible power consumption during the Stop Grant state, the system designer must ensure the input signals with pull-up resistors are not driven Low, and the input signals with pull-down resistors are not driven High.

All inputs except data bus pins must be driven to the power supply rails to ensure the lowest possible current consumption during Stop Grant or Stop Clock modes. For compatibility, data pins must be driven Low to achieve the lowest possible power consumption.

Table 2. Pin State During Stop Grant Bus State

Signal	Type	State
A3–A2	O	Previous State
A31–A4	I/O	Previous State
D31–D0	I/O	Floated
$\overline{\text{BE}}3–\overline{\text{BE}}0$	O	Previous State
DP3–DP0	I/O	Floated
$W/\overline{R}, D/\overline{C}, M/\overline{IO}, \overline{\text{CACHE}}$	O	Previous State
$\overline{\text{ADS}}$	O	Inactive
$\overline{\text{LOCK}}, \overline{\text{PLOCK}}$	O	Inactive
BREQ	O	Previous State
HLDA	O	As per HOLD
$\overline{\text{BLAST}}$	O	Previous State
$\overline{\text{FERR}}$	O	Previous State
$\overline{\text{PCHK}}$	O	Previous State
$\overline{\text{SMIACT}}$	O	Previous State
$\overline{\text{HITM}}$	O	Previous State
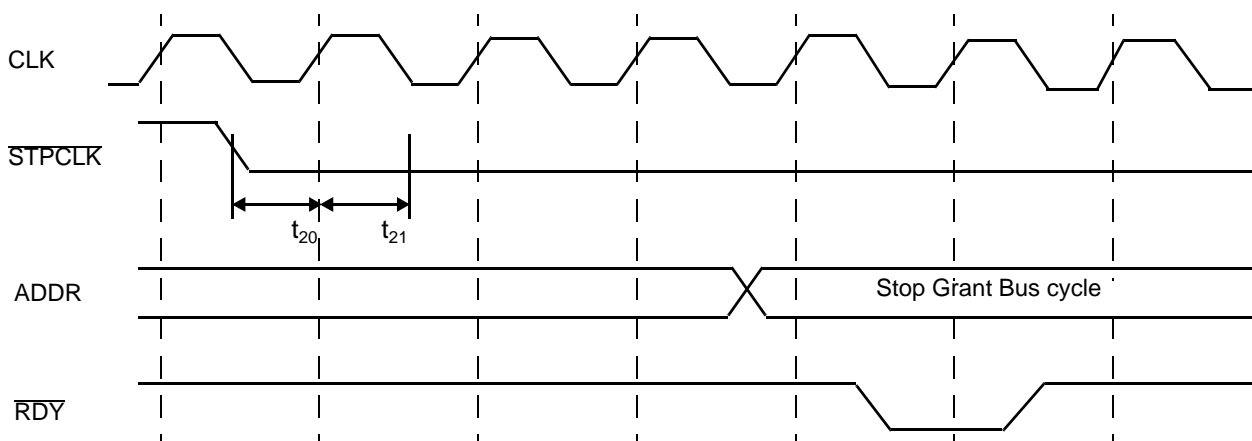
Figure 1. Entering Stop Grant State

AMD🗗

Clock Control State Diagram

Figure 2 shows the state transitions during a Stop Clock cycle.

Normal State

This is the normal operating state of the CPU. While in the normal state, the CLK input can be dynamically changed within the specified CLK period stability limits.

Stop Grant State

The Stop Grant state provides a low-power state that can be entered by simply asserting the external $\overline{STPCLK}$ interrupt pin. When the Stop Grant bus cycle has been placed on the bus, and either $\overline{RDY}$ or $\overline{BRDY}$ is returned, the CPU is in this state. The CPU returns to the normal execution state 10–20 clock periods after $\overline{STPCLK}$ has been deasserted.

While in the Stop Grant state, the pull-up resistors on $\overline{STPCLK}$ and $\overline{UP}$ are disabled internally. The system must continue to drive these inputs to the state they were in immediately before the CPU entered the Stop Grant state. For minimum CPU power consumption, all other input pins should be driven to their inactive level while the CPU is in the Stop Grant state.

A RESET or SRESET brings the CPU from the Stop Grant state to the Normal state. The CPU recognizes the inputs required for cache invalidations (HOLD, AHOLD, $\overline{BOFF}$, and $\overline{EADS}$), as explained later. The CPU does not recognize any other inputs while in the Stop Grant state. Input signals to the CPU are not recognized until 1 clock after $\overline{STPCLK}$ is deasserted (see Figure 3).

While in the Stop Grant state, the CPU does not recognize transitions on the interrupt signals ($\overline{SMI}$, NMI, and INTR). Driving an active edge on either $\overline{SMI}$ or NMI does not guarantee recognition and service of the interrupt request following exit from the Stop Grant state.

However, if one of the interrupt signals ($\overline{SMI}$, NMI or INTR) is driven active while the CPU is in the Stop Grant state, and held active for at least one CLK after $\overline{STPCLK}$ is deasserted, the corresponding interrupt will be ser-

viced. The Am486DE2 processor requires INTR to be held active until the CPU issues an interrupt acknowledge cycle to guarantee recognition. This condition also applies to the existing Am486 CPUs.

In the Stop Grant state, the system can stop or change the CLK input. When the clock stops, the CPU enters the Stop Clock state. The CPU returns to the Stop Grant state immediately when the CLK input is restarted. You must hold the $\overline{STPCLK}$ input Low until a stabilized frequency has been maintained for at least 1 ms to ensure that the PLL has had sufficient time to stabilize.

The CPU generates a Stop Grant bus cycle when entering the state from the Normal or the Auto Halt Power-Down state. When the CPU enters the Stop Grant state from the Stop Clock state or the Stop Clock Snoop state, the CPU does not generate a Stop Grant bus cycle.

Stop Clock State

Stop Clock state is entered from the Stop Grant state by stopping the CLK input (either logic High or logic Low). None of the CPU input signals should change state while the CLK input is stopped. Any transition on an input signal (except INTR) before the CPU has returned to the Stop Grant state may result in unpredictable behavior. If INTR goes active while the CLK input is stopped, and stays active until the CPU issues an interrupt acknowledge bus cycle, it is serviced in the normal manner. System design must ensure the CPU is in the correct state prior to asserting cache invalidation or interrupt signals to the CPU.

Auto Halt Power-Down State

A HALT instruction causes the CPU to enter the Auto Halt Power-Down state. The CPU issues a normal HALT bus cycle, and only transitions to the Normal state when INTR, NMI, $\overline{SMI}$, RESET, or SRESET occurs.

The system can generate a $\overline{STPCLK}$ while the CPU is in the Auto Halt Power-Down state. The CPU generates a Stop Grant bus cycle when it enters the Stop Grant state from the HALT state. When the system deasserts the $\overline{STPCLK}$ interrupt, the CPU returns execution to the HALT state. The CPU generates a new HALT bus cycle when it reenters the HALT state from the Stop Grant state.

SRESET FUNCTION

The Am486DE2 microprocessor supports a soft reset function through the SRESET pin. SRESET forces the processor to begin execution in a known state. The processor state after SRESET is the same as after RESET except that the internal caches, CD and NW in CR0, write buffers, SMBASE registers, and floating-point registers retain the values they had prior to SRESET, and

cache snooping is allowed. The processor starts execution at physical address FFFFFFF0h. SRESET can be used to help performance for DOS extenders written for the 80286 processor. SRESET provides a method to switch from Protected to Real mode while maintaining the internal caches, CR0, and the FPU state. SRESET may not be used in place of RESET after power-up.
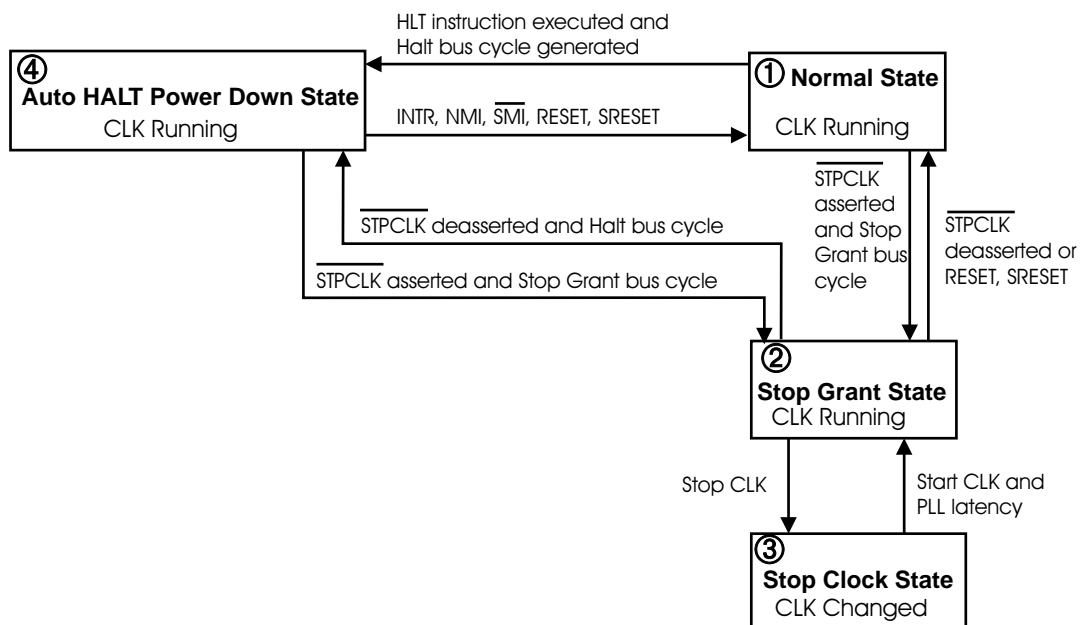
Figure 2. Stop Clock State Machine
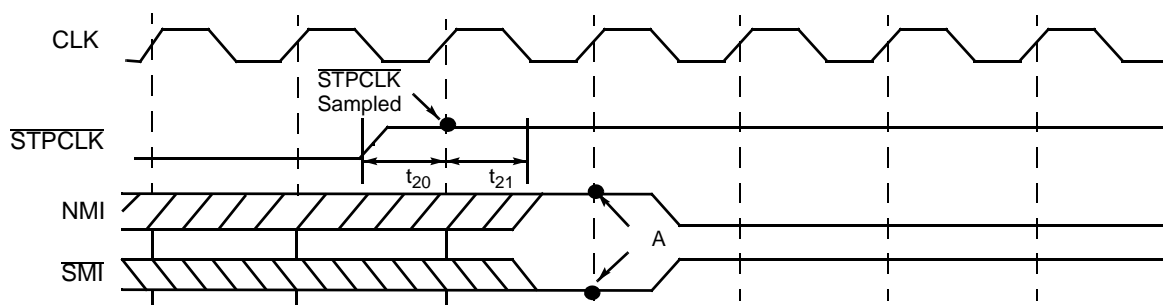
Note: A = Earliest time at which NMI or $\overline{\text{SMI}}$ is recognized.

Figure 3. Recognition of Inputs when Exiting Stop Grant State

AMD

SYSTEM MANAGEMENT MODE

Overview

The Am486DE2 microprocessor supports four modes: Real, Virtual, Protected, and System Management Mode (SMM). As an operating mode, SMM has a distinct processor environment, interface, and hardware/software features. SMM lets the system designer add new software-controlled features to the computer products that always operate transparent to the operating system (OS) and software applications. SMM is intended for use only by system firmware, not by applications software or general-purpose systems software.

The SMM architectural extension consists of the following elements:

1. System Management Interrupt (SMI) hardware interface

2. Dedicated and secure memory space (SMRAM) for SMI handler code and CPU state (context) data with a status signal for the system to decode access to that memory space, $\overline{SMIACT}$

3. Resume (RSM) instruction, for exiting SMM

4. Special features, such as I/O Restart and I/O instruction information, for transparent power management of I/O peripherals, and Auto Halt Restart

Terminology

The following terms are used throughout the discussion of System Management Mode.

- **SMM:** System Management Mode. This is the operating environment that the processor (system) enters when servicing a System Management Interrupt.

- **SMI:** System Management Interrupt. This is the trigger mechanism for the SMM interface. When $\overline{SMI}$ is asserted ($\overline{SMI}$ pin asserted Low), it causes the processor to invoke SMM. The $\overline{SMI}$ pin is the only means of entering SMM.

- **SMI handler:** System Management Mode handler. This is the code that is executed when the processor is in SMM. An example application that this code might implement is a power-management-control or a system-control function.

- **RSM:** Resume instruction. This instruction is used by the SMI handler to exit the SMM and return to the interrupted OS or application process.

- **SMRAM:** This is the physical memory dedicated to SMM. The SMI handler code and related data reside in this memory. The processor also uses this memory to store its context before executing the SMI handler. The operating system and applications should not have access to this memory space.

- **SMBASE:** This is a control register that contains the base address that defines the SMRAM space.

- **Context:** This term refers to the processor state. The SMM discussion refers to the context, or processor state, just before the processor invokes SMM. The context normally consists of the CPU registers that fully represent the processor state.

- **Context Switch:** A context switch is the process of either saving or restoring the context. The SMM discussion refers to the context switch as the process of saving/restoring the context while invoking/exiting SMM, respectively.

- **SMSAVE:** A mechanism that saves and restores all internal registers to and from SMRAM.

System Management Interrupt Processing

The system interrupts the normal program execution and invokes SMM by generating a System Management Interrupt (SMI) to the CPU. The CPU services the SMI by executing the following sequence (see Figure 4).

1. The CPU asserts the $\overline{SMIACT}$ signal, instructing the system to enable the SMRAM.

2. The CPU saves its state (internal register) to SMRAM. It starts at the SMBASE relative address location (see "SMRAM" on page 24), and proceeds downward in a stack-like fashion.

3. The CPU switches to the SMM processor environment (an external pseudo-Real mode).

4. The CPU then jumps to the absolute address of SMBASE + 8000h in SMRAM to execute the SMI handler. This SMI handler performs the system management activities.

Note: If the SMRAM shares the same physical address location with part of the system RAM, it is "overlaid" SMRAM. To preserved cache consistency and correct SMM operation in systems using overlaid SMRAM, the cache must be flushed via the $\overline{FLUSH}$ pin when entering SMM.

5. The SMI handler then executes the RSM instruction, which restores the CPU's context from SMRAM, deasserts the $\overline{SMIACT}$ signal, and then returns control to the previously interrupted program execution.
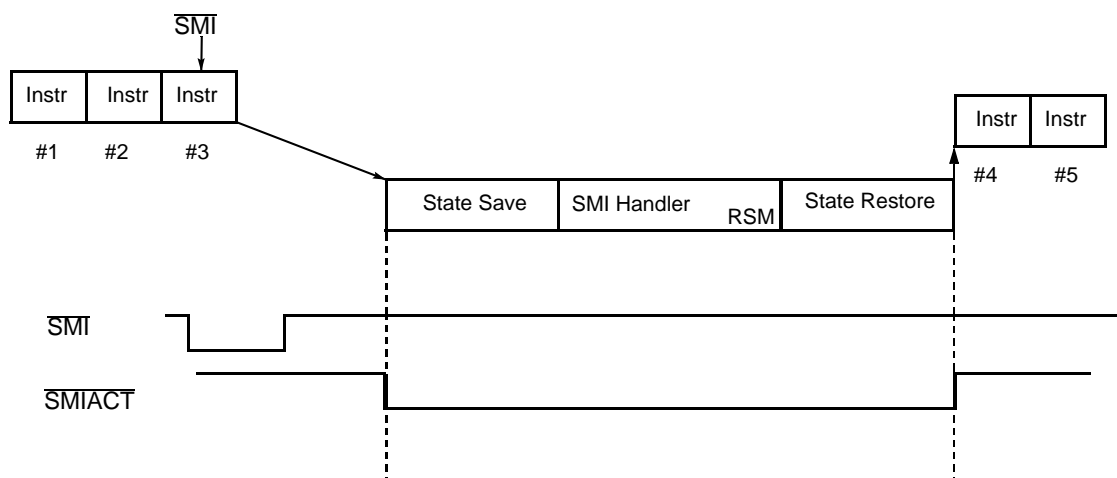
Figure 4. Basic SMI Interrupt Service

For uses such as fast enabling of external I/O devices, the SMSAVE mode permits the restarting of the I/O instructions and the HALT instruction. This is accomplished through I/O Trap Restart and Halt/Auto Halt Restart slots. Only I/O and HALT opcodes are restartable. Attempts to restart any other opcode may result in unpredictable behavior.

The System Management Interrupt hardware interface consists of the SMI request input and the SMIACT output used by the system to decode the SMRAM (see Figure 5).
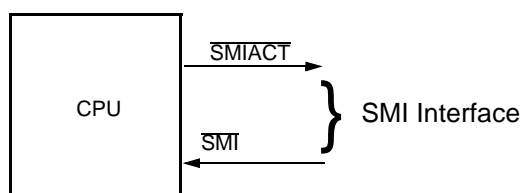
Figure 5. Basic SMI Hardware Interface

System Management Interrupt Processing

SMI is a falling-edge-triggered, non-maskable interrupt-request signal. SMI is an asynchronous signal, but setup and hold times must be met to guarantee recognition in a specific clock. The SMI input does not have to remain active until the interrupt is actually serviced. The SMI input needs to remain active for only a single clock if the required setup and hold times are met. SMI also works correctly if it is held active for an arbitrary number of clocks (see Figure 6).

The SMI input must be held inactive for at least four clocks after it is asserted to reset the edge-triggered logic. A subsequent SMI may not be recognized if the SMI input is not held inactive for at least four clocks after being asserted. SMI, like NMI, is not affected by the IF bit in the EFLAGS register and is recognized on an instruction boundary. SMI does not break locked bus cycles. SMI has a higher priority than NMI and is not masked during an NMI. After SMI is recognized, the SMI signal is masked internally until the RSM instruction is executed and the interrupt service routine is complete.

Masking SMI prevents recursive calls. If another SMI occurs while SMI is masked, the pending SMI is recognized and executed on the next instruction boundary after the current SMI completes. This instruction boundary occurs before execution of the next instruction in the interrupted application code, resulting in back-to-back SMI handlers. Only one SMI signal can be pending while SMI is masked. The SMI signal is synchronized internally and must be asserted at least three clock periods prior to asserting the RDY signal to guarantee recognition on a specific instruction boundary. This is important for servicing an I/O trap with an SMI handler.

SMI Active (SMIACT)

SMIACT indicates that the CPU is operating in SMM. The CPU asserts SMIACT in response to an SMI interrupt request on the SMI pin. SMIACT is driven active after the CPU has completed all pending write cycles

AMD

(including emptying the write buffers), and before the first access to SMRAM when the CPU saves (writes) its state (or context) to SMRAM. $\overline{SMIACT}$ remains active until the last access to SMRAM when the CPU restores (reads) its state from SMRAM. The $\overline{SMIACT}$ signal does not float in response to HOLD. The $\overline{SMIACT}$ signal is used by the system logic to decode SMRAM. The number of clocks required to complete the SMM state save and restore is dependent on system memory performance. The values shown in Figure 7 assume 0 wait-state memory writes (two clock cycles), 2 – 1 – 1 – 1 burst read cycles, and 0 wait-state non-burst reads (two clock cycles). Additionally, it is assumed that the data read during the SMM-state-restore sequence is not cacheable. The minimum time required to enter an SMSAVE SMI handler routine for the CPU (from the completion of the interrupted instruction) is given by:

Latency to start of SMI handler = A + B + C = 161 clocks

and the minimum time required to return to the interrupted application (following the final SMM instruction before RSM) is given by:

Latency to continue application = E + F + G = 258 clocks

SMRAM

The CPU uses the SMRAM space for state-save and state-restore operations during an SMI. The SMI handler, which also resides in SMRAM, uses the SMRAM space to store code, data, and stacks. In addition, the SMI handler can use the SMRAM for system management information such as the system configuration, configuration of a powered-down device, and system designer-specific information.

Note: Access to SMRAM is through the CPU internal cache. To ensure cache consistency and correct operation, always assert the $\overline{FLUSH}$ pin in the same clock as $\overline{SMI}$ for systems using overlaid SMRAM.

The CPU asserts $\overline{SMIACT}$ to indicate to the memory controller that it is operating in System Management Mode. The system logic should ensure that only the CPU and SMI handler have access to this area. Alternate bus masters or DMA devices trying to access the SMRAM space when $\overline{SMIACT}$ is active should be directed to system RAM in the respective area. The system logic is minimally required to decode the physical memory address range from 38000h–3FFFFh as SMRAM area. The CPU saves its state to the state-save area from 3FFFFh downward to 3FE00h. After saving its state, the CPU jumps to the address location 38000h to begin executing the SMI handler. The system logic can choose to decode a larger area of SMRAM as needed. The size of this SMRAM can be between 32 Kbyte and 4 Gbyte.The system logic should provide a manual method for switching the SMRAM into system memory space when the CPU is not in SMM. This enables initialization of the SMRAM space (i.e., loading SMI handler) before executing the SMI handler during SMM (see Figure 8).
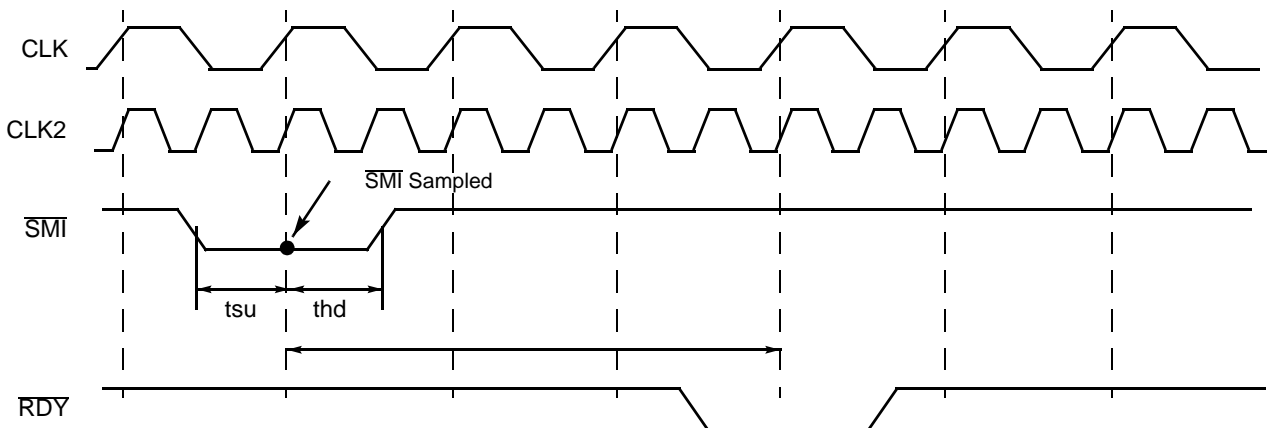
Figure 6. $\overline{SMI}$ Timing for Servicing an I/O Trap

Clock-Doubled CPU
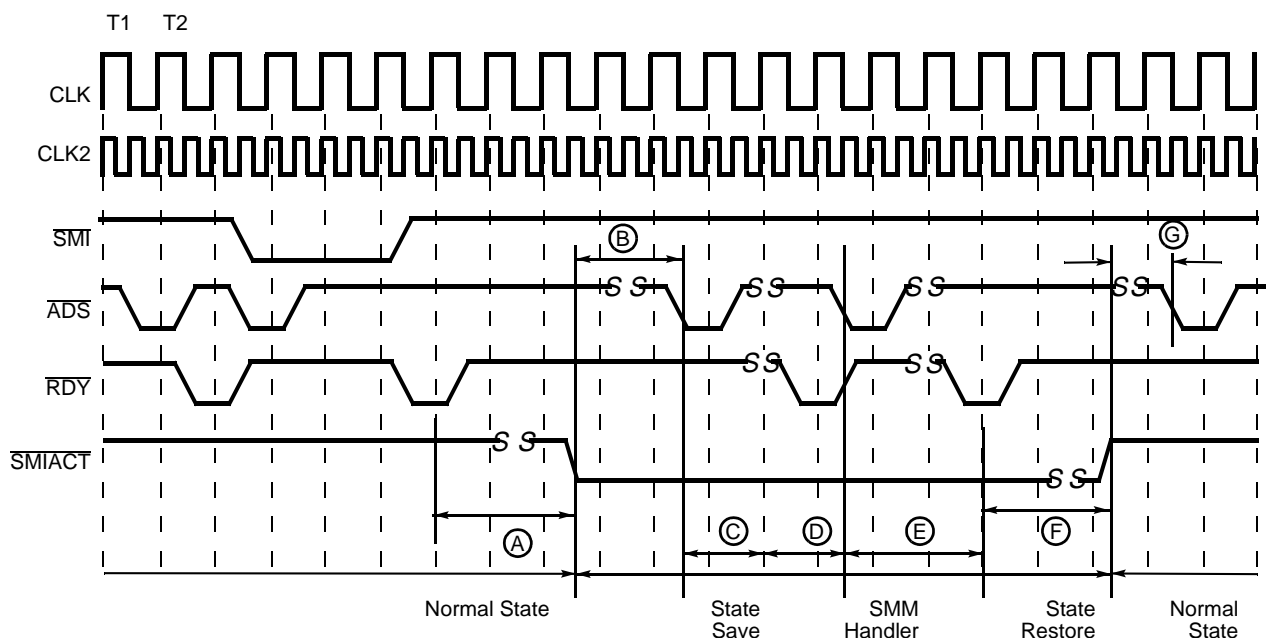
A: Last $\overline{RDY}$ from non-SMM transfer to $\overline{SMIACT}$ assertion — 2 CLKs minimum

B: $\overline{SMIACT}$ assertion to first $\overline{ADS}$ for SMM state save — 20 CLKs minimum

C: SMM state save (dependent on memory performance) — 139 CLKs

D: SMI handler — User-determined

E: SMM state restore (dependent on memory performance) — 236 CLKs

F: Last $\overline{RDY}$ from SMM transfer to deassertion of $\overline{SMIACT}$ — 2 CLKs minimum

G: $\overline{SMIACT}$ deassertion of first non-SMM $\overline{ADS}$ — 20 CLKs minimum

Figure 7. $\overline{SMIACT}$ Timing
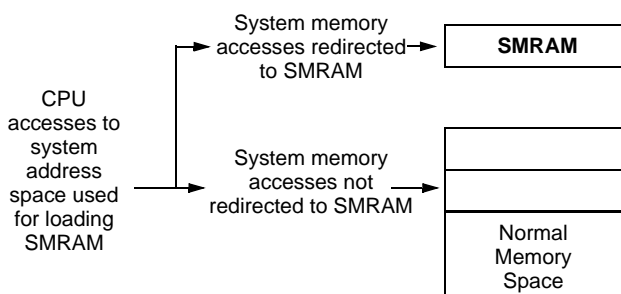
Figure 8. Redirecting System Memory Address to SMRAM

SMRAM State Save Map

When $\overline{SMI}$ is recognized on an instruction boundary, the CPU core first sets the $\overline{SMIACT}$ signal Low, indicating to the system logic that accesses are now being made to the system-defined SMRAM areas. The CPU then writes its state to the state save area in the SMRAM. The state save area starts at SMBASE + [8000h + 7FFFh]. The default CS Base is 30000h; therefore, the default state save area is at 3FFFFh. In this case, the CS Base is also referred to as the SMBASE.

If the SMBASE relocation feature is enabled, the SMRAM addresses can change. The following formula is used to determine the relocated addresses where the context is saved: SMBASE + [8000h + Register Offset], where the default initial SMBASE is 30000h and the Register Offset is listed in Table 3. Reserved spaces are for new registers in future CPUs. Some registers in the SMRAM state save area may be read and changed by

AMD🠶

the SMI handler, with the changed values restored to the processor register by the RSM instruction. Some register images are read-only, and must not be modified. (Modifying these registers results in unpredictable behavior.) The values stored in the *reserved* areas may change in future CPUs. An SMI handler should not rely on values stored in a reserved area.

The following registers are written out during SMSAVE mode to the RESERVED memory locations (7FA7h–7F98h, 7F93h–7F8Ch, and 7F87h–7F08h), but are not visible to the system software programmer:

■ DR3–DR0

■ CR2

■ CS, DS, ES, FS, GS, and SS hidden descriptor registers

■ EIP_Previous

■ GDT Attributes and Limits

■ IDT Attributes and Limits

■ LDT Attributes, Base, and Limits

■ TSS Attributes, Base, and Limits

If an SMI request is issued to power down the CPU, the values of all reserved locations in the SMM state save must be saved to nonvolatile memory.

The following registers are not automatically saved and restored by SMI and RSM:

■ TR7–TR3

■ FPU registers:

— STn

— FCS

— FSW

— Tag Word

— FP instruction pointer

— FP opcode

— Operand pointer

Note: *You can save the FPU state by using an FSAVE or FNSAVE instruction.*

For all SMI requests except for power-down suspend/resume, these registers do not have to be saved because their contents will not change. During a power-down suspend/resume, however, a resume reset clears these registers back to their default values. In this case, the suspend SMI handler should read these registers directly to save them and restore them during the power-up resume. Anytime the SMI handler changes these registers in the CPU, it must also save and restore them.

Table 3. SMRAM State Save Map

Register Offset*	Register	Writable?
7FFCh	CRO	No
7FF8h	CR3	No
7FF4h	EFLAGS	Yes
7FF0h	EIP	Yes
7FECh	EDI	Yes
7FE8h	ESI	Yes
7FE4h	EBP	Yes
7FE0h	ESP	Yes
7FDCh	EBX	Yes
7FD8h	EDX	Yes
7FD4h	ECX	Yes
7FD0h	EAX	Yes
7FCCh	DR6	No
7FC8h	DR7	No
7FC4h	TR*	No
7FC0h	LDTR*	No
7FBCh	GS*	No
7FB8h	FS*	No
7FB4h	DS*	No
7FB0h	SS*	No
7FACh	CS*	No
7FA8h	ES*	No
7FA7h–7F98h	Reserved	No
7F94h	IDT Base	No
7F93h–7F8Ch	Reserved	No
7F88h	GDT Base	No
7F87h–7F08h	Reserved	No
7F04h	I/O Trap Word	No
7F02h	Auto Halt Restart	Yes
7F00h	I/O Trap Restart	Yes
7EFCh	SMM Revision Identifier	Yes
7EF8h	State Dump Base	Yes
7EF7h–7E00h	Reserved	No

Note: *Upper 2 bytes are not modified.*

Entering System Management Mode

SMM is one of the major operating modes, along with Protected mode, Real mode, and Virtual mode. Figure 9 shows how the processor can enter SMM from any of the three modes and then return.

The external signal $\overline{SMI}$ causes the processor to switch to SMM. The RSM instruction exits SMM. SMM is transparent to applications programs and operating systems for the following reasons:

- The only way to enter SMM is via a type of nonmaskable interrupt triggered by an external signal.

- The processor begins executing SMM code from a separate address space, referred to earlier as system management RAM (SMRAM).

- Upon entry into SMM, the processor saves the register state of the interrupted program (depending on the save mode) in a part of SMRAM called the SMM context save space.

- All interrupts normally handled by the operating system or applications are disabled upon SMM entry.

- A special instruction, RSM, restores processor registers from the SMM context save space and returns control to the interrupted program.

Similar to Real mode, SMM has no privilege levels or address mapping. SMM programs can execute all I/O and other system instructions and can address up to 4 Gbyte of memory.

Exiting System Management Mode

The RSM instruction (opcode 0F AAh) leaves SMM and returns control to the interrupted program. The RSM instruction can be executed only in SMM. An attempt to execute the RSM instruction outside of SMM generates an invalid opcode exception. When the RSM instruction is executed and the processor detects invalid state information during the reloading of the save state, the processor enters the shutdown state. This occurs in the following situations:

- The value in the State Dump base field is not a 32-Kbyte aligned address.

- A combination of bits in CR0 is illegal: PG=1 and PE=0, or NW=1 and CD=0.

In shutdown mode, the processor stops executing instructions until an NMI interrupt is received or reset initialization is invoked. The processor generates a shutdown bus cycle.

Three SMM features can be enabled by writing to control slots in the SMRAM state save area:

1. **Auto Halt Restart**. It is possible for the SMI request to interrupt the HALT state. The SMI handler can tell the RSM instruction to return control to the HALT instruction or to return control to the instruction following the HALT instruction by appropriately setting the Auto Halt Restart slot. The default operation is to restart the HALT instruction.
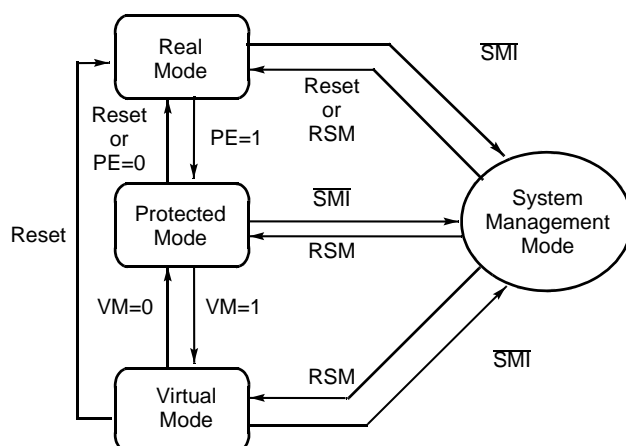
Figure 9. Transition to and from SMM

2. **I/O Trap Restart**. If the SMI was generated on an I/O access to a powered-down device, the SMI handler can instruct the RSM instruction to re-execute that I/O instruction by setting the I/O Trap Restart slot.

3. **SMBASE Relocation**. The system can relocate the SMRAM by setting the SMBASE Relocation slot in the state save area. The RSM instruction sets SMBASE in the processor based on the value in the SMBASE relocation slot. The SMBASE must be aligned on 32-Kbyte boundaries.

A RESET also causes execution to exit from SMM.

Processor Environment

When an $\overline{SMI}$ signal is recognized on an instruction execution boundary, the processor waits for all stores to complete, including emptying the write buffers. The final write cycle is complete when the system returns $\overline{RDY}$ or $\overline{BRDY}$. The processor then drives $\overline{SMIACT}$ active, saves its register state to SMRAM space, and begins to execute the SMI handler.

$\overline{SMI}$ has greater priority than debug exceptions and external interrupts. This means that if more than one of these conditions occur at an instruction boundary, only the SMI processing occurs. Subsequent SMI requests are not acknowledged while the processor is in SMM. The first SMI request that occurs while the processor is in SMM is latched, and serviced when the processor exits SMM with the RSM instruction. Only one $\overline{SMI}$ signal is latched by the CPU while it is in SMM. When the CPU invokes SMM, the CPU core registers are initialized as indicated in Table 4.

AMD

Table 4. SMM Initial CPU Core Register Settings

Register	SMM Initial State
General Purpose Registers	Unmodified
EFLAGS	0000 0002h
CR0	Bits 0, 2, 3, and 31 cleared (PE, EM, TS, and PG); remainder unmodified
DR6	Unpredictable state
DR7	0000 0400h
GDTR, LDTR, IDTR, TSSR	Unmodified
EIP	0000 8000h

Note: *Interrupts from INT and NMI are disabled on SMM entry.*

Table 5. Segment Register Initial States

Segment Register	Selector	Base	Attributes	Limit[1]
CS[2]	3000h	30000h	16-bit, expand up	4 Gbyte
DS	0000h	00000000h	16-bit, expand up	4 Gbyte
ES	0000h	00000000h	16-bit, expand up	4 Gbyte
FS	0000h	00000000h	16-bit, expand up	4 Gbyte
GS	0000h	00000000h	16-bit, expand up	4 Gbyte
SS	0000h	00000000h	16-bit, expand up	4 Gbyte

1. The segment limit check is 4 Gbyte instead of the usual 64K.

2. The Selector value for CS remains at 3000h even if the SMBASE is changed.

The following is a summary of the key features in the SMM environment:

- Real mode style address calculation

- 4-Gbyte limit checking

- IF flag is cleared

- NMI is disabled

- TF flag in EFLAGS is cleared; single step traps are disabled

- DR7 is cleared; debug traps are disabled

- The RSM instruction no longer generates an invalid op code error

- Default 16-bit op code, register, and stack use

- All bus arbitration (HOLD, AHOLD, $\overline{BOFF}$) inputs and bus sizing ($\overline{BS8}$, $\overline{BS16}$) inputs operate normally while the CPU is in SMM.

Executing System Management Mode Handler

The processor begins execution of the SMI handler at offset 8000h in the CS segment. The CS Base is initially 30000h, as shown in Table 5.

The CS Base can be changed using the SMM Base relocation feature. When the SMI handler is invoked, the CPU's PE and PG bits in CR0 are reset to 0. The processor is in an environment similar to Real mode, but without the 64-Kbyte limit checking. However, the default operand size and the default address size are set to 16 bits. The EM bit is cleared so that no exceptions are generated. (If the SMM was entered from Protected mode, the Real mode interrupt and exception support is not available.) The SMI handler should not use floating-point unit instructions until the FPU is properly detected (within the SMI handler) and the exception support is initialized.

Because the segment bases (other than CS) are cleared to 0 and the segment limits are set to 4 Gbyte, the address space may be treated as a single, flat 4-Gbyte linear space that is unsegmented. The CPU is still in Real mode and when a segment selector is loaded with a 16-bit value, that value is then shifted left by 4 bits and loaded into the segment base cache.

In SMM, the CPU can access or jump anywhere within the 4-Gbyte logical address space. The CPU can also indirectly access or perform a near jump anywhere within the 4-Gbyte logical address space.

Exceptions and Interrupts with System Management Mode

When the CPU enters SMM, it disables INTR interrupts, debug, and single-step traps by clearing the EFLAGS, DR6, and DR7 registers. This prevents a debug application from accidentally breaking into an SMI handler.

This is necessary because the SMI handler operates from a distinct address space (SMRAM) and the debug trap does not represent the normal system memory space.

For an SMI handler to use the debug trap feature of the processor to debug SMI handler code, it must first ensure that an SMM-compliant debug handler is available. The SMI handler must also ensure DR3–DR0 is saved to be restored later. The debug registers DR3–DR0 and DR7 must then be initialized with the appropriate values.

For the processor to use the single-step feature of the processor, it must ensure that an SMM-compliant single-step handler is available and then set the trap flag in the EFLAGS register. If the system design requires the processor to respond to hardware INTR requests while in SMM, it must ensure that an SMM-compliant interrupt handler is available, and then set the interrupt flag in the EFLAGS register (using the STI instruction). Software interrupts are not blocked on entry to SMM, and the system software designer must provide an SMM-compliant interrupt handler before attempting to execute any software interrupt instructions. Note that in SMM mode the interrupt vector table has the same properties and location as the Real mode vector table.

NMI interrupts are blocked on entry to the SMI handler. If an NMI request occurs during the SMI handler, it is latched and serviced after the processor exits SMM. Only one NMI request is latched during the SMI handler. If an NMI request is pending when the processor executes the RSM instruction, the NMI is serviced before the next instruction of the interrupted code sequence.

Although NMI requests are blocked when the CPU enters SMM, they may be enabled through software by executing an IRET instruction. If the SMI handler requires the use of NMI interrupts, it should invoke a dummy interrupt service routine to execute an IRET instruction. When an IRET instruction is executed, NMI interrupt requests are serviced in the same Real mode manner in which they are handled outside of SMM.

SMM Revisions Identifier

The 32-bit SMM Revision Identifier specifies the version of SMM and the extensions that are available on the processor. The fields of the SMM Revision Identifiers and bit definitions are shown in Table 6 and Table 7. Bit 17 or 16 indicates whether the feature is supported (1 = supported, 0 = not supported). The processor always reads the SMM Revision Identifier at the time of a restore. The I/O Trap Extension and SMM Base Relocation bits are fixed. The processor writes these bits out at the time it performs a save state.

Note: Changing the state of the reserved bits may result in unpredictable processor behavior.

Table 6. System Management Mode Revision Identifier

31–18	17	16	15–0
Reserved	SMM Base Relocation	I/O Trap Extension	SMM Revision Level
00000000000000	1	1	0000h

Table 7. SMM Revision Identifier Bit Definitions

Bit Name	Description	Default State	State at SMM Entry	State at SMM Exit	Notes
SMM Base Relocation	1=SMM Base Relocation Available 0=SMM Base Relocation Unavailable	1	1 0	1 0	No Change in State No Change in State
I/O Trap Extension	1=I/O Trapping Available 0=I/O Trapping Unavailable	1	1 0	1 0	No Change in State No Change in State

AMD🡒

Auto Halt Restart

The Auto Halt Restart slot at register offset (word location) 7F02h in SMRAM indicates to the SMI handler that the SMI interrupted the CPU during a HALT state; bit 0 of slot 7F02h is set to 1 if the previous instruction was a HALT (see Figure 10). If the $\overline{\text{SMI}}$ did not interrupt the CPU in a HALT state, then the SMI microcode sets bit 0 of the Auto Halt Restart slot to 0. If the previous instruction was a HALT, the SMI handler can choose to either set or reset bit 0. If this bit is set to 1, the RSM microcode execution forces the processor to reenter the HALT state. If this bit is set to 0 when the RSM instruction is executed, the processor continues execution with the instruction just after the interrupted HALT instruction. If the HALT instruction is restarted, the CPU will generate a memory access to fetch the HALT instruction (if it is not in the internal cache), and execute a HALT bus cycle.
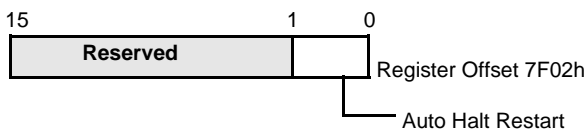
Figure 10. Auto Halt Restart Register Offset

Table 8 shows the possible restart configurations. If the interrupted instruction was not a HALT instruction (bit 0 is set to 0 in the Auto Halt Restart slot upon SMM entry), setting bit 0 to 1 will cause unpredictable behavior when the RSM instruction is executed.

Table 8. Auto Halt Restart Configuration

Value at Entry	Value at Exit	Processor Action on Exit
0	0	Return to next instruction in interrupted program
0	1	Unpredictable
1	0	Returns to instruction after HALT
1	1	Returns to interrupted HALT instruction

I/O Trap Restart

The I/O instruction restart slot (register offset 7F00h in SMRAM) gives the SMI handler the option of causing the RSM instruction to automatically reexecute the interrupted I/O instruction (see Figure 11).
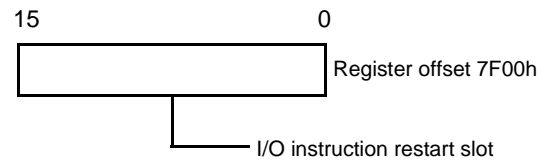
Figure 11. I/O Instruction Restart Register Offset

When the RSM instruction is executed, if the I/O instruction restart slot contains the value 0FFh, then the CPU automatically reexecutes the I/O instruction that the $\overline{\text{SMI}}$ signal trapped. If the I/O instruction restart slot contains the value 00h when the RSM instruction is executed, then the CPU does not reexecute the I/O instruction. The CPU automatically initializes the I/O instruction restart slot to 00h during SMM entry. The I/O instruction restart slot should be written only when the processor has generated an SMI on an I/O instruction boundary. Processor operation is unpredictable when the I/O instruction restart slot is set when the processor is servicing an SMI that originated on a non-I/O instruction boundary.

If the system executes back-to-back SMI requests, the second SMI handler must not set the I/O instruction restart slot. The second back-to-back $\overline{\text{SMI}}$ signal will not have the I/O Trap Word set.

I/O Trap Word

The I/O Trap Word contains the address of the I/O access that forced the external chipset to assert $\overline{\text{SMI}}$, whether it was a read or write access, and whether the instruction that caused the access to the I/O address was a valid I/O instruction. Table 9 shows the layout.

Table 9. I/O Trap Word Configuration

31–16	15–2	1	0
I/O Address	Reserved	Valid I/O Instruction	R/$\overline{\text{W}}$

Bits 31–16 contain the I/O address that was being accessed at the time $\overline{\text{SMI}}$ became active. Bits 15–2 are reserved.

If the instruction that caused the I/O trap to occur was a valid I/O instruction (IN, OUT, INS, OUTS, REP INS, or REP OUTS), the Valid I/O Instruction bit is set. If it was not a valid I/O instruction, the bit is saved as a 0. For REP instructions, the external chip set should return a valid $\overline{\text{SMI}}$ within the first access.

Bit 0 indicates whether the opcode that was accessing the I/O location was performing either a read (1) or a write (0) operation as indicated by the R/$\overline{W}$ bit.

If an SMI occurs and it does not trap an I/O instruction, the contents of the I/O address and R/$\overline{W}$ bit are unpredictable and should not be used.

SMM Base Relocation

The Am486DE2 processor provides a control register not in the standard Am486DX processor: SMBASE. The SMRAM address space can be modified by changing the SMBASE register before exiting an SMI handler routine. SMBASE can be changed to any 32K-aligned value. (Values that are not 32K-aligned cause the CPU to enter the Shutdown state when executing the RSM instruction.) SMBASE is set to the default value of 30000h on RESET. If SMBASE is changed by an SMI handler, all subsequent SMI requests initiate a state save at the new SMBASE.

The SMBASE slot in the SMM state-save area indicates and changes the SMI jump-vector location and SMRAM-save area. When bit 17 of the SMM Revision Identifier is set, then this feature exists and the SMRAM base and consequently, the jump vector, are as indicated by the SMM Base slot (see Table 7). During the execution of the RSM instruction, the CPU reads this slot and initializes the CPU to use the new SMBASE during the next SMI. During an SMI, the CPU does its context save to the new SMRAM area pointed to by the SMBASE, stores the current SMBASE in the SMM Base slot (offset 7EF8h), and then starts execution of the new jump vector based on the current SMBASE (see Figure 12).
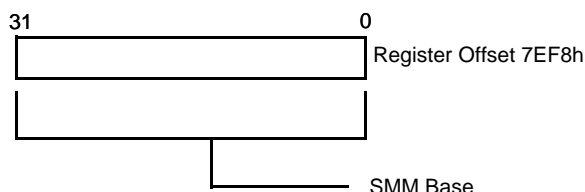
Figure 12. SMM Base Slot Offset

The SMBASE must be a 32-Kbyte aligned, 32-bit integer that indicates a base address for the SMRAM context save area and the SMI jump vector. For example, when the processor first powers up, the minimum SMRAM area is from 38000h–3FFFFh. The default SMBASE is 30000h.

As illustrated in Figure 13, the starting address of the jump vector is calculated by:

$$SMBASE + 8000h$$
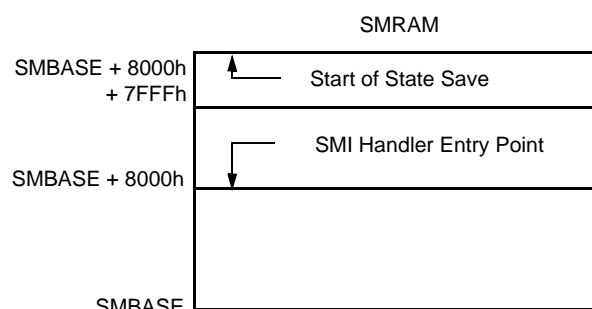
Figure 13. SRAM Usage

The starting address for the SMRAM state-save area is calculated by:

$$SMBASE + [8000h + 7FFFh]$$

When this feature is enabled, the SMRAM register map is addressed according to the above formula.

To change the SMRAM base address and SMI jump vector location, the SMI handler modifies the SMBASE slot. Upon executing an RSM instruction, the processor reads the SMBASE slot and stores it internally. Upon recognition of the next SMI request, the processor uses the new SMBASE slot for the SMRAM dump and SMI jump vector. If the modified SMBASE slot does not contain a 32-Kbyte aligned value, the RSM microcode causes the CPU to enter the Shutdown state.

SMM System Design Considerations

SMRAM Interface

The hardware designed to control the SMRAM space must follow these guidelines:

■ Initialize SMRAM space during system boot up. Initialization must occur before the first $\overline{SMI}$ occurs. Initialization of SMRAM space must include installation of an SMI handler and may include installation of related data structures necessary for particular SMM applications. The memory controller interfacing SMRAM should provide a means for the initialization code to open the SMRAM space manually.

■ The memory controller must decode a minimum initial SMRAM address space of 38000h–3FFFFh.

■ Alternate bus masters (such as DMA controllers) must not be able to access SMRAM space. The system should allow only the CPU, either through $\overline{SMI}$ or during initialization, to access SMRAM.

AMD

■ To implement a 0-V suspend function, the system must have access to all normal system memory from within an SMI handler routine. If the SMRAM overlays normal system memory (see Figure 14), there must be a method to access overlaid system memory independently.
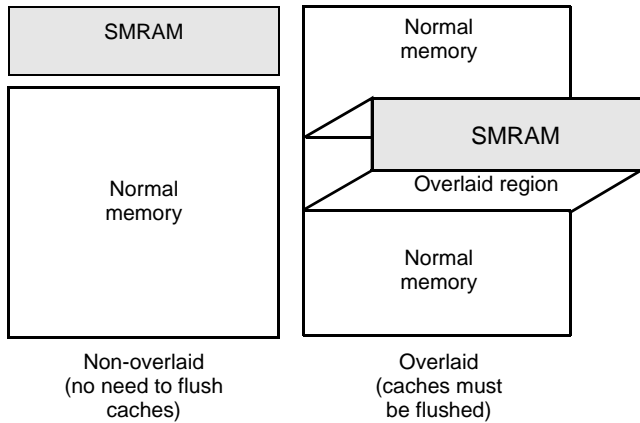

Figure 14. SMRAM Location

The recommended configuration is to use a separate (nonoverlaid) physical address for SMRAM. This non-overlaid scheme prevents the CPU from improperly accessing the SMRAM or system RAM directly or through the cache. Figure 15 shows the relative SMM timing for nonoverlaid SMRAM for systems configured in Write-through mode.

When the default SMRAM location is used, however, SMRAM is overlaid with system main memory (at 38000h–3FFFFh). For simplicity, system designers may want to use this default address, or they may select another overlaid address range. However, in this case the system control circuitry must use $\overline{\text{SMIACT}}$ to distinguish between SMRAM and main system memory, and must restrict SMRAM space access to the CPU only. To maintain cache coherency and to ensure proper system operation in systems configured in Write-through mode, the system must flush both the CPU internal cache and any second-level caches in response to $\overline{\text{SMIACT}}$ going Low. A system that uses cache during SMM must flush the cache a second time in response to $\overline{\text{SMIACT}}$ going High (see Figure 16). If $\overline{\text{KEN}}$ is driven High when $\overline{\text{FLUSH}}$ is asserted, the cache is disabled and a second flush is not required (see Figure 17).

Cache Flushes

The CPU does not unconditionally flush its cache before entering SMM. Therefore, the designer must ensure that, for systems using overlaid SMRAM, the cache is flushed upon SMM entry, and SMM exit if caching is enabled.

If the flush at SMM entry is not done, the first SMM read could hit in a cache that contains normal memory space code/data instead of the required SMI handler and the handler could not be executed. If the cache is not disabled and cache is not flushed at SMM exit, the normal read cycles after SMM may hit in a cache that may contain SMM code/data instead of the normal system memory contents.

In Write-through mode, assert the $\overline{\text{FLUSH}}$ signal in response to the assertion of $\overline{\text{SMIACT}}$ at SMM entry, and if required because the cache is enabled, assert $\overline{\text{FLUSH}}$ again in response to the deassertion of $\overline{\text{SMIACT}}$ at SMM exit (see Figure 16 and Figure 17).

Reloading the state registers at the end of SMM restores cache functionality to its pre-SMM state.
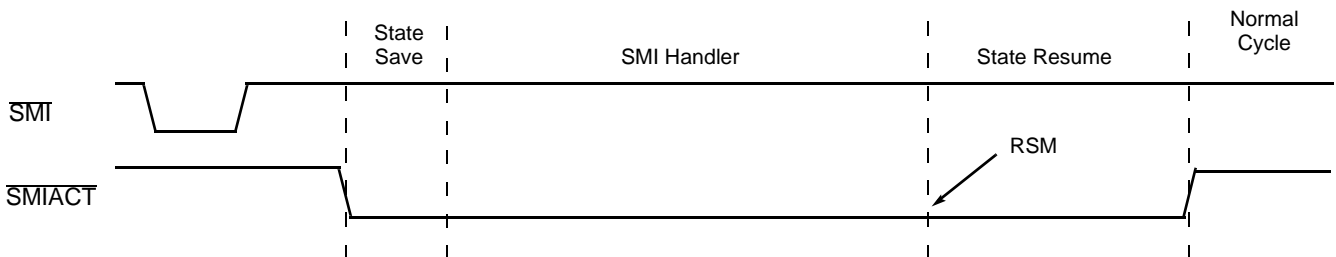

Figure 15. SMM Timing in Systems Using Non-Overlaid Memory Space and Write-Through Mode with Caching Enabled During SMM
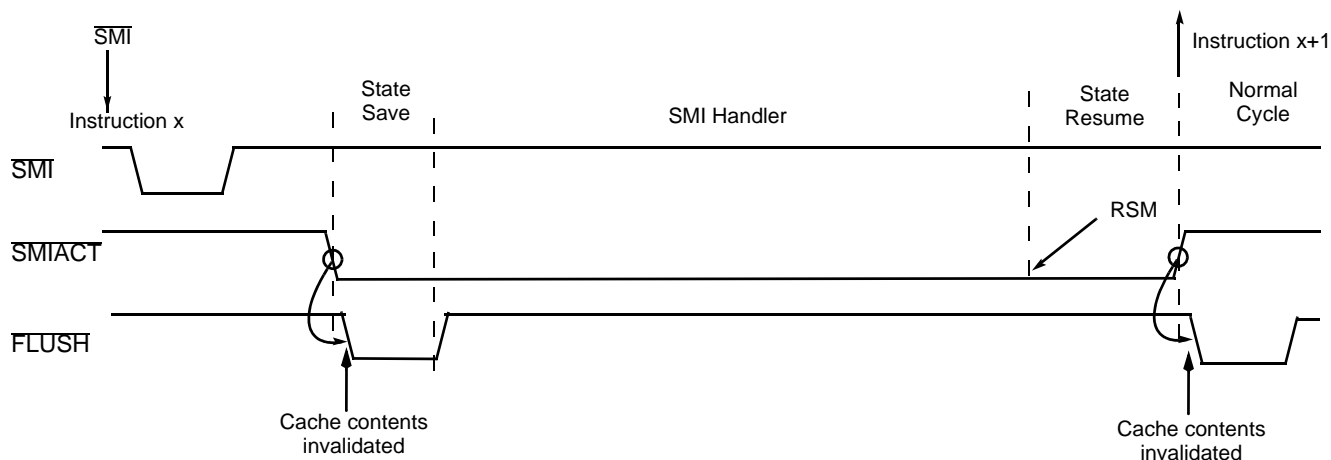
Figure 16. SMM Timing in Systems Using Overlaid Memory Space and Write-Through Mode with Caching *Enabled* During SMM
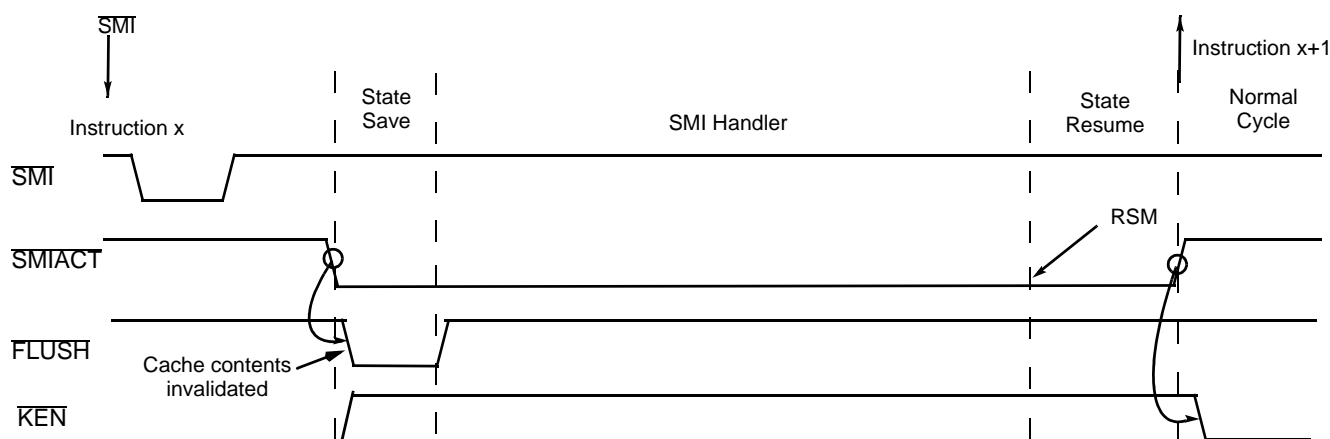

Figure 17. SMM Timing in Systems Using Overlaid Memory Space and Write-Through Mode with Caching *Disabled* During SMM

A20M Pin

Systems based on the MS-DOS operating system contain a feature that enables the CPU address bit A20 to be forced to 0. This limits physical memory to a maximum of 1 Mbyte, and is provided to ensure compatibility with those programs that relied on the physical address wraparound functionality of the original IBM PC. The A20M pin on Am486DE2 CPUs provides this function. When A20M is active, all external bus cycles drive A20 Low, and all internal cache accesses are performed with A20 Low.

The A20M pin is recognized while the CPU is in SMM. The functionality of the A20M input must be recognized in two instances:

1. If the SMI handler needs to access system memory space above 1 Mbyte (for example, when saving memory to disk for a zero-volt suspend), the A20M pin must be deasserted before the memory above 1 Mbyte is addressed.

2. If SMRAM has been relocated to address space above 1 Mbyte and A20M is active upon entering SMM, the CPU attempts to access SMRAM at the relocated address, but with A20 Low. This could cause the system to crash, because there would be no valid SMM interrupt handler at the accessed location.

AMD

To account for these two situations, the system designer must ensure that $\overline{A20M}$ is deasserted on entry to SMM. $\overline{A20M}$ must be driven inactive before the first cycle of the SMM state save, and must be returned to its original level after the last cycle of the SMM state restore. This can be done by blocking the assertion of $\overline{A20M}$ when $\overline{SMIACT}$ is active.

CPU Reset During SMM

The system designer should take into account the following restrictions while implementing the CPU Reset logic:

■ When running software written for the 80286 CPU, a CPU RESET switches the CPU from Protected mode to Real mode. RESET and SRESET have a higher priority than $\overline{SMI}$. When the CPU is in SMM, the SRESET to the CPU during SMM should be blocked until the CPU exits SMM. SRESET must be blocked beginning from the time when $\overline{SMI}$ is driven active. Care should be taken not to block the global system RESET, which may be necessary to recover from a system crash.

■ During execution of the RSM instruction to exit SMM, there is a small time window between the deassertion of $\overline{SMIACT}$ and the completion of the RSM microcode. If a Protected mode to Real mode SRESET is asserted during this window, it is possible that the SMRAM space will be violated. The system designer must guarantee that SRESET is blocked until at least 20 CPU clock cycles after $\overline{SMIACT}$ has been driven inactive or until the start of a bus cycle.

■ Any request for a CPU RESET for the purpose of switching the CPU from Protected mode to Real mode must be acknowledged after the CPU has exited SMM. To maintain software transparency, the system logic must latch any SRESET signals that are blocked during SMM.

For these reasons, the SRESET signal should be used for any soft resets, and the RESET signal should be used for all hard resets.

SMM and Second-Level Write Buffers

Before the processor enters SMM, it empties its internal write buffers. This is to ensure that the data in the write buffers is written to normal memory space, not SMM space. When the CPU is ready to begin writing an SMM state save to SMRAM, it asserts $\overline{SMIACT}$. $\overline{SMIACT}$ may be driven active by the CPU before the system memory controller has had an opportunity to empty the second-level write buffers.

To prevent the data from these second-level write buffers from being written to the wrong location, the system memory controller needs to direct the memory write cycles to either SMM space or normal memory space. This can be accomplished by saving the status of $\overline{SMIACT}$ with the address for each word in the write buffers.

Nested SMI and I/O Restart

Special care must be taken when executing an SMI handler for the purpose of restarting an I/O instruction. When the CPU executes a Resume (RSM) instruction with the I/O restart slot set, the restored EIP is modified to point to the instruction immediately preceding the SMI request, so that the I/O instruction can be reexecuted. If a new SMI request is received while the CPU is executing an SMI handler, the CPU services this $\overline{SMI}$ request before restarting the original I/O instruction. If the I/O restart slot is set when the CPU executes the RSM instruction for the second SMI handler, the RSM microcode decrements the restored EIP again. EIP then points to an address different from the originally interrupted instruction, and the CPU begins execution at an incorrect entry point. To prevent this from occurring, the SMI handler routine must not set the I/O restart slot during the second of two consecutive SMI handlers.

SMM Software Considerations

SMM Code Considerations

The default operand size and the default address size are 16 bits; however, operand-size override and address-size override prefixes can be used as needed to directly access data anywhere within the 4-Gbyte logical address space.

With operand-size override prefixes, the SMI handler can use jumps, calls and returns to transfer a control to any location within the 4-Gbyte space. Note, however, the following restrictions:

■ Any control transfer that does not have an operand-size override prefix truncates EIP to 16 Low-order bits.

■ Due to the Real mode style of base-address formation, a long jump or call cannot transfer control segment with a base address of more than 20 bits (1 Mbyte).

Exception Handling

Upon entry into SMM, external interrupts that require handlers are disabled (the IF in EFLAGS is cleared). This is necessary because, while the processor is in SMM, it is running in a separate memory space. Consequently, the vectors stored in the interrupt descriptor table (IDT) for the prior mode are not applicable. Before allowing exception handling (or software interrupts), the SMM program must initialize new interrupt and excep-

tion vectors. The interrupt vector table for SMM has the same format as for Real mode. Until the interrupt vector table is correctly initialized, the SMI handler must not generate an exception (or software interrupt). Even though hardware interrupts are disabled, exceptions and software interrupts can still occur. Only a correctly written SMI handler can prevent internal exceptions. When new exception vectors are initialized, internal exceptions can be serviced. The restrictions follow:

■ Due to the Real mode style of base address formation, an interrupt or exception cannot transfer control to a segment with a base address of more than 20 bits.

■ An interrupt or exception cannot transfer control to a segment offset of more than 16 bits.

■ If exceptions or interrupts are allowed to occur, only the Low order 16 bits of the return address are pushed onto the stack. If the offset of the interrupted procedure is greater than 64 Kbytes, it is not possible for the interrupt/exception handler to return control to that procedure. (One workaround is to perform software adjustment of the return address on the stack.)

■ The SMBASE Relocation feature affects the way the CPU returns from an interrupt or exception during an SMI handler.

Note: The execution of an IRET instruction enables Non-Maskable Interrupt (NMI) processing.

HALT during SMM

HALT should not be executed during SMM, unless interrupts have been enabled. Interrupts are disabled on entry to SMM. INTR and NMI are the only events that take the CPU out of HALT within SMM.

Relocating SMRAM to an Address above 1 Mbyte

Within SMM (or Real mode), the segment base registers can be updated only by changing the segment register. The segment registers contain only 16 bits, which allows only 20 bits to be used for a segment base address (the segment register is shifted left 4 bits to determine the segment base address). If SMRAM is relocated to an address above 1 Mbyte, the segment registers can no longer be initialized to point to SMRAM.

These areas can still be accessed by using address override prefixes to generate an offset to the correct address. For example, if the SMBASE has been relocated immediately below 16M, the DS and ES registers are still initialized to 0000 0000h. Data in SMRAM can still be accessed by using 32-bit displacement registers:

```
move esi,OOFFxxxxh;64K segment immediately
          below 16M
move ax,ds:[esi]
```

AMD

TEST REGISTERS 4 AND 5 MODIFICATIONS

The Cache Test Registers for the Am486DE2 microprocessor are the same test registers (TR3, TR4, and TR5) provided in earlier Am486DX and DX2 microprocessors. TR3 is the cache test data register. TR4, the cache test status register, and TR5, the cache test control register, operate together with TR3.

When WB/$\overline{\text{WT}}$ meets the necessary setup timing and is sampled Low on the falling edge of RESET, the processor is placed in Write-through mode and the test register function is identical to the earlier Am486 microprocessors. Table 10 and Table 11 show the individual bit functions of these registers. "TR4 Definition" on page 36 and "TR5 Definition" on page 36 provide a detailed description of the field functions.

Table 10. Test Register (TR4)

	31	30–29	28	27–26	25–24	23–22	21–20	19	18	17	16	15–11	10	9–7	6–3	2–0
EXT = 0	Tag												Valid	LRU	Valid (rd)	Not used

Table 11. Test Register (TR5)

	31–20	19	18–17	16	15–11	10–4	3–2	1–0
Write-Through	Not used					Index	Entry	Control

TR4 Definition

This section includes a detailed description of the bit fields defined for TR4.

Note: Bits listed in Table 10 as Not used are not included in these descriptions.

- *Tag (bits 31–11):* Read/Write, always available in Write-through mode. For a cache write, this is the tag that specifies the address in memory. On a cache look-up, this is the tag for the selected entry in the cache.

- *Valid (bit 10):* Read/Write. This is the Valid bit for the accessed entry. On a cache look-up, Valid is a copy of one of the bits reported in bits 6–3. On a cache-write in Write-through mode, Valid becomes the new valid bit for the selected entry and set.

- *LRU (bits 9–7):* Read Only, independent of the Ext bit in TR5. On a cache look-up, these are the three LRU bits of the accessed set. On a cache write, these bits are ignored; the LRU bits in the cache are updated by the pseudo-LRU cache replacement algorithm. Write operations to these locations have no effect on the device.

- *Valid (bits 6–3):* Read Only. On a cache look-up, these are the four Valid bits of the accessed set. Write operations to these locations have no effect on the device.

TR5 Definition

This section includes a detailed description of the bit fields in TR5.

Note: Bits listed in Table 11 as Not used are not included in the descriptions.

- *Index (bits 10–4):* Read/Write. Index selects one of the 128 sets.

- *Entry (bits 3–2):* Read/Write. Entry selects between one of the four entries in the set addressed by the Set Select during a cache read or write. During cache-fill buffer writes or cache-read buffer reads, the value in the Entry field selects one of the four doublewords in a cache line.

- *Control (bits 1–0):* Read/Write. The control bits determine which operation is to be performed. The following is a definition of the control operations:

  — 00 = Write to cache fill buffer, or read from cache read buffer.
  — 01 = Perform cache write.
  — 10 = Perform cache read.
  — 11 = Flush the cache (mark all entries invalid)

Am486DE2 Microprocessor

Am486DE2 MICROPROCESSOR FUNCTIONAL DIFFERENCES

The Am486DE2 microprocessor is a new member of the AMD Am486 family, which also includes the Enhanced Am486 and the Am486DX microprocessors.

Although the Am486DE2 is based on and compatible with the Enhanced Am486 microprocessors, it has no support for write-back cache.

Several important differences exist between the Am486DE2 and the Am486DX processors:

■ The Am486DE2 ID register contains a different version signature than the Am486DX. It has the same ID register as the Enhanced Am486DX2 in Write-through mode.

■ A burst write feature is available for copy-backs. The $\overline{FLUSH}$ pin and WBINVD instruction copy-back all modified data to external memory prior to issuing the special bus cycle or reset.

■ The RESET state is invoked either after power up or after the RESET signal is applied according to the standard Am486DX microprocessor specification.

■ After reset, the STATUS bits of all lines are set to 0. The LRU bits of each set are placed in a starting state.

In addition, the differences in the processors are highlighted in Table 12.

Table 12. Am486DE2 Microprocessor Functional Differences

Processor	Cache	Clock	Major Enhancements	Package
Am486DX2-66	8 Kbyte, Write-through	1x, 2x		168-Pin PGA
Enhanced Am486DX2-66	8-Kbyte, Write-through/ Write-back	2x, 3x	SMI, write-back	168-Pin PGA
Am486DE2-66	8-Kbyte, Write-through	2x	SMI	168-Pin PGA, 208-Lead SQFP

Am486DE2 MICROPROCESSOR IDENTIFICATION

The Am486DE2 microprocessor supports two standard methods for identifying the CPU in a system. The reported values are dynamically assigned based on the CPU type and the status of the WB/$\overline{WT}$ pin input at RESET.

DX Register at RESET

The DX register always contains a component identifier at the conclusion of RESET. The upper byte of DX (DH) contains 04 and the lower byte of DX (DL) contains a CPU type/stepping identifier (see Table 13).

Table 13. CPU ID Codes

CPU Type and Cache Mode	Component ID (DH)	Revision ID (DL)
DE2 in Write-through mode	04	3x

CPUID Instruction

The Am486DE2 implements a new instruction that makes information available to software about the family, model, and stepping of the microprocessor on which it is executing. Support of this instruction is indicated by the presence of a user-modifiable bit in position EFLAGS.21, referred to as the EFLAGS.ID bit. This bit is reset to zero at device reset (RESET or SRESET) for compatibility with existing processor designs.

CPUID Timing

CPUID execution timing depends on the selected EAX parameter values (see Table 14).

AMD

Table 14. CPUID Instruction Description

OP Code	Instruction	EAX Input Value	CPU Core Clocks	Description
0F A2	CPUID	0 1 >1	41 14 9	AMD string CPU ID Register null registers

CPUID Operation

The CPUID instruction requires the user to pass an input parameter to the CPU in the EAX register. The CPU response is returned to the user in registers EAX, EBX, ECX, and EDX.

When the parameter passed in EAX is zero, the register values returned upon instruction execution are:

EAX[31:0] ← 00000001h
EBX[31:0] ← 68747541h
ECX[31:0] ← 444D4163h
EDX[31:0] ← 69746E65h

The values in EBX, ECX, and EDX indicate an AMD microprocessor. When taken in the proper order:

■ EBX (least significant bit to most significant bit)

■ EDX (least significant bit to most significant bit)

■ ECD (least significant bit to most significant bit)

they decode to:

'AuthenticAMD'

When the parameter passed in EAX is 1, the register values returned are:

EAX[3:0]   ← Stepping ID*
EAX[7:4]   ← Model:
                 Am486DE2CPU—
                   Write-through mode = 3h
EAX[11:8]   ← Family
                 Am486 CPU = 4h
EAX[15:12] ← 0000
EAX[31:16] ← RESERVED
EBX[31:0]   ← 00000000h
ECX[31:0]   ← 00000000h
EDX[31:0]   ← 00000001h = all versions
                 The 1 in bit 0 indicates that the
                 FPU is present

Note: *Please contact AMD for stepping ID details.*

The value returned in EAX after CPUID instruction execution is identical to the value loaded into EDX upon device reset. Software must avoid any dependency upon the state of reserved processor bits.

When the parameter passed in EAX is greater than one, register values returned upon instruction execution are:

EAX[31:0] ← 00000000h
EBX[31:0] ← 00000000h
ECX[31:0] ← 00000000h
EDX[31:0] ← 00000000h

Flags affected: No flags are affected.
Exceptions: None

ELECTRICAL DATA

The following sections describe recommended electrical connections for the Am486DE2 microprocessor, and electrical specifications.

Power Connections

The Am486DE2 microprocessor has modest power requirements. However, the high clock-frequency output buffers can cause power surges as multiple output buffers drive new signal levels simultaneously. For clean, on-chip power distribution at high frequency, 23 V_{CC} pins and 28 V_{SS} pins feed the microprocessor in the 168-pin PGA package. The 208-lead SQFP package includes 53 V_{CC} pins and 38 V_{SS} pins.

Power and ground connections must be made to all external V_{CC} and V_{SS} pins of the microprocessors. On a circuit board, all V_{CC} pins must connect to a V_{CC} plane. Likewise, all V_{SS} pins must connect to a common GND plane.

The Am486DE2 microprocessor requires only 3.3 V as input power. Unlike other 3-V 486 processors, the Am486DE2 microprocessor does *not* require a V_{CC5} input of 5 V to indicate the presence of 5-V I/O devices on the system motherboard. For socket compatibility, this pin is INC (Internal No Connect), allowing the Am486DE2 CPU to operate in 3-V sockets in systems that use 5-V I/O.

Power Decoupling Recommendations

Liberal decoupling capacitance should be placed near the microprocessor. The microprocessor, driving its 32-bit parallel address and data buses at high frequencies, can cause transient power surges, particularly when driving large capacitive loads.

Low-inductance capacitors and interconnects are recommended for best high-frequency electrical performance. Inductance can be reduced by shortening circuit board traces between the microprocessor and the decoupling capacitors. Capacitors designed specifically for use with PGA packages are commercially available.

Other Connection Recommendations

For reliable operation, always connect unused inputs to an appropriate signal level. Active Low inputs should be connected to V_{CC} through a pull-up resistor. Pull-ups in the range of 20 kΩ are recommended. Active High inputs should be connected to GND.

AMD🗖

ABSOLUTE MAXIMUM RATINGS

Case Temperature under Bias . . . . . – 65°C to +110°C
Storage Temperature . . . . – 65°C to +150°C
Voltage on any pin
  with respect to ground . . . . . . – 0.5 V to V_{cc} +2.6 V
Supply voltage with
  respect to V_{SS} . . . . – 0.5 V to +4.6 V

*Stresses above those listed under Absolute Maximum Ratings
may cause permanent device failure. Functionality at or above
these limits is not implied. Exposure to Absolute Maximum
Ratings for extended periods may affect device reliability.*

OPERATING RANGES

Commercial (C) Devices

T_{CASE} · 0°C to 85°C
V_{CC} · 3.3 V ± 0.3 V

*Operating Ranges define those limits between which the func-
tionality of the device is guaranteed.*

DC CHARACTERISTICS OVER COMMERCIAL OPERATING RANGES

V_{CC} = 3.3 V ± 0.3 V; T_{CASE} = 0°C to + 85°C

Symbol	Parameter	Min	Max	Notes
V_{IL}	Input Low Voltage	– 0.3 V	+0.8 V	
V_{IH}	Input High Voltage	2.0 V	V_{CC} + 2.4 V	
V_{OL}	Output Low Voltage		0.45 V	Note 1
V_{OH}	Output High Voltage	2.4 V		Note 2
I_{CC}	Power Supply Current: 66 MHz		660 mA	Typical supply current: 528 mA @ 66 MHz Inputs at rails, outputs unloaded.
$I_{CCSTOPGRANT}$ or $I_{CCAUTOHALT}$	Input Current in Stop Grant or Auto Halt mode: 66 MHz		66 mA	Typical supply current for Stop Grant or Auto Halt mode: 20 mA @ 66 MHz and 75 MHz, 30 mA @ 80 MHz, 50 mA @ 100 MHz, and 60 mA @ 120 MHz.
$I_{CCSTPCLK}$	Input Current in Stop Clock mode		5 mA	Typical supply current in Stop Clock mode is 600 μA.
I_{LI}	Input Leakage Current		± 15 μA	Note 3
I_{IH}	Input Leakage Current		200 μA	Note 4
I_{IL}	Input Leakage Current		– 400 μA	Note 5
I_{LO}	Output Leakage Current		± 15 μA	
C_{IN}	Input Capacitance		10 pF	F_C = 1 MHz (Note 6)
C_O	I/O or Output Capacitance		14 pF	F_C = 1 MHz (Note 6)
C_{CLK}	CLK Capacitance		12 pF	F_C = 1 MHz (Note 6)

Notes:

1. *This parameter is measured at: Address, Data, $\overline{BE}$n = 4.0 mA; Definition, Control = 5.0 mA*
2. *This parameter is measured at: Address, Data, $\overline{BE}$n = - 1.0 mA; Definition, Control = - 0.9 mA*
3. *This parameter is for inputs without internal pull-ups or pull-downs and $0 \leq V_{IN} \leq V_{CC}$.*
4. *This parameter is for inputs with internal pull-downs and V_{IH} = 2.4 V.*
5. *This parameter is for inputs with internal pull-ups and V_{IL} = 0.45 V.*
6. *Not 100% tested.*

SWITCHING CHARACTERISTICS OVER COMMERCIAL OPERATING RANGES

The AC specifications, provided in the AC characteristics table, consist of output delays, input setup requirements, and input hold requirements. All AC specifications are relative to the rising edge of the CLK signal. AC specifications measurement is defined by Figure 36. All timings are referenced to 1.5 V unless otherwise specified.

Am486DE2 microprocessor output delays are specified with minimum and maximum limits, measured as shown. The minimum microprocessor delay times are hold times provided to external circuitry. Input setup and hold times are specified as minimums, defining the smallest acceptable sampling window. Within the sampling window, a synchronous input signal must be stable for correct microprocessor operation.

AMD

Switching Characteristics for 33-MHz Bus (66-MHz Microprocessor)
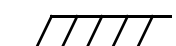
V_{CC} = 3.3 V ±0.3 V; T_{CASE} = 0°C to + 85°C; C_L = 50 pF unless otherwise specified[1]

Symbol	Parameter	Min	Max	Unit	Figure	Notes
	Frequency	8	33	MHz		Note 2
t_1	CLK Period	30	125	ns	39	
t_{1a}	CLK Period Stability		0.1%	Δ		Adjacent Clocks Notes 3 and 4
t_2	CLK High Time at 2 V	11		ns	39	Note 3
t_3	CLK Low Time at 0.8 V	11		ns	39	Note 3
t_4	CLK Fall Time (2 V–0.8 V)		3	ns	39	Note 3
t_5	CLK Rise Time (0.8 V–2 V)		3	ns	39	Note 3
t_6	A31–A2, PWT, PCD, $\overline{BE3}$–$\overline{BE0}$, M/$\overline{IO}$, D/$\overline{C}$, $\overline{CACHE}$, W/$\overline{R}$, $\overline{ADS}$, $\overline{LOCK}$, FERR, BREQ, HLDA, $\overline{SMIACT}$, $\overline{HITM}$ Valid Delay	3	14	ns	40	Note 5
t_7	A31–A2, PWT, PCD, $\overline{BE3}$–$\overline{BE0}$, M/$\overline{IO}$, D/$\overline{C}$, $\overline{CACHE}$, W/$\overline{R}$, $\overline{ADS}$, $\overline{LOCK}$ Float Delay	3	20	ns	41	Note 3
t_8	$\overline{PCHK}$ Valid Delay	3	14	ns	42	
t_{8a}	$\overline{BLAST}$, $\overline{PLOCK}$, Valid Delay	3	14	ns	40	
t_9	$\overline{BLAST}$, $\overline{PLOCK}$, Float Delay	3	20	ns	41	Note 3
t_{10}	D31–D0, DP3–DP0 Write Data Valid Delay	3	14	ns	40	
t_{11}	D31–D0, DP3–DP0 Write Data Float Delay	3	20	ns	41	Note 3
t_{12}	$\overline{EADS}$, INV, WB/$\overline{WT}$ Setup Time	5		ns	43	
t_{13}	$\overline{EADS}$, INV, WB/$\overline{WT}$ Hold Time	3		ns	43	
t_{14}	$\overline{KEN}$, $\overline{BS16}$, $\overline{BS8}$ Setup Time	5		ns	43	
t_{15}	$\overline{KEN}$, $\overline{BS16}$, $\overline{BS8}$ Hold Time	3		ns	43	
t_{16}	$\overline{RDY}$, $\overline{BRDY}$ Setup Time	5		ns	44	
t_{17}	$\overline{RDY}$, $\overline{BRDY}$ Hold Time	3		ns	44	
t_{18}	HOLD, AHOLD Setup Time	6		ns	43	
t_{18a}	$\overline{BOFF}$ Setup Time	7		ns	43	
t_{19}	HOLD, AHOLD, $\overline{BOFF}$ Hold Time	3		ns	43	
t_{20}	RESET, $\overline{FLUSH}$, $\overline{A20M}$, NMI, INTR, $\overline{IGNNE}$, $\overline{STPCLK}$, SRESET, $\overline{SMI}$ Setup Time	5		ns	43	Note 5
t_{21}	RESET, $\overline{FLUSH}$, $\overline{A20M}$, NMI, INTR, $\overline{IGNNE}$, $\overline{STPCLK}$, SRESET, $\overline{SMI}$ Hold Time	3		ns	43	Note 5
t_{22}	D31–D0, DP3–DP0, A31–A4 Read Setup Time	5		ns	43, 44	
t_{23}	D32–D0, DP3–DP0, A31–A4 Read Hold Time	3		ns	43, 44	

Notes: 1. Specifications assume C_L = 50 pF. I/O Buffer model must be used to determine delays due to loading (trace and component). First-order I/O buffer models for the processor are available.

2. 0-MHz operation guaranteed during stop clock operation or 1x Static Clock mode.

3. Not 100% tested. Guaranteed by design characterization.

4. For faster transitions (>0.1% between adjacent clocks), use the Stop Clock protocol to switch operating frequency.

5. All timings are referenced at 1.5 V (as illustrated in the listed figures) unless otherwise noted.

Switching Waveforms

Key to Switching Waveforms

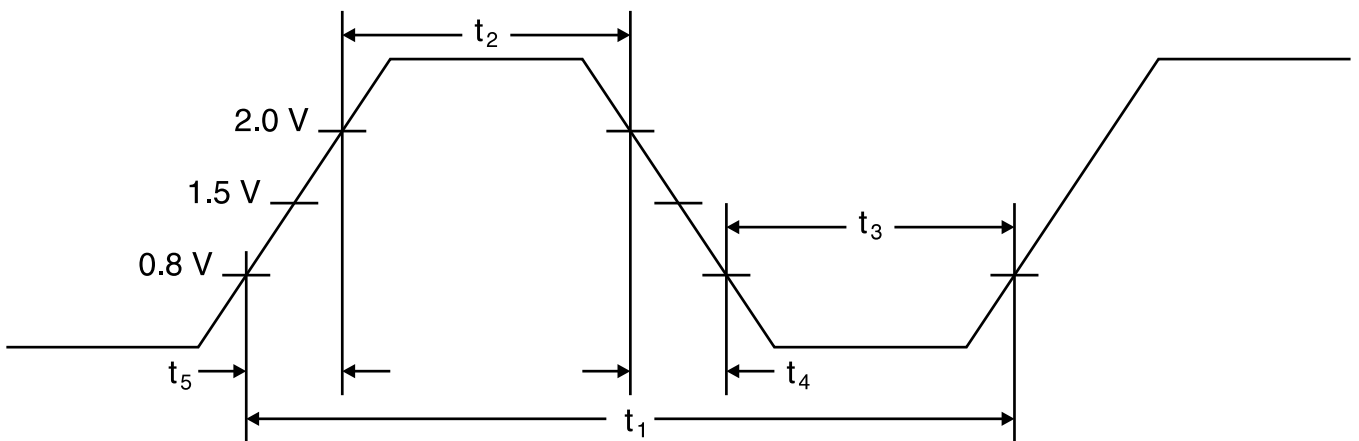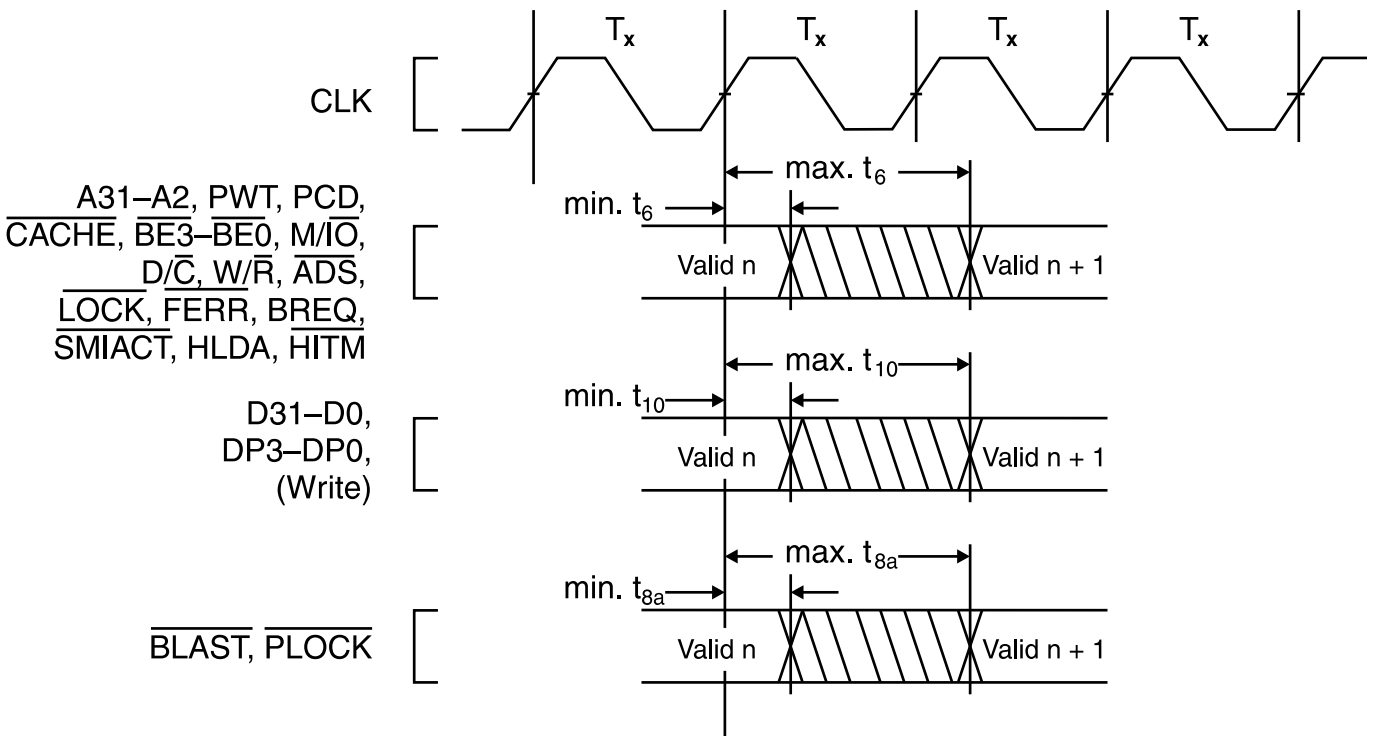| Waveform	Inputs	Outputs
	Must be steady	Will be steady
	May change from H to L	Will change from H to L
	May change from L to H	Will change from L to H
	Don't care; any change permitted	Changing; state unknown
	Does not apply	Center line is High-impedance "Off" state

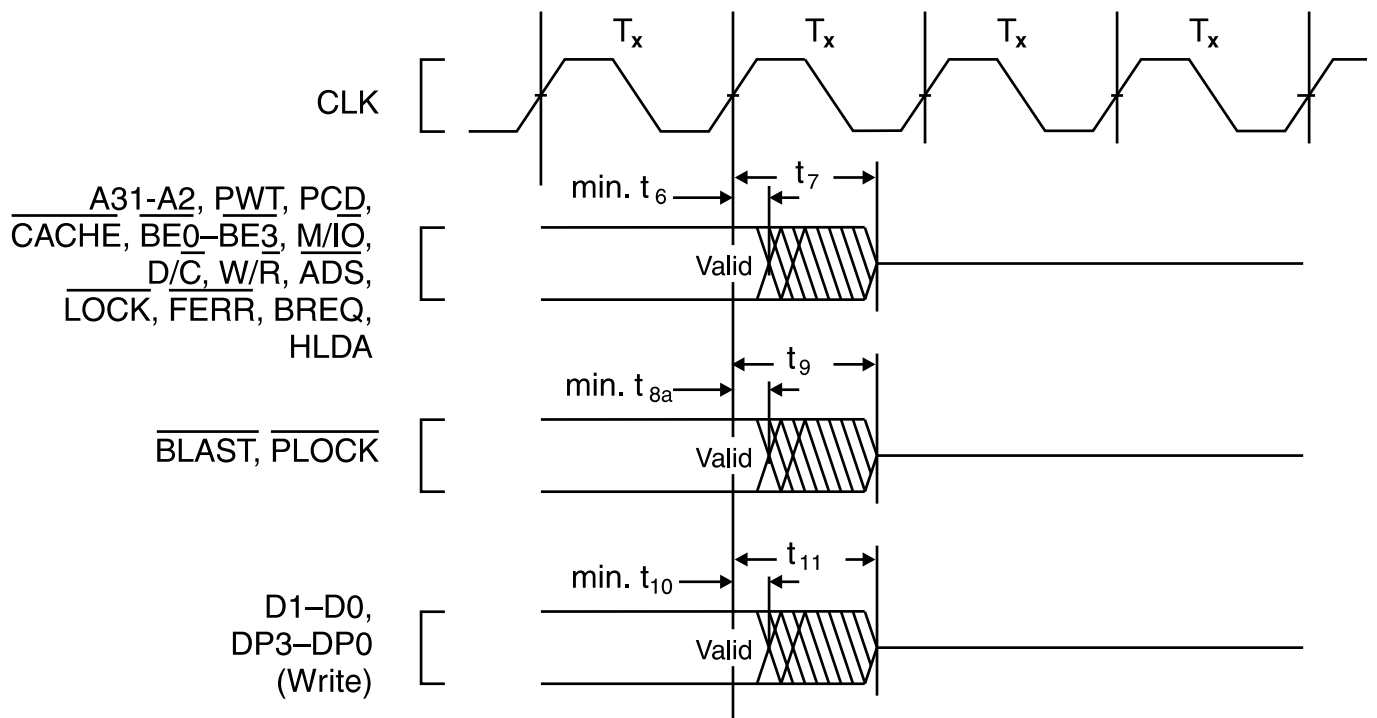**Figure 18. CLK Waveforms**

Figure 19. Output Valid Delay Timing

Figure 20. Maximum Float Delay Timing
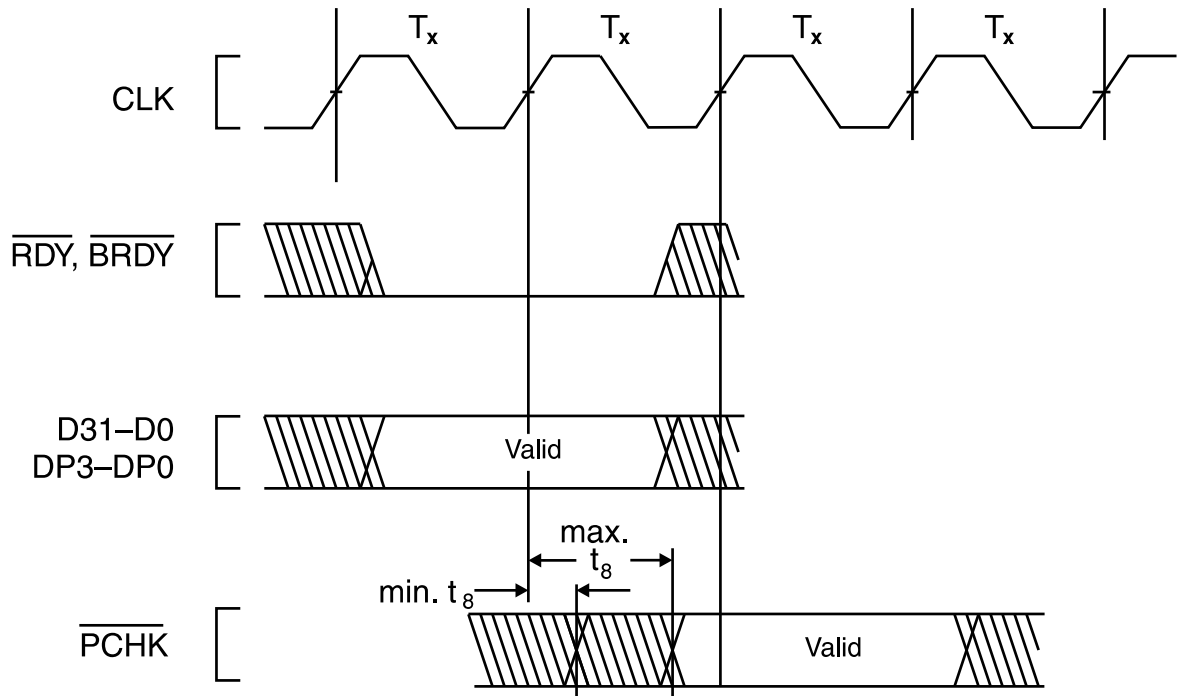
Figure 21. $\overline{\text{PCHK}}$ Valid Delay Timing
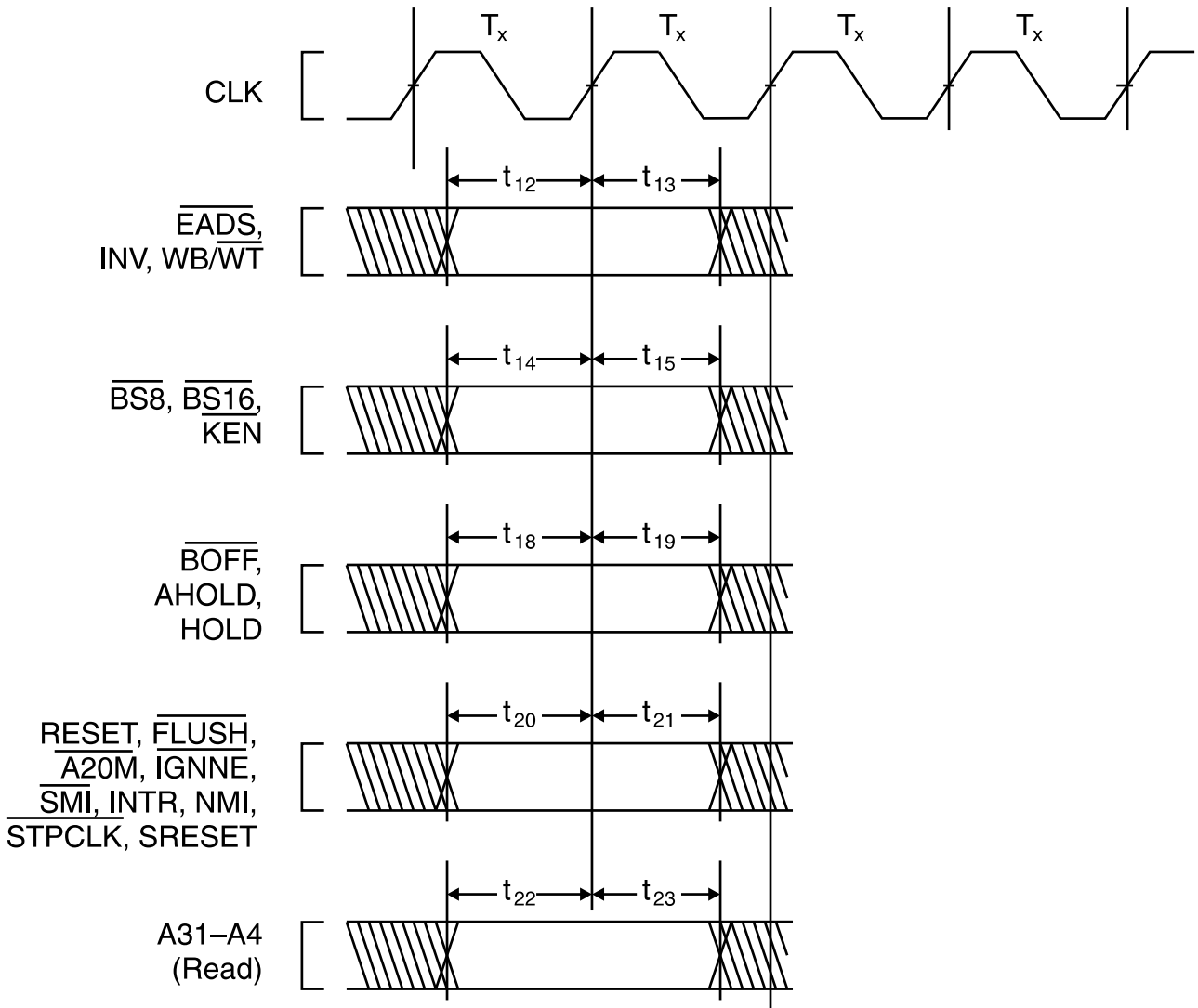
Figure 22. Input Setup and Hold Timing
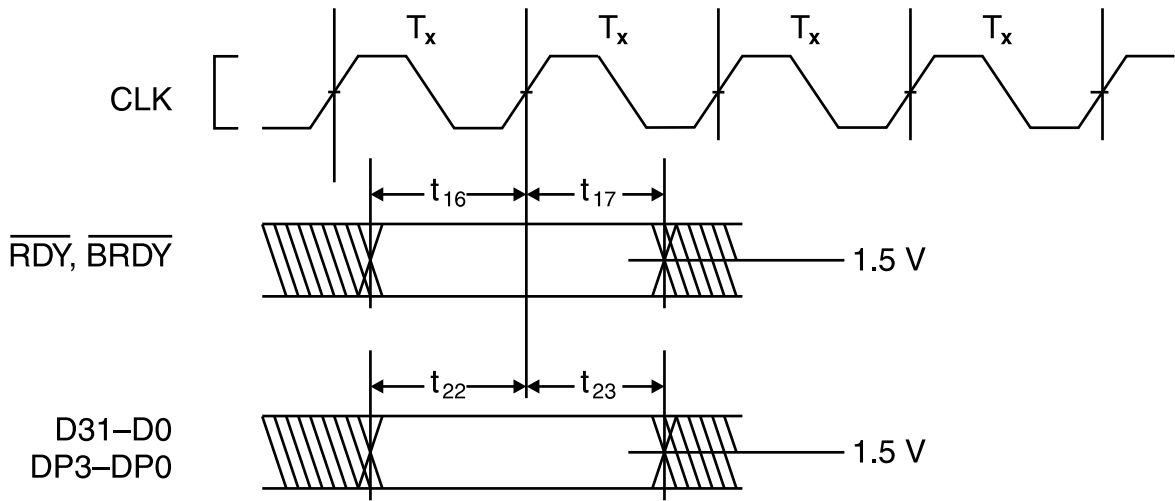
Figure 23. $\overline{RDY}$ and $\overline{BRDY}$ Input Setup and Hold Timing
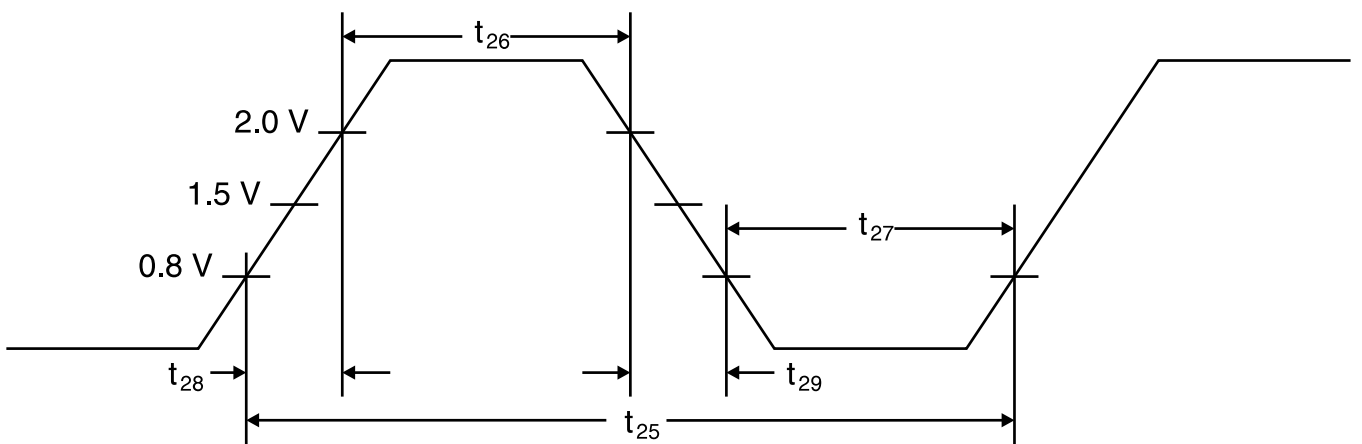
Figure 24. TCK Waveforms
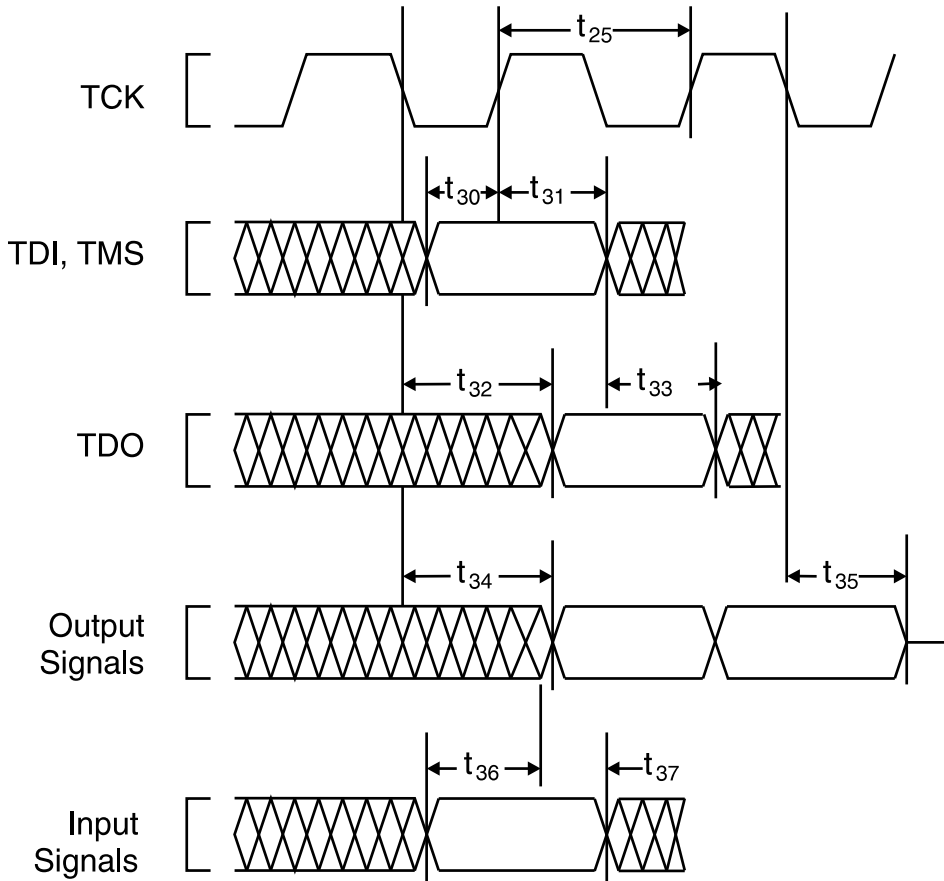
Figure 25. Test Signal Timing Diagram

PACKAGE THERMAL SPECIFICATIONS

The Am486 microprocessor is specified for operation when T_{CASE} (the case temperature) is within the range of 0°C to +85°C. T_{CASE} can be measured in any environment to determine whether the Am486DE2 microprocessor is within the specified operating range. The case temperature should be measured at the center of the top surface opposite the pins.

The ambient temperature (T_A) is guaranteed as long as T_{CASE} is not violated. The ambient temperature can be calculated from θ_{JC} and θ_{JA} and from the following equations:

$$T_J = T_{CASE} + P \cdot \theta_{JC}$$

$$T_A = T_J - P \cdot \theta_{JA}$$

$$T_{CASE} = T_A + P \cdot [\theta_{JA} - \theta_{JC}]$$

where:

T_J, T_A, T_{CASE} = Junction, Ambient, and Case Temperature.

θ_{JC}, θ_{JA} = Junction-to-Case and Junction-to-Ambient Thermal Resistance, respectively

P = Maximum Power Consumption

The values for θ_{JA} and θ_{JC} are given in Table 15 for the 1.75 sq. in., 168-pin, ceramic PGA. For the 208-lead SQFP plastic package, $\theta_{JA} = 14.0$ and $\theta_{JC} = 1.5$.

Table 16 shows the T_A allowable (without exceeding T_{CASE}) at various airflows and operating frequencies (Clock). Note that T_A is greatly improved by attaching fins or a heat sink to the package. Heat sink dimensions are shown in Figure 47. P (the maximum power consumption) is calculated by using the maximum I_{CC} at 3.3 V as tabulated in the *DC Characteristics.*

Table 15. Thermal Resistance (°C/W) θ_{JC} and θ_{JA} for the Am486DE2 in 168-Pin PGA Package

Cooling Mechanism	θ_{JC}	θ_{JA} vs. Airflow-ft/min. (m/sec)					
		0 (0)	200 (1.01)	400 (2.03)	600 (3.04)	800 (4.06)	1000 (5.07)
No Heat Sink	1.5	16.5	14.0	12.0	10.5	9.5	9.0
Heat Sink*	2.0	12.0	7.0	5.0	4.0	3.5	3.25
Heat Sink* and fan	2.0	5.0	4.6	4.2	3.8	3.5	3.25

0.350″ high unidirectional heat sink (Al alloy 6063-T5, 40 mil fin width, 155 mil center-to-center fin spacing).

17852B-113
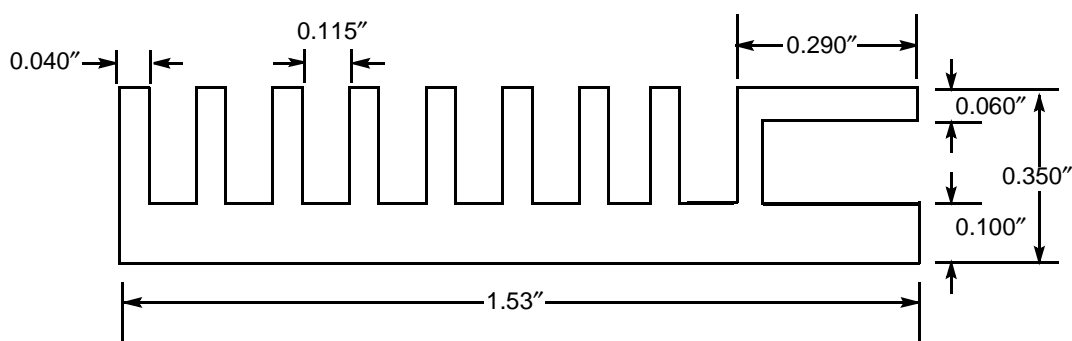
Figure 26. . Heat Sink Dimensions

Table 16. Maximum T_A at Various Airflows in °C

T_A by Cooling Type	Clock	Airflow-ft/min. (m/sec)					
		0 (0)	200 (1.01)	400 (2.03)	600 (3.04)	800 (4.06)	1000 (5.07)
T_A without Heat Sink	66 MHz	49.0	55.0	59.8	63.4	65.8	67.0
T_A with Heat Sink	66 MHz	61.0	73.0	77.8	80.2	81.4	82.0
T_A with Heat Sink and fan	66 MHz	77.8	78.8	79.7	80.7	81.4	82.0

AMD🡒

PHYSICAL DIMENSIONS

168-Pin PGA

Ceramic Pin Grid Array, CGM 168

Index Corner

1.735
1.765

1.595
1.605

1.595
1.605

1.735
1.765
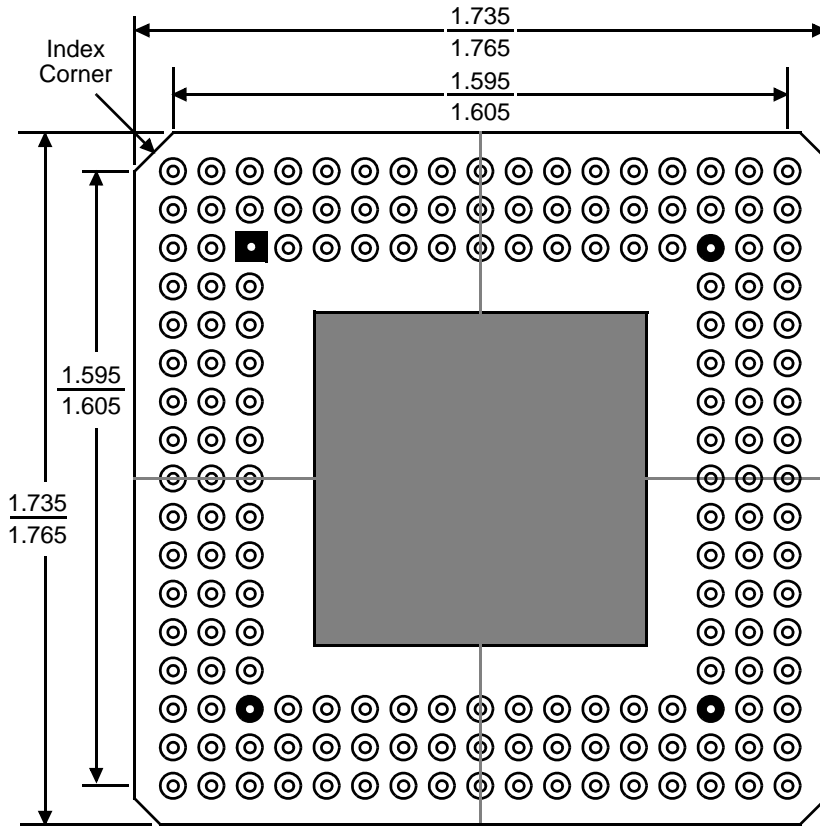
Bottom View (Pins Facing Up)

Base Plane

Seating Plane

0.017
0.020

0.090
0.110

0.105
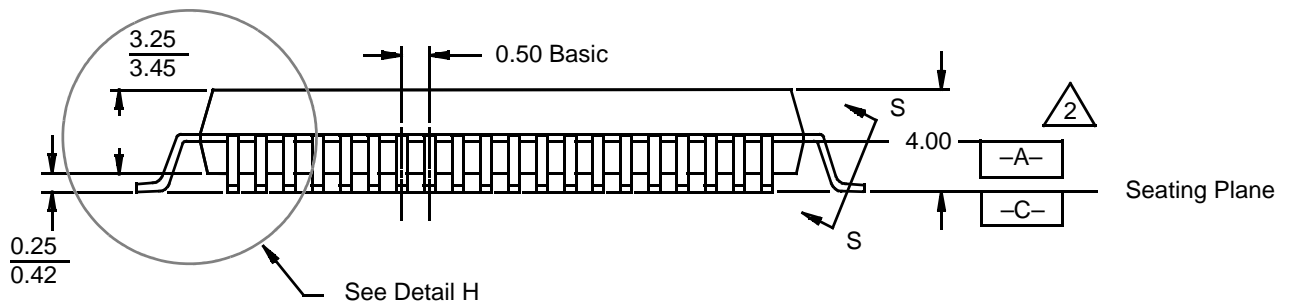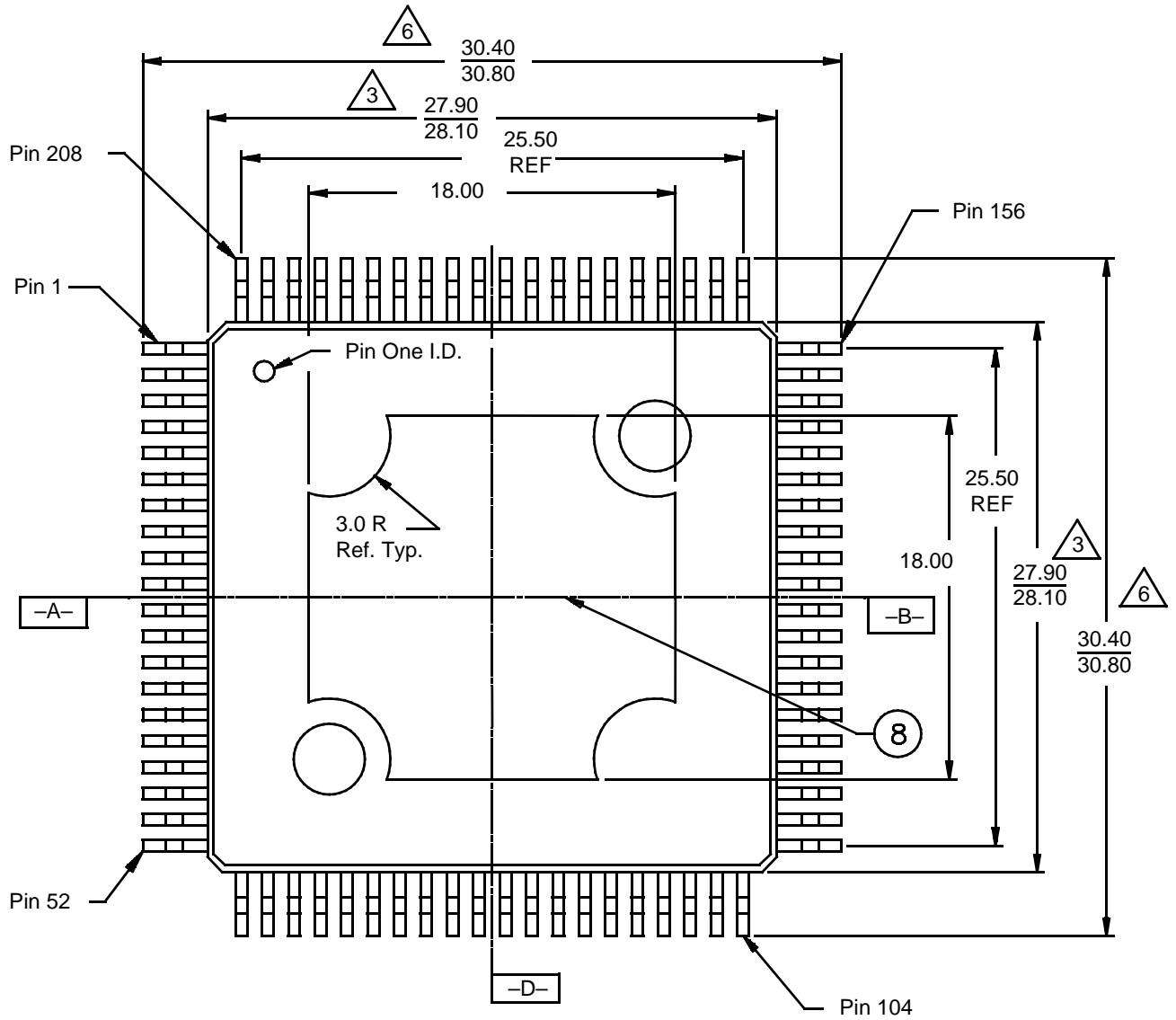0.125

0.025
0.045
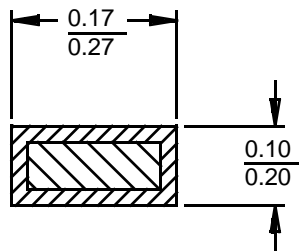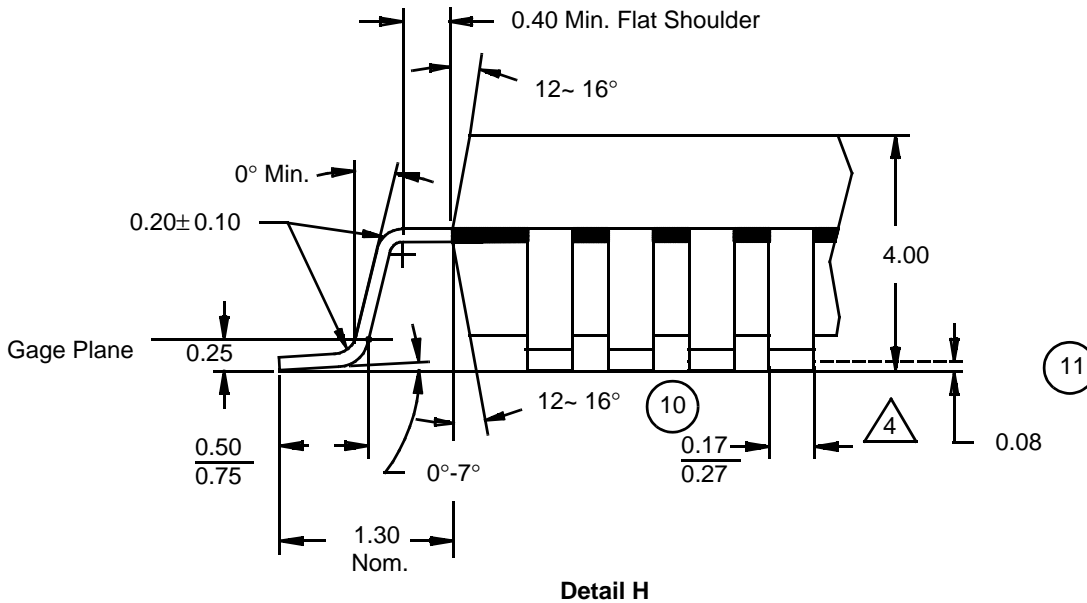
0.140
0.180

0.110
0.140
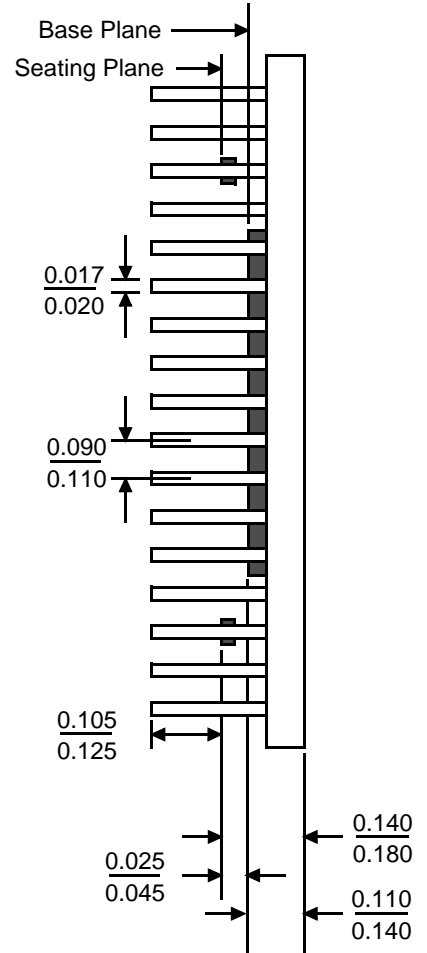
Side View

Notes:

1. All measurements are in inches.

2. Not to scale. For reference only.

3. BSC is an ANSI standard for Basic Space Centering.

208-Lead SQFP

Shrink Quad Flat Pack, PDE 208

AMD

0.40 Min. Flat Shoulder

12~ 16°

0° Min.

0.20± 0.10

4.00

Gage Plane   0.25

11

12~ 16°   10

0.17
0.27

4

0.08

0.50
0.75

0°-7°

1.30
Nom.

Detail H

0.17
0.27

0.10
0.20

Section S-S

Notes:

1. *All dimensions and tolerances conform to ANSI Y14.5M-1982.*

2. *DATUM Plane -A- is located at the mold parting line, and is coincedent with the bottom of the lead where the lead exits the plastic body.*

3. *Dimensions "D1" and "E1" do not include mold protrusions. Allowable protrusion is 0.25mm per side. Dimensions "D1" and "E1" include mold mismatch, and are determined at DATUM plane -A-.*

4. *Dimension "b" does not include DAMBAR protrusion.*

5. *Controlling dimensions: Millimeter.*

6. *Dimensions "D" and "E" are measured from outermost points.*

7. *Pin No. 1 ID may be inside top ejector mark, or separate.*

8. *Heatsink centerline to be aligned with package centerline±0.30.*

9. *Half span (Center of package to lead tip) shall be 15.30±0.165[0.602"±0.0065"]*

10. *No lead distortion (bent leads etc...) shall cause deviation from true position greater than±0.04[0.0016"] at "b max".*

11. *Lead coplanarity with respect to the seating plane shall not exeed 0.08[0.0031"].*