FEATURES

- **Available in 56- and 40-MHz speed grades**
- **System-on-a-chip solution**
  - — 32-bit ARM7 processor with MMU
  - — 4K unified cache
  - — FPU (floating point unit)
  - — Graphics controller drives CRT or LCD
  - — CD-quality sound audio controller
  - — DRAM controller
  - — ROM/Flash controller
  - — Three-channel DMA for video, cursor, and sound
  - — PC-style I/O bus
  - — Two-state power management
  - — Eight general-purpose I/O lines
- **Performance**
  - — 50 Vax™-MIPS (Dhrystone®) at 56 MHz
  - — Up to 12 Mflops, double-precision FP (LINPACK)
- **FPU**
  - — Implements ANSI/IEEE Std 754-1985
  - — Single, double, and extended precision

System-on-a Chip for Internet Appliance

OVERVIEW

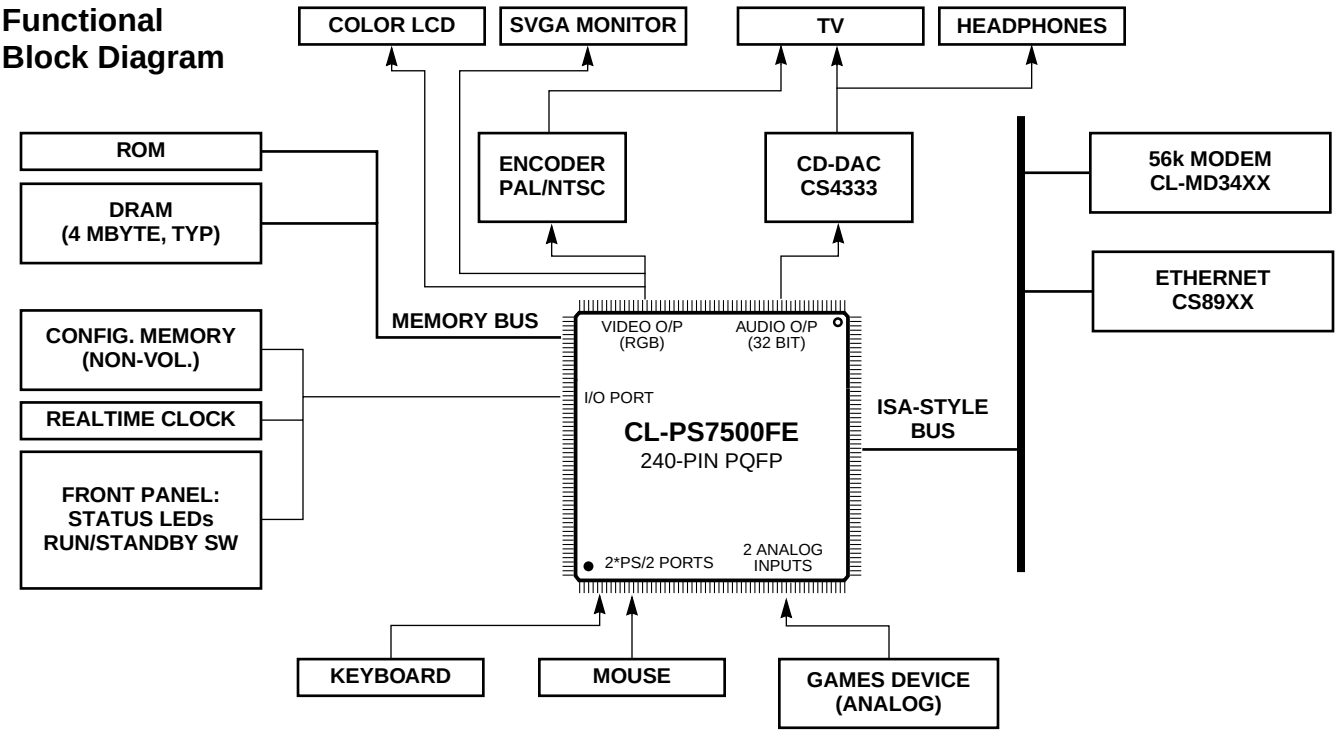
The Cirrus Logic CL-PS7500FE is designed to be used in internet appliances such as the network computer, smart-TV, intranet terminal, screen phones, DVD players, and so on.

The massively integrated CL-PS7500FE offers a complete system-on-a-chip solution that includes a 32-bit ARM CPU with cache and MMU, CRT and LCD controller, memory controller, FPU, CD-quality sound controller, interface to the Cirrus Logic DSP device for 56K modem and speakerphone, and a PC-type I/O bus. To handle streaming of audio and video data on the Internet, the CL-PS7500FE includes a double-precision FPU to accelerate software codecs.

(cont.)

(cont.)

Functional Block Diagram



FEATURES (cont.)

- **CRT or color/monochrome LCDs**
  - Resolutions up to 1024 × 768
  - 256-entry 28-bit video palette
  - Single- and dual-scan panel LCDs (16-bit grayscale)
- **Serial CD digital sound (32-bit) output**
- **Supports EDO and Fast page mode DRAMs**
  - Up to 132 Mbytes/sec. (peak) using 64-MHz memory clock and 32-bit-wide DRAM
  - Programmable 16- or 32-bit-wide memory system
  - Speed-critical paths are pipelined
- **PC-style I/O bus (40-MHz) for connection to any Cirrus Logic peripheral device**
  - 56k fax/modem chipset
  - CS89XX Ethernet controller
  - Can be expanded to 32 bits with external transceivers
- **ROM/FLASH**
  - Supports two 16-Mbyte banks
  - Individual read timings
  - Burst mode reads
  - Allows for writes under register control for FLASH

OVERVIEW (cont.)

The video controller features RGB drive of a SVGA monitor or a color LCD. It also incorporates various sync inputs, which when combined with an external encoder, permit the use of interlaced TV displays. The device incorporates a digital audio controller with a 32-bit serial interface for connection to the Cirrus Logic, CS4333 CD-DAC device. The CL-PS7500FE can also interface to the Cirrus Logic 56K, FastPath™ modem chipset ideal for Internet access over a POTS line.

The CL-PS7500FE is available in two speed grades:

- ARM CPU running at 40 MHz; memory clock running up to 64 MHz
- ARM CPU running at 56 MHz; memory clock running up to 64 MHz

The CL-PS7500FE supports UMA (unified memory architecture); EDO DRAMs can be used to achieve high-memory bandwidth.

The CL-PS7500FE is the main computing engine in the NC platform defined by Oracle®, and runs the NC operating system and applications.

The device is available in a 240-pin PQFP (plastic quad flat pack) package.

TABLE OF CONTENTS

Appendixes

CONVENTIONS

Abbreviations

Symbol	Units of measure
bpp	bits-per-pixel
°C	degree Celsius
Hz	hertz (cycles per second)
Kbyte	kilobyte (1,024 bytes)
kHz	kilohertz
kΩ	kilohm
Mbyte	megabyte (1,048,576 bytes)
MHz	megahertz (1,000 kilohertz)
µF	microfarad
µs	microsecond (1,000 nanoseconds)
mA	milliampere
ms	millisecond (1,000 microseconds)
ns	nanosecond
pV	picovolt
qword	quad word

The use of ‘tbd’ indicates values that are ‘to be determined’; ‘n/a’ designates ‘not available’; ‘n/c’ indicates a pin that is a ‘no connect’.

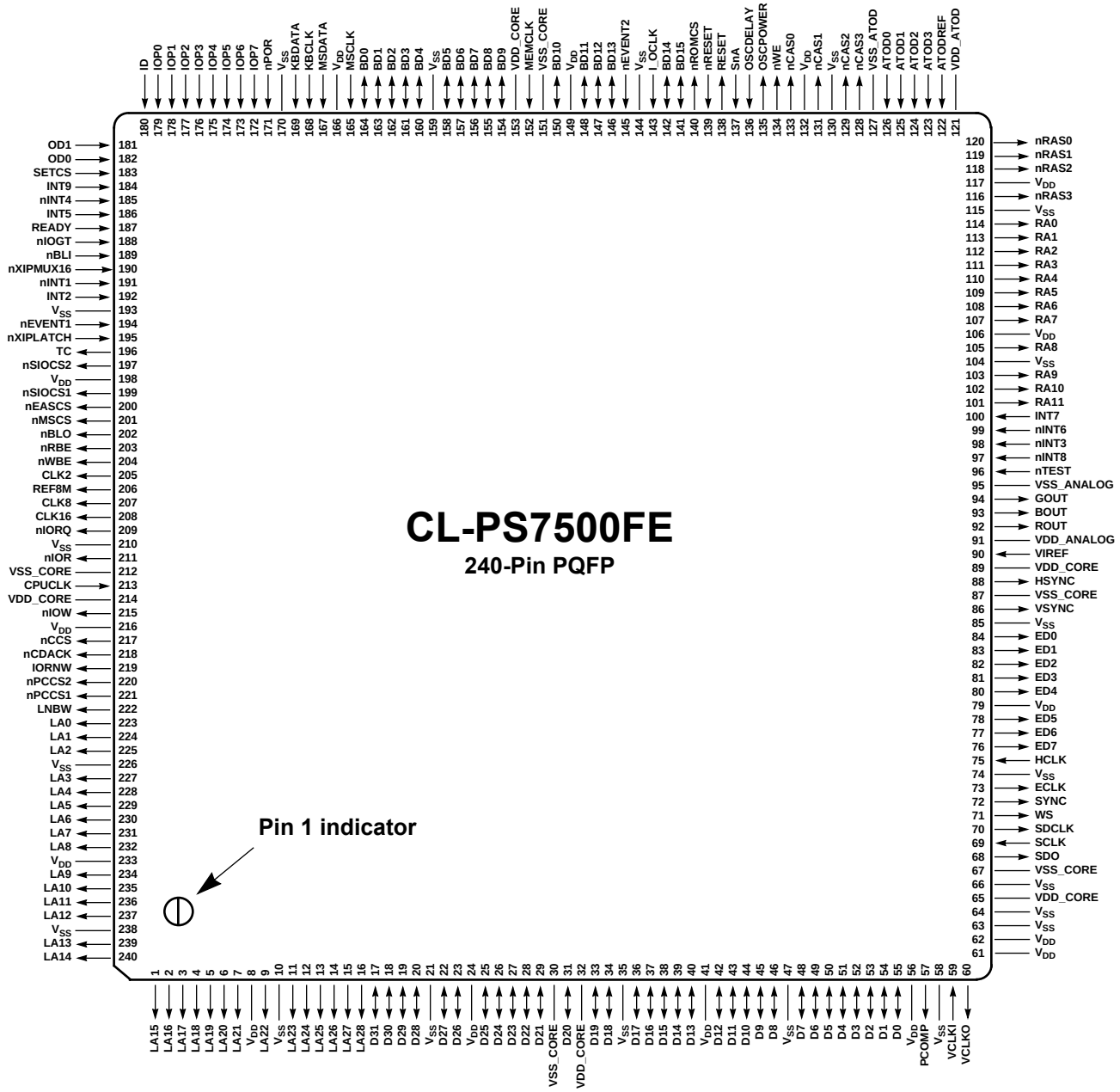
Acronyms

Acronym	Definition
AC	alternating current
ALU	arithmetic logic unit
AP	access permissions
A-to-D	analog-to-digital
BIOS	basic input/output system
CISC	complex instruction set computer
CMOS	complementary metal-oxide semi-conductor
CRT	cathode ray tube
DAC	digital-to-analog converter
DC	direct current
DMA	direct memory access
DRAM	dynamic random-access memory
DVD	digital video disk
EDO	extended data out
FIFO	first in/first out
FPA	floating point accelerator
FPU	floating point unit
GPIO	general-purpose IO
HBM	human body model
IC	integrated circuit
IDC	instruction and data cache
ISA	industry standard architecture
LSB	least-significant bit
LUT	lookup table
MFLOPS	million floating points per second
MMU	memory management unit
MSB	most-significant bit
MUX	multiplexer
NaN	not a number
PC	program counter
PLL	phased-locked loop
PQFP	plastic quad-flat pack
RAM	random-access memory
RISC	reduced instruction set computer
ROM	read-only memory
R/W	read/write
SRAM	static random-access memory
SWI	software interrupt instruction
TLB	translation look-aside buffer
TTB	translation table base
UMA	unified memory architecture
VCO	voltage-controlled oscillator
VRAM	video random-access memory
WB	write buffer
XIP	execute-in-place

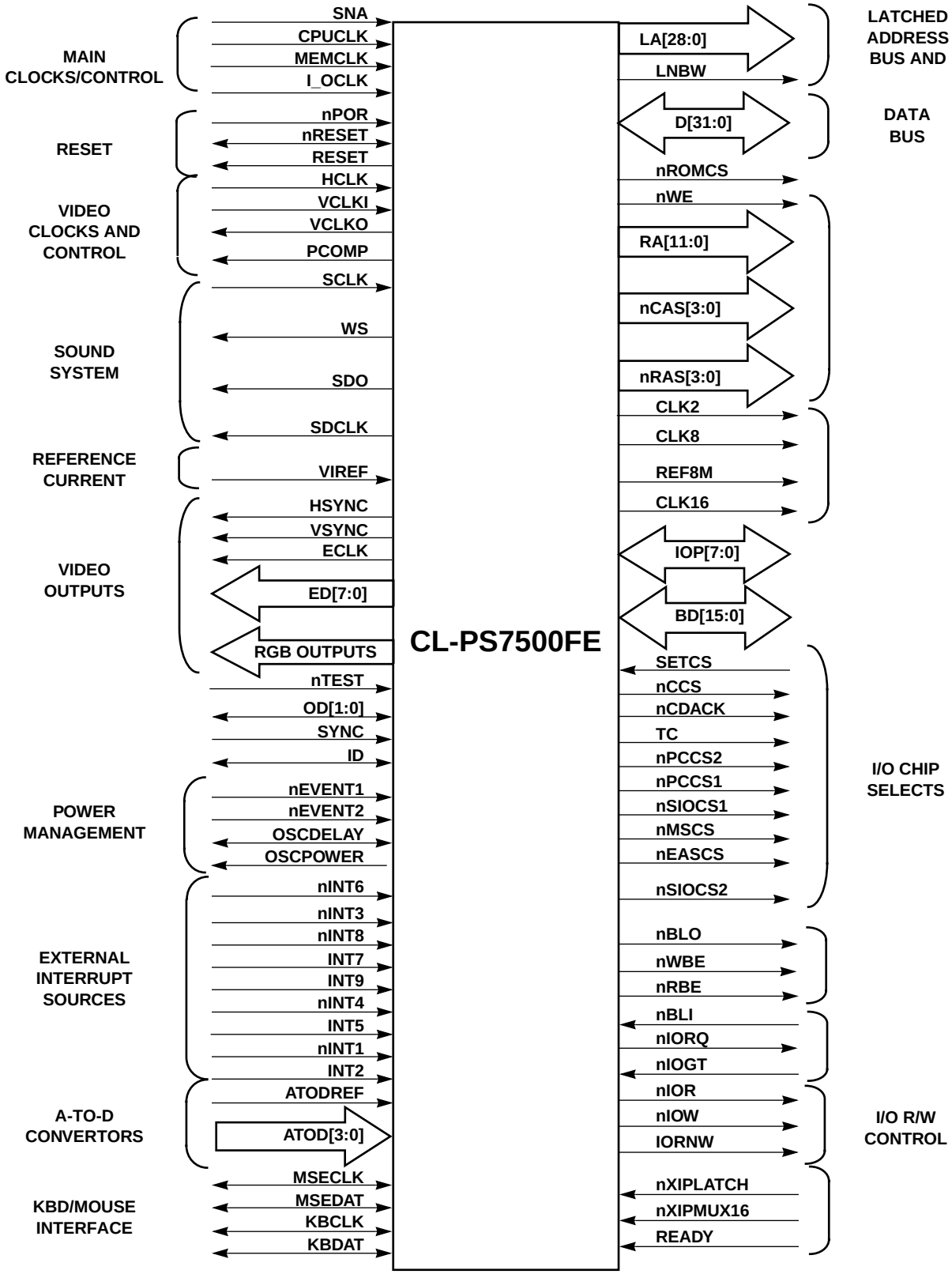
1. PIN INFORMATION

The CL-PS7500FE is available in a 240-pin PQFP (plastic quad flat pack) configuration.

1.1 Pin Diagram



1.2 Block Diagram



2. PIN DESCRIPTIONS

This chapter gives the name, type, and relevant details of each of the CL-PS7500FE signals.

NOTE: When output signals are placed in the high-impedance state for long periods, ensure that they do not 'float' to an undefined logic level.

The following abbreiviations are used to indicate signal types.

IC	Input, CMOS threshold
OCZ	Output, CMOS levels, tristate
IT	Input, TTL threshold
ICS	Input, CMOS Schmitt
IA	Input, analog
OA	Output, analog
BTZ	Bidirectional, CMOS output, TTL threshold input level
TOD	Open-drain, TTL input
CSOD	Open-drain, CMOS schmitt input
IAOD	Input, analog with programmable internal pull-down transistor

For outputs and bidirectional signals, drive strength is classified 1, 2, or 3. See Chapter 22 for DC and AC characteristics.

2.1 CL-PS7500FE Pin Descriptions

Name	Type	Drive Strength	Description
LA[28:0]	OCZ	2	**LATCHED ADDRESS BUS:** This bus is the latched version of the ARM address for memory accesses, changing on the falling edge of the internal MCLK signal.
LNBW	OCZ	2	**LATCHED NOT BYTE WORD:** This is a latched version of the internal NBW signal from the ARM processor, changing on the falling edge of the internal MCLK signal.
D[31:0]	BTZ	2	**DATA:** The main data bus for the CL-PS7500FE. All external data transfers happen through this bus. When the CL-PS7500FE is configured for operation in 16-bit mode, only the lower 16 bits are used.
SnA	IC		**SYNCHRONOUS/NOT ASYNCHRONOUS**: This pin is set according to the relationship required between the internal clock signals, MCLK and FCLK, for the ARM. If this pin is set high, both the memory system and the CPU are driven from the MEMCLK pin, and the required synchronous timing relationship between the ARM processor clocks is generated automatically on-chip. If different clocks are to be used for the MEMCLK and CPUCLK inputs, the SnA pin must be set low.

2.1 CL-PS7500FE Pin Descriptions *(cont.)*

Name	Type	Drive Strength	Description
BOUT	AO		**BLUE ANALOG OUTPUT:** The video signal analog outputs are designed to drive doubly-terminated 75-Ω lines.
ECLK	OCZ	3	**EXTERNAL CLOCK:** When enabled, this clock validates the data on ED[7:0]. In normal video mode, it runs at the pixel rate, but when LCD data is being produced, it runs at a quarter of the pixel rate.
ED[7:0]	OCZ	2	**EXTERNAL DATA:** This is the digital video output port of the CL-PS7500FE. From this, the digital equivalent of the analog output can be produced in any color, or data from the external palette may be produced. This can be used for a variety of purposes such as fading or supremacy. Also, data for driving LCD panels is output from this port. Data produced is validated by ECLK.
GOUT	AO		**GREEN ANALOG OUTPUT:** The video signal analog outputs are designed to drive doubly-terminated 75-Ω lines.
HCLK	IT		**HIGH SPEED CLOCK:** This is the clock used with video subsystem.
HSYNC	OCZ	3	**HORIZONTAL SYNCHRONIZATION:** There are two synchronization outputs on the CL-PS7500FE, HSYNC and VSYNC. Dependent on the state of bits 17 and 16 in the video External register, either a horizontal or a composite (NOR) sync can be output on this pin, in either polarity. The width of the HSYNC pulse is definable in units of 2 pixels.
PCOMP	OCZ	1	**PHASE COMPARATOR OUTPUT:** Used with VCLK pins.
ROUT	AO		**RED ANALOG OUTPUT:** The video signal analog outputs are designed to drive doubly-terminated 75-Ω lines.
SCLK	IT		**SOUND CLOCK:** This signal can be used to clock the sound system, when a clock asynchronous to the internal video reference clock is required.
SDCLK	OCZ	2	**SERIAL DATA CLOCK:** This clock validates serial sound data on its rising edge.
SDO	OCZ	2	**SERIAL DATA OUT:** Serial sound data is output from this pin.
SYNC	IT		**EXTERNAL SYN:** This signal is used to synchronize CL-PS7500FE with another video system.
VCLKI	IC		**PHASE COMPARATOR CLOCK IN** (for video subsystem).
VCLKO	OCZ	2	**PHASE COMPARATOR CLOCK OUT** (for video subsystem).

2.1 CL-PS7500FE Pin Descriptions *(cont.)*

Name	Type	Drive Strength	Description
VIREF	IA		**VIDEO REFERENCE CURRENT:** The video DACs need a reference current to calibrate them. A constant current source is recommended, although a resistor up to V_{DD} is sufficient for many applications. This current also generates the constant source for the A-to-D comparators.
VSYNC	OCZ	3	**VERTICAL SYNCHRONIZATION:** Dependent on the state of bits 19 and 18 in the external register, either a vertical or a composite (XNOR) sync can be output on this pin, in either polarity. The width of the **VSYNC** pulse can be defined in units of a raster.
WS	OCZ	2	**WORD SELECT:** This signal denotes whether the output serial data is for the left-hand or right-hand stereo channel.
nTEST	IT		**TEST MODE INPUT:** This pin should be held permanently high. It is only intended to be used during production test of the CL-PS7500FE. An on-chip pull-up resistor is included, but it is advised to apply an external pull-up resistor to this pin.
nWE	OCZ	2	**WRITE ENABLE:** This is a active-low signal.
RA[11:0]	OCZ	2	**DRAM ROW/COLUMN MULTIPLEXED ADDRESS BUS:** Addresses for this bus are decoded from the ARM processor address for normal memory accesses, and are generated by the DMA controller for DMA.
nRAS[3:0]	OCZ	3	**DRAM ROW ADDRESS STROBES:** Each of these selects one of the four banks of DRAM available.
nCAS[3:0]	OCZ	3	**DRAM COLUMN ADDRESS STROBES:** These select the byte within the word for DRAM accesses.
ATOD[3:0]	IAOD		**ANALOG-TO-DIGITAL:** These are the four A-to-D channel input voltages.
ATODREF	IA		**ANALOG-TO-DIGITAL REFERENCE:** This is the reference voltage for the A-to-D converter comparators.
OSCPOWER	OCZ	1	**OCILLATOR POWER:** This is the enable signal for the system oscillator(s). When low, this signal can be used to disable the external oscillator(s).
OSCDELAY	CSOD	1	**OCILLATOR DELAY:** This signal requires an RC network to generate a fixed delay when restarting the system oscillator(s) on exit from STOP mode.

2.1 CL-PS7500FE Pin Descriptions *(cont.)*

Name	Type	Drive Strength	Description
RESET	OCZ	2	**RESET OUTPUT:** This is the synchronized version of internal system reset signal.
nRESET	CSOD	2	**RESET:** This is an open-drain output and a 'soft' reset input. This pin is sampled every 1µs for reset events, so to guarantee a successful reset, a reset pulse applied to this pin must be longer than 1µs (1µs, assuming the internal I/O clock is 32 MHz).
nROMCS	OCZ	1	**ROM CHIP SELECT:** This signal goes low to indicate a ROM access.
I_OCLK	IC		**I/O SYSTEM CLOCK:** This clock input should always be 32 MHz when in Divide-by-1 mode, and 64 MHz in Divide-by-2 mode.
MEMCLK	IC		**MEMORY SYSTEM CLOCK:** In synchronous mode, the ARM processor FCLK is also driven from this clock.
CPUCLK	IC		This clock creates FCLK for the ARM CPU in asynchronous mode. When SnA is high, this signal should be permanently tied high or low.
BD[15:0]	BTZ	2	This is the main external 16-bit I/O bus.
MSCLK	TOD	2	**MOUSE CLOCK:** An open-drain pin for the mouse PS/2 interface.
MSDATA	TOD	2	**MOUSE DATA:** An open-drain pin for the mouse PS/2 interface.
KBCLK	TOD	2	**KEYBOARD CLOCK:** An open-drain pin for the keyboard PS/2 interface.
KBDATA	TOD	2	**KEYBOARD DATA:** An open-drain pin for the keyboard PS/2 interface.
nPOR	ICS		**POWER ON RESET:** Any low transitions on this pin are detected and stretched to ensure a full reset.

CPUCLK [MHz]	MEMCLK [MHz]	SnA	Notes
Low	40	High	
40	56	Low	
40	64	Low	
Low	56	High	Recommended
56	64	Low	

2.1 CL-PS7500FE Pin Descriptions *(cont.)*

Name	Type	Drive Strength	Description
IOP[7:0]	TOD	1	**I/O PORT:** This is the 8-bit-wide I/O port. Each bit is directly controllable through an CL-PS7500FE register, and can be used as an interrupt source if required.
ID	TOD	1	**ID:** This pin activates a system ID chip. It is forced low during the power-on reset sequence.
OD[1:0]	TOD	1	**OPEN DRAIN 1:0:** These are the two open-drain pins, which (unlike the IOP[7:0] bus) cannot be used to generate interrupts, but can be used as general-purpose I/O pins (for example to communicate with a realtime clock chip).
SETCS	IC		This signal selects between two address decoding options for the three main I/O chip selects. It affects the outputs nEASCS, nMSCS, and nSIOCS2.
nINT1	IT		This is a falling-edge-triggered interrupt. The nINT1 value can be read directly in the IOCR I/O control register.
INT2	IT		This is a rising-edge-triggered interrupt pin that can generate an IRQ interrupt.
nINT3	IT		This is an active-low interrupt that can generate an IRQ interrupt.
nINT4	IT		This is an active-low interrupt that can generate an IRQ interrupt.
INT5	IT		This is an active-high interrupt that can generate either an IRQ or a FIQ interrupt, depending on the status of the relevant mask register bits.
nINT6	IT		This is an active-low interrupt that can generate either an IRQ or a FIQ interrupt, depending on the programming of the mask registers.
INT7	IT		This is an active-high interrupt that can generate an IRQ interrupt.
nINT8	IT		This is an active-low interrupt that can generate either a FIQ or an IRQ interrupt.
INT9	IT		This is an active-high interrupt that can only generate a FIQ (highest priority) interrupt.
nEVENT1	IT		This is the active-low asynchronous event pin 1. A falling edge terminates the STOP or SUSPEND power-saving modes.
nEVENT2	IT		This is the active-low asynchronous event pin 2. A falling edge terminates the STOP or SUSPEND power-saving modes.

2.1 CL-PS7500FE Pin Descriptions *(cont.)*

Name	Type	Drive Strength	Description
READY	IT		**READY:** This pin can stretch I/O accesses when set low during a 16-MHz PC-type I/O cycle.
nIORQ	OCZ	2	**I/O REQUEST:** This signal is for the module-type I/O for handshaking, together with nIOGT.
nIOGT	IT		**I/O GRANT:** This signal is for the module-type I/O for handshaking, together with nIORQ.
nBLI	IT		This input is used during module-type I/O reads to cause the latching of data from the BD port.
nBLO	OCZ	1	This signal is the latching signal for use with external latches on the upper 16 bits of the external datapath to create a 32-bit-wide I/O bus.
nRBE	OCZ	1	This active-low read enable is used to create a 32-bit-wide I/O bus for an external transceiver attached to the upper 16 bits of the I/O bus.
nWBE	OCZ	1	This active-low write enable is used to create a 32-bit-wide I/O bus for an external transceiver attached to the upper 16 bits of the I/O bus.
nXIPMUX16	IT		This signal is for XIP (execute in place) support. This signal multiplexes 16 bits of data from the upper or lower halfword of the CL-PS7500FE internal data bus to the 16-bit I/O bus, depending on its state during writes.
nXIPLATCH	IT		This signal is for XIP support and latches the upper 16 bits of data from the I/O bus while the lower 16 bits are being read. This signal is used in conjunction with nXIPMUX16 to enable XIP (for example, from a 16-bit PCMCIA card).
nSIOCS1	OCZ	1	This is the active-low chip select for simple I/O.
nSIOCS2	OCZ	1	This is the active-low chip select for simple I/O, with address decode modified according to the state of SETCS.
nMSCS	OCZ	2	This is the active-low chip select for module-type I/O, with address decode modified according to the state of SETCS.
nEASCS	OCZ	1	This is the active-low chip select for extended 16-MHz PC-type I/O, with address decode modified according to the state of SETCS.
nCCS	OCZ	1	**NOT COMBO CHIP SELECT:** This is the chip select signal for a PC Combo chip.

2.1 CL-PS7500FE Pin Descriptions *(cont.)*

Name	Type	Drive Strength	Description
nCDACK	OCZ	1	**NOT COMBO DACK:** This is the chip select and DACK signal for a PC Combo chip.
TC	OCZ	1	**TERMINAL COUNT:** This active-high signal is used in conjunction with the nCDACK signal for pseudo DMA to a PC Combo chip.
nPCCS1	OCZ	1	This is the active-low chip select for an area of 16-MHz PC-type I/O space.
nPCCS2	OCZ	1	This is the active-low chip select for an area of 16-MHz PC-type I/O space.
LNBW	OCZ	2	**LATCHED NOT BYTE WORD:** This is a latched version of the internal NBW signal from the ARM processor, which transitions on the falling edge of the internal MCLK signal.
IORNW	OCZ	2	**I/O READ/NOT WRITE:** This signal goes high during an I/O read, and low during an I/O write.
nIOR	OCZ	2	**NOT I/O READ:** This signal has two functions: **1)** It transitions low during simple and PC-type I/O reads; not used for module-type I/O. **2)** It is asserted low during ROM read cycles to act as an output enable.
nIOW	OCZ	2	**NOT I/O WRITE:** This signal has two functions: **1)** It transitions low during simple and PC-type I/O reads; not used for module-type I/O. **2)** It is asserted low during writes to ROM space, to act as a write enable, if writes are enabled in the ROMCR register.
CLK2	OCZ	2	This is the 2-MHz I/O clock output.
CLK8	OCZ	2	This is the 8-MHz I/O clock output, the inverted version of REF8M.
REF8M	OCZ	2	This is the 8-MHz I/O clock output.
CLK16	OCZ	2	This is the 16-MHz I/O clock output, for PC-type I/O.

2.2 Power and Ground Pins

Name	Pin No.	Description
V_{DD}	8 24 41 56 61 62 79 106 117 132 149 166 198 216 233	These 15 pins supply +5 volts to the digital logic of the CL-PS7500FE. Each pin must be connected to the V_{CC} plane.
V_{SS}	10 21 35 47 58 63 64 66 74 85 104 115 130 144 159 170 193 210 226 238	These 20 pins supply the ground reference to the digital logic of the CL-PS7500FE. Each pin must be connected to the ground plane.
VDD_CORE	32 65 89 153 214	These five pins supply +5 volts to the core logic of the CL-PS7500FE. Each pin must be connected to the V_{CC} plane.

2.2 Power and Ground Pins *(cont.)*

Name	Pin No.	Description
VSS_CORE	30 67 87 151 212	These five pins supply the ground reference to the core logic. Each pin must be connected to the ground plane.
VDD_ATOD	121	This pin is the positive (+5V) supply for the A-to-D converter comparators. This pin must be connected directly to the V_{CC} plane.
VSS_ATOD	127	This pin is the analog ground for the A-to-D converter comparators. This pin must be connected to the ground plane.
VDD_Analog	91	This pin supplies the positive (+5V) supply for analog video system. This pin must be connected directly to the V_{CC} plane.
VSS_Analog	95	This pin supplies ground for analog video system. This pin must be connected to the ground plane.

2.3 Numerical Pin Listing

The following table is a numeric listing of the pins of the CL-PS7500FE.

Pin number	Signal name
1	LA[15]
2	LA[16]
3	LA[17]
4	LA[18]
5	LA[19]
6	LA[20]
7	LA[21]
8	V_{DD}
9	LA[22]
10	V_{SS}
11	LA[23]
12	LA[24]
13	LA[25]
14	LA[26]
15	LA[27]
16	LA[28]
17	D[31]
18	D[30]
19	D[29]
20	D[28]
21	V_{SS}
22	D[27]
23	D[26]
24	V_{DD}
25	D[25]
26	D[24]
27	D[23]
28	D[22]
29	D[21]

Pin number	Signal name
30	VSS_CORE
31	D[20]
32	VDD_CORE
33	D[19]
34	D[18]
35	V_{SS}
36	D[17]
37	D[16]
38	D[15]
39	D[14]
40	D[13]
41	V_{DD}
42	D[12]
43	D[11]
44	D[10]
45	D[9]
46	D[8]
47	V_{SS}
48	D[7]
49	D[6]
50	D[5]
51	D[4]
52	D[3]
53	D[2]
54	D[1]
55	D[0]
56	V_{DD}
57	PCOMP
58	V_{SS}

Pin number	Signal name
59	VCLKI
60	VCLKO
61	V_{DD}
62	V_{DD}
63	V_{SS}
64	V_{SS}
65	VDD_CORE
66	V_{SS}
67	VSS_CORE
68	SDO
69	SCLK
70	SDCLK
71	WS
72	SYNC
73	ECLK
74	V_{SS}
75	HCLK
76	ED[7]
77	ED[6]
78	ED[5]
79	V_{DD}
80	ED[4]
81	ED[3]
82	ED[2]
83	ED[1]
84	ED[0]
85	V_{SS}
86	VSYNC
87	VSS_CORE

Pin number	Signal name
88	HSYNC
89	VDD_CORE
90	VIREF
91	VDD_ANALOG
92	ROUT
93	BOUT
94	GOUT
95	VSS_ANALOG
96	nTEST
97	nINT8
98	nINT3
99	nINT6
100	INT7
101	RA[11]
102	RA[10]
103	RA[9]
104	V_{SS}
105	RA[8]
106	V_{DD}
107	RA[7]
108	RA[6]
109	RA[5]
110	RA[4]
111	RA[3]
112	RA[2]
113	RA[1]
114	RA[0]
115	V_{SS}
116	nRAS[3]
117	V_{DD}
118	nRAS[2]
119	nRAS[1]

Pin number	Signal name
120	nRAS[0]
121	VDD_ATOD
122	ATODREF
123	ATOD[3]
124	ATOD[2]
125	ATOD[1]
126	ATOD[0]
127	VSS_ATOD
128	nCAS[3]
129	nCAS[2]
130	V_{SS}
131	nCAS[1]
132	V_{DD}
133	nCAS[0]
134	nWE
135	OSCPOWER
136	OSCDELAY
137	SnA
138	RESET
139	nRESET
140	nROMCS
141	BD[15]
142	BD[14]
143	I_OCLK
144	V_{SS}
145	nEVENT2
146	BD[13]
147	BD[12]
148	BD[11]
149	V_{DD}
150	BD[10]
151	VSS_CORE

Pin number	Signal name
152	MEMCLK
153	VDD_CORE
154	BD[9]
155	BD[8]
156	BD[7]
157	BD[6]
158	BD[5]
159	V_{SS}
160	BD[4]
161	BD[3]
162	BD[2]
163	BD[1]
164	BD[0]
165	MSCLK
166	V_{DD}
167	MSDATA
168	KBCLK
169	KBDATA
170	V_{SS}
171	nPOR
172	IOP[7]
173	IOP[6]
174	IOP[5]
175	IOP[4]
176	IOP[3]
177	IOP[2]
178	IOP[1]
179	IOP[0]
180	ID
181	OD[1]
182	OD[0]
183	SETCS

Pin number	Signal name
184	INT9
185	nINT4
186	INT5
187	READY
188	nIOGT
189	nBLI
190	nXIPMUX16
191	nINT1
192	INT2
193	V_{SS}
194	nEVENT1
195	nXIPLATCH
196	TC
197	nSIOCS2
198	V_{DD}
199	nSIOCS1
200	nEASCS
201	nMSCS
202	nBLO
203	nRBE
204	nWBE
205	CLK2
206	REF8M
207	CLK8
208	CLK16
209	nIORQ
210	V_{SS}
211	nIOR
212	VSS_CORE
213	CPUCLK
214	VDD_CORE
215	nIOW

Pin number	Signal name
216	V_{DD}
217	nCCS
218	nCDACK
219	IORNW
220	nPCCS2
221	nPCCS1
222	LNBW
223	LA[0]
224	LA[1]
225	LA[2]
226	V_{SS}
227	LA[3]
228	LA[4]
229	LA[5]
230	LA[6]
231	LA[7]
232	LA[8]
233	V_{DD}
234	LA[9]
235	LA[10]
236	LA[11]
237	LA[12]
238	V_{SS}
239	LA[13]
240	LA[14]

3. FUNCTIONAL DESCRIPTION

The CL-PS7500FE is a high-performance, low-power RISC-based single-chip computer built around an ARM microprocessor core. To maximize the potential of the ARM processor macrocell, the CL-PS7500FE contains memory and I/O control on-chip, enabling the direct connection of external memory devices and peripherals with the minimum of external components. The FPA (floating-point accelerator) is also integrated, resulting in outstanding math performance.

NOTE: The CL-PS7500FE is based on the ARM7500FE architecture from ARM Ltd., U.K. (http://www.arm.com).

The CL-PS7500FE includes features that make it particularly suitable for low-power portable applications. Both 32- and 16-bit-wide memory systems are supported, allowing the design of a lower-cost 16-bit-based system. The CL-PS7500FE drives color CRT or color LCD panels. Monochrome single- or dual-panel LCDs with 16 levels of greyscaling can also be driven. Power-management circuitry is included with two power-saving states. The high level of integration achieved allows significant PCB area saving, and results in a very cost-competitive system.

The CL-PS7500FE is also particularly suited to any application requiring high-quality video, sound, and general I/O requirements, such as multimedia. The video controller provides up to 16 million colors from a 256-entry palette, running at up to a 120-MHz pixel clock rate. The sound subsystem includes a serial sound interface for CD-quality 32-bit sound. Four on-chip A-to-D converters allow the connection of analog joysticks or similar control devices. The clocking scheme is very flexible, allowing either a very cheap system to be built using a single oscillator, or separate asynchronous clocks to be used for the CPU, memory and I/O subsystems, which gives an extremely flexible system, able to take advantage of the fastest available DRAM memory.

3.1 Functional Block Diagram

Figure 3-1 on page 28 presents a more detailed view of the functionality of the CL-PS7500FE single-chip computer.

3.2 ARM Processor Macrocell

The ARM processor contains an ARM7 core with MMU, 4-Kbyte cache, and write buffer.

3.3 FPA Macrocell

The FPA is a fully IEEE-754 compliant floating-point accelerator, and supports single, double, and extended precision formats. It is connected to the ARM through the coprocessor interface and provides the same floating-point functionality as the FPA11.

Concurrent load/store and arithmetic units, and speculative execution are employed to give good floating-point performance.

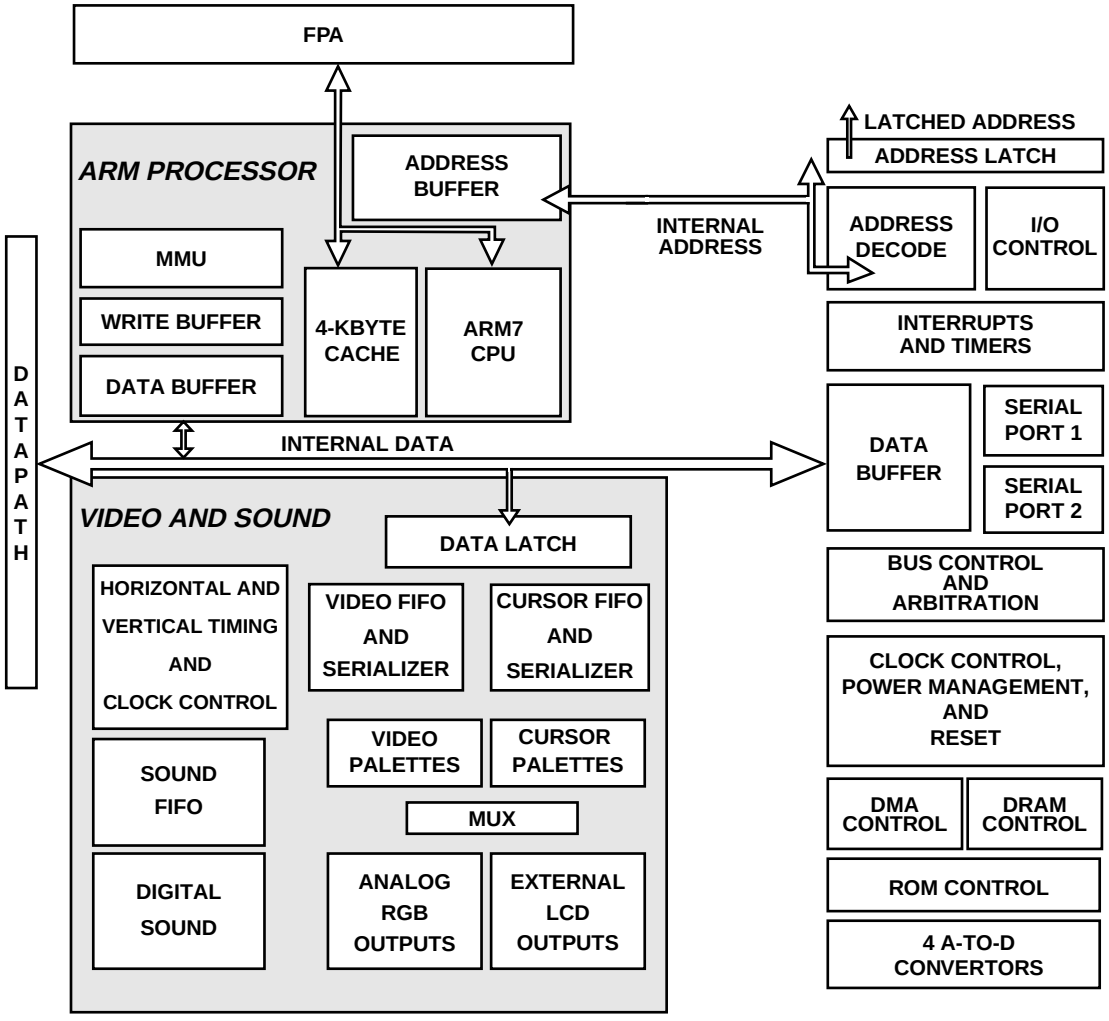


Figure 3-1. Functional Block Diagram of the CL-PS7500FE

3.4 Video and Sound Macrocell

The video and sound macrocell gives the CL-PS7500FE the flexibility to drive high analog CRT or low power LCD displays, and features the following:

- Up to 120-MHz pixel clock rate
- Resolutions of up to 1024 × 768 pixels are directly supported (greater if external serialization is used)
- Fully programmable display parameters
- 256-entry by 28-bit video palette
- Red, green and blue 8-bit linear DACs to drive CRT
- 1-, 2-, 4-, 8-, 16-, and 32-bpp CRT modes
- Up to 16 million colors
- External bits in palette for supremacy, fading, HiRes
- Single- or dual-panel LCD driving
- 16-level gray scalar for LCD
- Power management features
- Hardware cursor for all display modes
- Sound system — serial CD digital output

3.5 Clock Control and Power Management

The clocking strategy for CL-PS7500FE has been designed for maximum flexibility, and includes separate clock inputs for the:

- CPU core clock
- Memory system clock
- I/O system clock (in addition to the video clock inputs).

Each of the three clock inputs has a selectable divide-by-two prescalar to generate an internal 50/50 mark-space ratio if required. Throughout this data book, all timing diagrams assume that CPUCLK, MEMCLK, and I_OCLK are divided by one.

There are two levels of power management included:

- **SUSPEND mode:** The clock to the CPU is stopped, but the display continues to work normally, that is, DMA unaffected.
- **STOP mode:** All clocks are stopped. Two asynchronous wake-up event pins are provided to terminate stop mode. Circuitry is included on-chip to stop external oscillators and restart them cleanly when required.

3.6 Memory System

The memory system interface control logic is completely asynchronous in operation to the I/O control logic. This means that the clock to the memory controller can be increased in frequency to allow faster memory to be used. This implementation gives maximum system flexibility.

The CL-PS7500FE can control a 32- or 16-bit-wide memory system. The width of each bank of ROM or DRAM is selectable by programming appropriate register bits. Fast page mode or EDO DRAMs are supported.

A DRAM controller is included that can directly drive up to four banks of DRAM. Four nRAS strobes individually select one of the four banks, and four nCAS strobes provide individual byte selection. The DRAM address multiplexing option provided allows a wide variety of DRAM sizes to be used – from 256 Kbytes to beyond 16 Mbytes. Up to 256 page mode transfers can occur in one sequential burst. When configured for operation with a 16-bit-DRAM system, the DRAM controller converts the access into two DRAM cycles to access the two halves of the 32-bit word. Byte transfers only take one DRAM access cycle, even in 16-bit mode.

A programmable register allows one of four DRAM refresh rates to be selected. In addition, a register is provided to enable direct software control of the nCAS and nRAS lines for setting DRAM into a self-refresh state.

A ROM controller supports two 16-Mbyte banks of ROM with individually programmable read cycle timings. Support is provided for burst mode reads. Each ROM bank can be programmed to operate in 16-bit-wide mode and, like the DRAM controller, converts accesses into two ROM cycles for the two halves of the 32-bit word. The ROM controller can be programmed to allow write cycles through this interface, allowing FLASH to be programmed.

3.6.1 DMA

Three fully programmable DMA channels are included, for video, cursor, and sound data. The DMA controller includes additional support for dual-panel LCDs.

3.6.2 I/O Control

The I/O bus of the CL-PS7500FE is 16-bits wide, but for some types of access can be expanded to 32 bits by the use of external transceivers. The input clock I_OCLK provides a reference for the I/O subsystem nominally 32 MHz. The I/O features of this device can be separated into three distinct cycle types:

- Simple I/O with fixed 8-MHz timings
- Module I/O with variable length 8-MHz timings
- PC bus style I/O with fixed 16-MHz timings and support for 32-bit data

Simple I/O

The Simple I/O type of access is 16-bit-only and has a selection of four different cycle speeds selectable by address. When writing, the upper half-word of the ARM data bus is written out on the I/O bus. When reading, the I/O bus data is read back onto the lower half-word of the ARM data bus. During these accesses, a chip select is asserted with the appropriate nIOR/nIOW read or write strobe, based on the 8-MHz clock CLK8.

Module I/O

The Module I/O type of access is 16 bits only and timing is controlled by a handshake mechanism with the external hardware. The signals nIORQ (output) and nIOGT (input) are used for this handshaking and are referenced to REF8M. When writing, the upper half-word of the ARM data bus is written out on the I/O bus. When reading, the I/O bus data is read back onto the lower half-word of the ARM data bus.

During these accesses, a chip select is asserted but the nIOR/nIOW read and write strobes are not used, although the IORNW signal is active.

PC Bus Style I/O

The PC bus style I/O type of access routes the lower half-word of the ARM bus through the device providing a direct 16-bit interface. Signals are generated to support the addition of external latches/drivers to extend the I/O data by 16 bits. The upper half-word of the ARM data bus is routed through these external devices if present.

There are five different address areas that generate five different chip selects using the same type of access. There are four fixed-cycle types based on the 16-MHz clock, although the largest area only supports two of these cycle types. Any access can be held up by external circuitry removing the READY signal before the end of the cycle.

During these accesses, the relevant chip select is asserted, as well as the appropriate read or write strobes.

Two special inputs are provided to allow external circuitry to route the full 32 bits through the 16-bit I/O bus using multiplexing. This allows, for example, the execution of code from a 16-bit PCMCIA card with suitable external controller. On a read I/O, if this latching signal is used, the data read back onto the ARM data bus comes from the I/O bus instead of the external extension latches.

3.7 Other Features

The CL-PS7500FE includes four analog comparators used to create four A-to-D converter channels, and two serial keyboard/mouse ports.

There are eight general-purpose, open-drain I/O lines that can be used as inputs or open drain outputs and, if required, as interrupt sources.

An interrupt handler processes a variety of internal and external interrupt sources to generate the IRQ and FIQ interrupts for the ARM processor.

3.8 Test Modes

The CL-PS7500FE has an nTEST pin used to invoke various test modes. When nTEST is set low, the functionality of many of the pins change depending on the values applied to the nINT3, nINT6 and nINT8 pins. The nTEST pin includes an on-chip pull-up resistor, but it is recommended that the pin be also pulled up to V_{DD} externally. See Appendix F.

NOTE: The nTEST pin should never be forced low during normal operation.

3.9 Structure of the CL-PS7500FE

The CL-PS7500FE includes three modified ARM macrocells:

- ARM processor
- FPA
- Video and sound macrocells

These macrocells are self-contained and the relevant control registers are contained within them. This has the effect that there are four sets of programmable registers within the CL-PS7500FE, which are accessed in different ways depending on their location.

3.9.1 Register Programming

ARM processor register programming is described in Chapter 4. FPA register programming is described in Chapter 18.

The video and sound macrocell registers are programmed using only the internal CL-PS7500FE data bus (the address bus is not passed to the macrocell). The address 0x03400000 is decoded to provide a write strobe for the video macrocell registers, and the addressing of registers within the macrocell is decoded from the upper four or eight bits of the data word. This system is described in more detail in Chapter 16.

The remaining CL-PS7500FE registers, associated with Memory, I/O, and general miscellaneous control, form a separate group and are programmed between addresses 0x03200000 and 0x032001F8. The majority of the registers are only 8 bits wide, although all register addresses are word-aligned. These registers are described in Chapter 7.

3.9.2 Interaction Between Macrocells

Interaction between the macrocells occurs mainly across the internal 32-bit data bus of the CL-PS7500FE, which is routed to the ARM and video/sound macrocells, and most of the other memory and I/O control logic. The address bus of the ARM processor is routed to an internal address decoder where memory space is decoded to determine required cycle types and register addresses. The same address bus is latched and exported from the device as the LA[28:0] bus. Only these 29 bits of the address bus are available externally.

3.10 Resetting CL-PS7500FE Systems

The CL-PS7500FE is designed to operate with both 16- and 32-bit-wide ROMs, which means that it must be capable of booting from either. To achieve this, the device is always reset into 16-bit mode, which might be expected to cause difficulty when the device is being booted up from 32-bit ROM. However, Appendix A describes a simple code sequence that allows the device to be started up without difficulty under these circumstances.

4. THE ARM PROCESSOR MACROCELL

The CL-PS7500FE contains a 32-bit RISC ARM processor, similar to the ARM710C macrocell. It has a 4-Kbyte cache, write buffer, and an MMU. The ARM processor macrocell offers high-level RISC performance, yet its fully static design ensures minimal power consumption. This makes it ideal for incorporation into the CL-PS7500FE. The CL-PS7500FE aims to make maximum use of the performance and flexibility offered by the ARM processor.

This section describes the features of the ARM processor macrocell available to the user in its embedded state within the CL-PS7500FE single-chip computer.

4.1 Architecture

The ARM processor architecture is based on RISC principles, and the instruction set and related decode mechanism are greatly simplified compared with microprogrammed CISCs.

The mixed data and instruction cache, together with the write buffer, substantially raise the average execution speed and reduce the average amount of memory bandwidth required by the processor. This allows the CL-PS7500FE bus structure to support DMA channels with minimal performance loss.

The MMU supports a conventional two-level page-table structure and a number of extensions, making it ideal for embedded control, UNIX, and object-oriented systems.

4.2 Instruction Set

The instruction set comprises ten basic instruction types:

- Two instruction types make use of the on-chip ALU, barrel shifter, and multiplier to perform high-speed operations on the data in a bank of 31 registers, each 32 bits wide.
- Three classes of instruction control data transfers between memory and the registers:
  - — one optimized for flexibility of addressing,
  - — another for rapid context switching, and
  - — the third for swapping data.
- Two instructions control the flow and privilege level of execution.
- Three instruction types are dedicated to the control of coprocessors that allow the functionality of the instruction set to be extended in an open and uniform way; the on-chip FPA is one such processor.
  However, the facility to add external coprocessors to the CL-PS7500FE is not available, and software emulation of coprocessor activity is required if instructions, other than those for the on-chip FPA or control coprocessor #15, are to perform a defined function.

The ARM instruction set is a good target for compilers of many different high-level languages. Where required for critical code segments, assembly code programming is also straightforward, unlike some RISC processors that depend on sophisticated compiler technology to manage complicated instruction interdependencies.

4.3 Memory Interface

The memory interface has been designed to allow the performance potential to be realized without incurring high costs in the memory system. Speed-critical control signals are pipelined to allow system control functions to be implemented in standard low-power logic, and these control signals permit the CL-PS7500FE to exploit the page mode access offered by industry-standard DRAMs.

4.4 Clocks and Synchronous/Asynchronous Modes

The ARM processor uses two independent clock sources, MCLK and FCLK. Both are generated internally to the CL-PS7500FE from MEMCLK and CPUCLK. The ARM7 core CPU switches between MCLK and FCLK according to the operation being carried out. For example, if the ARM7 core CPU is reading data from the cache, it is clocked by FCLK; if the core CPU is reading data from uncached memory then it is clocked by MCLK. The control logic of the ARM processor ensures that the correct clock is used internally and switches between the two clocks automatically.

When SnA is tied high, MEMCLK creates both FCLK and MCLK; MCLK has half the frequency of FCLK. This synchronous mode ensures that there are no synchronization penalties whenever the ARM 7 core is switched between FCLK and MCLK.

When SnA is tied low, MEMCLK creates MCLK and CPUCLK must be driven to supply FCLK. MEMCLK and CPUCLK can be of unrelated frequency. There is a synchronization penalty whenever the ARM7 core clock switches between MCLK and FCLK. This penalty is symmetric, and varies between nothing and a whole period of the clock where the core resynchronized. Thus, when changing there is an average resynchronization penalty of one-half a clock period, MCLK or FCLK as appropriate.

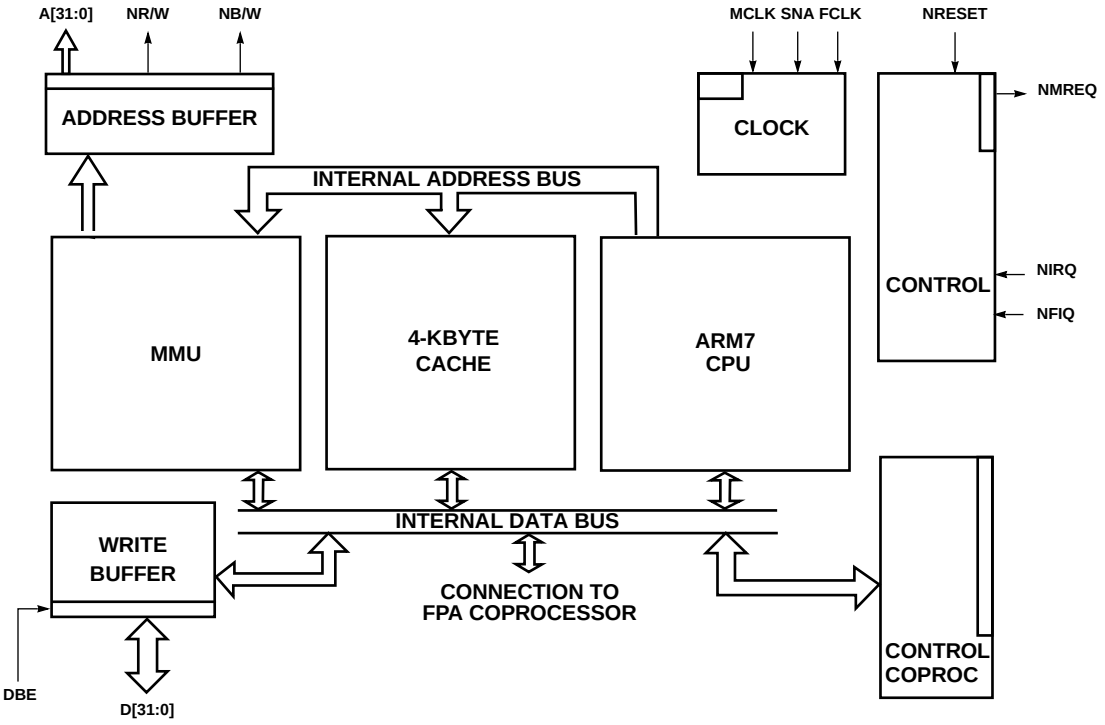


Figure 4-1. ARM Processor Block Diagram

5. IDC

ARM processor contains a 4-Kbyte mixed-instruction and data cache. The IDC has 256 lines of 16 bytes (4 words), organized as a four-way set associative cache, and uses the virtual addresses generated by the processor core. The IDC is always reloaded one line at a time (four words). It can be enabled or disabled through the ARM processor Control register and is disabled on nRESET.

The operation of the cache is further controlled by the *cacheable* or C bit stored in the Memory Management Page table (see the Section 6 on page 38). For this reason, to use the IDC the MMU must be enabled. However, the two functions can be enabled simultaneously with a single write to the Control register.

5.1 Cacheable Bit

The C bit determines whether data being read can be placed in the IDC and used for subsequent read operations. Typically, main memory is marked as cacheable to improve system performance, and I/O space as non-cacheable to stop the data being stored in the cache of the CL-PS7500FE. (For example, if the processor is polling a hardware flag in I/O space, it is important that the processor is forced to read data from the external peripheral, and not a copy of initial data held in the cache.) The Cacheable bit can be configured for both pages and sections.

5.2 IDC Operation

In the ARM processor the cache is searched regardless of the state of the C bit, only reads that miss the cache are affected.

- Cacheable reads
  - C = 1: A line fetch of 4 words is performed and randomly placed in a cache bank.
- Uncacheable reads
  - C = 0: An external memory access is performed and the cache is not written.

5.2.1 IDC Validity

The IDC operates with virtual addresses, so ensure that the contents remain consistent with the virtual-to-physical mappings performed by the MMU. If the memory mappings are changed, the IDC validity must be ensured.

5.2.2 Software IDC Flush

The entire IDC can be marked as invalid by writing to the ARM processor IDC Flush register (register 7). The cache is flushed immediately the register is written, but note that the next two instruction fetches may come from the cache before the register is written.

5.2.3 Doubly-Mapped Space

Since the cache works with virtual addresses, it is assumed that every virtual address maps to a different physical address. If the same physical location is accessed by more than one virtual address, the cache cannot maintain consistency, since each virtual address has a separate entry in the cache, and only one entry is updated on a processor write operation. To avoid any cache inconsistencies, both doubly-mapped virtual addresses should be marked as uncacheable.

5.2.4 Read-Locked-Write

The IDC treats the Read-Locked-Write instruction as a special case. The read phase always forces a read of external memory, regardless of whether the data is contained in the cache. The write phase is treated as a normal write operation (and if the data is already in the cache, the cache is updated). Externally the two phases are flagged as indivisible by asserting the LOCK signal.

5.3 IDC Enable/Disable and Reset

The IDC is automatically disabled and flushed on nRESET. Once enabled, cacheable read accesses cause lines to be placed in the cache.

5.3.1 Enable the IDC

To enable the IDC, make sure that the MMU is enabled first by setting Control register, bit 0, then enable the IDC by setting Control register, bit 2. The MMU and IDC can be simultaneously enabled with a single control register write.

5.3.2 Disable the IDC

To disable the IDC, clear Control register, bit 2 and perform a flush by writing to the flush register.

5.4 Write Buffer (WB)

The ARM processor WB is provided to improve system performance. It can buffer up to 8 words of data, and four independent addresses. It can be enabled or disabled through the W bit (bit 3) in the ARM processor Control register and the buffer is disabled and flushed on reset.

The operation of the write buffer is further controlled by one bit, B or *bufferable*, stored in the Memory Management Page Tables. For this reason, to use the write buffer the MMU must be enabled.

The two functions may, however, be enabled simultaneously, with a single write to the Control register. For a write to use the write buffer, both the W bit in the Control register, and the B bit in the corresponding page table must be set.

5.4.1 Bufferable Bit

This bit controls whether a write operation may or may not use the write buffer. Typically main memory is bufferable and I/O space unbufferable. The B bit can be configured for both pages and sections.

5.4.2 Write Buffer Operation

When the CPU performs a write operation, the translation entry for that address is inspected and the state of the B bit determines the subsequent action. If the write buffer is disabled through the ARM processor Control register, bufferable writes are treated in the same way as unbuffered writes.

Bufferable Write

If the write buffer is enabled and the processor performs a write to a bufferable area, the data is placed in the write buffer at FCLK speeds and the CPU continues execution. The write buffer then performs the external write in parallel. If the write buffer is full (either because there are already 8 words of data in the buffer, or because there is no slot for the new address) then the processor is stalled until there is sufficient space in the buffer.

Unbufferable Writes

If the write buffer is disabled or the CPU performs a write to an unbufferable area, the processor is stalled until the write buffer empties and the write completes externally. This process may require synchronization and several external clock cycles.

Read-Locked Write

The write phase of a read-locked-write sequence is treated as an unbuffered write, even if it is marked as buffered.

NOTE: A single write requires one address slot and one data slot in the write buffer; a sequential write of *n* words requires one address slot and *n* data slots. The entire eight data slots in the buffer can be used as required. So for instance there could be three non-sequential writes and one sequential write of 5 words in the buffer, and the processor could continue as normal: a fifth write or a sixth word in the fourth write would stall the processor until the first write is complete.

5.4.3 Enable the Write Buffer

To enable the WB, ensure the MMU is enabled by setting Control register, bit 0, then enable the write buffer by setting Control register, bit 3. The MMU and write buffer can be enabled simultaneously with a single write to the Control register.

5.4.4 Disable the Write Buffer

To disable the WB, clear the control register, bit 3.

NOTE: Any writes already in the write buffer completes normally.

5.5 Coprocessors

The on-chip FPA is a coprocessor and its operation are described later in this manual.

The ARM processor also has an internal coprocessor designated #15 for internal control of the device.

However, the CL-PS7500FE has no external coprocessor bus, so it is not possible to add further external coprocessors to this device. All coprocessor operations, other than those implemented by the FPA, MRC, or MCR to registers 0–7 on coprocessor #15, cause the undefined instruction trap to be taken.

6. ARM PROCESSOR MMU

The MMU performs two primary functions: it translates virtual addresses into physical addresses, and it controls memory access permissions. The MMU hardware required to perform these functions consists of a TLB, access control logic, and translation table walking logic.

The MMU supports memory accesses based on Sections or Pages:

- **Sections** are comprised of 1-Mbyte blocks of memory.
- **Pages –** two different page sizes are supported:
  - **Small pages** consist of 4-Kbyte blocks of memory. Additional access control mechanisms are extended within small pages to 1-Kbyte subpages.
  - **Large pages** consist of 64-Kbyte blocks of memory. Additional access control mechanisms are extended within large pages to 16-Kbyte subpages. Large pages are supported to allow mapping of a large region of memory while using only a single entry in the TLB.

The MMU also supports the concept of domains — areas of memory that can be defined to possess individual access rights. The Domain Access Control register specifies access rights for up to 16 separate domains.

The TLB caches 64 translated entries. During most memory accesses, the TLB provides the translation information to the access control logic. If the TLB contains a translated entry for the virtual address, the access control logic determines whether access is permitted. If access is permitted, the MMU outputs the appropriate physical address corresponding to the virtual address. If access is not permitted, the MMU signals the CPU to abort.

If the TLB misses (that is, it does not contain a translated entry for the virtual address), the translation table walk hardware is invoked to retrieve the translation information from a translation table in physical memory. Once retrieved, the translation information is placed into the TLB, possibly overwriting an existing value. The entry to be overwritten is chosen by cycling sequentially through the TLB locations.

When the MMU is turned off (for example, on reset), the virtual address is output directly onto the physical address bus.

6.1 MMU Program-Accessible Registers

The ARM processor provides several 32-bit registers that determine the operation of the MMU. The format for these registers and a brief description is shown in Figure 6-1 on page 39. Each register is discussed in more detail within the section that describes its use.

Data is written to and read from the MMU registers using the ARM CPU MRC and MCR coprocessor instructions.

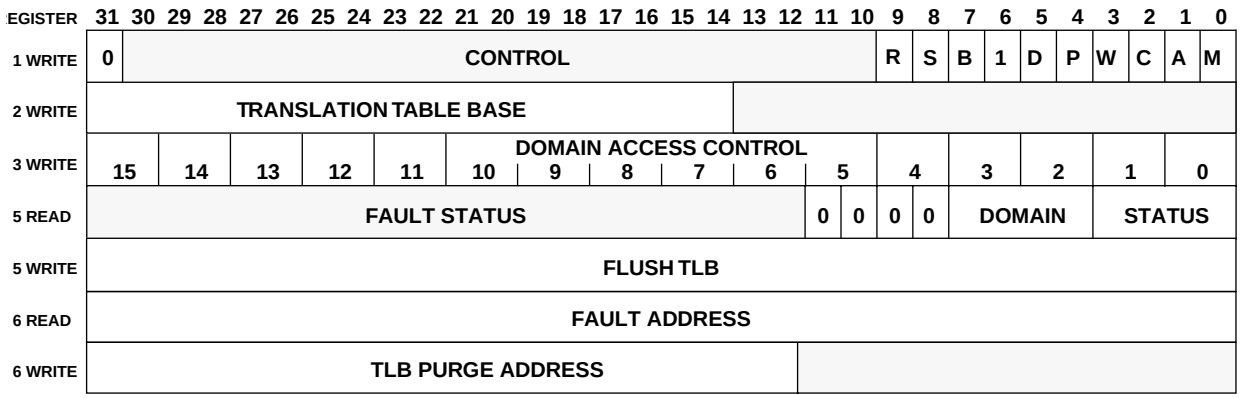


Figure 6-1. MMU Register Summary

6.1.1 Translation Table Base Register

The Translation Table Base register contains the physical address of the base of the translation table maintained in main memory. Note that this base must reside on a 16-Kbyte boundary.

6.1.2 Domain Access Control Register

The Domain Access Control register consists of sixteen 2-bit fields, each defines the access permissions for one of the sixteen domains (D[15:0]).

NOTE: The registers not shown are reserved and should not be used.

6.1.3 Fault Status Register

The Fault Status register indicates the domain and type of access being attempted when an abort occurred. Bits 7:4 specify the accessed of the sixteen domains (D[15:0]) during a fault. Bits 3:1 indicate the type of access being attempted. The encoding of these bits is different for internal and external faults (as indicated by bit 0 in the register) and is shown in Table 6-4 on page 47. A write to this register flushes the TLB.

6.1.4 Fault Address Register

The Fault Address register holds the virtual address of the access attempted when a fault occurred. A write to this register causes the data written to be treated as an address and, if it is found in the TLB, the entry is marked as invalid. (This operation is known as a TLB ‘purge’.) The Fault Status and Fault Address registers are only updated for data faults, not for prefetch faults.

6.2 Address Translation

The MMU translates virtual addresses generated by the CPU into physical addresses to access external memory, and also derives and checks the access permission. Translation information, consisting of both the address translation data and the access permission data, resides in a translation table located in physical memory. The MMU provides the logic needed to traverse this translation table, obtain the translated address, and check the access permission.

There are three routes for address translation and permission checking. The selected route depends on if the address in question is marked as a section-mapped access or a page-mapped access; there are two sizes of page-mapped access (large pages and small pages). However, the translation process always starts out in the same way, as described in Section 6.3.2, with a Level One fetch. A section-mapped access only requires a Level One fetch, but a page-mapped access also requires a Level Two fetch.

6.3 Translation Process

6.3.1 TTB (Translation Table Base)

The translation process is initiated when the on-chip TLB does not contain an entry for the requested virtual address. The TTB register points to the base of a table in physical memory that contains Section and/or Page descriptors. The 14 low-order bits of the TTB register are set to zero as illustrated in Figure 6-2, the table must reside on a 16-Kbyte boundary.

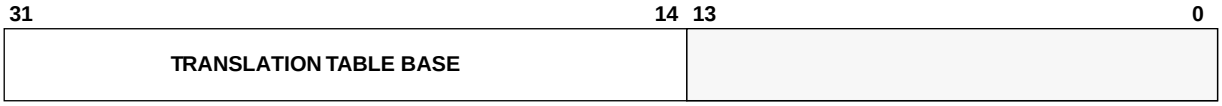


Figure 6-2. Translation Table Base Register

6.3.2 Level One Fetch

Bits 31:14 of the TTB register are concatenated with bits 31:20 of the virtual address to produce a 30-bit address, as illustrated in Figure 6-3. This address selects a 4-byte translation table entry that is a First Level descriptor for either a Section or a Page (bit 1 of the descriptor returned specifies if it is for a Section or Page).

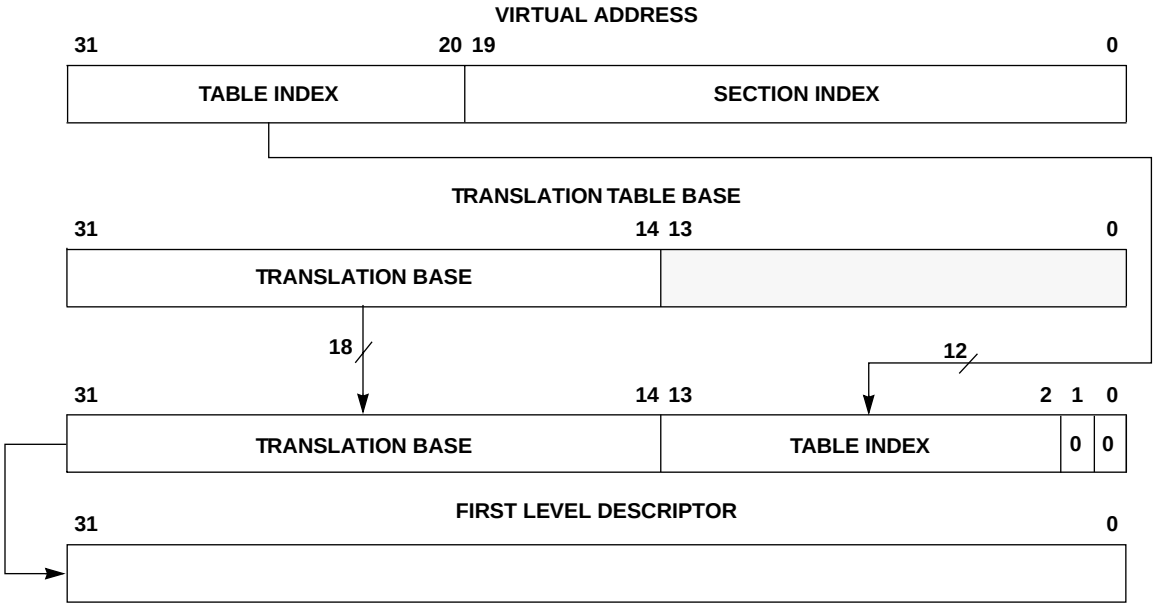


Figure 6-3. Accessing the Translation Table First Level Descriptors

6.3.3 Level One Descriptor

The Level One descriptor returned is either a Page Table descriptor or a Section descriptor, and the format varies accordingly. Figure 6-4 illustrates the format of Level One descriptors.

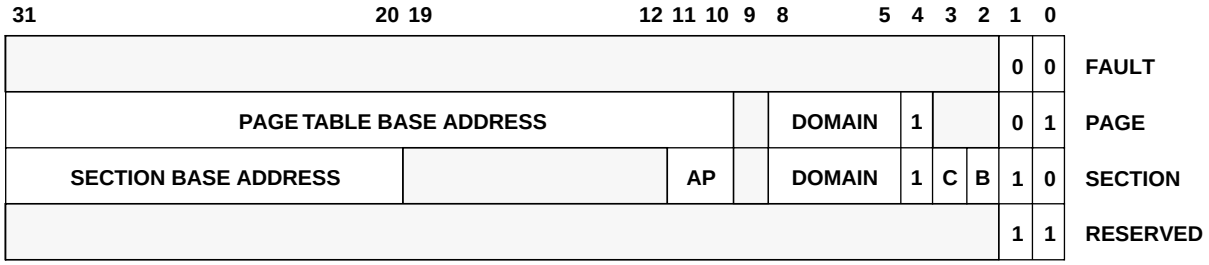


Figure 6-4. Level One Descriptors

The two least-significant bits indicate the descriptor type and validity, and are interpreted as in Table 6-1.

Table 6-1. Interpreting Level One Descriptor Bits 1:0

Value	Meaning	Notes
00	Invalid	Generates a Section Translation fault.
01	Page	Indicates that this is a Page descriptor.
10	Section	Indicates that this is a Section descriptor.
11	Reserved	Reserved for future use.

6.3.4 Page Table Descriptor

Bit	Description
3:2	Always write as ‘0’.
4	Write as ‘1’ for backward compatibility.
8:5	Specify one of the sixteen possible domains (held in the Domain Access Control register) that contain the primary access controls.
31:10	Form the base for referencing the Page Table entry. (The page table index for the entry is derived from the virtual address as illustrated in Figure 6-7 on page 45.)

If a Page Table descriptor is returned from the Level One fetch, a Level Two fetch is initiated, as described in the following section.

6.3.5 Section Descriptor

Bit	Description
3:2 (C, B)	Control the cache- and write-buffer-related functions as follows: • C – Cacheable: data at this address is placed in the cache (if the cache is enabled). • B – Bufferable: data at this address is written through the write buffer (if enabled).
4	Write as '1' for backward compatibility.
8:5	Specify one of the sixteen possible domains (held in the Domain Access Control register) that contain the primary access controls.
11:10 (AP)	Specify the access permissions for this section (see Table 6-2). The interpretation depends upon the setting of the S and R bits (control register bits 8 and 9). Note that the Domain Access Control specifies the primary access control; the AP bits only have an effect in Client mode. Refer to Section 6.9.
19:12	Always write as '0'.
31:20	Form the corresponding bits of the physical address for the 1-Mbyte section.

Table 6-2. Interpreting Access Permission (AP) Bits

AP	S	R	Supervisor Permissions	User Permissions	Notes
00	0	0	No access	No access	Any access generates a permission fault.
00	1	0	Read only	No access	Supervisor read-only permitted.
00	0	1	Read only	Read only	Any write generates a permission fault.
00	1	1	Reserved		
01	X[a]	X	Read/write	No access	Access allowed only in Supervisor mode.
10	X	X	Read/write	Read only	Writes in User mode cause permission fault.
11	X	X	Read/write	Read/write	All access types permitted in both modes.
XX	1	1	Reserved		

[a] 'X' indicates a don't care state.

6.4 Translating Section References

Figure 6-6 on page 44 illustrates the complete Section translation sequence. Note that the access permissions contained in the Level One descriptor must be checked before the physical address is generated. The sequence for checking access permissions is described in Section 6.10.4 on page 50.

6.4.1 Level Two Descriptor

If the Level One fetch returns a Page Table descriptor, this provides the base address of the page table to be used. The page table is then accessed as described in Figure 6-7 on page 45, and a Page Table

entry, or Level Two descriptor, is returned. This in turn may define either a small page or large page access. Figure 6-5 shows the format of Level Two descriptors.

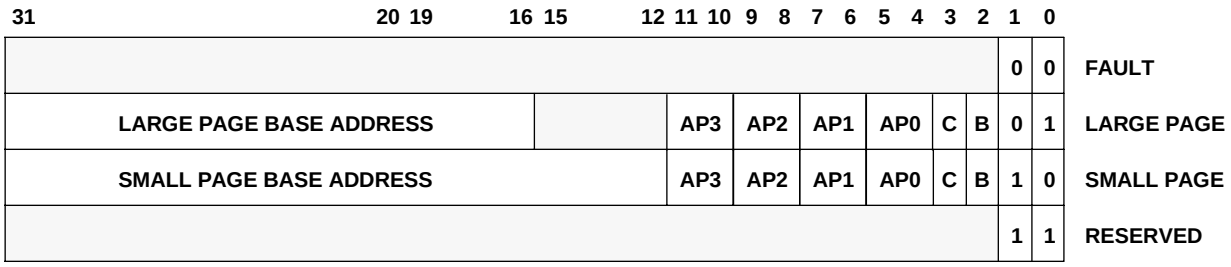


Figure 6-5. Page Table Entry (Level Two Descriptor)

The two least-significant bits indicate the page size and validity, and are interpreted as shown in Table 6-3.

Table 6-3. Interpreting Page Table Entry Bits 1:0

Value	Meaning	Notes
00	Invalid	Generates a Page Translation fault.
01	Large Page	Indicates that this is a 64-Kbyte page.
10	Small Page	Indicates that this is a 4-Kbyte page.
11	Reserved	Reserved for future use.

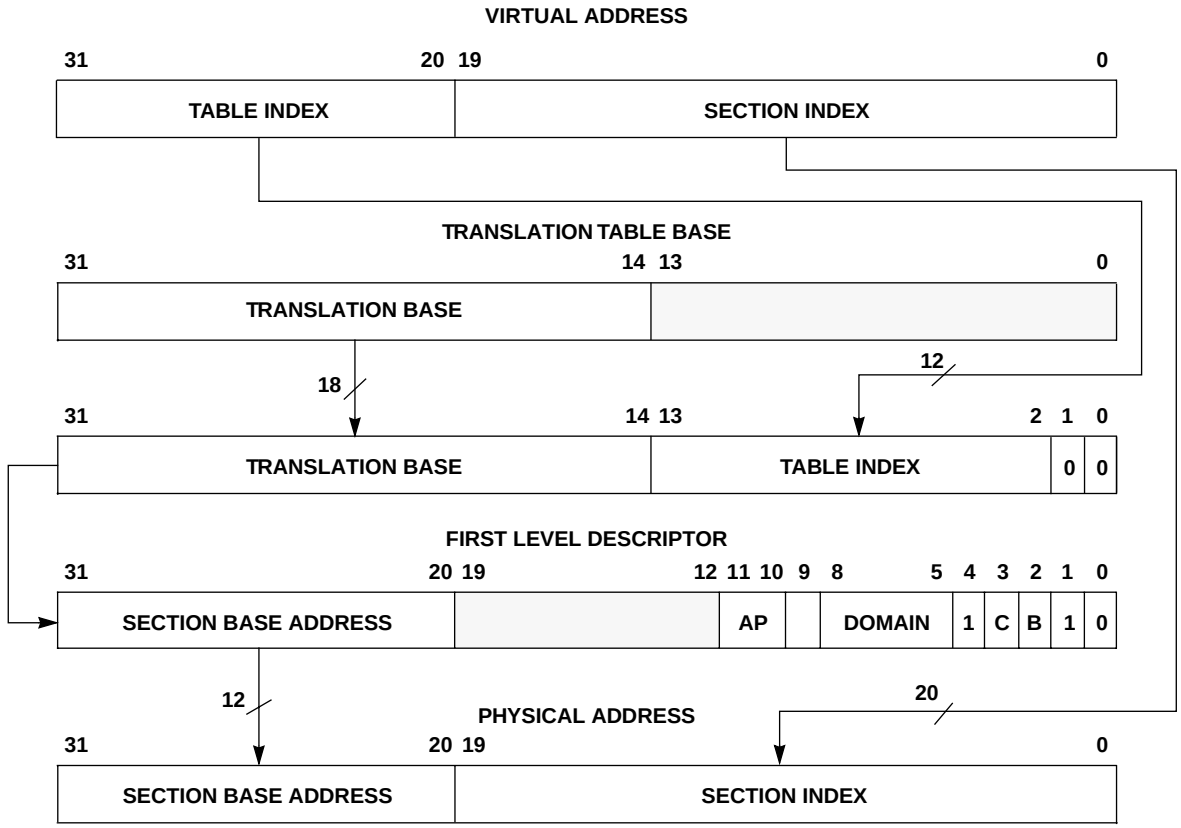


Figure 6-6. Section Translation

Bit	Description
2	(B – Bufferable) Indicates that data at this address is written through the WB (if the write buffer is enabled).
3	(C – Cacheable) Indicates that data at this address is placed in the IDC (if the cache is enabled).
11:4	Specify the access permissions (AP[3:0]) for the four subpages and interpretation of these bits is described earlier in Table 6-1 on page 41.
15:12	For large pages, these bits are programmed as '0'.

Bits 31:12 (small pages) or bits 31:16 (large pages) form the corresponding bits of the physical address — the *physical page number*. (The page index is derived from the virtual address as illustrated in Figure 6-7 and Figure 6-8.)

6.5 Translating Small Page References

Figure 6-7 illustrates the complete translation sequence for a 4-Kbyte small page. Page translation involves one additional step beyond that of a section translation: the Level One descriptor is the Page Table descriptor, and this is used to point to the Level Two descriptor, or Page Table entry. (Note that the access permissions are now contained in the Level Two descriptor and must be checked before the physical address is generated. The sequence for checking access permissions is described in Section 6.10.4 on page 50.)

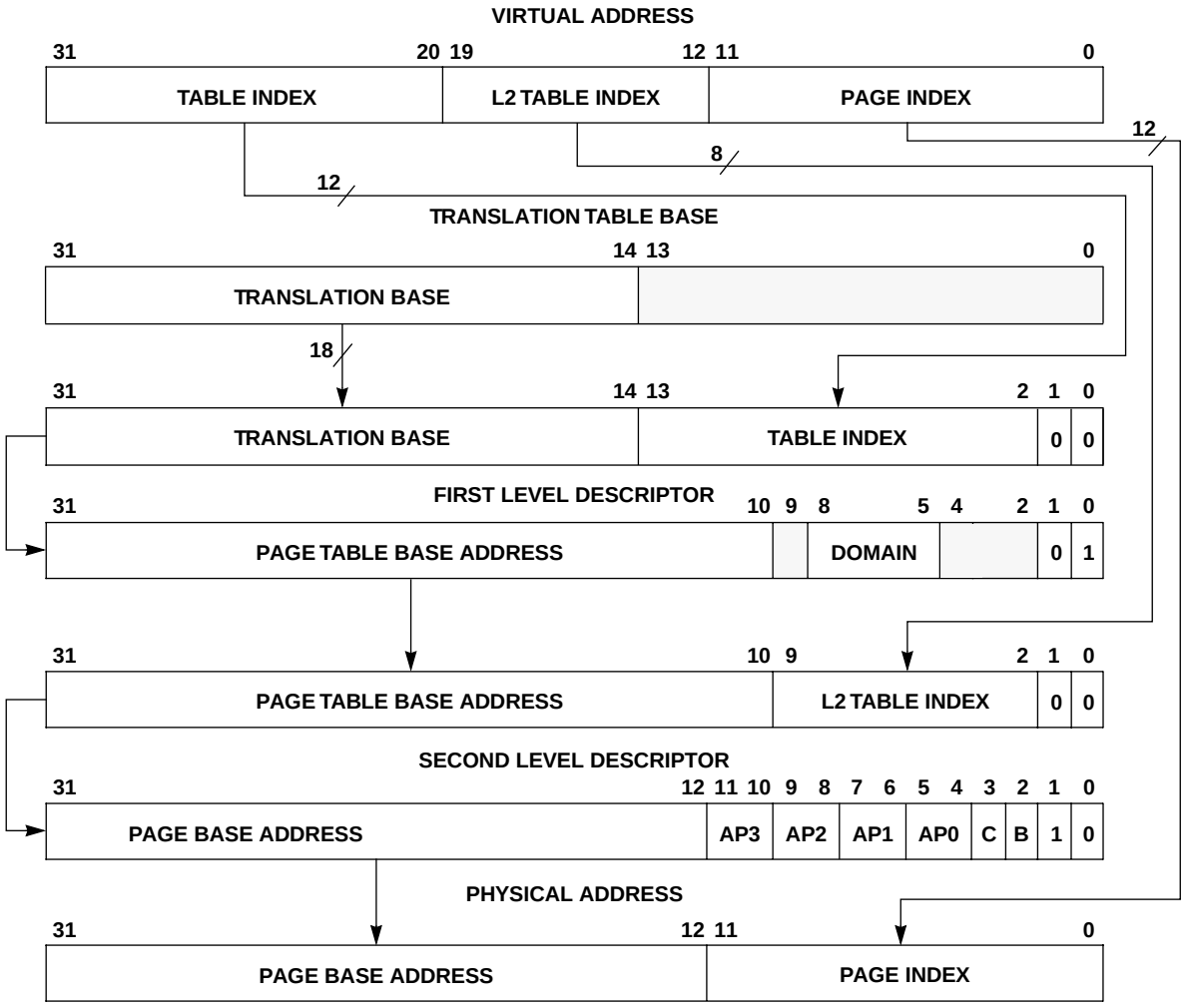


Figure 6-7. Small Page Translation

6.6 Translating Large Page References

Figure 6-8 illustrates the complete translation sequence for a 64-Kbyte large page. Note that since the upper 4 bits of the Page Index and low-order 4 bits of the Page Table index overlap, each Page Table entry for a large page must be duplicated 16 times (in consecutive memory locations) in the Page Table.

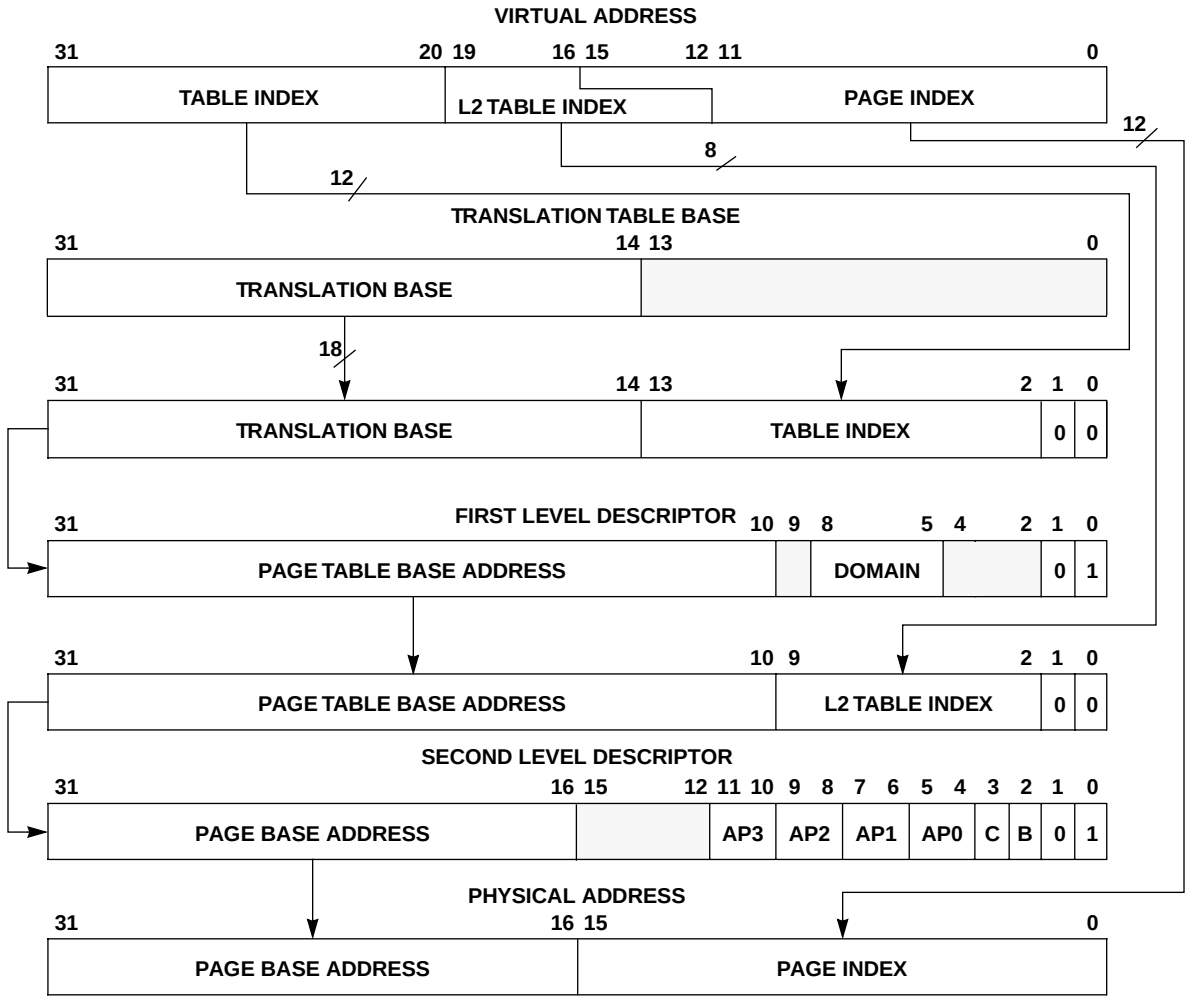


Figure 6-8. Large Page Translation

6.7 MMU Faults and CPU Aborts

The MMU generates four types of faults:

- Alignment
- Translation
- Domain
- Permission

The access control mechanisms of the MMU detect the conditions that produce these faults. If a fault is detected as the result of a memory access, the MMU aborts the access and signal the fault condition to the CPU. The MMU is also capable of retaining status and address information about the abort. The CPU recognizes two types of abort, data and prefetch, and these are treated differently by the MMU.

If the MMU detects an access violation, it detects it before the external memory access occurs, and inhibit the access.

6.8 Fault Address and Fault Status Registers (FAR, FSR)

Aborts resulting from data accesses (data aborts) are acted upon by the CPU immediately, and the MMU places an encoded 4-bit value FS[3:0], along with the 4-bit encoded Domain number, in the FSR. In addition, the virtual processor address that caused the data abort is latched into the FAR. If an access violation simultaneously generates more than one source of abort, they are encoded in the priority given in Table 6-4.

On the other hand, CPU instructions are prefetched so a prefetch abort simply flags the instruction as it enters the instruction pipeline. Only when (and if) the instruction is executed does it cause an abort; an abort is not acted upon if the instruction is not used (that is, it is branched around). Because instruction prefetch aborts may or may not be acted upon, the MMU status information is not preserved for the resulting CPU abort; for a prefetch abort, the MMU does not update the FSR or FAR.

The following sections describe the various access permissions and controls supported by the MMU and detail how these are interpreted to generate faults.

In Table 6-4 X indicates a don't care state and can read as '0' or '1'.

NOTE: Any abort masked by the priority encoding can be regenerated by fixing the primary abort and restarting the instruction. In fact, this register contains bits 8:5 of the Level One entry, which are undefined, but would encode the domain in a valid entry.

Table 6-4. Priority Encoding of Fault Status Register

Priority	Source	FS[3:0]	Domain [3:0]	FAR
Highest	Alignment	00x1	X	Valid
	Translation (Section)	0101	Note 2	Valid
	Translation (Page)	0111	Valid	Valid
	Domain (Section)	1001	Valid	Valid
	Domain (Page)	1011	Valid	Valid
	Permission (Section)	1101	Valid	Valid
Lowest	Permission (Page)	1111	Valid	Valid

6.9 Domain Access Control

MMU accesses are primarily controlled through domains. There are 16 domains, and each has a 2-bit field to define it. Two basic kinds of users are supported:

- Clients use a domain, and
- Managers control the behavior of the domain.

The domains are defined in the Domain Access Control register. Figure 6-9 illustrates how the 32 bits of the register are allocated to define the sixteen 2-bit domains.

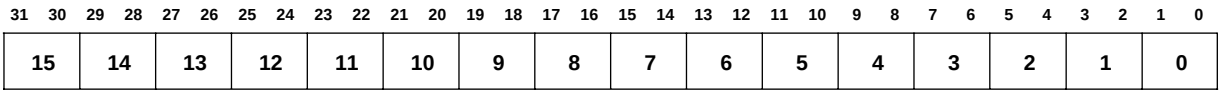


Figure 6-9. Domain Access Control Register Format

Table 6-5 defines how the bits within each domain are interpreted to specify the access permissions.

Table 6-5. Interpreting Access Bits in Domain Access Control Register

Value	Meaning	Notes
00	No access	Any access generates a Domain fault.
01	Client	Accesses are checked against the access permission bits in the Section or Page descriptor.
10	Reserved	Reserved. Currently behaves like the No Access mode.
11	Manager	Accesses are *not* checked against the access Permission bits, so a Permission fault cannot be generated.

6.10 Fault-Checking Sequence

The sequence used by the MMU to check for access faults is slightly different for Sections and Pages. Figure 6-9 illustrates the sequence for both types of accesses. The following sections and figures describe the conditions that generate each of the faults.

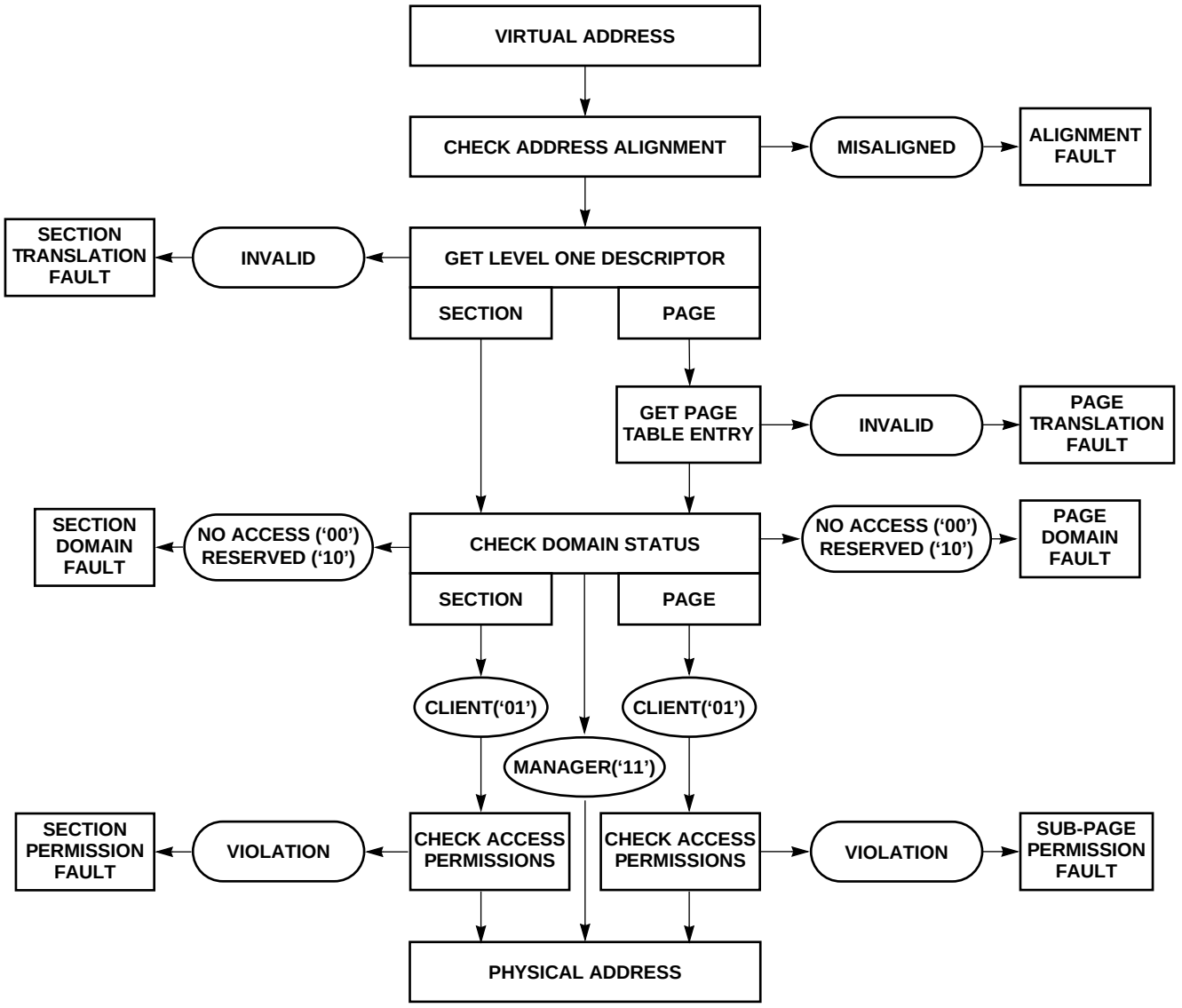


Figure 6-10. Sequence for Checking Faults

6.10.1 Alignment Fault

When Alignment fault is enabled (bit 1 in Control register is set), the MMU generates an Alignment fault on any data word access where the address is not word-aligned irrespective of whether the MMU is enabled or not; in other words, if either of virtual address bits 1:0 are not '0'.

An Alignment fault is not generated on any instruction fetch or on any byte access. Note that if the access generates an Alignment fault, the access sequence aborts without reference to further permission checks.

6.10.2 Translation Fault

There are two types of Translation fault:

- **Section** is generated if the Level One descriptor is marked as invalid. This happens if bits 1:0 of the descriptor are both '0' or both '1'.
- **Page** is generated if the Page Table entry is marked as invalid. This happens if bits 1:0 of the entry are both '0' or both '1'.

6.10.3 Domain Fault

There are two types of Domain fault:

- Section
- Page

In both cases the Level One descriptor holds the 4-bit Domain field that selects one of the sixteen 2-bit domains in the Domain Access Control register. The two bits of the specified domain are then checked for access permissions as detailed in Table 6-2 on page 42. For a Section fault, the domain is checked once the Level One descriptor is returned. For a Page fault, the domain is checked once the Page Table entry is returned.

If the specified access is either 'no access' ('00') or 'reserved' ('10'), then either a Section Domain or Page Domain fault occurs.

6.10.4 Permission Fault

There are two types of Permission fault:

- Section
- Subpage

Permission faults are checked at the same time as Domain faults. If the 2-bit domain field returns Client (01), then the permission access check is invoked as follows:

1) Section

If the Level One descriptor defines a section-mapped access, then the AP bits of the descriptor define whether or not the access is allowed according to Table 6-2. Their interpretation is dependent upon the setting of the S bit (Control register, bit 8). If the access is not allowed, a Section Permission fault is generated.

2) Subpage

If the Level One descriptor defines a page-mapped access, then the Level Two descriptor specifies four access permission fields (AP3–AP0) each corresponding to one quarter of the page. For small pages, AP3 is selected by the top 1 Kbyte of the page, and AP0 is selected by the bottom 1 Kbyte of the page; for large pages, AP3 is selected by the top 16 Kbytes of the page, and AP0 is selected by the bottom 16 Kbyte of the page. The selected AP bits are then interpreted in exactly the same way as for a section (see Table 6-2), the only difference is that the fault generated is a Subpage Permission fault.

6.11 External Aborts

The CL-PS7500FE does not support external aborts.

6.11.1 Interaction of the MMU, IDC, and Write Buffer

The MMU, IDC, and WB can be enabled/disabled independently. However there are only five valid combinations. There are no hardware interlocks on these restrictions, so invalid combinations cause undefined results.

Table 6-6. Valid MMU, IDC, and WB Combinations

MMU	IDC	WB
Off	Off	Off
On	Off	Off
On	On	Off
On	Off	On
On	On	On

The following procedures must be observed.

Enable the MMU

1. **Program the TTB and Domain Access Control registers.**
2. **Program Level One and Level Two page tables as required.**
3. **Enable the MMU by setting bit 0 in the Control register.**

NOTE: Care must be taken if the translated address differs from the untranslated address as the two instructions following the enabling of the MMU are fetched using 'flat translation' and enabling the MMU may be considered as a branch with delayed execution. A similar situation occurs when the MMU is disabled. Consider the following example code sequence:

```
MOV      R1, #0x1
MCR      15,0,R1,0,0                    ; Enable MMU
Fetch Flat
Fetch Flat
Fetch Translated
```

Disable the MMU

1. **Disable the WB by clearing bit 3 in the Control register.**
2. **Disable the IDC by clearing bit 2 in the Control register.**
3. **Disable the MMU by clearing bit 0 in the Control register.**

NOTE: If the MMU is enabled, then disabled, then subsequently re-enabled the contents of the TLB are preserved. If these are now invalid, the TLB should be flushed before re-enabling the MMU.

Disabling of all three functions can be done simultaneously.

7. REGISTER DESCRIPTIONS

The CL-PS7500FE supports a variety of operating configurations. Some are controlled by register bits and are known as the *configurations*. Other configurations can be controlled by software and are known as *operating modes*.

7.1 Register Configuration

The CL-PS7500FE provides three register configuration settings that can be changed while the processor is running. These are discussed below.

7.1.1 Big and Little Endian (the Bigend Bit)

The Bigend bit in the Control register sets whether the CL-PS7500FE treats words in memory as being stored in big endian or little endian format. Memory is viewed as a linear collection of bytes numbered upwards from zero. Bytes 0–3 hold the first stored word; bytes 4–7 the second, and so on.

Little Endian

In the little endian scheme, the lowest-numbered byte in a word is considered to be the least-significant byte of the word; the highest-numbered byte is the most-significant byte.

In this scheme, byte 0 of the memory system should be connected to D[7:0] (data lines 7 through 0).

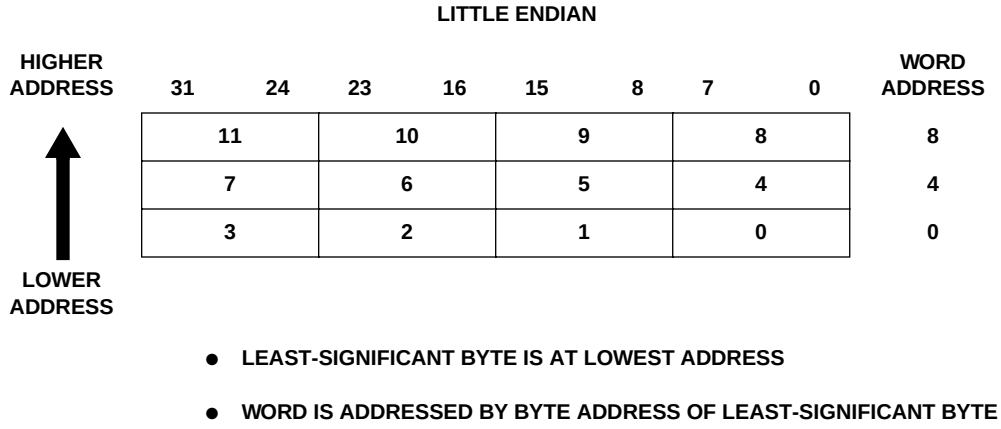


Figure 7-1. Little Endian Addresses of Bytes within Words

Big Endian

In the big endian scheme, the most-significant byte of a word is stored at the lowest-numbered byte, and the least-significant byte is stored at the highest-numbered byte.

Byte 0 of the memory system should therefore be connected to D[31:24] (data lines 31 through 24).

Load and store are the only instructions affected by the endianism.

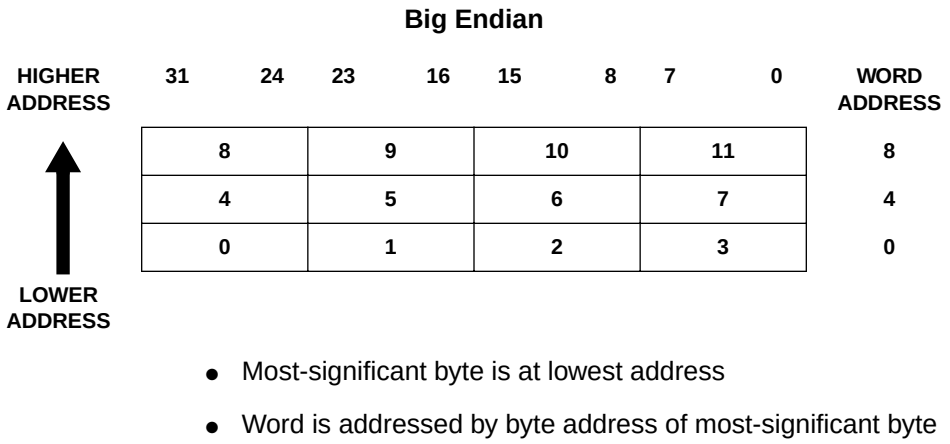


Figure 7-2. Big Endian Addresses of Bytes within Words

7.1.2 Configuration Bits for Backward Compatibility

Two register bits, PROG32 and DATA32, select one of three processor configurations:

1) 26-bit program and data space

PROG32 LOW, DATA32 LOW
This configuration forces the ARM processor to operate like the earlier ARM processors with 26-bit address space. The programmer's model for these processors applies, but the new instructions to access the CPSR and SPSR registers operate as detailed in the *CL-PS7500FE Programmer's Guide*. In this configuration, it is impossible to select a 32-bit operating mode, and all exceptions (including address exceptions) enter the exception handler in the appropriate 26-bit mode.

2) 26-bit program space and 32-bit data space

PROG32 LOW, DATA32 HIGH
This is the same as the 26-bit program and data space configuration, but with address exceptions disabled to allow data transfer operations to access the full 32-bit address space.

3) 32-bit program and data space

PROG32 HIGH, DATA32 HIGH
This configuration extends the address space to 32 bits, introduces major changes in the programmer's model and provides support for running existing 26-bit programs in the 32-bit environment.

NOTE: Do not select the fourth processor configuration, 26-bit data space and 32-bit program space.

26-bit Program Space

When configured for 26-bit program space, the CL-PS7500FE is limited to operating in one of four modes known as the 26-bit modes. These modes correspond to the modes of the earlier ARM processors and are known as:

- User26
- FIQ26
- IRQ26
- Supervisor26

NOTE: The PROG32 and DATA32 bits are only used for backward compatibility with earlier ARM processors and should normally be set to '1'. The 32-bit mode is recommended for compatibility with future ARM processors and all new code should be written to use only the 32-bit operating modes.

Because the original ARM instruction set is modified to accommodate 32-bit operation, there are certain additional restrictions that programmers must note. The following sections discuss these restrictions.

7.2 Operating Mode Selection

The ARM processor has a 32-bit data bus and a 32-bit address bus. However, only 29 of the address bits are available at the CL-PS7500FE pins. The data types the processor supports are:

- Bytes (8-bits)
- Words (32-bits) that must be aligned to 4-byte boundaries.

Instructions are exactly one word, and data operations (for example ADD) are only performed on word quantities. Load and store operations can transfer either bytes or words.

ARM processor supports six modes of operation:

- User mode (usr) — The normal program execution state.
- FIQ mode (fiq) — Designed to support a data transfer or channel process.
- IRQ mode (irq) — Used for general-purpose interrupt handling.
- Supervisor mode (svc) — A protected mode for the operating system.
- Abort mode (abt) — Entered after a data or instruction prefetch abort.
- Undefined mode (und) — Entered when an undefined instruction is executed.

Mode changes can be made under software control or brought about by external interrupts or exception processing. Most application programs execute in User mode. The other modes, known as *Privileged modes*, are entered to service interrupts or exceptions, or to access protected resources.

7.3 Registers

The processor macrocell has a total of 37 registers made up of:

- 31 general 32-bit registers
- Six status registers

At any one time, 16 general registers (R0 to R15) and one or two status registers are visible to the programmer. The visible registers depend on the processor mode, and the other registers (the *banked registers*) are switched in to support IRQ, FIQ, Supervisor, Abort, and Undefined mode processing.

The register bank organization is shown in Figure 7-3. The banked registers are shaded.

General Registers and Program Counter Modes

User32	FIQ32	Supervisor32	Abort32	IRQ32	Undefined32
R0	R0	R0	R0	R0	R0
R1	R1	R1	R1	R1	R1
R2	R2	R2	R2	R2	R2
R3	R3	R3	R3	R3	R3
R4	R4	R4	R4	R4	R4
R5	R5	R5	R5	R5	R5
R6	R6	R6	R6	R6	R6
R7	R7	R7	R7	R7	R7
R8	R8_fiq	R8	R8	R8	R8
R9	R9_fiq	R9	R9	R9	R9
R10	R10_fiq	R10	R10	R10	R10
R11	R11_fiq	R11	R11	R11	R11
R12	R12_fiq	R12	R12	R12	R12
R13	R13_fiq	R13_svc	R13_abt	R13_irq	R13_und
R14	R14_fiq	R14_svc	R14_abt	R14_irq	R14_und
R15 (PC)	R15 (PC)	R15 (PC)	R15 (PC)	R15 (PC)	R15 (PC)

Program Status Registers

CPSR	CPSR	CPSR	CPSR	CPSR	CPSR
	SPSR_fiq	SPSR_svc	SPSR_abt	SPSR_irq	SPSR_und

Figure 7-3. Register Organization

In all modes, 16 registers (R0 to R15) are directly accessible. All registers except R15 are general-purpose and can be used to hold data or address values. Register R15 holds the PC. When R15 is read, bits 1:0 are '0', and bits 31:2 contain the PC. A seventeenth register, CPSR (Current Program Status register), is also accessible. It contains condition code flags and the current mode bits and can be thought of as an extension to the PC.

R14 is used as the subroutine link register and receives a copy of R15 when a Branch and Link instruction is executed. It can be considered a general-purpose register at all other times. R14_svc, R14_irq, R14_fiq, R14_abt, and R14_und are used to hold the return values of R15 when interrupts and exceptions arise, or when Branch and Link instructions are executed within interrupt or exception routines.

FIQ mode has seven banked registers mapped to R8–14 (R8_fiq–R14_fiq). Many FIQ programs do not need to save any registers.

User mode, IRQ mode, Supervisor mode, Abort mode, and Undefined mode each have two banked registers mapped to R13 and R14. The two banked registers allow these modes to each have a private stack pointer and link register.

Supervisor, IRQ, Abort, and Undefined mode programs require more than these two banked registers and are expected to save some or all of the caller's registers (R0 to R12) on their respective stacks. They are then free to use these registers that they restore before returning to the caller.

In addition, there are also five SPSRs (Saved Program Status registers) that are loaded with the CPSR when an exception occurs. There is one SPSR for each privileged mode.

7.3.1 PSRs (Program Status Registers)

The format of the PSRs is shown in Figure 7-4.

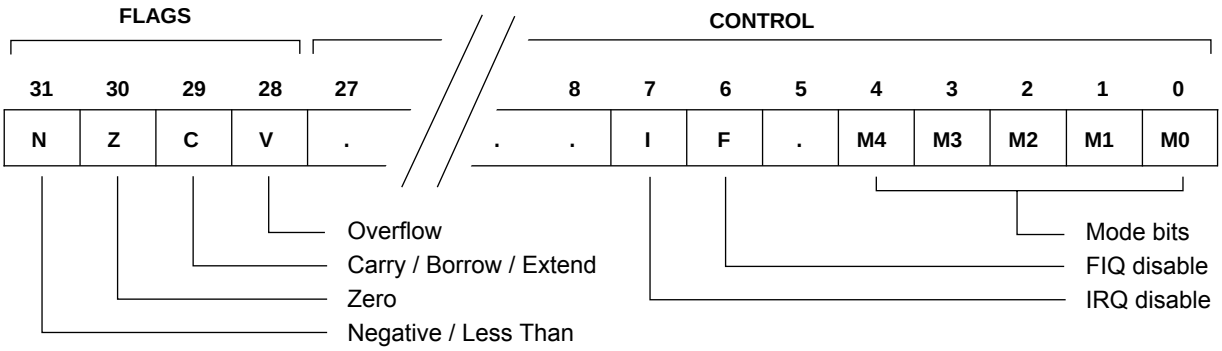


Figure 7-4. Format of the PSRs

7.3.1.1 Condition Code Flags

The N, Z, C, and V bits are the *condition code* flags. The condition code flags in the CPSR can be changed as a result of arithmetic and logical operations in the processor and can be tested by all instructions to determine if the instruction is to be executed.

Interrupt Disable Bits

The I and F bits are the *interrupt disable* bits. The I bit disables IRQ interrupts when set; the F bit disables FIQ interrupts when set.

7.3.1.2 Mode Bits

Bits M[4:0] are the *mode* bits that determine the processor operating mode. The interpretation of the mode bits is shown in Table 7-1. Not all combinations of the mode bits define a valid processor mode. Only those explicitly described are used.

Table 7-1. Mode Bits

M[4:0]	Mode	Accessible Register Set	
10000	User	PC, R14–R0	CPSR
10001	FIQ	PC, R14_fiq–R8_fiq, R7–R0	CPSR, SPSR_fiq
10010	IRQ	PC, R14_irq–R13_irq, R12–R0	CPSR, SPSR_irq
10011	Supervisor	PC, R14_svc–R13_svc, R12–R0	CPSR, SPSR_svc
10111	Abort	PC, R14_abt–R13_abt, R12–R0	CPSR, SPSR_abt
11011	Undefined	PC, R14_und–R13_und, R12–R0	CPSR, SPSR_und

7.3.1.3 Control Bits

The bottom 28 bits of a PSR (incorporating I, F, and M[4:0]) are known collectively as the *control* bits. The control bits change when an exception arises and, in addition, can be manipulated by software when the processor is in a privileged mode. Unused bits in the PSRs are reserved and their state must be preserved when changing the flag or control bits. Programs must not rely on specific values from the reserved bits when checking the PSR status, since they can read as '1' or '0' in future processors.

7.4 Exceptions

Exceptions arise whenever there is a need to break the normal flow of program execution. For example, the processor can be diverted to handle an interrupt from a peripheral. The processor state just prior to handling the exception must be preserved so that the original program can be resumed when the exception routine is complete. Many exceptions can arise at the same time.

The ARM processor handles exceptions by making use of the banked registers to save state. The old PC and CPSR contents are copied into the appropriate R14 and SPSR, and the PC and mode bits in the CPSR bits are forced to a value that depends on the exception. Interrupt disable flags are set where required to prevent otherwise unmanageable nestings of exceptions. In the case of a re-entrant interrupt handler, R14 and the SPSR should be saved onto a stack in main memory before re-enabling the interrupt.

NOTE: When transferring the SPSR to and from a stack, it is important to transfer the whole 32-bit value, and not just the flag or control fields.

When simultaneous multiple exceptions arise, a fixed priority determines their order. The priorities are listed in Section 7.4.7 on page 61.

7.4.1 FIQ

The FIQ (Fast Interrupt reQuest) exception is generated by the interrupt handler within the CL-PS7500FE. This input is delayed by one clock cycle for synchronization before it can affect the processor execution flow. It is designed to support a data transfer or channel process, and has sufficient private registers to

remove the need for register saving in such applications (thus minimizing the overhead of context switching).

NOTE: The FIQ exception can be disabled by setting the F flag in the CPSR (but note that this is not possible from User mode).

If the F flag is clear, the ARM processor checks for a low level on the output of the FIQ synchronizer at the end of each instruction. When a FIQ is detected, the ARM processor performs the following:

1) Saves the address of the next instruction to be executed plus 4 in R14_fiq; saves CPSR in SPSR_fiq.

2) Forces M[4:0]=10001 (FIQ mode) and sets the F and I bits in the CPSR.

3) Forces the PC to fetch the next instruction from address 0x1C.

Returning from FIQ

To return normally from FIQ, use SUBS PC, R14_fiq, #4 to restore both the PC (from R14) and the CPSR (from SPSR_fiq) and resume execution of the interrupted code.

7.4.2 IRQ

The IRQ (Interrupt ReQuest) exception is a normal interrupt caused by the interrupt handler within the CL-PS7500FE. It has a lower priority than FIQ, and is masked out when a FIQ sequence is entered. Its effect can be masked out at any time by setting the I bit in the CPSR (but note that this is not possible from User mode).

If the I flag is clear, the ARM processor checks for a low level on the output of the IRQ synchronizer at the end of each instruction. When an IRQ is detected, the ARM processor performs the following:

1) Saves the address of the next instruction to be executed plus 4 in R14_irq; saves CPSR in SPSR_irq.

2) Forces M[4:0]=10010 (IRQ mode) and sets the I bit in the CPSR.

3) Forces the PC to fetch the next instruction from address 0x18.

Returning from IRQ

To return normally from IRQ, use SUBS PC,R14_irq, #4 to restore both the PC and the CPSR and resume execution of the interrupted code.

7.4.3 Abort

An abort is signalled by the internal MMU and indicates that the current memory access cannot be completed. For instance, in a virtual memory system the data corresponding to the current address may have been moved out of memory onto a disk, and considerable processor activity may be required to recover the data before the access can be performed successfully.

The abort mechanism allows a *demand paged virtual memory system* to be implemented when suitable memory management software is available. The processor is allowed to generate arbitrary addresses, and when the data at an address is unavailable, the MMU signals an abort. The processor traps into system software that must then work out the cause of the abort, make the requested data available, and retry the aborted instruction. The application program needs no knowledge of the amount of memory available to it, nor is its state in any way affected by the abort.

The ARM processor checks for ABORT during memory access cycles. When successfully aborted the ARM processor responds in one of two ways:

- a prefetch abort
- a data abort

Prefetch Abort

If the abort occurred during an instruction prefetch (a *prefetch abort*), the prefetched instruction is marked as invalid but the abort exception does not occur immediately. If the instruction is not executed, for example, as a result of a branch being taken while it is in the pipeline, no abort occurs. An abort takes place if the instruction reaches the head of the pipeline and is about to be executed.

Data Abort

If the abort occurred during a data access (a *data abort*), the action depends on the instruction type:

- Single data transfer instructions (LDR and STR) write back modified base registers and the Abort handler must be aware of this.
- The swap instruction (SWP) is aborted as though it had not executed, though externally the read access can occur.
- Block data transfer instructions (LDM and STM) complete and, if write-back is set, the base updates. If the instruction would normally have overwritten the base with data (such as, LDM with the base in the transfer list), this overwriting is prevented. All register overwriting is prevented after the abort is indicated, which particularly means that R15 (always last to transfer) is preserved in an aborted LDM instruction.

Abort Sequence

When either a prefetch or data abort occurs, the ARM processor performs the following:

1) Saves the address of the aborted instruction plus four (for prefetch aborts) or eight (for data aborts) in R14_abt; saves CPSR in SPSR_abt.

2) Forces M[4:0]=10111 (Abort mode) and sets the I bit in the CPSR.

3) Forces the PC to fetch the next instruction from either:

a) address 0x0C (prefetch abort), or

b) address 0x10 (data abort).

Returning from an Abort

To return after fixing the reason for the abort, use SUBS PC, R14_abt, #4 (for a prefetch abort) or SUBS PC, R14_abt, #8 (for a data abort). This restores both the PC and the CPSR, and retry the aborted instruction.

7.4.4 Software Interrupt

The SWI gets into Supervisor mode, usually to request a particular supervisor function. When a SWI is executed, the ARM processor performs the following:

1) Saves the address of the SWI instruction plus four in R14_svc; saves CPSR in SPSR_svc.

2) Forces M[4:0]=10011 (Supervisor mode) and sets the I bit in the CPSR.

3) Forces the PC to fetch the next instruction from address 0x08.

Returning from a SWI

To return from a SWI, use MOVS PC, R14_svc. This restores the PC and CPSR, and return to the instruction following the SWI.

7.4.5 Undefined Instruction Trap

When the ARM processor comes across an instruction that it cannot manage, it takes the undefined instruction trap. This includes all coprocessor instructions, except MCR and MRC operations that access the internal control coprocessor.

The trap can be used for software emulation of a coprocessor in a system that does not have the coprocessor hardware, or for general-purpose instruction set extension by software emulation.

When the ARM processor takes the undefined instruction trap, it performs the following:

1) Saves the address of the Undefined or coprocessor instruction plus 4 in R14_und; saves CPSR in SPSR_und.

2) Forces M[4:0]=11011 (Undefined mode) and sets the I bit in the CPSR.

3) Forces the PC to fetch the next instruction from address 0x04.

Returning from an Undefined Instruction Trap

To return from this trap after emulating the failed instruction, use MOVS PC,R14_und. This restores the CPSR and returns to the instruction following the undefined instruction.

7.4.6 Vector Summary

These are byte addresses and normally contain a branch instruction pointing to the relevant routine.

The FIQ routine might reside at 0x1C onwards, thereby avoiding the need for (and execution time of) a branch instruction.

Table 7-2. Vector Summary

Address	Exception	Mode on Entry
0x00000000	Reset	Supervisor
0x00000004	Undefined instruction	Undefined
0x00000008	Software interrupt	Supervisor
0x0000000C	Abort (prefetch)	Abort
0x00000010	Abort (data)	Abort
0x00000014	Reserved	–
0x00000018	IRQ	IRQ
0x0000001C	FIQ	FIQ

7.4.7 Exception Priorities

When simultaneous multiple exceptions arise, a fixed-priority system determines their order as:

1) Reset (highest priority)

2) Data abort

3) FIQ

4) IRQ

5) Prefetch abort

6) Undefined instruction, software interrupt (lowest priority)

NOTE: Not all exceptions can occur at once. Undefined instruction and software interrupt are mutually exclusive since they each correspond to particular (non-overlapping) decodings of the current instruction.

If a data abort occurs at the same time as a FIQ and FIQs are enabled (that is, the F flag in the CPSR is clear), the ARM processor enters the data abort handler and then immediately proceed to the FIQ vector. A normal return from FIQ causes the data abort handler to resume execution. Placing data abort at a higher priority than FIQ is necessary to ensure that the transfer error does not escape detection; the time for this exception entry should be added to worst-case FIQ latency calculations.

7.4.8 Interrupt Latencies

Calculating the worst-case interrupt latency for the ARM processor is quite complex due to the cache, MMU and write buffer and is dependent on the configuration of the whole system.

7.4.9 Reset

When the CL-PS7500FE is reset, the ARM processor abandons the executing instruction and then performs idle cycles from incrementing word addresses.

When the CL-PS7500FE comes out of reset, the ARM processor does the following:

1) Overwrites R14_svc and SPSR_svc by copying the current values of the PC and CPSR into them. The value of the saved PC and CPSR is not defined.

2) Forces M[4:0] = 10011 (Supervisor mode) and sets the I and F bits in the CPSR.

3) Forces the PC to fetch the next instruction from address 0x00.

End of Reset Sequence

At the end of the reset sequence:

- The MMU disabled and the TLB flushed, forcing ‘flat’ translation (that is, the physical address is the virtual address, and there is no permission checking)
- Alignment faults also disabled
- The cache disabled and flushed
- The write buffer disabled and flushed
- The ARM7 CPU core placed into 26-bit Data and Address mode, with early abort timing and Little Endian mode

7.5 Configuration Control Registers

The operation and configuration of the ARM processor is controlled both directly via coprocessor instructions and indirectly through the Memory Management Page tables.

The coprocessor instructions manipulate a number of on-chip registers that control the configuration of the Cache, write buffer, MMU, and several other configuration options.

7.5.1 Backward Compatibility

To ensure backward compatibility of future CPUs:

- Program all reserved or unused bits in registers and coprocessor instructions to ‘0’.
- Invalid registers must not be read/written.
- Program the following bits to ‘0’:
  - register 1, bits 31:11
  - register 2, bits 13:0
  - register 5, bits 31:0
  - register 6, bits 11:0
  - register 7, bits 31:0

NOTE: Program the areas marked ‘reserved’ in the register and translation diagrams to ‘0’ for future compatibility.

7.5.2 Internal Coprocessor Instructions

The on-chip registers can be read using MRC instructions and written using MCR instructions. These operations are only allowed in non-user modes and the undefined instruction trap is taken if accesses are attempted in user mode. Refer to the *CL-PS7500FE Programmer’s Guide*.

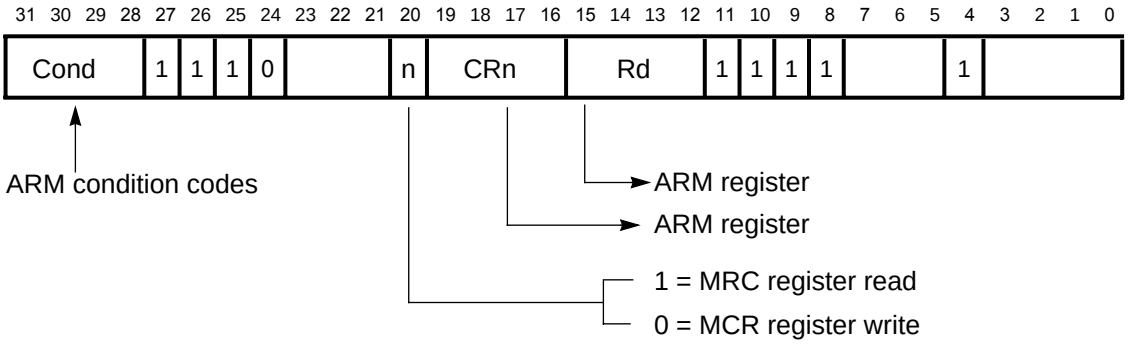


Figure 7-5. Format of Internal Coprocessor Instructions MRC and MCR

7.5.3 Registers

The ARM processor contains registers that control the cache and MMU operation. These registers are accessed using CPRT instructions to coprocessor #15 with the processor in a privileged mode.

Only some of registers 0–7 are valid:

- An access to an invalid register causes neither the access or an undefined instruction trap, and therefore should never be carried out.
- An access to any of the registers 8–15 cause the undefined instruction trap to be taken.

Table 7-3. Cache and MMU Control Registers

Register	Register Reads	Register Writes
0	CPU ID	Reserved
1	Reserved	Control
2	Reserved	Translation Table Base
3	Reserved	Domain Access Control
4	Reserved	Reserved
5	Fault Status	Flush TLB
6	Fault Address	Purge TLB
7	Reserved	Flush IDC
8–15	Reserved	Reserved

7.6 Register 1: Control (Write only)

MSB	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	LSB
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	R	S	B	1	D	P	W	C	A	M	

This register is write-only and contains control bits. All bits in this register are forced low at reset.

M Bit 0 — **ENABLE/DISABLE:** When this bit is ‘0’, the on-chip MMU is turned off. When this bit is ‘1’, the on-chip MMU is turned on.

A Bit 1 — **ADDRESS FAULT ENABLE/DISABLE:** When this bit is ‘0’, the alignment fault is disabled. When this bit is ‘1’, the alignment fault is enabled.

C Bit 2 — **CACHE ENABLE/DISABLE:** When this bit is ‘0’, the instruction/data cache is turned off. When this bit is ‘1’, the instruction/data cache is turned on

W Bit 3 — **WRITE BUFFER ENABLE/DISABLE:** When this bit is ‘0’, the WB is turned off. When this bit is ‘1’, the WB is turned on.

P Bit 4 — **ARM 32/26-BIT PROGRAM SPACE:** When this bit is ‘0’, the 26-bit program space is selected. When this bit is ‘1’, the 32-bit program space is selected.

D Bit 5 — **ARM 32/26-BIT DATA SPACE:** When this bit is ‘0’, the 26-bit data space is selected. When this bit is ‘1’, the 32-bit data space is selected.

B Bit 7 — **BIG/LITTLE ENDIAN:** When this bit is ‘0’, the CL-PS7500FE is in little-endian operation. When this bit is ‘1’, the CL-PS7500FE is in big-endian operation.

S Bit 8 — This system bit controls the ARM processor permission system.

R Bit 9 — This ROM bit controls the ARM processor permission system.

7.7 Register 2: Level One Page Table (Write only)

This register is write only and holds the base of the currently active Level One page table.

7.8 Register 3: Domain Access Control (Write only)

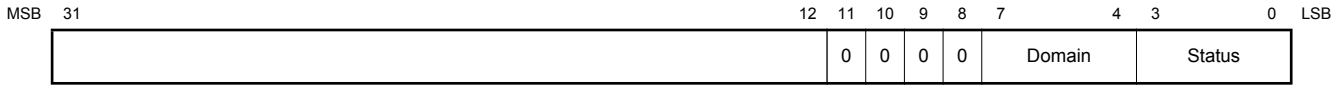
MSB	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	LSB
	15		14		13		12		11		10		9		8		7		6		5		4		3		2		1		0		

This register is write only and holds the current access control for domains 0–15. See Section 6.9 on page 48 for the access permission definitions and other details.

7.9 Register 4: Reserved

This register is reserved. Access of this register has no effect and should never be attempted.

7.10 Register 5: Fault Status/Translation Lookaside Buffer Flush



Read: Fault Status

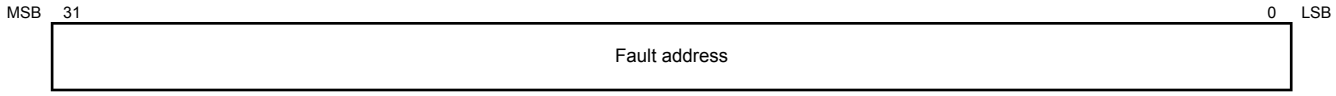
A read of this register returns the status of the last data fault. It is not updated for a prefetch fault. (See Section 6 on page 38 for more details.)

Note that only the bottom 12 bits are returned. The upper 20 bits are the last value on the internal data bus, and therefore have no meaning. Bits 11:8 are always returned as zero.

Write: Translation Look-aside Buffer Flush

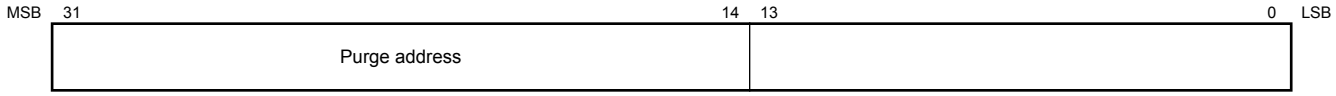
A write to this register flushes the TLB. The data written is discarded.

7.11 Register 6: Fault Address/TLB Purge



Read: Fault Address

A read of this register returns the virtual address of the last data fault.



Write: TLB Purge

A write of this register purges the TLB; the data is treated as an address, and the TLB is searched for a corresponding page table descriptor. If a match is found, the corresponding entry is marked as invalid. This allows the page table descriptors in main memory to be updated and invalid entries in the on-chip TLB to be purged without requiring the entire TLB to be flushed.

7.12 Register 7: IDC Flush (Write only)

This register is write only. Data written to this register is discarded and the IDC is flushed.

7.13 Registers 8–15: Reserved

An access of any of these registers causes the undefined instruction trap to be taken.

8. MEMORY MAP

All addresses featured in the CL-PS7500FE memory map table are physical addresses.

Table 8-1. CL-PS7500FE Memory Map Table

Memory (Mbytes)	Address (Hex)	To (Hex)	Device
0	00000000	00FFFFFF	ROM bank 0
16	01000000	01FFFFFF	ROM bank 1
32	02000000	02FFFFFF	Reserved
48	03000000	0300FFFF	Module I/O space
	03010000	0302BFFF	16-MHz PC-style I/O
	0302C000	0302FFFF	Reserved
	03030000	0303FFFF	Further module I/O space
	03040000	031FFFFF	Reserved
	03200000	0320FFFF	CL-PS7500FE registers
	03210000	033FFFFF	Simple I/O space
	03400000	034FFFFF	Video registers
	03500000	03FFFFFF	Reserved
64	04000000	07FFFFFF	Reserved
128	08000000	0FFFFFFF	Extended I/O space
256	10000000	DRAM bank 0	
320	14000000	DRAM bank 1	
384	18000000	DRAM bank 2	
448	1C000000	DRAM bank 3	
512	20000000	ROM bank 0 (repeated)	

9. MEMORY SUBSYSTEMS

9.1 ROM Interface

The CL-PS7500FE ROM interface supports both non sequential and burst mode read and write cycles, with a range of programmable timings for each type. A single chip select signal, nROMCS, is generated for addresses between 0x00000000 and 0x01FFFFFF that can be split externally to provide separate chip selects for two 16-Mbyte banks of ROM. Each bank of ROM can be 16- or 32-bits wide. The ROM access time depends on the MEMCLK frequency and to enable slow ROMs to be used with a high-frequency MEMCLK, there is a half-speed bit available that causes all ROM timings to slow to twice as many MEMCLK cycles, when the half-speed bit is set to '0'.

The ROM interface of CL-PS7500FE can also support write cycles with the generation of an output and write enable. The feature is disabled on reset so that write cycles do not:

- produce a chip select – nROMCS,
- write enable, or
- drive the data out onto the external data bus

When this feature is disabled, an output enable is still generated on read cycles.

The ability to write data to ROM space devices is primarily intended to allow the programming of FLASH devices directly. With only one write enable, byte writes to 16- or 32-bit-wide devices are not handled directly. External logic can decode address bits LA[1:0], and the write enable can enable a full SRAM interface to be generated (if required). However, the interface is not designed to provide a high-performance interface to SRAM.

Assuming a MEMCLK frequency of 32 MHz, the access time for a non-sequential cycle can be varied from 220 to 62.5 ns in 31.25-ns increments. For burst mode cycles, the two LSBs of the latched address from the CL-PS7500FE increment to allow up to four sequential reads. The access time for burst mode cycles can be programmed from 125 ns down to 62.5 ns, again in 31.25-ns increments.

NOTE: Due to the timing of the write enable, the smallest cycle length for a write cycle is three MEMCLK cycles – that is, 93.75 ns.

If a frequency other than 32 MHz is used for MEMCLK, these timings scale accordingly.

Support for 16-bit-wide ROMs is provided through a programmable bit in each of the ROM control registers. If a 16-bit-wide device is selected, then two memory system cycles are required to fetch the full 32-bit word required by the ARM. If burst mode is disabled for that bank, then CL-PS7500FE performs two non-sequential fetches using the programmed non-sequential timing, latch the intermediate 16-bit value, and present the full 32-bit word to the ARM processor macrocell.

If the burst mode timing bits are programmed into an enabled state, then the first 16-bit read is a standard non-sequential cycle, but the second is a burst mode cycle to minimize the total access time.

When a 16-bit-wide ROM bank is being addressed, the ROM address is shifted up by one bit such that the LSB appears on LA[2], thus allowing the same PCB layout to be used for 16-bit or 32-bit ROM banks.

When using a 16-bit-wide ROM device, data must be stored so that the least-significant bytes of a 32-bit word are stored at the lower memory address:

Contents																Address
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0x00000000
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0x00000001

When this is read, the ARM sees:

MSB LSB

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.1.1 ROM Bank Configuration and Timing

There are two identical registers that control the configuration and timing of the two ROM banks. Both registers default to read-only 16-bit mode and the slowest possible non-sequential timings on reset. This means that one of the first actions when using 32-bit-wide ROM must be to reprogram these registers for 32-bit-wide operation. A detailed description of how to boot up a CL-PS7500FE system using 32-bit-wide ROM is contained in Appendix A, “Initialization and Boot Sequence”.

7	6	5	4	3	2	1	0
W	S	H	B	B	N	N	N

To program these registers, write a byte to 0x03200080 for the ROMCR0 register (address range 0x00000000 to 0x00FFFFFF) or to 0x03200084 for the ROMCR1 register (address range 0x01000000 to 0x0FFFFFFF). The details of these registers are shown below.

N non-sequential access time (H = 1):
- 000 7 MEMCLK cycles
- 001 6 MEMCLK cycles
- 010 5 MEMCLK cycles
- 011 4 MEMCLK cycles
- 100 3 MEMCLK cycles
- 101 2 MEMCLK cycles

B burst mode access time (H = 1):
- 00 Burst Off
- 01 4 MEMCLK cycles
- 10 3 MEMCLK cycles
- 11 2 MEMCLK cycles

H half-speed select, that is, double the above cycle time when H=0

S 16/32-bit mode

W Write enable

Write bit[7]
- 0 writes disabled
- 1 writes enabled

	bit[6]	
	0	32-bit
	1	16-bit
	bit[5]	
	0	half speed mode
	1	normal speed
Read	return above values	
Reset	set to 0x40, that is, 16-bit, slowest access time, and writes disabled.	

The output and write enable signals are output on the pins nIOR and nIOW respectively. This reuse of I/O signals is not expected to cause any difficulties since I/O chip selects is not active during accesses to ROM space.

9.2 DRAM Interface

The DRAM interface can directly drive four banks of DRAM to give a maximum of 64 Mbytes in each DRAM bank:

- Four nRAS strobes to select the bank
- Four nCAS strobes to select the byte within the word
- Twelve multiplexed row/column address lines RA[11:0]

The nRAS strobes are decoded directly from bits 27 and 26 of the address. This means that the DRAM address space is non-contiguous if the entire 64 Mbytes is not used for each bank.

The DRAM controller supports page mode burst cycles with up to 255 sequential accesses in a burst. Each of the four banks can be a 16- or 32-bit-wide device.

The interface can be programmed to support either Fast Page or EDO type DRAMs. When EDO DRAM has been selected, the data is latched into CL-PS7500FE one cycle later, taking advantage of the data latches resident in the output stage of the DRAM. The memory clock frequency can then be increased to realize the greater sequential access bandwidth available with EDO DRAMs.

NOTE: With a lower frequency memory clock, the interface may support EDO DRAM even without the configuration bit being set.

Support is provided for CAS-before-RAS refresh, and direct programmability of the nRAS and nCAS outputs through a special register allows software to directly control self-refresh DRAM.

DRAM cycle speed is controlled by the frequency of MEMCLK. Non-sequential DRAM cycles require between five and nine MEMCLK cycles, depending on the selected mode and RAS pre-charge requirements. Page mode sequential cycles require two MEMCLK cycles.

9.2.1 DRAM Control Registers

There are three registers associated with DRAM control:

DRAMCR has seven bits, including four (one for each bank) to allow selection between 16- and 32-bit modes of operation for each bank. Of the three remaining bits:

- One selects EDO memory support
- One inserts an extra wait state between falling nRAS and falling nCAS on read cycles to preserve t_{RAC}

- The final bit selects between 3 and 4 MEMCLK cycles of minimum nRAS[x] precharge time, t_{RP}

SELFREF allows direct forcing of the nRAS and nCAS outputs. The default state of each of these bits is '0', allowing normal operation of the nRAS and nCAS outputs. But, when a bit is set high, the relevant nCAS or nRAS output is immediately forced active (low).

REFCR controls the refresh rate for CAS-before-RAS refresh. There are four possible refresh periods from 128–16 µs.

9.2.2 DRAM Address Multiplexing

The multiplexing of the DRAM address onto the RA[11:0] outputs is slightly different for 32- and 16-bit modes. The DRAM address requested by the ARM or DMA controller must be shifted up by one bit in 16-bit mode, to enable two locations to be accessed to read or write one 32-bit word. The row/column address multiplexing arrangements are shown below, where the numbers in the table refer to the address bits provided by the ARM or DMA controller.

32-bit-wide DRAM Bank

RA[11:0]	11	10	9	8	7	6	5	4	3	2	1	0
Row address	24	22	19	18	17	16	15	14	13	12	11	10
Column address	25	23	21	20	9	8	7	6	5	4	3	2

16-bit-wide DRAM Bank

RA[11:0]	11	10	9	8	7	6	5	4	3	2	1	0
Row address	23	21	18	17	16	15	14	13	12	11	10	9
Column address	24	22	20	19	8	7	6	5	4	3	2	•

- This bit is generated separately by DRAM controller to access each 16-bit half-word sequentially.

9.2.3 Selection Between 16- and 32-bit DRAM

7	6	5	4	3	2	1	0
X	P	R	E	S	S	S	S

The DRAMCR at address 0x032000D0 allows the width of each of the four DRAM banks to be defined for CL-PS7500FE. On reset, all banks are defined as 32-bits wide, so if a 16-bit system is being used it is necessary to program this register before any writes to DRAM occur. It is not possible to write to DRAM in 16-bit mode and read back from the same bank in 32-bit mode, or vice versa.

S	16- or 32-bit mode select, one for each bank
Write	bit[3] bank 3 DRAM width
0	32-bit
1	16-bit

bit[2] bank 2 DRAM width
- 0 32-bit
- 1 16-bit

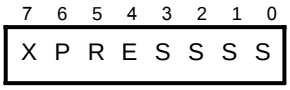
bit[1] bank 1 DRAM width
- 0 32-bit
- 1 16-bit

bit[0] bank 0 DRAM width
- 0 32-bit
- 1 16-bit

Read reads above values

Reset set bits to zero (32-bit)

9.2.4 EDO and Timing Mode Selection



The DRAMCR at address 0x032000D0 also controls EDO mode and some other timing features. On reset all these bits are set low, that is, inactive. To ensure reliable operation in many systems after reset, these register bits must be correctly programmed before the DRAM is used.

Write:

P Precharge RAS control
- 0 3 MEMCLK cycles minimum RAS precharge
- 1 4 MEMCLK cycles minimum RAS precharge

R RAS to CAS delay
- 0 2 MEMCLK cycles RAS to CAS delay on reads
- 1 3 MEMCLK cycles RAS to CAS delay on reads

E EDO Control
- 0 Fast Page DRAMs selected
- 1 EDO DRAMs selected

Read reads above values

Reset set all bits to zero (Fast page, no extra delays)

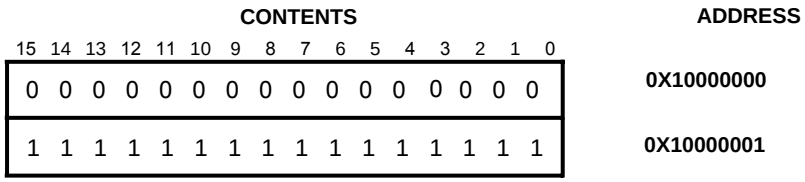
To take advantage of the faster page mode accesses of EDO DRAMs, the memory clock frequency should be increased accordingly. For example, a system using 80-ns Fast Page DRAMs require a memory clock of ~32 MHz; 80-ns EDO DRAMs could use a memory clock of ~50 MHz. This improves the asymptotic DRAM bandwidth from 64 to 100 Mbytes for a 32-bit-wide system.

However, the increase in memory clock may cause the Trac time and the Trp times to be violated at 4 and 3 MEMCLK cycles respectively (when EDO selected). The register configuration bits R and P allow each of these to be increased by one MEMCLK cycle when appropriate.

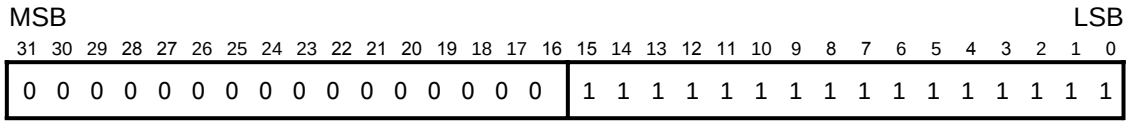
16-bit Mode

In 16-bit mode CL-PS7500FE must perform two reads or writes for each 32-bit word DRAM access requested by the ARM processor or the DMA controller. Only nCAS[1] and nCAS[0] are used, to access the two bytes of each word. nCAS[3:2] are held at logic one. In 16-bit mode, the same number of physical

addresses are available as for 32-bit mode, meaning that only 32 Mbytes of DRAM is supported per bank. Words are stored in DRAM with the upper half-word at the lower address.



When this is read, the ARM sees:

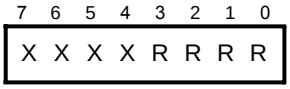


In 16-bit mode, byte reads and writes only require a single DRAM access, and the LSB of the column address is decoded in conjunction with the nCAS[1:0] outputs to select a single byte from four. For this reason byte reads and writes for 16-bit-wide DRAMs have the same timing as for the non-sequential 32-bit case.

16-bit mode word accesses involve a non-sequential access for the upper halfword, followed by a sequential access for the lower half word at the next memory location. A non sequential 16-bit mode word access thus requires 7 MEMCLK cycles, then sequential accesses can continue until a page boundary is reached, taking two cycles for each half word.

9.2.5 DRAM Refresh

DRAM refresh is controlled by a small state machine and counter within CL-PS7500FE. The refresh interval timer is clocked by a clock derived from the fixed frequency I_OCLK, and thus the refresh intervals remain the same even if the frequency of MEMCLK is increased for use with faster DRAM. There are four timings available for refresh, controlled by the REFCR refresh control register at address 0x0320008C. During reset, the refresh timer is reset to the fastest value (16 µs), and the counter and state machine are clocked such that refresh continues even during reset.



R	refresh period
Write	bit[3:0]
	0000 refresh off
	0001 16 µs
	0010 32 µs
	0100 64 µs
	1000 128 µs
	all others are undefined
Read	return above values
Reset	set to ‘0001’ (fastest available refresh rate)

9.2.6 DRAM Self-Refresh

7	6	5	4	3	2	1	0
C	C	C	C	R	R	R	R

The nCAS and nRAS lines can be forced active by programming bits in the SELFREF register at address 0x032000D4. This is intended for use with self refresh DRAM, and particularly in conjunction with STOP mode so that DRAM can retain state when all the CL-PS7500FE clocks have been stopped. All DMA must be stopped and the code that writes to this register must be executing from ROM.

C	force all nCAS low	
R	force all nRAS low	
Write	bits[7:4]	
	0	normal
	1	force to zero
	bits[3:0]	
	0	normal
	1	force to zero
Read	reads above values	
Reset	set bits to zero (normal)	

9.2.7 Non-Sequential Access Time and RAS Precharge

At the end of one DRAM access, the earliest the next access can start is two memory clock cycles later. The new access must be to a different DRAM bank than the previous for this to be allowed. If the new access is to the same bank as the previous, to maintain the RAS precharge time (t_{RP}), an extra clock cycle is inserted before the nRAS[x] signal is asserted again.

Thus, the minimum RAS precharge time is guaranteed to be 3 MEMCLK cycles. This can be increased to 4 MEMCLK cycles by setting DRAMCR[7] high. These wait states increase the access time of a non-sequential DRAM access by 1 or 2 cycles.

To provide adequate RAS access delay (t_{RAC}) at higher memory clock frequencies, DRAMCR[6] can be set. This inserts a wait state between the falling nRAS and the first falling nCAS of a read cycle.

Setting bit 5 of the DRAMCR delays the latching of data into CL-PS7500FE by one cycle to support EDO DRAMs and so increases non-sequential access time by one cycle.

Table 9-1 shows how to calculate the non-sequential DRAM read access time:

Table 9-1. Non-Sequential DRAM Read Access Times

DRAMCR		
	Bit 6 = 0	**Bit 6 = 1**
Fast Page (bit 5 = 0)	5	6
EDO (bit 5 = 1)	6	7

The non-sequential write access remains at 5 cycles for the above conditions.

To preserve minimum RAS precharge times when one access closely follows another to the same DRAM bank, the following must be added to these values

if bit 7 is low	0 or 1 cycles
if bit 7 is high	0, 1, or 2 cycles

9.3 DMA Channels

The CL-PS7500FE supports video, cursor and sound DMA to enable direct transfer of qwords of data from DRAM to the video and sound processing interfaces. All DMA is in units of four words (qwords) and data can be read from any of the four banks of DRAM in either 16- or 32-bit mode. CL-PS7500FE contains a DMA address generator that has a number of programmable control registers associated with each channel. Most of these registers contain 28-bit physical addresses. The DMA controller also includes support for DMA to dual-panel LCD screens.

All three of the DMA channels have at least one CURRENT register that contains the address in memory of the next data to be fetched from DRAM on that channel. Each channel uses START, INIT, and END registers to define the size and location of the buffer in memory where DMA occurs. However, all three channels have slightly different methods of using these registers. Exact details of the contents of all these registers can be found in Chapter 10, “MEMORY AND I/O PROGRAMMERS’ MODEL”.

9.3.1 Video DMA

The video DMA channel can be used in two modes. Duplex mode fetches DMA data for use with a dual-panel LCD display, and involves fetching a qword of data for the top half of the display, followed by a qword of data for the bottom half of the display, then the next qword for the top half and so on. This is implemented using two parallel sets of registers that must be programmed accordingly. A description of how to use the CL-PS7500FE with a dual panel LCD display can be found in Appendix B, “Dual-Panel Liquid Crystal Displays”.

Normal mode is used for standard CRT and LCD displays and data is fetched sequentially from the frame buffer. Selection between normal and duplex mode of operation is achieved through VIDCR[7] at location 0x032001E0. VIDCR[5] enables the video DMA channel and should not be enabled until the other address registers are programmed to sensible values.

The registers associated with video DMA should only be programmed during the FLYBACK period, to avoid corrupting data while DMA is in progress or while the display is half way through a raster. The state of the internal FLYBACK signal is available for polling in the IOCR, and can create an interrupt by programming the IRQA mask register appropriately.

There is a single VIDSTART register that should be programmed with the location in memory of the first qword of video data at the start of the frame buffer. The VIDEND register is programmed with the location in memory of the start of the last qword in the frame buffer image.

For normal mode operation, the VIDINITA register should be programmed with the address in memory of the data that creates the pixels at the top-left corner of the display. This need not necessarily be at the same address as that programmed into the VIDSTART register, thus allowing hardware scrolling by moving the address in the VIDINITA register through the frame buffer. The value in the VIDINITA register is automatically transferred into the VIDCURA register during the FLYBACK period, so there is no need to program the current register separately.

For normal operation, the VIDINITB register should be programmed to 0x00000000, so that the value in the VIDCURB register is defined. All video channel registers should be programmed with addresses that are qword aligned (that is, bits 3:0 are '0').

There is an extra bit (30) in the VIDINITA register that must be programmed high if the address in the VIDINITA register is the same as the address in the VIDEND register. At all other times it should be programmed low.

Once all bits have been programmed, the enable bit in the VIDCR can be written to, and the video DMA channel becomes operational. The channel is then controlled by a video request signal from the video controller part of CL-PS7500FE. When a request for more video data arrives and the current bus cycle finishes, the bus controller arbitrates in favor of the DMA (having the highest priority on the bus) to fetch a qword of data for the video sub system. Immediately after each DMA access, the address in the current register is incremented by 16 (one qword) and the address is compared with the address in the VIDEND register. If they are the same, the DMA controller knows that the next DMA is the last one in the buffer, and after the next DMA, the current register is reloaded from the VIDSTART register. During the FLYBACK period, the current register is automatically reloaded with the value in the VIDINITA register.

Programming of the DMA and video subsystem for use with dual panel LCDs is described in full in Appendix B, "Dual-Panel Liquid Crystal Displays", and uses identical principles, except there are two Current registers and two Init registers – one for each panel. On each successive DMA access, the CL-PS7500FE toggles between the two sets of registers providing data first for the upper panel and then from the lower panel. This means that the two init registers should always be programmed with addresses with are equidistantly spaced through the wrapped-around frame buffer.

9.3.2 Cursor DMA

There are only two registers associated with the cursor channel, the CURSCUR current register and the CURSINIT register. The channel is enabled under the control of the video enable bit in the VIDCR. The operation of the channel is the same for normal or duplex modes, but it is necessary to program the cursor differently depending on the mode being used. Details of the programming required can be found in Appendix B, "Dual-Panel Liquid Crystal Displays".

The CURSINIT register should be programmed with the address of the first word of cursor data in memory. There is no END register as the width of the cursor is predetermined (32 pixels) and the height of the cursor is defined by programming the VCSR and VCER in the video subsystem. Each qword fetches results in two rasters worth of cursor data being transferred (except in HiRes mode). At the end of each fetch, the value in the CURSCUR is increased by 16, to address the start of the next qword. The value programmed into the CURSINIT register must be qword-aligned.

9.3.3 Sound DMA

The Sound DMA channel provides data for the CL-PS7500FE sound interface. There are two sets of pointer registers so that data transfers can be double buffered to ensure that DMA data is always available even when the data in one buffer is exhausted. One set of registers can be reprogrammed while the others are being used.

Sound DMA transfers are constrained to a single 4-Kbyte page, as only the lowest 12 bits of the DMA address are incremented and compared to check for the end of the buffer. All sound DMA is qword and must be from qword aligned addresses, so the lowest four bits of the registers are not used and should be programmed to zero. Bit 30 of each of the END registers is the 'last' bit and must be programmed high if the initial value in the current register is the same as the end register for that buffer (that is, for a single transfer).

There is also an interrupt mask and status bit for the sound channel that allows the status of the sound DMA state machine to be monitored. The state machine generates an interrupt when the end of the current buffer is reached, and it is up to the system software to take appropriate action to reprogram that channel as required while DMA continues from the location pointed to by the other set of buffers.

Sound data is requested by the CL-PS7500FE sound subsystem that asserts a request signal, and the bus controller arbitrates in favor of the sound DMA when the current bus cycle has completed as long as there is not an outstanding video or cursor DMA request.

9.3.4 The Sound DMA State Machine

The sound DMA channel is controlled by a simple state machine. The state machine remains in an idle state when the enable bit in the sound DMA control register has not been set. The state bits of the state machine are directly mapped to the Sound DMA status register, where they are named Overrun, Int and A/B. On reset, the state machine is set to '110', setting the Overrun and Int bits. The Overrun bit indicates when a channel has stopped because it has finished a transfer and the other pointer pair is not programmed. The Int bit indicates when the channel is requesting an interrupt. The A/B bit indicates the pair of current/end pointers in use.

The state machine diagram in Figure 9-1 on page 77 shows how the state machine transfers between buffers A and B to allow DMA to continue uninterrupted when both sets of DMA address registers have been programmed. The transitions between states occur either when the ARM processor programs an pointer register pair, or when a buffer is completed. To ensure correct operation, the current pointer must be programmed before the end pointer as it is the action of programming the end pointer that causes the state transition. The 'stop' bit in the end register terminates a sequence of DMA, by forcing the state machine back into one of the idle states at the end of the last buffer.

During operation of the state machine, when the end of one buffer is reached, an interrupt is generated that can be used to signal to the ARM processor that it is time to reprogram that pair of pointers. If one buffer's address pointers have not been reprogrammed before the other buffer is exhausted, then both the Int and Overrun bits are set, and DMA cannot continue until the pointers are reprogrammed.

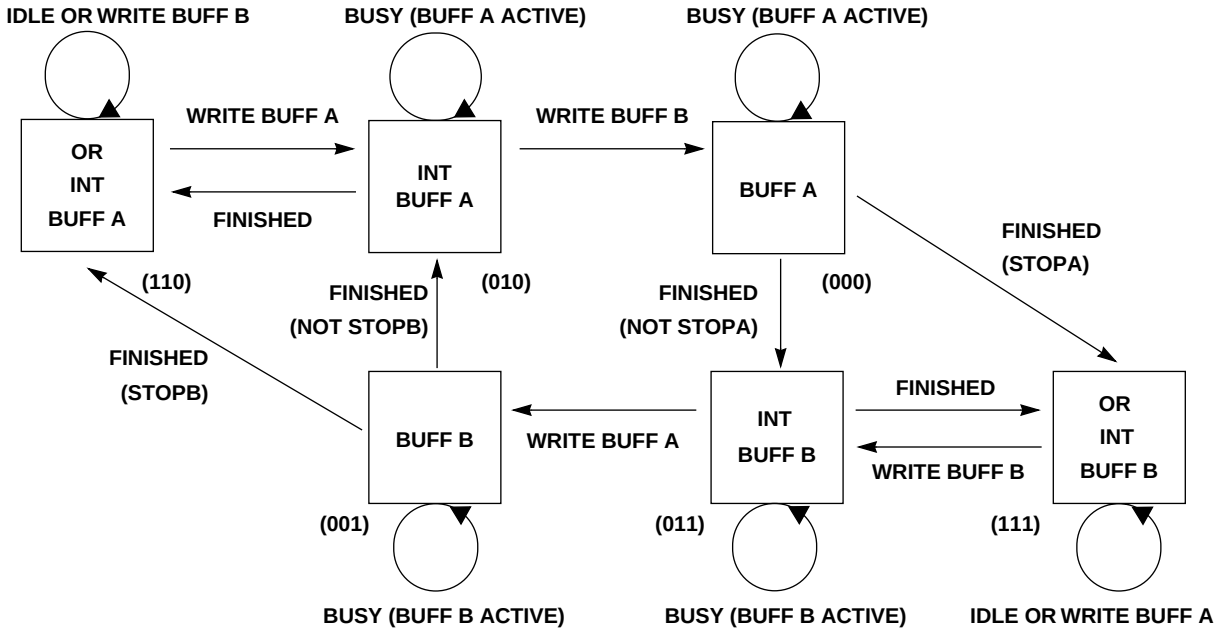


Figure 9-1. Hardware DMA State Machine Diagram

10. MEMORY AND I/O PROGRAMMERS' MODEL

10.1 Introduction

The CL-PS7500FE contains over 100 programmable registers (in addition to those in the ARM processor and the 256 video palette entries), grouped into three sets. Registers inside the ARM processor are described in Those inside the video and sound macrocell are all programmed by writing to memory locations 0x03400000 to 0x034FFFFF, with the upper bits of the programmed data determining which video/sound register is to be programmed. All these registers are write-only, and are described in Chapter 12 and Chapter 13. The remaining CL-PS7500FE registers are programmed by writing a full 32-bit data word to an address between 0x03200000 and 0x032001F8. Although most of these registers are only 8- or 16-bits wide, all the register addresses are word-aligned. All the CL-PS7500FE registers that do not form part of the ARM processor or the video and sound macrocell are described in the following sections.

10.2 Register Summary

In Table 10-1, all addresses are hex and relative to the base address 0x03200000.

Table 10-1. CL-PS7500FE Registers

Name	Address	Size (Bits)	Read	Write	Function	Page
IOCR	00	8	✓[a]	✓	I/O control	81
KBDDAT	04	8	✓	✓	Keyboard data	81
KBDCR	08	8	✓	✓	Keyboard control	82
IOLINES	0C	8	✓	✓	General-purpose I/O lines	83
IRQSTA	10	8	✓	✗[b]	IRQA status	83
IRQRQA	14	8	✓	✓	IRQA request/clear	84
IRQMSKA	18	8	✓	✓	IRQA mask	84
SUSMODE	1C	8	✓	SUSPEND	Enter SUSPEND mode	85
IRQSTB	20	8	✓	✗	IRQB status	85
IRQRQB	24	8	✓	✗	IRQB request	86
IRQNSKB	28	8	✓	✓	IRQB mask	86
STOPMODE	2C	8	✗	STOP	Enter STOP mode	87
FIQST	30	8	✓	✗	FIQ status	87
FIQRQ	34	8	✓	✗	FIQ request	87
FIQMSK	38	8	✓	✓	FIQ mask	88
CLKCTL	3C	8	✓	✓	Clock divider control	88
T0low	40	8	✓	✓	Timer 0 low bits	89

Table 10-1. CL-PS7500FE Registers *(cont.)*

Name	Address	Size (Bits)	Read	Write	Function	Page
T0high	44	8	✓	✓	Timer 0 high bits	89
T0GO	48	8	✗	GO	Timer 0 go command	89
T0LAT	4C	8	✗	LATCH	Timer 0 latch command	89
T1low	50	8	✓	✓	Timer 1 low bits	89
T1high	54	8	✓	✓	Timer 1 high bits	89
T1GO	58	8	✗	GO	Timer 1 go command	90
T1LAT	5C	8	✗	LATCH	Timer 1 latch command	90
IRQSTC	60	8	✓	✗	IRQC status	90
IRQRQC	64	8	✓	✗	IRQC request	90
IRQMSKC	68	8	✓	✓	IRQC mask	90
VIDMUX	6C	8	✓	✓	LCD and IIS control bits	91
IRQSTD	70	8	✓	✗	IRQD status	91
IRQRQD	74	8	✓	✗	IRQD request	92
IRQMSKD	78	8	✓	✓	IRQD mask	92
ROMCR0	80	8	✓	✓	ROM control bank 0	93
ROMCR1	84	8	✓	✓	ROM control bank 1	93
REFCR	8C	8	✓	✓	Refresh period	94
ID0	94	8	✓	✗	Chip ID number low byte	94
ID1	98	8	✓	✗	Chip ID number high byte	94
VERSION	9C	8	✓	✗	Chip version number	94
MSEDAT	A8	8	✓	✓	Mouse data	94
MSECR	AC	8	✓	✓	Mouse control	95
IOTCR	C4	8	✓	✓	I/O timing control	95
ECTCR	C8	8	✓	✓	Expansion card timing control	95
ASTCR	CC	8	✓	✓	Asynchronous I/O timing control	96
DRAMCR	D0	8	✓	✓	DRAM control	96
SELFREF	D4	8	✓	✓	Force CAS/RAS lines low individually for self refresh	97

Table 10-1. CL-PS7500FE Registers *(cont.)*

Name	Address	Size (Bits)	Read	Write	Function	Page
ATODICR	E0	8	✓	✓	A-to-D interrupt control	97
ATODSR	E4	8	✓	✗	A-to-D status	98
ATODCC	E8	8	✓	✓	A-to-D convertor control	98
ATODCNT1	EC	16	✓	✗	A-to-D counter 1	99
ATODCNT2	F0	16	✓	✗	A-to-D counter 2	99
ATODCNT3	F4	16	✓	✗	A-to-D counter 3	99
ATODCNT4	F8	16	✓	✗	A-to-D counter 4	99
SD0CURA	180	32	✓	✓	Sound DMA 0 current A	99
SD0ENDA	184	32	✓	✓	Sound DMA 0 end A	100
SD0CURB	188	32	✓	✓	Sound DMA 0 current B	100
SD0ENDB	18C	32	✓	✓	Sound DMA 0 end B	101
SD0CR	190	8	✓	✓	Sound DMA control	102
SD0ST	194	8	✓	✗	Sound DMA status	102
CURSCUR	1C0	32	✓	✓	Cursor DMA current	103
CURSINIT	1C4	32	✓	✓	Cursor DMA INIT	103
VIDCURB	1C8	32	✓	✓	Duplex LCD video current B	103
VIDCURA	1D0	32	✓	✓	Video DMA current A	104
VIDEND	1D4	32	✓	✓	Video DMA end	104
VIDSTART	1D8	32	✓	✓	Video DMA start	104
VIDINITA	1DC	32	✓	✓	Video DMA INIT A	105
VIDCR	1E0	8	✓	✓	Video cursor DMA control	106
VIDINITB	1E8	32	✓	✓	Duplex LCD video INIT B	107
DMAST	1F0	8	✓	✗	DMA interrupt status	108
DMARQ	1F4	8	✓	✗	DMA interrupt request	108
DMASK	1F8	8	✓	✓	DMA interrupt mask	108

[a] '✓' indicates read/writable

[b] '✗' indicates do not write or read

10.3 Register Descriptions

10.3.1 IOCR (0x00) — I/O Control

7	6	5	4	3	2	1	0
F	N	1	1	I	1	C	D

This register controls various I/O functions. The value of the FLYBACK signal from the video subsystem can be examined by reading bit 7 of this register, this is important for GENLOCK as FLYBACK provides information about the vertical timing of the display. The FLYBACK bit also gives information about when the video palette registers can safely be reprogrammed without causing any visual effects. This can only be done during the FLYBACK period, when this bit is set high. Control of the open drain OD[1:0] and ID pins is provided from this register. It is also possible to read the status of the nINT1 pin.

F	FLYBACK value
N	nINT1 value
I	ID open drain pin control
C	OD[1] open drain pin control
D	OD[0] open drain pin control
Write	bits[7:4, 2] ignored
	bit[3, 1:0] open drain pin controls: 0 force pin low 1 pin is input only
Read	bit[7] reads current FLYBACK value from video and sound macrocell
	bit[6] reads current nINT1 pin value
	bits[5:4, 2] read one
	bit[3] reads current ID pin value
	bit[1] reads current OD[1] pin value
	bit[0] reads current OD[0] pin value
Reset	bits[3, 1:0] set as inputs (high)

10.3.2 KBDDAT (0x04) — Keyboard Data

7	6	5	4	3	2	1	0
D	D	D	D	D	D	D	D

D	keyboard data
Write	next byte to be sent over serial interface to keyboard
Read	last byte of data received from keyboard

10.3.3 KBDCR (0x08) — Keyboard Control

7	6	5	4	3	2	1	0
T	T	R	R	E	P	D	C

T transmit status

R receive status

E enable

P received parity

D data pin status

C clock pin status

Write bits[7:4, 2] ignored

bit[3] enable:
- 0 state machine cleared
- 1 state machine enabled

bit[1] force KBDATA pin low:
- 0 don't force low
- 1 force low

bit[0] force KBCLK pin low:
- 0 don't force low
- 1 force low

Read bit[7] TXE shift register empty:
- 0 not ready
- 1 enabled and ready to transmit

bit[6] TXB, transmitter busy:
- 0 not busy
- 1 currently sending data

bit[5] RXF, receive shift register full:
- 0 not full
- 1 ready to read

bit[4] RXB, receiver busy:
- 0 not busy
- 1 currently receiving data

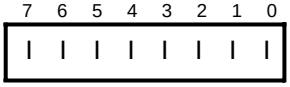
bit[3] ENA, state machine enable:
- 0 disabled
- 1 enabled

bit[2] RXP, receive parity bit, odd parity bit for last received data

bit[1] SKD, KBDATA pin value after synchronization

bit[0] SKC, KBCLK pin value after synchronization

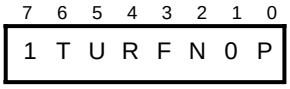
10.3.4 IOLINES (0x0C) — IOP[7:0] Port Control



This register is the control for the 8-bit I/O port included in the CL-PS7500FE. Each bit independently controls the state of one of the open drain I/O pins IOP[7:0]. On reset, all the bits are configured to be inputs.

I	IOP open drain pin
Write	corresponding pin: 0 force corresponding pin low 1 corresponding pin becomes an input
Read	read value on corresponding pin
Reset	set all as inputs

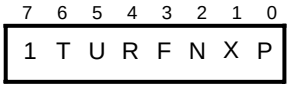
10.3.5 IRQSTA (0x10) — IRQ A Interrupts Status



This is the first of four sets of IRQ interrupt control, masking and status registers in CL-PS7500FE. Not all the bits in each register are used. Note that this status register contains a bit (7) that is always active, and can force an interrupt from software by programming the corresponding bit in the IRQA mask register high.

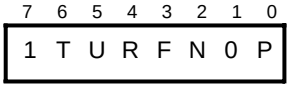
1	always active bit
T	2-MHz timer 1, rising-edge triggered
U	2-MHz timer 0, rising-edge triggered
R	power on reset
F	FLYBACK, rising-edge triggered
N	nINT1, falling-edge triggered
P	INT2, rising-edge triggered
Write	ignored
Read	status bit[7] is always ‘1’ bits[6:2, 0] 0 not triggered since last cleared 1 triggered since last cleared bit[1] is always 0
Reset	clear bits[6:5, 3:2, 0] to ‘0’ power on reset sets bit[4] to ‘1’ push button reset maintains the current bit[4] value

10.3.6 IRQRQA (0x14) — IRQ A Interrupts Request/Clear



1	always active bit
T	2-MHz timer 1, rising-edge triggered
U	2-MHz timer 0, rising-edge triggered
R	power on reset
F	FLYBACK, rising edge triggered
N	nINT1, falling-edge triggered
P	INT2, rising-edge triggered
Write	clear triggered interrupts 0 do not clear interrupt 1 clear interrupt
Read	requests, as status, but bitwise AND'ed with mask

10.3.7 IRQMSKA (0x18) — IRQ A Interrupts Mask



1	always active bit
T	2-MHz timer 1, rising-edge triggered
U	2-MHz timer 0, rising-edge triggered
R	power on reset
F	FLYBACK, rising-edge triggered
N	nINT1, falling-edge triggered
P	INT2, rising-edge triggered
Write	set mask for each interrupt source 0 do not form part of nIRQ 1 form part of nIRQ
Read	value set by write
Reset	set all '0' (none affect nIRQ)

10.3.8 SUSMODE (0x1C) — SUSPEND Mode

7	6	5	4	3	2	1	0
X	X	X	X	X	X	X	S

This register allows the CPU to set the CL-PS7500FE into SUSPEND mode. Only one bit (0) is used, and a write to this bit causes SUSPEND mode to be entered. The value written to bit 0 determines if the external I/O clocks, normally output from the device, are also disabled during SUSPEND mode. The value programmed depends on the peripherals being driven by these clocks.

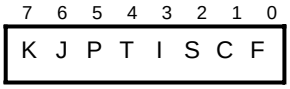
S	SUSPEND mode control of external I/O clocks. Enter SUSPEND mode with MEMCLK, FCLK, I/O clocks, and some internal clocks stopped. DMA continues and the write to this location completes on either wakeup event, nIRQ or nFIQ or reset.
Write	turn off external I/O clocks when in this mode
	0 turn off
	1 do not turn off
Read	return above value
Reset	set to '0'

10.3.9 IRQSTB (0x20) — IRQ B Interrupts Status

7	6	5	4	3	2	1	0
K	J	P	T	I	S	C	F

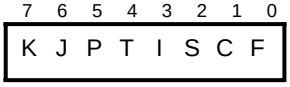
K	keyboard receive interrupt
J	keyboard transmit interrupt
P	nINT3, active-low
T	nINT4, active-low
I	INT5, active-high
S	nINT6, active-low
C	INT7, active-high
F	nINT8, active-low
Write	ignored
Read	status
	0 inactive
	1 active

10.3.10 IRQRQB (0x24) — IRQ B Interrupts Request



K	keyboard receive interrupt
J	keyboard transmit interrupt
P	nINT3, active-low
T	nINT4, active-low
I	INT5, active-high
S	nINT6, active-low
C	INT7, active-high
F	nINT8, active-low
Write	ignored
Read	request, status bitwise AND'ed with mask

10.3.11 IRQMSKB (0x28) — IRQ B Interrupts Mask



K	keyboard receive interrupt
J	keyboard transmit interrupt
P	nINT3, active-low
T	nINT4, active-low
I	INT5, active-high
S	nINT6, active-low
C	INT7, active-high
F	nINT8, active-low
Write	set mask for each interrupt source: 0 do not form part of nIRQ 1 form part of nIRQ
Read	value set by write
Reset	set all '0' (none affect nIRQ)

10.3.12 STOPMODE (0x2C) — STOP Mode

7	6	5	4	3	2	1	0
X	X	X	X	X	X	X	X

This register exists only as an address decode and is used to enter STOP mode. It is very important that DMA activity is stopped before this register is written to. The value written to the register is permanently forced out on the main data bus during STOP mode, and for most systems it is desirable to ensure that this value is 0xFFFFFFFF. The address bus is automatically forced high during STOP mode.

Write	(any value), enter STOP mode with OSCPOWER set low. The write to this register completes on either wakeup event, nEVENT, nEVENT2, or reset
Read	ignored

10.3.13 FIQST (0x30) — FIQ Interrupts Status

7	6	5	4	3	2	1	0
1	F	0	S	0	0	I	D

The FIQ control registers take a similar form to the IRQ registers previously described. Again, bit 7 is always active so that a FIQ interrupt can be forced via software.

1	always active
F	nINT8, active-low
S	nINT6, active-low
I	INT5, active-high
D	INT9, active-high
Write	ignored
Read	status
	0 inactive
	1 active

10.3.14 FIQRQ (0x34) — FIQ Interrupts Request

7	6	5	4	3	2	1	0
1	F	0	S	0	0	I	D

1	always active
F	nINT8, active-low
S	nINT6, active-low
I	INT5, active-high
D	INT9, active-high
Write	ignored
Read	request, status bitwise AND'ed with mask

10.3.15 FIQMSK (0x38) — FIQ Interrupts Mask

7	6	5	4	3	2	1	0
1	F	0	S	0	0	I	D

1	always active
F	nINT8, active-low
S	nINT6, active-low
I	INT5, active-high
D	INT9, active-high
Write	set mask for each interrupt source: 0 do not form part of nFIQ 1 form part of nFIQ
Read	value set by write
Reset	set all '0' (none affect nFIQ)

10.3.16 CLKCTL (0x3C) — Clock Control

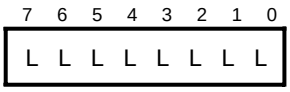
7	6	5	4	3	2	1	0
X	X	X	X	X	F	M	I

On system power up, the clock control register is reset so that all three main clocks have a Divide-by-2 prescale at the inputs to the chip. This register sometimes needs to be reprogrammed as part of the initial tasks of the operating system, to set the prescalars into Divide-by-1 mode.

Divide-by-2 mode allows faster oscillators to be used with less rigorous mark-space requirements.

F	FCLK divide control
M	MEMRFCK divide control
I	I/O clock divide control
Write	bit[2] 0 FCLK × 2 = CPUCLK 1 FCLK = CPUCLK bit[1] 0 MEMRFCK × 2 = MEMCLK 1 MEMRFCK = MEMCLK bit[0] 0 IOCK32 × 2 = I_OCLK 1 IOCK32 = I_OCLK
Read	return above value
Power On Reset only	set all to '0', that is, the Divide-by-2 clocks Push button reset does not affect this register

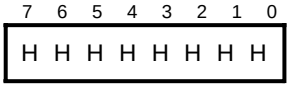
10.3.17 T0low (0x40) — Timer 0 Low Bits



There are eight registers associated with the two 16-bit timers in CL-PS7500FE.

L	low byte of timer
Write	set low byte latch value that is loaded into timer when it reaches end count
Read	read value of low count latched by the ‘Latch’ command T0LAT

10.3.18 T0high (0x44) — Timer 0 High Bits



H	high byte of timer
Write	set high byte latch value that is loaded into timer when it reaches end count
Read	read value of high count latched by the ‘Latch’ command T0LAT

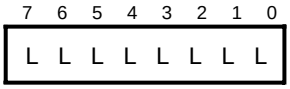
10.3.19 T0GO (0x48) — Timer 0 Go Command

Write	load counter with high and low latch values and start to decrement (value ignored)
Read	ignored

10.3.20 T0LAT (0x4C) — Timer 0 Latch Command

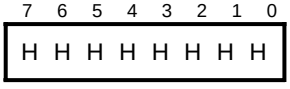
Write	latch timer value in high and low count latches (value ignored)
Read	ignored

10.3.21 T1low (0x50) — Timer 1 Low Bits



L	low byte of timer
Write	set low byte latch value loaded into timer when it reaches end count
Read	read value of low count latched by the ‘Latch’ command T1LAT

10.3.22 T1high (0x54) — Timer 1 High Bits



H	high byte of timer
Write	set high byte latch value that is loaded into timer when it reaches end count
Read	read value of high count latched by the ‘Latch’ command T1LAT

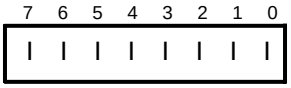
10.3.23 T1GO (0x58) — Timer 1 Go Command

Write	load counter with high and low latch values and start to decrement (value ignored)
Read	ignored

10.3.24 T1LAT (0x5C) — Timer 1 Latch Command

Write	latch timer value in high and low count latches (value ignored)
Read	ignored

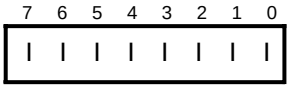
10.3.25 IRQSTC (0x60) — IRQ C Interrupts Status



The IRQC set of control registers control the effect of the IOP[7:0] I/O port bits on the main interrupts. Their functionality is identical to that described for IRQB.

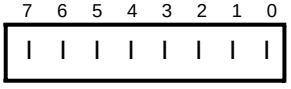
I	IOP[7:0] pins, active-low
Write	ignored
Read	status
	0 inactive
	1 active

10.3.26 IRQRQC (0x64) — IRQ C Interrupts Request



I	IOP[7:0] pins, active-low
Write	ignored
Read	request, status bitwise AND'ed with mask

10.3.27 IRQMSKC (0x68) — IRQ C Interrupts Mask



I	IOP[7:0] pins, active-low
Write	set mask for each interrupt source
	0 do not form part of nIRQ
	1 form part of nIRQ
Read	value set by write
Reset	set all '0' (none affect nIRQ)

10.3.28 VIDMUX (0x6C) — Video LCD and Serial Sound MUX Control

7	6	5	4	3	2	1	0
X	X	X	X	X	X	I	L

This register has two functions:

Bit 1	allows selection of the type of serial sound interface to be supported. The timing of the two possibilities is shown in the Chapter 13.
Bit 0	controls the color LCD multiplexer used with the video pixel clock to double the available bandwidth of color LCD data provided.

Further details of how to use this feature can be found in the video and sound macrocell chapters.

L	color LCD support MUX control
I	Serial Sound Format selection
Write	bit[0] 0 ESEL[0] = EREG[0] 1 ESEL[0] = ECLK bit[1] 0 normal format 1 Japanese format
Read	return above value
Reset	set to ‘0’ (normal)

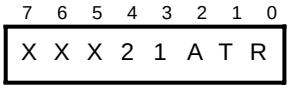
10.3.29 IRQSTD (0x70) — IRQ D Interrupts Status

7	6	5	4	3	2	1	0
X	X	X	2	1	A	T	R

The IRQD control registers are used in an identical way to the IRQB and C registers.

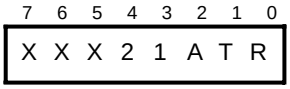
2	nEVENT2, reads back high during an active-low wakeup event 2
1	nEVENT1, reads back high during an active-low wakeup event 1
A	A-to-D, active-high
T	mouse transmit active-high
R	mouse receive active-high
Write	ignored
Read	status bits[7:5] unused bits[4:0] 0 inactive 1 active

10.3.30 IRQRQD (0x74) — IRQ D Interrupts Request



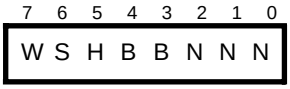
2	nEVENT2, active-low wakeup event 2
1	nEVENT1, active-low wakeup event 1
A	A-to-D, active-high
T	mouse transmit active-high
R	mouse receive active-high
Write	ignored
Read	request, status bitwise AND'ed with mask

10.3.31 IRQMSKD (0x78) — IRQ D Interrupts Mask



2	nEVENT2, active-low wakeup event 2
1	nEVENT1, active-low wakeup event 1
A	A-to-D, active-high
T	mouse transmit active-high
R	mouse receive active-high
Write	set mask for each interrupt source 0 do not form part of nIRQ 1 form part of nIRQ
Read	value set by write
Reset	set all '0' (none affect nIRQ)

10.3.32 ROMCR1:0 (0x80 and 0x84) — ROM Control



The ROM interface is very flexible, allowing the length of non sequential and burst cycles to be programmed. These two registers allow this programming to take place.

The half-speed select bit is included so the interface can be used with slow ROMs when fast DRAM is being used, and the memory system clock is running at a higher frequency. When the half-speed bit is set low, CL-PS7500FE doubles the length of all the timings and allows the ROM interface to function correctly with slower ROMs. In normal operation with sufficiently fast ROM devices, this bit should be programmed to '1'.

Each register also contains a bit (6) that, when set, allows a 16-bit wide ROM device to be used for that bank, by performing two 16-bit fetches to form the 32-bit word required by the CL-PS7500FE. Bit 7 allows writes to occur to this address space; the data is driven out, and a write enable generated, if enabled.

N — non-sequential access time (H = 1)

000	7 MEMCLK cycles
001	6 MEMCLK cycles
010	5 MEMCLK cycles
011	4 MEMCLK cycles
100	3 MEMCLK cycles
101	2 MEMCLK cycles

B — burst mode access time (H = 1)

00	Burst Off
01	4 MEMCLK cycles
10	3 MEMCLK cycles
11	2 MEMCLK cycles

H — half-speed select, that is, double the above delays when H = 0. Normally, the H bit should be programmed to '1' (normal speed)

S — 16- or 32-bit mode

W — Write Enable

Write — bit[7]

0	writing disabled
1	writing enabled

bit[6]

0	32-bit
1	16-bit

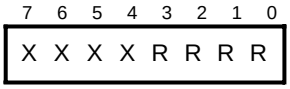
bit[5]

0	half-speed mode
1	normal speed

Read — return the above values

Reset — set to 0x40, that is, the 16-bit, slowest access time, to ensure all systems can be booted from reset.

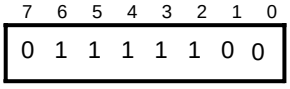
10.3.33 REFCR (0x8C) — Refresh Period



This register programs the DRAM refresh period. It is set to the fastest available rate during reset, as refresh continues during reset to ensure that the requirements of DRAM specification can be fully met.

R	refresh period
Write	bit[3:0] 0000 refresh off 0001 16 µs 0010 32 µs 0100 64 µs 1000 128 µs all others are undefined
Read	return the above values
Reset	set to '0001' (fastest available refresh rate)

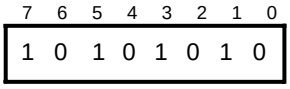
10.3.34 ID0 (0x94) — Chip ID Number (Low Byte)



The ID registers and the version register read back the CL-PS7500FE ID and version numbers. These registers are read-only and must *not* be written to, as they are used to set the CL-PS7500FE into special modes during production test.

Write	do not write to this location
Read	low byte of chip ID: 0x7C

10.3.35 ID1 (0x98) — Chip ID Number (High Byte)



Write	do not write to this location
Read	high byte of chip ID: 0xAA

10.3.36 VERSION (0x9C) — Chip Version Number

Write	ignored
Read	chip version number byte

10.3.37 MSEDAT (0xA8) — Mouse Data

The Mouse Data and Control registers are identical to the keyboard data and control registers, and are written to and read from in exactly the same way.

10.3.38 MSECR (0xAC) — Mouse Control

As KBDCR register.

10.3.39 IOTCR (0xC4) — I/O Timing Control

7	6	5	4	3	2	1	0
X	X	X	X	C	C	S	S

This register sets up the cycle types for two areas of I/O space.

C — combo area access speed

S — NPCCS 1/2 area access speed

Write — bits[7:4] reserved

bits[3:2]

- 00 Type A (slowest)
- 01 Type B
- 10 Type C
- 11 Type D (fastest)

bits[1:0]

- 00 Type A (slowest)
- 01 Type B
- 10 Type C
- 11 Type D (fastest)

Read — read back the above values

10.3.40 ECTCR (0xC8) — I/O Expansion Card Timing Control

7	6	5	4	3	2	1	0
E	E	E	E	E	E	E	E

This register sets up the access speed for eight portions of extended address space within the area of I/O space from 08FFFFFF to 0FFFFFFF. (Types A and C only.)

E — expansion card area access speed

Write — bit[7] (0F00 0000..0FFF FFFF)

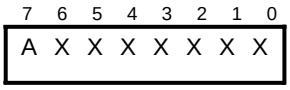
- 0 Type A
- 1 Type C

bit[0] (0800 0000..08FF FFFF)

- 0 Type A
- 1 Type C

Read — read back above values

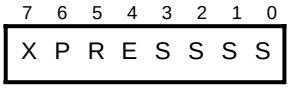
10.3.41 ASTCR (0xCC) — I/O Asynchronous Timing Control



This register is used where I/O is being used with a very fast memory system clock. Normally it is programmed to '0' to give the minimum delay for these cycles; however, in some configurations it may be necessary to program the register bit to '1' to slow down the internal synchronization between I/O clocks and memory clocks and thus ensure sufficient address hold time for the I/O address.

A asynchronous timing control
- 0 minimal delay to I/O cycles
- 1 wait states to ensure address hold time

10.3.42 DRAMCR (0xD0) — DRAM Control



This register selects between 16- and 32-bit modes of operation for each of the four available banks of DRAM. Each bank can be individually selected for 16 or 32-bit operation. This allows a mixed 16/32-bit-wide system to be built. It also controls EDO support and some timing options.

P RAS precharge time
- 0 3 memory clock cycles guaranteed RAS precharge
- 1 4 memory clock cycles guaranteed RAS precharge

R RAS-to-CAS delay on read cycles
- 0 2 memory clock cycles from falling nRAS to falling nCAS
- 1 3 memory clock cycles from falling nRAS to falling nCAS

E EDO memory
- 0 Fast Page memory
- 1 EDO memory

S 16- or 32-bit mode select, one for each bank

Write bit 3, bank 3 DRAM width
- 0 32-bit
- 1 16-bit

bit 2, bank 2 DRAM width
- 0 32-bit
- 1 16-bit

bit 1, bank 1 DRAM width
- 0 32-bit
- 1 16-bit

bit 0, bank 0 DRAM width
- 0 32-bit
- 1 16-bit

Read reads above values

Reset set bits to '0' (32-bit)

10.3.43 SELFREF (0xD4) — DRAM Self-Refresh Control

7	6	5	4	3	2	1	0
C	C	C	C	R	R	R	R

Direct software control of the external nRAS[3:0] and nCAS[3:0] lines is provided by this register. This is intended for use with self-refresh DRAMs, so that before the CL-PS7500FE is forced into STOP mode, the banks of DRAM can be set into a self-refresh state from software by forcing the nRAS and nCAS lines as specified in the DRAM data sheet.

C	force all nCAS low
R	force all nRAS low
Write	bits[7:4] 0 normal 1 force to '0' bits[3:0] 0 normal 1 force to '0'
Read	reads above values
Reset	set bits to '0' (normal)

10.3.44 ATODICR (0xE0) — A-to-D Interrupt Control

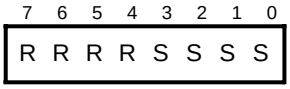
7	6	5	4	3	2	1	0
S	F	A	C	4	3	2	1

The A-to-D convertor interface is designed so that various combination of interrupts from the channels can be used to generate an interrupt request in the IRQD interrupt request register. Note that the logical OR of all four basic enables powers up the comparators. As the comparators consume static current, they must be powered down by disabling all the A-to-D channels using this register before STOP mode is entered.

1	channel 1 interrupt enable
2	channel 2 interrupt enable
3	channel 3 interrupt enable
4	channel 4 interrupt enable
C	any combination of channels generates nIRQ
A	only all channels enabled generates nIRQ
F	first pair enabled generates nIRQ
S	second pair enabled generates nIRQ
Write	bit[7:0] 0 disabled 1 enabled
Read	return above values
Reset	reset to 0x0F

NOTE: The OR of bit[3:0] powers up all the comparators. Thus they reset to the powered-up state.

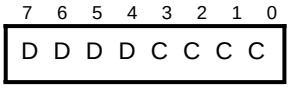
10.3.45 ATODSR (0xE4) — A-to-D Status



This register shows which of the A-to-D channels were triggered and can have their counters read to ascertain the analog value at the input to the channel. The interrupt request status bits are generated from the stop flags logically AND'ed with the interrupt enables from the interrupt control register.

R[3:0]	interrupt request state for channels 4–1	
S[3:0]	stop flag for channels 4–1	
Write	ignored	
Read	bit[7:4]	
	0	not requesting
	1	requesting
	bit [3:0]	
	0	not stopped
	1	stopped
Reset	set all '0' (not requesting or stopped)	

10.3.46 ATODCC (0xE8) — A-to-D Convertor Control



The lower 4 bits of this register directly reset each of the four counters, so that the counters can be set back to '0' before a new analog to digital conversion cycle takes place. The counter starts counting as soon as the relevant clear bit is set back to '0'. The discharge transistor controls causes the channel comparator input to be pulled firmly down to V_{SS}, thus discharging an external capacitor and ensuring zero volts across the capacitor until the discharge bit is programmed low again. With the system connected conventionally, the external capacitor begins charging as soon as the discharge bit is reset. The discharge bit should be reset at the same time as the counter clear bit for that channel to be re-enabled.

D[3:0]	discharge transistor control for channels 4–1	
C[3:0]	clear counter for channels 4–1	
Write	bit[7:4]	
	0	transistor off
	1	transistor on (discharge)
	bit[3:0]	
	0	clear counter
	1	enable counter
Read	return above values	
Reset	set all '0' (clear counters and don't discharge)	

10.3.47 ATODCNT1 (0xEC) — A-to-D Counter 1

Write	ignored
Read	returns 16-bit counter value

10.3.48 ATODCNT2 (0xF0) — A-to-D Counter 2

Write	ignored
Read	returns 16-bit counter value

10.3.49 ATODCNT3 (0xF4) — A-to-D Counter 3

Write	ignored
Read	returns 16-bit counter value

10.3.50 ATODCNT4 (0xF8) — A-to-D Counter 4

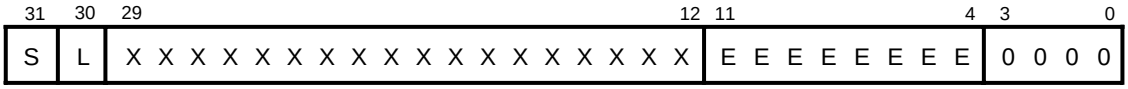
Write	ignored
Read	returns 16-bit counter value

10.3.51 SDCURA (0x180) — Sound DMA Current A

The operation of the sound DMA channel is described in Chapter 9. The sound current registers are programmed with a page address and the offset within that page to describe the precise location of the first DMA fetch. The value in the register is then increased by 16 following each DMA access.

P	page[16:0]
F	offset[11:0]
Write	bits[31:29] unused
	bits[28:12] page of next DMA fetch
	bits[11:4] offset within page of next DMA fetch
	bits[3:0] ignored
Read	bits[31:29] undefined
	bits[28:4] current DMA fetch location
	bits[3:0] always ‘0’

10.3.52 SDENDA (0x184) — Sound DMA End A



Program this register with the offset within the page of the final qword. Bit 30 should always be programmed to '0', unless the channel is being initialized for a single transfer – when it must be programmed high.

S stop bit

L last bit

E end[11:0]

Write bit[31] stop bit:

- 0 do not stop after reaching End
- 1 stop after reaching End

bit[30] last bit

- 0 not last transfer
- 1 last qword transfer

bits[11:4] last DMA location within page selected

bits[3:0] ignored

Read bits[31:30, 11:4] value written

bits[3:0] always '0'

10.3.53 SDCURB (0x188) — Sound DMA Current B

The B pair of registers for the sound DMA channel are used in exactly the same way as the A pair, to enable DMA to continue from the page addressed by one set of registers while the other set are being reprogrammed.

P page[16:0]

F offset[11:0]

Write bits[31:29] unused

bits[28:12] page of next DMA fetch

bits[11:4] offset within page of next DMA fetch

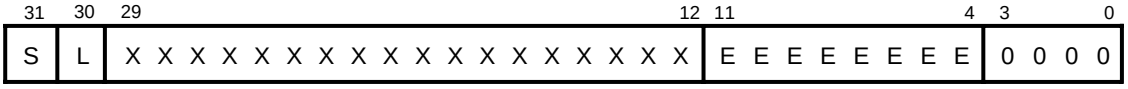
bits[3:0] ignored

Read bits[31:29] undefined

bits[28:4] current DMA fetch location

bits[3:0] always '0'

10.3.54 SDENDB (0x18C) — Sound DMA End B



This register is used in the same way as the SDENDA register.

S stop bit

L last bit

E end[11:0]

Write bit[31] stop bit

- 0 do not stop after reaching end
- 1 stop after reaching end

bit[30] last bit

- 0 not last transfer
- 1 last qword transfer

bits[11:4] last DMA location within page selected

bits[3:0] ignored

Read bits[31:30, 11:4] value written
bits[3:0] always ‘0’

10.3.55 SDCR (0x190) — Sound DMA Control

7	6	5	4	3	2	1	0
C	0	E	1	0	0	0	0

This register controls the sound DMA channel and its state machine. Only two bits can be written to:

- bit 7 clears the state machine into a state where it has overrun and is requesting an interrupt.
- bit 6 enables the sound DMA channel.

C clear

E enable

Write bit[7] clear
- 0 do not clear state machine
- 1 clear state machine. Self clearing

bit[6] not used

bit[5] enable
- 0 disabled
- 1 enabled

bits[4:0] not used

Read bit[7] always reads ‘0’

bit[6] always reads ‘0’

bit[5] enable
- 0 disabled
- 1 enabled

bits[4:0] read as ‘10000’ (binary), indicating a qword transfer

Reset enable set to ‘0’

10.3.56 SDST (0x194) — Sound DMA Status

7	6	5	4	3	2	1	0
X	X	X	X	X	O	I	W

The sound DMA status register shows the status of the state machine that controls sound DMA accesses. It cannot be written to.

O overrun

I interrupt request

W A or B buffer indication

Write ignored

Read bits[7:3] unused

bits[2:0] direct state machine state

Reset set to ‘110’ (binary)

10.3.57 CURSCUR (0x1C0) — Cursor DMA Current

The cursor current register need not normally be written to as the value in the Init register is transferred into it during the FLYBACK period. It is then updated automatically in qword increments during DMA.

C	Current fetch location
Write	bits[31:29] unused
	bits[28:4] cursor current DMA fetch location
	bits[3:0] ignored
Read	bits[31:29] undefined
	bits[28:4] cursor current DMA fetch location
	bits[3:0] always ‘0’

10.3.58 CURSINIT (0x1C4) — Cursor DMA INIT

This register is written with the initial location of the cursor data buffer.

I	initial fetch location
Write	bits[31:29] unused
	bits[28:4] cursor initial DMA fetch location
	bits[3:0] ignored
Read	bit[31:29] undefined
	bits[28:4] cursor initial DMA fetch location
	bits[3:0] always ‘0’

10.3.59 VIDCURB (0x1C8) — Duplex LCD Video DMA Current B

The B Video DMA Address registers are for use with dual-panel LCDs. The current registers do not normally need to be programmed as the value in the relevant INIT register is loaded into the current register during the FLYBACK period. This register gives the current location of the DMA data for the lower panel.

C	current fetch location B
Write	bits[31:29] unused
	bits[28:4] video current B DMA fetch location
	bits[3:0] ignored
Read	bits[31:29] undefined
	bits[28:4] video current B DMA fetch location
	bits[3:0] always ‘0’

10.3.60 VIDCURA (0x1D0) — Video DMA Current A

C	current fetch location A
Write	bits[31:29] unused
	bits[28:4] video current A DMA fetch location
	bits[3:0] ignored
Read	bits[31:29] undefined
	bits[28:4] video current A DMA fetch location
	bits[3:0] always ‘0’

10.3.61 VIDEND (0x1D4) — Video DMA End

Load the Video End register with the address of the final qword of the video frame buffer within memory

E	end location
Write	bits[31:24] unused
	bits[23:4] video end location
	bits[3:0] ignored
Read	bits[31:24] undefined
	bits[23:4] video end location
	bits[3:0] always ‘0’

10.3.62 VIDSTART (0x1D8) — Video DMA Start

Load the Video Start register with the location of the first qword at the start of the video frame buffer. All the DMA control registers can only be loaded with qword-aligned values.

S	start location
Write	bits[31:29] unused
	bits[28:4] video DMA start fetch location
	bits[3:0] ignored
Read	bit[31:29] undefined
	bits[28:4] video DMA start fetch location
	bits[3:0] always ‘0’

10.3.63 VIDINITA (0x1DC) — Video DMA INIT A

For normal CRT displays and single-panel LCD data, only the A registers are used. Load the INIT register with the address within the frame buffer of the first qword to be displayed in the first raster at the top of the screen. For dual-panel displays, load this register with the address of the first qword in the frame buffer to be displayed at the top left of the upper panel.

Set the last bit (30) only if the INIT A register is programmed to the same value as the VIDEND register. Using an INIT register allows hardware scrolling to be implemented by moving the position of the init register within the frame buffer.

I initial fetch location A

Write bits[31, 29] unused

bit[30] last bit

- 0 not last fetch location
- 1 last fetch location

bits[28:4] video initial A DMA fetch location

bits[3:0] ignored

Read bit[31] ‘0’

bit[30] last bit

- 0 not last fetch location
- 1 last fetch location

bit[29] ‘equal’ – output of comparator

bits[28:4] video initial A DMA fetch location

bits[3:0] always ‘0’

10.3.64 VIDCR (0x1E0) — Video DMA Control

7	6	5	4	3	2	1	0
D	1	E	1	0	0	0	0

This register allows overall control for video DMA. Bit 7 selects between dual- and single-panel modes for LCD driving; bit 5 enables video DMA.

NOTE: For driving normal CRT displays, set bit 7 to '0'.

D Dual-panel mode

E enable video/cursor DMA

Write bit[7]
- 0 normal
- 1 Dual-panel mode

bit[6] ignored

bit[5]
- 0 disable
- 1 enable DMA

bits[4:0] ignored

Read bits[7, 5] return above values

bit[6] always reads back '1', DRAM mode

bits[4:0] reads as '10000' (binary), indicates qword transfer

Reset set to '0' (disabled, Normal mode)

10.3.65 VIDINITB (0x1E8) — Duplex LCD Video DMA INIT B

For normal CRT displays and single-panel LCD data, only the A registers are used. Program this register with to '0's. For dual-panel displays, load this register with the address of the first qword in the frame buffer to be displayed at the top left of the lower panel. Set the last bit (30) only if the INIT B register is programmed to the same value as the VIDEND register.

I initial fetch location B

Write bits[31,29] unused

bit[30] last bit

- 0 not last fetch location
- 1 last fetch location

bits[28:4] video initial B DMA fetch location

bits[3:0] ignored

Read bit[31] '0'

bit[30] last bit

- 0 not last fetch location
- 1 last fetch location

bit[29] 'equal' – output of comparator

bits[28:4] video initial B DMA fetch location

bits[3:0] always '0'

10.3.66 DMAST/DMARQ/DMAMSK (0x1F0,0x1F4,0x1F8) — DMA Interrupt Control

The following three registers each contain only one bit relating to the status of the interrupt generated from the sound DMA state machine.

DMAST (0x1F0) — Sound DMA Interrupt Status

7	6	5	4	3	2	1	0
X	X	X	S	X	X	X	X

S — sound interrupt status

Write — ignored

Read — status

bits[7:5, 3:0] unused

bit[4]

- 0 inactive
- 1 active

DMARQ (0x1F4) — Sound Interrupt Request

7	6	5	4	3	2	1	0
X	X	X	S	X	X	X	X

S — sound interrupt request

Write — ignored

Read — request, status AND'ed with mask

bits[7:5, 3:0] unused

bit[4]

- 0 inactive
- 1 active

DMAMSK (0x1F8) — Sound Interrupt Mask

7	6	5	4	3	2	1	0
X	X	X	S	X	X	X	X

S — sound interrupt mask

Write — bits[7:5, 3:0] unused

bit[4]

- 0 do not affect nIRQ
- 1 affect nIRQ

Read — mask

bits[7:5, 3:0] unused

bit[4] read value written above

11. I/O SUBSYSTEMS

11.1 Introduction

CL-PS7500FE has a 16-bit-wide general I/O port, BD[15:0]. This allows slow I/O access to continue independently of DMA activity on the CL-PS7500FE data bus. There are three types of I/O access supported over the I/O bus:

- 16-MHz, PC-style I/O
- 8-MHz request/grant-based I/O
- simple 8-MHz-based fixed timing I/O

CL-PS7500FE also has a separate 8-bit-wide, general-purpose, open-drain I/O port, each bit of which can be configured as an interrupt source. There are four analog comparators, each with a 16-bit, 2-MHz timer that can be a four channel analog joystick interface. Two identical PS/2 serial mouse/keyboard ports are included. There are two general-purpose, 2-MHz, 16-bit counter timers that can be programmed to produce interrupts at timed intervals.

CL-PS7500FE includes an interrupt handler, with enable and mask bits for each interrupt source that can process potential interrupts from a number of internal and external sources.

The 16-MHz, PC-style I/O provides all the signals required to interface with a standard PC Combo chip, enabling an industry standard part to be used to complete the I/O interfaces to devices such as a floppy disc.

The facility is available to expand the width of the I/O bus externally by adding latches and buffers to the upper 16 bits of the main external data bus and control signals for these devices are provided from CL-PS7500FE.

Support is provided for XIP (execute-in-place) from a 16-bit-wide PCMCIA card attached to the I/O bus, using an external PCMCIA controller.

Because the I/O clocks can be completely asynchronous to the memory system clock (that controls the main bus arbitration state machine), there are additional synchronization penalties at the start and end of the I/O cycle. The exact additional delay depends on the actual phase of the clocks at the point in question. The timing diagrams do not attempt to show detail, however, worst-case synchronization delays are indicated.

11.2 I/O Address Space Usage

The main I/O address space is defined as being from address 0x03000000 to 0x03FFFFFF, as shown in Table 11-1.

In addition, there is an extended I/O address space for 16-MHz, PC-style I/O from address 0x08000000 up to 0x0FFFFFFF, divided into eight 16-Mbyte areas. The chip select generated throughout this area is nEASCS.

Table 11-1. I/O Address Space Usage

I/O address	Contents
0x03000000	Module space — asserts nMSCS
0x03010000	16-MHz I/O — asserts nCCS (Combo chip select)
0x03012000	16-MHz I/O — asserts nCDACK (Combo DACK)
0x0302A000	16-MHz I/O — asserts nCDACK and TC (Combo DACK and TC)
0x0302B000	16-MHz I/O — asserts nPCCS2
0x0302B800	16-MHz I/O — asserts nPCCS1
0x0302C000	Reserved
0x03030000	Module space — asserts nMSCS
0x03040000	Reserved
0x03200000	CL-PS7500FE internal I/O and memory control registers
0x03210000	Simple I/O space — asserts nSIOCS1/2
0x03400000	CL-PS7500FE internal video and sound control registers
0x03500000	Reserved

11.3 Additional I/O Chip Select Decode Logic

The SETCS input selects additional decode logic for some of the chip select outputs.

- When SETCS is high:

  nMSCS is asserted only in the following ranges of Module I/O space:

  0x03000000 -> 0x03003FFF
  0x03030000 -> 0x03033FFF

  nEASCS is asserted only in the following range of Extended I/O space:

  0x08000000 -> 0x08FFFFFF

  nSIOCS2 is asserted only in the following ranges of Simple I/O space:

  0x03240000 -> 0x03243FFF
  0x032C0000 -> 0x032C3FFF
  0x03340000 -> 0x03343FFF
  0x033C0000 -> 0x033C3FFF

- When SETCS is low:

  nMSCS is asserted over the whole of Module space

  nEASCS is asserted over the whole of Extended I/O address space

  nSIOCS2 is asserted only in the following ranges of simple I/O space:

  0x03240000 -> 0x0324FFFF
  0x032C0000 -> 0x032CFFFF
  0x03340000 -> 0x0334FFFF
  0x033C0000 -> 0x033CFFFF

11.4 Simple 8-MHz I/O

The Simple I/O type of access is 16-bit only and has a selection of 4 different cycle speeds selectable by bits 20 and 19 of the address. This type of I/O is selected for addresses in the range 0x3210000 to 0x32FFFFFF. When writing, the upper halfword of the CL-PS7500FE data bus is written out on the I/O bus. When reading, the I/O bus data is read back onto the lower halfword of the CL-PS7500FE data bus. This type of I/O cycle is not affected by the READY signal.

During these accesses, the signal nSIOCS1 is always asserted with a read or write strobe as appropriate based on the CLK8, 8-MHz clock. nSIOCS2 is asserted according to the decoding in the section above. The read and write strobes are the nIOR and nIOW output pins respectively. The four timings of the Simple 8-MHz I/O accesses are shown below:

Table 11-2. Timings of the Simple 8-MHz I/O accesses

Address [20:19]	Name	MIN CLK8 Cycles
00	Slow	7
01	Medium	6
10	Fast	5
11	Sync	5

The ‘sync’ timing is referenced to the 2-MHz CLK2 output, and there is an additional possible synchronization penalty of up to three CLK8 cycles, depending on the phase of CLK2 and CLK8 at the commencement of the I/O cycle. This is in addition to synchronization between the I/O and memory subsystem signals.

11.5 Module I/O

The Module I/O type of access is 16-bit only and its speed is controlled by a handshake mechanism with the external hardware. The signals nIORQ (output) and nIOGT (input) are used for this handshaking. When writing, the upper half-word of the CL-PS7500FE data bus is written out on the I/O bus. When reading, the I/O bus data is read back onto the lower half-word of the CL-PS7500FE data bus. The module type of I/O is initiated for addresses in the ranges 0x03000000 to 0x0300FFFF and 0x03030000 to 0x0303FFFF.

During these accesses, the signal nMSCS is asserted but read and write strobes are not used, although the IORNW signal is active. READY does not affect this type of access.

The nBLI is driven by the external hardware to indicate when the read or write data should be latched from the BD I/O bus.

The I/O cycle terminates when both nIORQ and nIOGT are low at the rising edge of REF8M.

11.6 PC Bus-Style I/O

This type of I/O is designed to function in conjunction with a standard PC Combo chip, and cycles are generated from a 16-MHz clock.

The PC bus-style I/O type of access routes the lower halfword of the CL-PS7500FE bus through the device providing a direct 16-bit interface. Additionally, signals are generated to support the addition of

external latches/drivers to extend the I/O data by 16 bits. The upper halfword of the CL-PS7500FE data bus is routed through these external devices if present. This type of I/O access is used for the address space from 03010000 to 0302CFFF (five sections), and in the larger extended address space from 0x08000000 to 0x0FFFFFFF (eight sections). There are four fixed cycle types based on the 16-MHz clock, although the larger extended address area only supports two of these cycle types. Any access may be held up by external circuitry removing the READY signal before the end of the cycle.

The signals that control the external buffers and latches required to implement 32-bit-wide I/O are:

- nWBE
- nRBE
- nBLO

For full details of the external circuitry and connections required to implement a 32-bit-wide I/O system using the CL-PS7500FE, refer to Appendix D.

Two additional inputs are provided to allow external circuitry to route a full 32-bit data word through the 16-bit I/O bus using multiplexing:

- nXIPLATCH
- nXIPMUX16

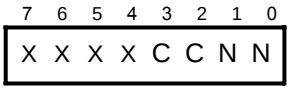
This would allow, for example, the execution of ARM code from a 16-bit-wide PCMCIA card with a suitable external controller. The nXIPMUX16 signal directly controls an internal multiplexer that maps either the upper or lower 16 bits of the internal data bus through to the 16-bit-wide I/O bus, for writes to an I/O peripheral.

When nXIPMUX16 is low, the upper 16 bits of the data bus are passed to BD[15:0], and when nXIPMUX16 is high, the lower 16 bits of the data bus are passed to BD[15:0].

For reads from an I/O peripheral, the falling edge of the nXIPLATCH signal causes the first 16 bits provided on the BD[15:0] bus to be latched as the upper halfword for the main internal data bus, then the lower 16 bits can be output from the peripheral and the I/O cycle can be allowed to complete normally. If nXIPLATCH has been driven low, the upper halfword of data is driven to the ARM processor internally and not from the external transceivers if present.

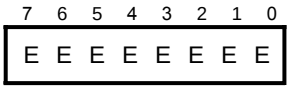
Depending on the cycle timing, it is usually necessary for the external controller to use the READY signal to stretch the I/O access to give sufficient time for both half words to be read or written as appropriate. If an I/O access is to be stretched, the READY signal must be set low before the end of the cycle as shown in the timing diagrams. This causes the nIOR or nIOW strobe and the chip select to be held low until READY is set back to high, when the I/O cycle complete as normal. READY is sampled on the rising edge of the first 16-MHz cycle before the I/O cycle is due to complete.

The four address areas for 16-MHz I/O within the main I/O address space can support any of the four available cycle types A-to-D. The IOTCR register can be programmed (at address 0x032000C4) to determine the type of cycle for each group of addresses. The addresses are grouped so that the nCCS and pseudo DMA address spaces form one group, and the nPCCS1 and nPCCS2 address area forms another group.



C	nCCS + pseudo DMA access speed	
N	nPCCS1 and nPCCS2 area access speed	
Write	bits[7:6] unused	
	bits[5:4] unused	
	bits[3:2]	
	00	Type A (slowest)
	01	Type B
	10	Type C
	11	Type D (fastest)
	bits[1:0]	
	00	Type A (slowest)
	01	Type B
	10	Type C
	11	Type D (fastest)
Read	read back above values	
Reset	set to ‘0’ (slowest)	

The extended address space from address 0x08000000 up for 16-MHz I/O accesses supports only cycle types A and C, and the ECTCR register should be programmed to specify the cycle type required for each of the eight 16-Mbyte areas within the extended address space. The details of this register, at address 0x032000C8, are shown below:



E = expansion card area access speed		
Write	bit[7] (0F00 0000..0FFF FFFF)	
	0	Type A
	1	Type C
	bit[0] (0800 0000..08FF FFFF)	
	0	Type A
	1	Type C
Read	read back above values	
Reset	set to ‘0’ (slowest)	

This type of I/O asserts a single chip select according to the area, except in Combo DACK + TC space, where both the nCDACK and TC outputs are asserted to signal to the PC Combo chip that the end of a pseudo DMA sequence has been reached. In the extended address space the nEASCS chip select is asserted.

11.7 DMA During I/O Cycles

DMA to the Video and Sound Macrocell can continue during I/O cycles. Write data from the ARM processor is latched early, so that the data bus can be used freely for DMA data. Thus, only the start of an I/O cycle needs to be added to any DMA latency calculations.

11.8 Clock Synchronization Conditions

7	6	5	4	3	2	1	0
A	X	X	X	X	X	X	X

In a system using a MEMCLK frequency greater than I_OCLK, it may be necessary to insert an extra I/O clock cycle to allow sufficient address hold time before the chip select is taken away. The problem arises because the chip select is generated from the fixed frequency I/O world clock, whereas the address changes according to the memory system clock. When a faster MEMCLK is used, it is possible for the synchronization to the memory clock to occur rapidly at the end of the cycle, and thus for the I/O address to change before the chip select has been removed. This may be a problem for some peripherals.

To avoid this, there is a register bit in the ASTCR register (address 0x032000CC) that is normally set to ‘0’, but can be programmed to ‘1’ to add an extra I/O clock period, ensuring that the address does not change before the chip select is deasserted.

A	asynchronous timing control	
	0	minimal delay
	1	wait states to ensure address hold time

See Appendix C for more information on the use of this register with high MEMCLK frequencies.

11.9 Keyboard/mouse Interface

7	6	5	4	3	2	1	0
T	T	R	R	E	P	D	C

The keyboard and mouse interfaces are identical, differing only in the names of the external pins. The interfaces are designed to communicate with a standard PS/2 keyboard or mouse, through a 2 pin serial link.

The keyboard interface uses the pins KBDATA, KBCLK, and the mouse interface uses the pins MSDATA and MSCLK, all are open-drain.

There is an 8-bit control register for each interface, providing direct access to the CLK and DATA outputs, an enable bit to enable the interface, and five status flags. The KBDCR is programmed at address 0x03200008, and the MSECR (mouse control register) at address 0x032000AC.

T	transmit status
R	receive status
E	enable
P	received parity
D	data pin status
C	clock pin status

Write bits[7:4, 2] ignored
bit[3] enable
0 state machine cleared
1 state machine enabled
bit[1] force KBDATA/MSDATA pin low
0 do not force low
1 force low
bit[0] force KBCLK/MSCLK pin low
0 do not force low
1 force low

Read bit[7] TXE, shift register empty
0 not ready
1 enabled and ready to transmit
bit[6] TXB, transmitter busy
0 not busy
1 currently sending data
bit[5] RXF, receive shift register full
0 not full
1 ready to read
bit[4] RXB, receiver busy
0 not busy
1 currently receiving data
bit[3] ENA, state machine enable
0 disabled
1 enabled
bit[2] RXP, receive parity bit, odd parity bit for last received data
bit[1] KBDATA/MSDATA pin value after synchronization
bit[0] KBCLK/MSCLK pin value after synchronization

There is also a data register (KBDAT) used for write bytes to be transmitted across the serial link and to read bytes received. The KBDAT register is programmed at address 0x03200004, and the MSEDAT (Mouse Data register) is programmed at address 0x032000A8.

The interfaces generate two interrupts each, one to indicate that the transmit buffer is empty and thus that another byte can be transmitted, and one to indicate that a byte has been received by the interface. These interrupt bits are processed by the IRQB register set (for keyboard) and the IRQD register set (for mouse).

The keyboard interface is held in reset until the enable bit in the control register is set. The interface can be controlled on the basis of the interrupts generated, or by polling the status flags in the control register. The Tx interrupt is generated when the transmit buffer has been emptied and the interface is ready to be programmed with another character for transmission. The Rx interrupt is set when a complete character has been received in the receive buffer, and the byte is ready to be read from the register. The received data parity bit, RXP, is available in the control register at bit 2. Odd parity is used. The keyboard and mouse interface state machines are clocked by the 8-MHz I/O system clock.

The KCLK/MSCLK signal is always driven by the keyboard/mouse, unless CL-PS7500FE wishes to prevent the peripheral from transmitting (because it is about to transmit some data itself). When data is

received from the peripheral, the KDATA/MSDATA line is pulled low as a start bit. Each data bit is set up to the falling edge of the clock. Eight data bits are transmitted from the keyboard/mouse, followed by a parity bit (odd parity) and a high stop bit.

When CL-PS7500FE transmits a byte to the peripheral, the KCLK/MSCLK line is pulled low, then allowed to float and the KDATA/MSDATA line is pulled low, as a request to send. The keyboard/mouse then drives the clock, causing CL-PS7500FE to put eight bits of serial data out onto the KDATA/MSDATA line. A parity bit is driven out, followed by a stop bit, and the stop bit may be acknowledged by the peripheral (the CL-PS7500FE does not check on the acknowledge).

11.10 Analog-to-Digital Converter Interface

CL-PS7500FE contains four analog comparators with 16-bit timers, designed primarily for the implementation of an analog joystick interface. Each converter is of the slope integration type, using an external RC network attached to the appropriate ATOD[3:0] pin to generate a variable ramp delay.

The time taken for the voltage at the input to the comparator to reach the comparator's threshold is measured by a 16-bit counter that stopped when the threshold of the comparator is reached. At this point an internal 'stop' flag for that channel is set. The value is held in the counter until it has been read and the channel is then reset.

Discharge transistors on the analog inputs are used to discharge the external capacitor and to initiate a new integration cycle.

11.10.1 Counters

Each of the four counters can be reset by programming one of four bits in the ATODCR register. The four counters cannot be written to but can be read at addresses as follows:

CNT1 (0x032000EC)	counter 1
CNT2 (0x032000F0)	counter 2
CNT3 (0x032000F4)	counter 3
CNT4 (0x032000F8)	counter 4

The four counters are implemented as simple asynchronous ripple counters, and it is therefore important that they should not be read until the 'stop' flag for that particular channel is set, as indicated in the status register, to indicate that the counter stopped and the read back value is stable.

11.10.2 Interrupt Control

There is a single bit in the main CL-PS7500FE interrupt handling registers (bit 2 of the IRQD set) that can accept an interrupt from the A-to-D converters. Thus, some interrupt pre-processing is done to determine how this main interrupt is to be generated. An interrupt control register is provided so that various combinations of channels can generate the final interrupt.

There are four possible interrupt sources, one for each channel, and each channel attempts to generate an interrupt when the comparator threshold is reached and the 'stop' flag is set internally.

Each of these interrupt sources can be individually enabled using the lower four bits of the Interrupt Control register, and the upper four bits determine the combination of bits that create the main interrupt passed to the IRQD registers.

Address 0x032000E0 — Interrupt Control

7	6	5	4	3	2	1	0
S	F	A	C	4	3	2	1

1	channel 1 interrupt enable
2	channel 2 interrupt enable
3	channel 3 interrupt enable
4	channel 4 interrupt enable
C	any combination of channels generates nIRQ
A	only all channels enabled generates nIRQ
F	first pair enabled generates nIRQ
S	second pair enabled generates nIRQ
Write	bit[7:0]
	0 disabled
	1 enabled
Read	return above values
Reset	reset to 0x0F

NOTE: The OR of bit[3:0] powers up all the comparators. Thus they reset to the powered-up state.

11.10.3 Status of Interface

The status of the 'stop' flag for each channel can be read directly from bits 0–3 of the status register, as can the interrupt status that is the logical AND of the 'stop' flag values and the corresponding channel enables from the interrupt control register.

This register should be read by the system software in a polled system to check whether a channel has reached its final count value and is thus waiting to be read before another conversion cycle can be initiated.

Address 0x032000E4 — Status

7	6	5	4	3	2	1	0
R	R	R	R	S	S	S	S

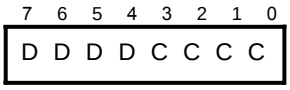
R[3:0]	interrupt request state for channels 4 to 1
S[3:0]	stop flag for channels 4 to 1
Write	ignored
Read	bit[7:4]
	0 not requesting
	1 requesting
Reset	set all zero (not requesting)

11.10.4 Control

The converter control register allows the discharge transistors and counters for each channel to be enabled and disabled, to give full control over the resetting of the counter and the timing of the start of a conversion cycle. Before a conversion can be started, the discharge bit and the counter clear bit for the channel in question should be forced one and zero respectively, and then the bits should be returned to zero and one respectively to actually initiate a conversion cycle. This causes the analog voltage across the external capacitor to begin to ramp up, and simultaneously the 2-MHz clock to the counters is enabled, thus starting the count.

Synchronization between the memory system clock used to program the registers, and the 2-MHz I/O world clock results in a small extra delay before the counter is really enabled, but this is negligible against the 0.5-µs period of the 2-MHz clock.

Address 0x032000E8 — Converter Control



D[3:0]	discharge transistor control for channels 4–1	
C[3:0]	clear counter for channels 4–1	
Write	bit[7:4]	
	0	transistor off
	1	transistor on (discharge)
	bit[3:0]	
	0	clear counter
	1	enable counter
Read	return above values	
Reset	set all zero (clear counters and don't discharge)	

11.10.5 Comparators

The comparators are accurate to 2.5-mV resolution and require a stable reference voltage of less than 2.5 V to function correctly. The reference voltage is applied at the ATODREF pin. The same reference voltage is routed to all four comparators.

In order for the comparators to function correctly, it is essential that the reference current to the Video DACs on the VIREF pin is present, as this current is used to generate the operating current used by the gain stages in the comparator. The comparator reference currents are disabled to save power if all the interrupt enables (bits 3:0 of the interrupt control register) are set to '0'. So, at least one channel must be enabled for any of the channels to function correctly.

11.10.6 Converter Operation

The values of the capacitance and variable resistance used in the external RC circuit determine the range of time delays seen from the moment the capacitor begins to charge to the moment that the comparator threshold is crossed.

The 16-bit counters are clocked by the 2-MHz internal clock (derived from the 32-MHz I_OCLK), and thus the counter counts for 65536 values over 32.7 ms before returning to zero. To provide a meaningful reading from the converter, it is important that the capacitor and variable resistor values are such that this time

is not exceeded under the worst case conditions. The A-to-D converter is effectively providing a digital count directly related to the value of the resistance in the RC circuit.

11.11 Timers

The CL-PS7500FE includes two general-purpose timers used as interrupt sources. Each timer is implemented as a 16-bit down counter, and has an input latch and an output latch associated with it. The counter decrements continuously, clocked at 2 MHz. When it reaches zero, it is reloaded from the input latch and the down count restarts.

There are four 8-bit-wide registers associated with the two timers. Each timer has:

- two eight bit registers corresponding to the 16-bits of the timer
- two further write-only registers that cause the GO and LATCH commands to be issued to the appropriate timer when written to

Figure 11-1 shows the timer configuration.

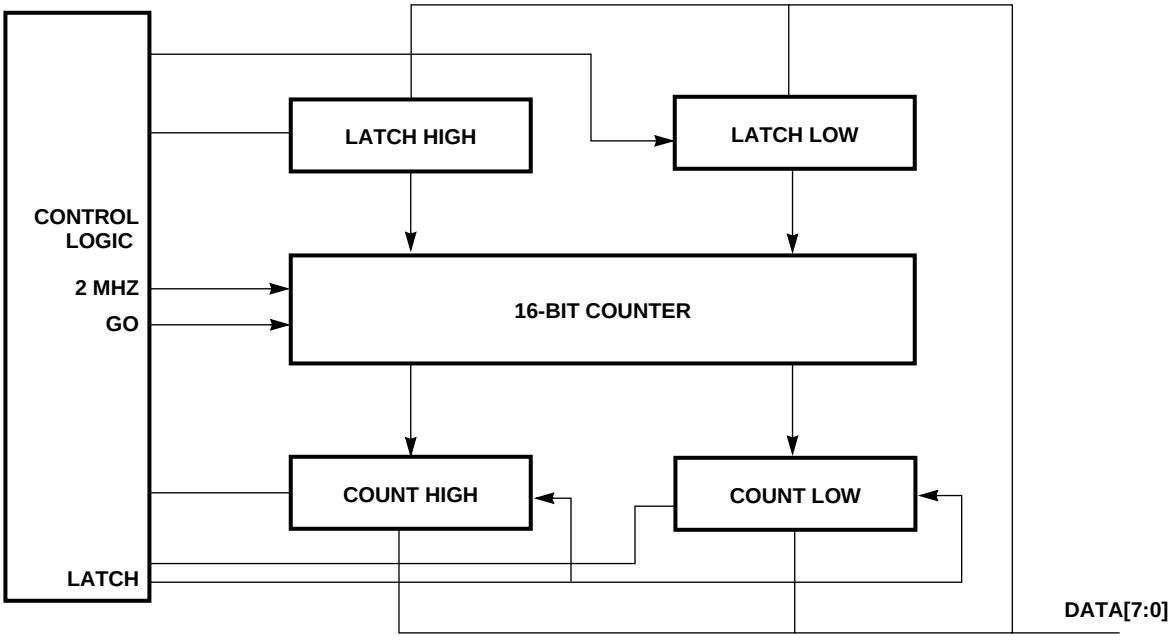


Figure 11-1. Timer Configuration

11.11.1 Programming the Timers

The locations of the registers can be found in Chapter 10.

A write to the following registers updates the values as described below:

T0low	updates the value in the lower half of the timer 0 input latch
T0high	updates the value in the upper half of the timer 0 input latch with the written value.
T0GO	loads the counters immediately with the value programmed into the input latch. If the counter is loaded with ‘0’ it is continuously reload.
T0LAT	places the current count value in the output latch.

A read of the following registers updates the values as described below:

T0high	returns the upper 8 bits of the count value
T0low	returns the lower 8 bits of the count value.

11.11.2 Timer interrupts

Each timer generates an interrupt when it reaches zero and is reloaded. These interrupts are handled by the IRQA set of interrupt processing registers (bits 5 and 6).

The timers can be used to generate timed interrupts at regular intervals, *T*, where:

$$T = (T0low + (256 \times T0high)) \times 0.5\ \mu s \qquad \textbf{Equation 11-1}$$

11.12 General-Purpose, 8-Bit-Wide I/O Port

A general-purpose 8-bit-wide I/O port is included in the CL-PS7500FE. The eight open drain output pins IOP[7:0] can be driven low or monitored as inputs by using the IOLINES register at address 0x0320000C.

When read, this register returns the current value seen at the IOP[7:0] pins. When written to, each bit controls the status of the corresponding IOP pin. When a one is written to a bit, that pin's output enable is switched off and it can be driven as an input. When a ‘0’ is written to a bit, the corresponding output pin is forced low.

There is a complete set of three interrupt control and status registers (IRQD) for the IOP pins that allow any bit to generate a unique interrupt. The interrupt is generated when the corresponding IOP bit is low.

11.13 ID and OD Open-Drain I/O Pins

There are three further open-drain I/O pins:

ID	Intended for use with an ID chip that outputs a unique system ID when the ID pin is forced low. During power-on reset, the ID output is forced low, and it then becomes tristate on leaving reset.
OD[1:0]	Could implement a simple serial link.

These are written to through the IOCR and are not capable of generating interrupts. Each pin is forced low by programming '0' to the appropriate bit in the IOCR. Programming '1' to any bit causes the corresponding pin to tristate, and the value of the input level applied to the pin can then be read back from the same bit of the IOCR.

NOTE: These three pins do not have pull-up resistors on-chip; be advised that to fit pull-up resistors externally if the pins are not connected to another device.

11.14 Version and ID Registers

The ID register is composed of two read-only, 8-bit hardwired registers. The lower byte is accessed at location 0x03200094, and the upper byte at location 0x03200098. Together they should return the value 0xAA7C.

The Version register is accessed at location 0x0320009C, and this is read back the version number of the device.

NOTE: Under no condition should either of these registers be written; this can cause the chip to enter a test mode.

11.15 Interrupt Control

The CL-PS7500FE interrupt handler takes interrupts from a variety of sources and generates the IRQ or FIQ interrupt signals required by the ARM processor, depending on the settings of the control and enable bits in the five sets of interrupt registers. The five sets are:

- FIQ
- IRQA
- IRQB
- IRQC
- IRQD

Each of these has a status, mask and request register associated with it, for a total of 15 registers.

Table 11-3 shows the interrupt sources featuring in each set of registers. The polarity entry refers to the level required at the external pin to set the interrupt. 'Internal' means that the interrupt is generated as a result of an internal state change, as opposed to change on an external pin.

When an interrupt signal is received from one of the interrupt sources, it causes the corresponding bit in the status register to go high. This bit is then logically AND'ed with the appropriate bit in the mask register, to create a value in the appropriate bit of the request register. If any of the bits in any of the IRQ request registers are high, then the CL-PS7500FE generates an internal IRQ interrupt to the ARM processor macrocell, causing the IRQ exception to be taken. If any of the bits in the FIQ request register are high, the CL-PS7500FE generates an internal FIQ interrupt to the ARM processor, causing the FIQ exception to be taken.

The system software can then read the request registers to determine the sources that requested an interrupt. Reading the status registers indicate the sources that requested interrupts, even if they were masked.

The IRQA request register is slightly different in that some of the interrupt flags are edge triggered and thus need to be cleared after they are read. All other request registers are read-only, but the IRQRQA register can be written to clear triggered interrupts. A write of '1' to a bit clears that interrupt. A write of '0' causes no action to be taken.

Table 11-3. Interrupts

Register	Bit	Polarity/Type	Name/Function
FIQ	7		Always active for software generated FIQ
	6	Low	nINT8 interrupt pin
	5		
	4	Low	nINT6 interrupt pin
	3		
	2		
	1	High	INT5 interrupt pin
	0	High	INT9 interrupt pin
IRQA	7		Always active for software generated IRQ
	6	Internal	2-MHz timer 1
	5	Internal	2-MHz timer 0
	4	Falling edge	nPOR power on reset
	3	Internal	Flyback from video subsystem
	2	Falling edge	nINT1 interrupt pin
	1		
	0	Rising edge	INT2 interrupt pin
IRQB	7	Internal	Keyboard Rx buffer full
	6	Internal	Keyboard Tx buffer empty
	5	Low	nINT3 interrupt pin
	4	Low	nINT4 interrupt pin
	3	High	INT5 interrupt pin
	2	Low	nINT6 interrupt pin
	1	High	INT7 interrupt pin
	0	Low	nINT8 interrupt pin
IRQC	7	Low	IOP[7] interrupt pin
	6	Low	IOP[6] interrupt pin
	5	Low	IOP[5] interrupt pin
	4	Low	IOP[4] interrupt pin

Table 11-3. Interrupts *(cont.)*

Register	Bit	Polarity/Type	Name/Function
	3	Low	IOP[3] interrupt pin
	2	Low	IOP[2] interrupt pin
	1	Low	IOP[1] interrupt pin
	0	Low	IOP[0] interrupt pin
IRQD	7		
	6		
	5		
	4	Low	nEVENT2 wake-up event
	3	Low	nEVENT1 wake-up event
	2	Internal	A-to-D convertor interrupt
	1	Internal	Mouse Tx buffer empty
	0	Internal	Mouse Rx buffer full

12. VIDEO FEATURES

This section discusses the video and sound macrocell video functionality including clocks, palette, cursor, synchronization, display support, and DACs.

12.1 Pixel Clock

The video and sound macrocell is capable of generating a display at any pixel rate up to 120 MHz. The pixel clock may be selected from one of three sources, and the frequency of this clock may be further divided down by a factor of between 1 and 8. These attributes are programmed by the lower 5 bits of the control register, CONREG.

If a maximum of three master frequencies are sufficient, then the clock inputs can be used directly. However, it is often a requirement to have many different master clock frequencies. In order to obviate the need for many crystals on the PCB, the video and sound macrocell is designed to drive a voltage controlled oscillator (VCO) to provide the master frequency. The VCO and filter are external to CL-PS7500FE, but everything else is built into the chip. Operation is described below:

An internal reference frequency of 32 MHz is supplied via the I_OCLK input of CL-PS7500FE. The signal from the VCO is input into CL-PS7500FE on the pin VCLKI. VCLKO is simply the inverse of VCLKI, and this may be used to bias the input signal about the threshold if the VCO output is not a full amplitude signal. To achieve operation at 120 MHz, the mark-space ratio of the VCO output should be as close as possible to 50-50.

The reference clock is divided by a programmable number set by the r-modulus in the fsynreg. The VCO clock is divided by a programmable number set by the v-modulus in the fsynreg. Each of the moduli can be a 6-bit number. The output of each of these dividers is routed into a phase comparator, and the result is output from CL-PS7500FE as PCOMP. This pin should then be filtered and used to control the VCO output frequency. In this way, the VCO can be set to have a frequency of v/r $\times$ Fref.

The phase comparator is of the phase-frequency type. The output PCOMP is normally tristate, but when the VCO frequency needs to decrease, the output is low and when the VCO frequency needs to increase, the output is high. When the two frequencies are in lock, PCOMP is be tristate, but is driven to the midpoint for a very short time (a few ns) every r/Fref + period. The output impedance of this pin when it is driven, is approximately 50 Ω (see Figure 12-1).

Filter and VCO selection is left to the user. It is important to avoid any low-frequency modulation of the VCO frequency. A suitable VCO is a 74AC04-inverter element with feedback, with the supply voltage controlled by the PCOMP output (see Appendix E).

With this approach, an enormous number of frequencies are possible. The 32-MHz reference frequency generated within CL-PS7500FE can be used to yield the common VCO frequencies shown in Table 12-1. There are many possible values of r and v. Select a set of values that favors filter response.

NOTE: Large moduli yield a lower comparison frequency.

Limit the VCO range and use the prescaler within the video and sound macrocell to obtain a lower pixel rate than the VCO frequency. The VCO range may require constrainment so much that it cannot provide the highest frequencies for operation of the video and sound macrocell. For this scenario, a single high-frequency clock can be fed into CL-PS7500FE on the HCLK pin and selected for the pixel clock.

Table 12-1. Synthesized VCO Frequency Settings

r-Modulus	v-Modulus	VCO Frequency (MHz)
8	2	8.0
16	6	12.0
4	2	16.0
8	6	24.0
2	2	32.0
8	9	36.0
16	35	70.0
4	15	120.0

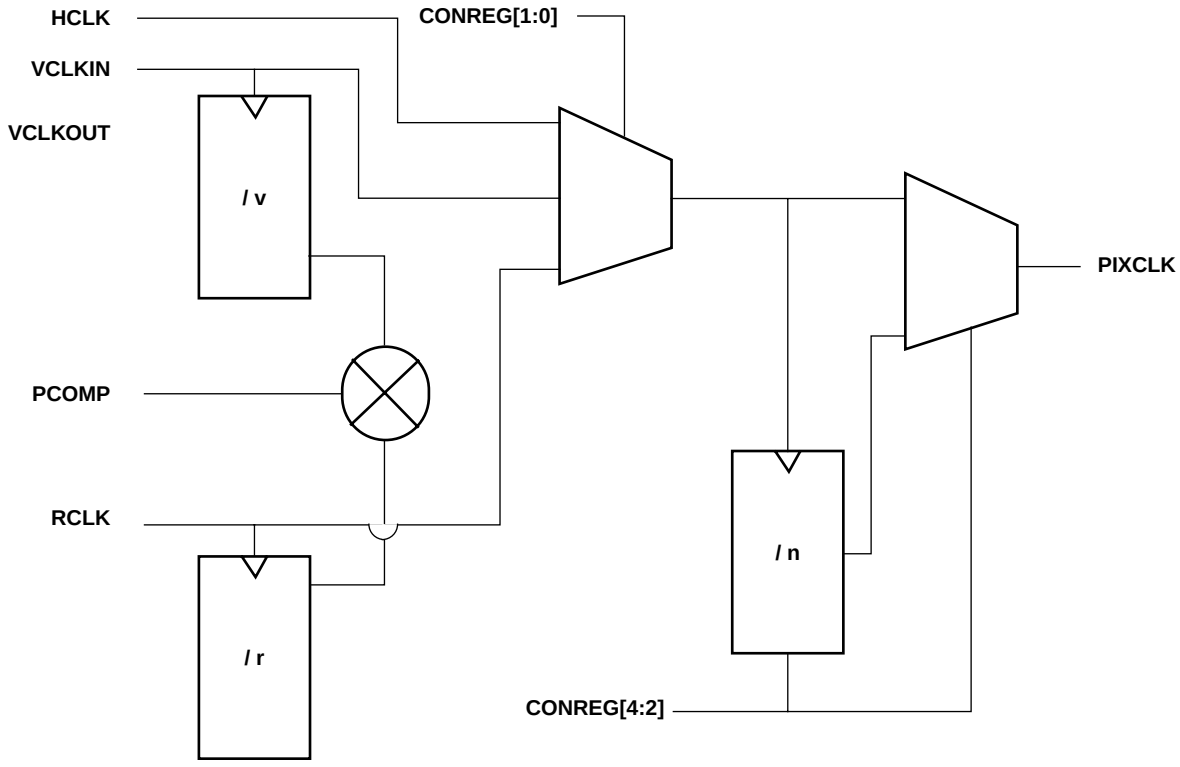


Figure 12-1. CL-PS7500FE Internal Subsystems for Pixel Clock Generation

12.2 Palette

The CL-PS7500FE has a 28-bit-wide, 256-entry palette constructed out of three 8-bit-wide LUTs, each has 256 entries, named Red, Green, and Blue and one 4-bit-wide LUT with 16 entries, named Ext. The Red, Green, and Blue LUTs each drive their respective DACs. The Ext LUT is normally configured to drive the ED[3:0] output port, except when HiRes mode or LCD mode is selected. These bits may be used outside the chip for a variety of purposes such as supremacy, fading, HiRes, and LCD driving. The ED[7:4] output port is normally driven from the External register – EREG[7:4] – that can be written at any time. These bits can be used as a DC control port.

The mapping of the logical colors through the LUTs is dependent on the mode in use, as follows:

- In 1-, 2-, 4-bits/pixel modes, the logical data is fed simultaneously to all four LUTs. This allows a completely flexible palette with any logical color mapped to any physical color and any ED[3:0] value. The palette gives 16 colors from a selection of 2^{24}.
- In 8-bits/pixel modes, the logical data is fed simultaneously to all 4 LUTs. This allows a completely flexible palette with any logical color being mapped to any physical color. Logical colors 0–15 access the Next LUT; logical colors 16–255 access location 0 of the Ext LUT. The Ext LUT again drives ED[3:0]. The palette gives 256 colors from a selection of 2^{24}.
- In 16-bpp mode, a patented technique has been developed. This approach is highly flexible and allows many different addressing modes (for example, 5:5:5, 5:6:5, and so on). In this mode, 2^{16} colors are available from a selection of 2^{24}.
- In 32-bpp mode, 24 bits from the logical field drive the 256 entries in each of the color LUTs (8 bits to each LUT); 4 bits drive the Ext LUT; the upper 4 bits are discarded.The palette gives the full range of 2^{24} colors.

NOTE: Where a logical field does not drive all the palette entries (for example, in 4-bpp mode), only the lower portion of the palette is used. Unused sections need not be programmed.

When HiRes or LCD mode is selected, the palette must be set up in a predetermined configuration. This is explained in Section 12.4 and Section 12.5.

12.2.1 Palette Updating

The FLYBK signal is an output in the CL-PS7500FE from the video and sound macrocell. FLYBK goes high at the start of the first raster that is not displayed, and goes low at the start of the first raster that is displayed. The rising edge of this signal can cause an interrupt through the CL-PS7500FE IRQA interrupt registers and the palette is updated at this time for flicker-free updating.

12.3 Cursor

The CL-PS7500FE has a hardware cursor 32-pixels wide and N pixels high. The 2-bpp allow four colors, including ‘transparent’, plus three other colors from a selection of 2^{24}. The cursor is displayable in the horizontal border, but not in the vertical border.

The cursor is a 28-bit-wide, three-entry palette, allowing each cursor logical color to be any physical color. There is also a 28-bit-wide border color register.

At the start of every frame, 16 bytes of cursor data transfer to the video subsystem during the horizontal retrace period. This is enough data for two rasters worth of cursor. After they are displayed, a request is made for another 16 bytes. In Normal mode, requests are made on every other raster where there is a cursor and enough data transfers for two rasters each. In HiRes mode, a request is made every raster.

NOTE: The cursor data always transfers in 4-word bursts.

12.3.1 Cursor in HiRes Mode

To allow µ-pixel resolution of the cursor in HiRes mode when operating at 4-µ-pixels-per-normal-pixel, define 2-bits-per-µ-pixel or 8-bits-per-normal-pixel. The 16 bytes of cursor data available for each raster can then generate 64 µ-pixels of cursor. In HiRes mode the cursor palette is not used (though the border may be programmed). Refer to Section 12.4 for more details on HiRes support.

The cursor is always positioned to align with a normal pixel. To position the cursor to a µ-pixel horizontally, four different copies of the cursor are required: each copy defines the cursor offset by a single µ-pixel. It is possible to define transparency to a resolution of a µ-pixel; selecting the correct cursor image, can achieve the required position.

12.3.2 Cursor in LCD Mode

The video subsystem is capable of displaying the hardware cursor in LCD mode. However, because of the split-screen nature of duplex LCDs, the cursor requires special attention. If the cursor is entirely in the upper or lower half-screen, program the cursor normally, but program VCSR[14:13] as:

- 0x10 = upper half-screen
- 0x01 = lower half-screen)

If the cursor straddles the split screen, the cursor image in memory must start at the top of the lower half-screen and end with the bottom of the upper half-screen. In this way, two contiguous images of the cursor image are required and the start pointer moves accordingly. Four images of the cursor are required to ensure that the resolution of one raster is maintained across the boundary. As the cursor moves from one panel to the other, the pointer to the cursor image in memory must move. For more details, refer to Appendix B.

When the cursor straddles the split screen, the meaning of the VCSR and VCER change. The VCER now defines the start of cursor in the upper half-screen; the VCSR defines the end of the cursor in the lower half-screen. The cursor actually displays in the lower half-screen from the start of display until VCSR, then again in the upper half-screen from VCER until the end of display. This mode is selected by programming VCSR[14:13] = 0x11. For more discussion on how to use CL-PS7500FE with dual-panel LCD screens, refer to Appendix B.

12.4 HiRes Support

The CL-PS7500FE can support color screens with resolutions above 1024 × 768 pixels. For higher resolutions, externally serializing the data is required to produce monochrome (or gray-level) pictures. In this scheme, one 16-ns pixel could theoretically be serialized to make eight 2-ns pixels (that is, approximately 500 MHz). This is dependent on the availability of external hardware capable of generating a serial bit-stream at this frequency.

12.4.1 CL-PS7500FE Support for HiRes Mode

When the hrm bit (EREG[14]) is set and EREG[1:0] is set to the value ‘0x10’, the CL-PS7500FE outputs 8 bits of data for every normal pixel on the ED[7:0] port. These bits can then be serialized to form a high-frequency, monochrome pixel stream; alternatively they can be serialized to 2 or 4 bits able to drive a high-speed, monochrome DAC for gray-level displays. With the pixel clock running at a fundamental frequency of approximately 100 MHz, the external serial clock can run at up to several hundred MHz. For the external circuit to be able to synchronize to the CL-PS7500FE output data, the CL-PS7500FE also outputs a pixel clock synchronous to the data stream when EREG[14] is set.

In this mode, with EREG[1:0] set to '0x10', the video data is driven from the Blue LUT, then outputting data BPD[7:0]. Depending on how the external serializer circuit is arranged, the LUT must be set up to give a one-one correlation between the logical address and the physical data value. For example, if 4 bits are externally serialized into a single bitstream, then select 4-bpp mode and use ED[6, 4, 2, 0]. Program the lower 16 words of the Blue LUT to give all 16 combinations of BPD[6, 4, 2, 0]. If 8 bits are externally serialized to give a single bitstream, then select 8-bpp mode and program all 256 values of the Blue LUT as a one–one mapping.

Hardware cursor support is:

- The cursor palette is not used, although the Blue border may be programmed.
- Eight bits of cursor data (CD[7:0]) are defined for each normal pixel. The 8 bits are divided into 4 pairs, with the LSB of each pair defining whether the video data (BPD) or the MSB of the cursor pair is displayed.
- Each cursor bit-pair operates on 2 bits of the BPD, according to Table 12-2 through Table 12-5.

If the external circuit serializes ED[6, 4, 2, 0] into a single bitstream or ED[7:0] into a 2-bit data stream, the cursor can be positioned and defined to any µ-pixel: in each case the cursor can be transparent, black, or white. If all 8 bits are serialized into a single very high-frequency bitstream, the cursor can only be positioned and defined to units of 2 µ-pixels.

Table 12-2. Deriving High-Speed 2-bit Cursor Data from the Normal 8-bit Output — CD[7:6]

CD[7]	CD[6]	ED[7]	ED[6]
0	0	BPD[7]	BPD[6]
0	1	0	0
1	0	BPD[7]	BPD[6]
1	1	1	1

Table 12-3. Deriving High-Speed 2-bit Cursor Data from the Normal 8-bit Output — CD[5:4]

CD[5]	CD[4]	ED[5]	ED[4]
0	0	BPD[5]	BPD[4]
0	1	0	0
1	0	BPD[5]	BPD[4]
1	1	1	1

Table 12-4. Deriving High-Speed 2-bit Cursor Data from the Normal 8-bit Output — CD[3:2]

CD[3]	CD[2]	ED[3]	ED[2]
0	0	BPD[3]	BPD[2]
0	1	0	0
1	0	BPD[3]	BPD[2]
1	1	1	1

Table 12-5. Deriving High-Speed 2-bit Cursor Data from the Normal 8-bit Output — CD[1:0]

CD[1]	CD[0]	ED[1]	ED[0]
0	0	BPD[1]	BPD[0]
0	1	0	0
1	0	BPD[1]	BPD[0]
1	1	1	1

12.5 Liquid Crystal Displays

The CL-PS7500FE can drive single-panel LCDs at 1-, 2-, 4-, 8-, 16-, or 32-bpp and dual-panel LCDs at 1-, 2-, or 4-bpp. Grayscaling is provided at up to 16 shades. The CL-PS7500FE can also drive single-panel color LCDs with no grayscaling in the Normal (video) mode. Two bits control LCD operation:

- lcd (EREG[13]): configures the external data port ED[7:0] for LCD operation and enables the grayscaling logic (EREG[1:0] must be set to '0x01')
- dup (CONREG[13]): enables Duplex mode and should be set for dual-panel LCDs.

12.5.1 LCD Grayscaling

To obtain a grayscaled output from the CL-PS7500FE, EREG[13] must be set. This configures the External port for LCD operation. Disable the DACs to save power since the CL-PS7500FE cannot simultaneously drive both CRT and LCD displays. To obtain this data out of the ED[7:0] port, EREG[1:0] must be set to value '0x01'.

The CL-PS7500FE provides a grayscaling algorithm that modulates the data output. Grayscaling is possible at 1-, 2-, or 4-bpp. The data is output from the device as one or two 4-bit quantities, depending on whether single- or dual-panel LCDs are used, at one quarter the pixel rate. The lower 4 bits of the Green LUT control the upper panel (ED[7:4]) and the 4 bits of the Ext LUT control the lower panel (ED[3:0]). The palette can still provide a mapping of logical-to-physical color. The cursor palette is similar, though the programming of the cursor position needs special treatment (refer to Appendix B). If a single-panel LCD is used, use ED[7:4] and program the Green LUT accordingly (ED[3:0] held low in this mode). The grayscaling logic lies between the output of the video MUX and the external port and works as described below.

There are effectively 16 physical gray levels available and, in 1-, 2-, or 4-bpp mode, the palettes are programmed to give a mapping of the logical color-to-physical shade. The resultant 4-bit pixel value out of the video MUX is modulated according to its value, the raster number, and the point on the raster where it is generated. The result is a single bit that is, on average, high for a time equal to the actual 4-bit value. For a single-panel screen, four of these bits are collected together and output as a nibble at one quarter the pixel rate on ED[7:4]. ED[4] represents the fourth pixel; and ED[7] represents the first pixel.

If Duplex mode is selected, the pixel stream for the upper half-screen is obtained from the Green LUT; that for the lower half-screen is obtained from the Ext LUT. Both pixel streams are simultaneously passed through the grayscale logic and output as two nibbles on ED[7:4] (upper half-screen) and ED[3:0] (lower half-screen).

12.5.2 Dual-Panel LCDs (Duplex Mode)

Duplex mode is configured by setting CONREg[13] and EREG[13]. The screen parameters are set according to the requirements of the LCD panel.

NOTE: Since the upper and lower panels are simultaneously driven, the CL-PS7500FE only produces data for half the total number of lines on the dual panel. The vertical registers must be programmed as if there were only one panel.

The CL-PS7500FE requests data in two qword units. The first qword the memory controller delivers is for the upper half-screen; the second qword is for the lower half-screen. The ARM processor then serializes the data into two simultaneous bitstreams, as previously described. One-, 2-, or 4-bpp can be selected.

For details of the CL-PS7500FE register programming requirements for duplex DMA, see Chapter 10.

12.5.3 Single-Panel Color LCDs

If neither CONREG[13] nor EREG[13] are set, then the ED[7:0] port can be used to gain access to all of the physical bits out of the video MUX. This allows many other types of display to be driven.

12.6 External Support

The CL-PS7500FE has an 8-bit output port, ED[7:0], and a synchronous clock, ECLK, that have different functions in different modes. The port is controlled by EREG[1:0] in the control register, that select the bytes from the video MUX. Additionally, an CL-PS7500FE register bit (bit 1 of the VIDMUX register — see Section 10.3.28 on page 91) can cause the data selection for the ED port to be modified according to the state of the ECLK output. This feature is intended to increase the bandwidth for driving color LCD screens. When this control bit is set low, the behavior of the ED port is as shown below. The bit is intended to be used with EREG[13] set low. When the VIDMUX bit is high and EREG[1:0] is set low:

- if ECLK is also low – the Red LUT is output on ED[7:0]
- if ECLK is high – the Green LUT is output on ED[7:0]

EREG[1:0] = 0

The Red LUT is output on ED[7:0].

EREG[1:0] = 1

If EREG[13] = 0, the Green LUT is output on ED[7:0].

If EREG[13] = 1, the grayscaled LCD signals are output. ED[7:4] carries the data for the upper half-screen from the Green LUT, and ED[3:0] carries the data for the lower half-screen from the Ext LUT.

NOTE: Since the data output actually represents 4 pixels for each half-screen, if EREG[13] = 1, data is output at one quarter of the ARM processor pixel rate.

When EREG[1:0] = 2

if EREG[14] = 0, the Blue LUT is output on ED[7:0].

If EREG[14] = 1, the multiplexed Blue LUT and HiRes cursor data is output on ED[7:0]. See Section 12.4. Also ED[7:0] is re-timed and delayed by one extra pixel.

When EREG[1:0] = 3

If EREG[12] = 0, ED[3:0] are driven by the Ext LUT and ED[7:4] are driven by the value of EREG[7:4]. This is intended as a DC control port in this mode.

If EREG[12] = 1, ED[3:0] are delayed by one pixel, so that they are exported from the CL-PS7500FE in the same pixel as the corresponding analog data. In this configuration, ED[3:0] bits can be used for supremacy to overlay pictures on a pixel-by-pixel basis. Because several bits are output, analog fading and mixing on a pixel basis is also possible.

12.6.1 ECLK

ECLK is output along with the data ED[7:0], so that the data can be externally latched and MUX'ed. ECLK is controlled by EREG[13] and EREG[2]. If EREG[2] is '0', then ECLK is output as logic 0. This is the power-saving configuration whenever ECLK is not required. If EREG[2] is '1' and EREG[13] is '0', ECLK is the PIXCLK output synchronously with the data stream. If EREG[13] is '1', then ECLK is the LCD clock that runs at one quarter the pixel rate. The LCD clock is only enabled while horizontal display data is being output and is synchronous to the data stream.

12.6.2 Power Saving Considerations

The External port can be configured to minimize power usage. It is very important not to load the signals heavily, especially ECLK that can clock at the pixel rate. When it is not in use, ECLK should not output the raw pixel data, but should output static signals. This is done by selecting EREG[1:0] = 3 and setting all entries of the Ext LUT to be one value. Turn off ECLK by setting EREG[2] to '0'.

If an LCD is fitted, but not operated, it may be necessary to power down the input signals to it. This can be achieved by setting EREG[13] low, disabling the grayscaler, and by disabling the external port as described above.

12.6.3 Vertical and Horizontal Synchronization

Software control over the polarities of the synchronization pulses is provided. Two types of composite sync signals can be output, each of either polarity. The logical OR of the horizontal and vertical syncs can be output on the HSYNC pin; the XOR of can be output on the VSYNC pin. Equalization pulses in the composite synchronization signal are supported for Interlace mode. When LCD mode is selected, the external HSYNC and VSYNC pulses are modified to the requirements of an LCD screen.

The HSYNC and VSYNC pins are programmed with EREG[19:16].

12.6.4 GENLOCK

GENLOCK is supported by the CL-PS7500FE. The SYNC pin can reset the vertical counter to the first raster (SYNC).

12.7 Analog Outputs

The CL-PS7500FE outputs analog R, G, and B signals. It is designed to drive doubly-terminated, 75-Ω lines directly.

12.7.1 DAC Control

There are four control bits in the Ext register associated with the DACs. These are EREG[12] and EREG[10:8] (pedon[2:0]).

Power-Save Mode

When EREG[12] is high, the DACs are enabled and generate a current proportional to the digital values from the video MUX. When EREG[12] is low, the reference current into all three DACs is turned off, so the DACs generate no output current, consuming much less power. This is useful when operating in LCD mode or at any time the screen should be blanked.

12.7.2 Pedestal Current

The DACs can be programmed to generate a pedestal offset of 20 LSB-equivalent currents. These are controlled individually with pedon[2:0] (EREG[10:8]), though they are typically programmed on or off together, depending on the monitor characteristics: pedon[0] controls the red pedestal; pedon[1] the green pedestal; pedon[2] the blue pedestal. If pedon[n] is high, the pedestal current switches on as the border starts and turns off as the border ends.

12.7.3 Video DAC Currents

The DACs are each 8-bit resolution and source 256 units of current according to the digital value from the video MUX. The current step is set by a common reference current, VIREF. The recommended reference current is 0.56 mA. This allows a DAC step of 69 μA. The digital value 0 gives 0 current and the digital value '0xFF' gives an output current of 255 × 69 = 17.6 mA. If EREG[10:8] is set during display time, the digital value 0 generates 20 × 69 = 1.38 mA, and the digital value '0xFF' generates 275 × 69 = 18.98 mA. A 4.3-kΩ resistor connected between VIREF and V_{DD} provides the desired 0.56 mA at 2.6 V.

DAC Accuracy

At 120 MHz, the DACs are accurate to 8-bits absolute resolution. They are always monotonic.

12.7.4 Monochrome Output

The CL-PS7500FE does not generate a separate composite monochrome signal. If required, this can be generated by resistively mixing the R, G, and B externally.

13. SOUND FEATURES

13.1 Sound

The video and sound macrocell has a digital sound system. This is a 32-bit serial sound interface suitable for driving external CD DACs.

13.2 The Sound FIFO

At the core of the sound system is a 4-word FIFO and a byte-wide latch. When empty, the FIFO fills completely by a DMA request. Data is then clocked out of the FIFO, one byte at a time, through the latch.

13.3 The Digital Serial Sound Interface

The serial sound interface offers a high quality 32-bit stereo sound, needing only a small amount of external circuitry. The serial sound system consists of a three-pin serial interface:

- SDCLK is the Serial Data Clock output
- SDO is the Serial Data output
- WS is the Word Select output

When no sound is required (SCTL[2:1] = 0), these outputs are stable (SDCLK = 0, SDO = 0, WS = 1).

In this mode, bytes from the sound FIFO are output in most-significant-first order. This is because the serial sound output must be MSB-first to be compatible with other serial sound devices. Each byte of data is loaded into a parallel-in, serial-out register and clocked out under control of the bit clock.

13.3.1 Timing Formats

There are two timing formats available for the interface:

- Normal
- Japanese

This selection is controlled by bit 0 of the VIDMUX register in the main part of the CL-PS7500FE.

Normal Format

When configured for Normal mode (VIDMUX bit 0 = low), each 32-bit sample consists of 16 bits for the left-hand channel and 16 bits for the right-hand channel. To distinguish between them, a WS signal is produced. This signal changes when the LSB of the previous word is output. When WS is high, the right-hand channel is being output.

Japanese Format

In Japanese format, the WS signal changes when the MSB of the new word is output. In addition, the polarity of WS is reversed.

The serial sound output can be used with any DAC with a serial sound input. Many DACs require a 11.2896-MHz input clock and, to reduce the required number of on-board crystals, the video and sound macrocell can cope with this frequency on the SCLK input. When using this configuration, the following parameters must be programmed in the registers:

- Serial sound (SCTL[1]) = 1
- clksel (SCTL[0]) = 1
- Sound Frequency register = 2

The sound system is not limited to operating with this frequency. However, the Sound Frequency register must be set accordingly to produce the necessary bit rate.

14. VIDEO AND SOUND MACROCELL

The CL-PS7500FE single-chip computer contains a high-performance video and sound controller, capable of meeting the requirements of a wide range of configurations.

The video and sound macrocell handles all the video processing aspects of the CL-PS7500FE functionality, making the CL-PS7500FE suitable for incorporation into a wide range of end products ranging from portable hand-held LCD systems through to higher performance Super VGA desktop products.

The flexible bus interface provides hardware support for interfacing to DRAM memory systems in conjunction with the CL-PS7500FE memory controller. The video and sound macrocell obtains data from external DRAM under DMA control. The macrocell also incorporates a stereo digital sound system, with a serial sound output port suitable for connection to an external CD DAC.

These features include:

- VGA, Super VGA, XGA resolution
- Three 8-bit DACs giving 16M colors
- Direct driving of LCD or CRT screens
- 1-, 2-, 4-, 8-, 16-, 32-bpp modes
- Up to 120-MHz pixel rate
- Very low power consumption

14.1 Features

14.1.1 Flexible Video System

The video and sound macrocell contains 288 write-only registers that offer a high degree of flexibility to the system programmer. Two hundred fifty-six of these registers are used as the 28-bit video palette entries. These registers are programmed through an auto-incrementing address pointer. The remaining registers are specific control registers and allow the user to program display parameters.

14.1.2 Hardware Cursor

The video and sound macrocell has a hardware cursor for all display modes:

- Normal
- HiRes
- LCD

By offering cursor support on chip, the designer benefits in speed and lower software overhead. The cursor is 32-pixels wide and any number of pixels high, and can be displayed in four colors, including transparent from its own 28-bit-wide palette. A cursor of any shape and size can be defined within the 32-pixel-wide limit.

14.1.3 Palette

The video subsystem has a 28-bit-wide, 256-entry palette where each entry uses 8 bits for Red, Green and Blue and 4 bits for external data. These external bits can be used outside the device for a variety of purposes (such as supremacy, fading, HiRes, and LCD driving).

The LUTs allow for logical-to-physical translation and gamma correction. The Red, Green, and Blue LUTs each drive respective DACs and the Ext LUT is normally configured to drive the 4-bit output port.

There are three 8-bit, linear monotonic DACs (Red, Green, and Blue), giving a total of 16 M possible colors. The DACs are designed to operate up to 120 MHz and directly drive doubly-terminated, 75-Ω lines.

14.1.4 Pixel Clock

The CL-PS7500FE is capable of generating a display at any pixel rate up to 120 MHz. The pixel clock can be selected from one of three sources, then the selected frequency of this clock can be further divided by a factor between 1 and 8.

The video and sound macrocell contains an on-chip phase comparator that, when used in conjunction with an external VCO, forms a PLL. This configuration allows a single reference clock to generate all required frequencies for any display mode, obviating the need for multiple external crystals.

14.1.5 Display Modes

Irrespective of the memory configuration used, the video subsystem is capable of many different display formats. In addition to the normal linear CRT display, the video subsystem can generate a display suitable for either very high resolution displays, single or dual-panel LCDs.

For CRT displays, the video and sound macrocell is capable of operating in a variety of pixel modes: 1-, 2-, 4-, 8-, 16-, 32-bpp, and can also directly drive LCD displays in 1-, 2-, or 4-bpp through a patented, internal, 16-level gray scalar algorithm.

14.1.6 Power Management

The macrocell is designed for power-sensitive applications and incorporates design features to minimize power consumption. Power-down mode allows power savings when the device is not in use (for example, in conjunction with a battery powered LCD system). Additional power-sensitive features include the powering down of functions currently not in use by the device (such as, the video DACs and the LCD gray scalar). The palette design is segmented so that only one-eighth of the palette is enabled and clocked at any one time. Power down mode can be used in conjunction with the STOP mode of the CL-PS7500FE to ensure minimum power consumption when clocks are stopped.

14.1.7 On-chip Sound System

The CL-PS7500FE supports a 32-bit serial sound output suitable for driving external CD DACs. Enhanced 32-bit stereo sound is offered by the serial sound output, consisting of a three-pin serial interface. Each 32-bit sample consists of 16 bits for the left channel and 16 bits for the right channel.

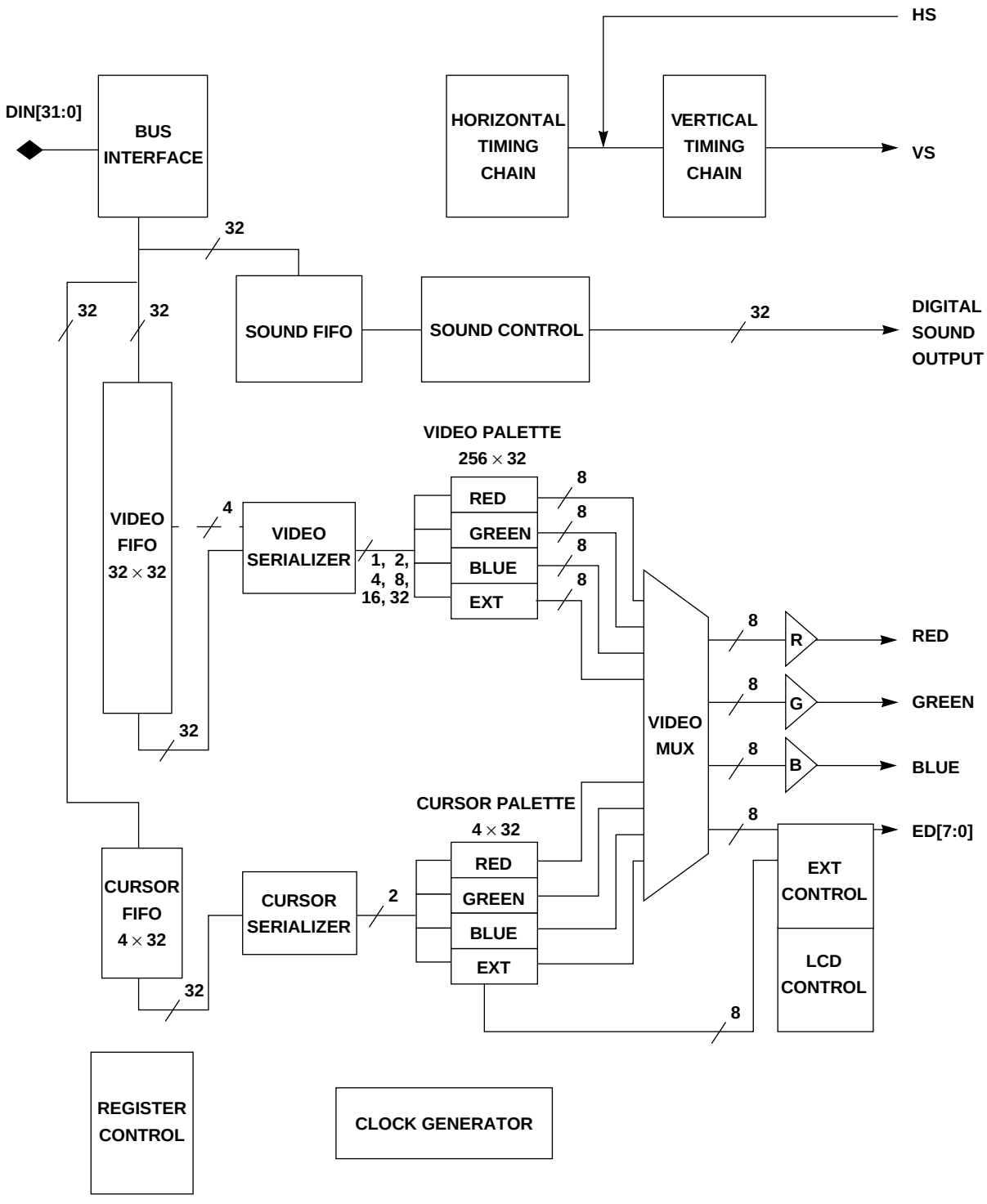


Figure 14-1. Video and Sound Macrocell Block Diagram

15. VIDEO MACROCELL INTERFACE

15.1 Bus Interface

The video macrocell does not use the ARM address bus. The address for programming video and sound registers (0x03400000 to 0x034FFFFF) is decoded elsewhere in CL-PS7500FE and the internal nPROG signal is generated as a general register write strobe. The specific register to be programmed is selected according to the state of the upper bits of the 32-bit input data bus.

All video and sound data is then obtained by DMA under the control of the nVIDRQ internal request signal. This signals to the main CL-PS7500FE bus arbitration logic that a DMA request is pending, and the request will be serviced at the first available opportunity. All DMA is qword, so four complete data words will be read from memory and stored in the appropriate video, cursor, or sound FIFO for each DMA burst. Note that video DMA may be read from memory in bursts of more than 4 words allowing almost continuous DRAM page mode access to occur.

The system software should create a video frame buffer in DRAM memory, and program the DMA address pointers to the start, end, and desired initial location within the buffer. All DMA pointer addresses should be qword aligned. Once the display is enabled, the video registers should only be programmed during the FLYBACK period to ensure flicker free updating of the screen. See Chapter 10 for details of how to program the DMA controller.

15.2 Setting the FIFO Preload Value

The video FIFO is a 32-entry, 32-bit-wide FIFO. Words of video data are clocked into the top of the FIFO under control of the internal CL-PS7500FE signals, BUSCLK and nVIDAK. Words are clocked out of the bottom of the FIFO as the video system displays the data, controlled by the pixel clock.

The FIFO is flushed during vertical flyback time, so before the start of the frame the FIFO is empty. At the start of the frame a video request is made to the memory subsystem by asserting the internal CL-PS7500FE signal, nVIDRQ. When a predetermined number of words have been loaded into the FIFO the request is removed. As the data in the FIFO is displayed, further video requests are made to refill the FIFO to the desired level.

The Control register (CONREG) includes a 3-bit field (CONREG[10:8]) to set the preload value of the video FIFO. In this way the FIFO can be programmed to load 4, 8, 12, 16, 20, 24, or 28 words of data into the FIFO at the start of a frame. After the start of a frame, the FIFO will request more data when the number of words it contains falls below the preloaded value.

The point where the FIFO requests more data to be loaded is dependent upon system considerations:

- If the FIFO is reloaded too late, there is a danger that it will run out of data (underflow)
- If the FIFO is reloaded too early, then there is a danger that the data will not fit into the FIFO (overflow)

In general, the higher the bandwidth of the screen, the more words required to preload into the FIFO. In a low bandwidth screen mode, it is not always desirable to have a large preload value, as the bus traffic has long bursts of data transferred at the start of the frame.

The optimum value to preload depends upon the screen mode in use (that is, the rate the data is read from the FIFO), and both the latency of the memory controller and the rate that data is provided to the CL-PS7500FE. It is generally prudent to program the minimum value possible to keep the bus traffic even.

For Equation 15-1, let:

n is the value programmed into the control register

v (words/µs) is the rate the video data is displayed

Lmax (µs) is the maximum latency in the memory system (this is the maximum time between the CL-PS7500FE requesting more video data and the memory system delivering the first word of that data)

If the FIFO is almost empty, it takes 0.025 µs for a word of data to reach the bottom of the FIFO before it can be used. The minimum value for *n* is deduced from the following condition to avoid the FIFO underflowing: There are four *n* words in the FIFO when the FIFO requests more data and, if not refilled, the FIFO is empty in 4n/v µs. Therefore, *n* must be set as in Equation 15-1.

$$4n/v > (Lmax + 0.025) \qquad \textbf{Equation 15-1}$$

The maximum value for *n* is deduced from the following condition to avoid the FIFO overflowing: *n* can have a maximum value of 7 and the FIFO can never overflow as there are always 4 words available in the top of the FIFO, even if the video request is serviced immediately.

15.2.1 Example

For the CL-PS7500FE, the value of *v* (words/µs) changes depending on the video mode and the pixel clock rate selected. In the worst-case DMA latency, *Lmax* alters depending on if ROM, DRAM accesses, or internal programming bursts are slowest, and the MEMCLK frequency used.

Chapter 9 discusses how to calculate worst-case DMA latencies for a particular system using the CL-PS7500FE. The value calculated should be imported as *Lmax* into Equation 15-2.

Assume an 8-bpp mode with a pixel clock rate of 60 MHz (period = 16.7 ns). In each pixel clock tick, 1/4 of a word is used – in a whole µs, $0.25 \times 1 \div 0.0167 = 14.9$ words are required.

Hence the value of *n* must be:

$$4n/v > (Lmax + 0.025) \qquad \textbf{Equation 15-2}$$

So, assuming an *Lmax* value of 1.0 µs:

$$n > 3.74 \times (1.0 + 0.025) \geq n > 3.83 \qquad \textbf{Equation 15-3}$$

So in this case the minimum value for *n* to prevent FIFO underflow is four.

16. THE VIDEO SOUND AND PROGRAMMER'S MODEL

16.1 The Video and Sound Macrocell Registers

The video and sound macrocell contains 296 write-only registers. These are split into 2 categories: the 256 28-bit video palette entries, and the remaining control registers. The video palette entries are written through an auto-incrementing address pointer. All the other registers (including the 28-bit cursor palette) are written directly with the address encoded in the top 4 or 8 bits of the data word. To program the registers, set the CL-PS7500FE address bus to between 0x03400000 and 0x034FFFFF and the data word written should include the individual register address in the upper 4 or 8 bits, as appropriate.

To define the display format correctly, eleven registers must be programmed as shown in Figure 16-1.

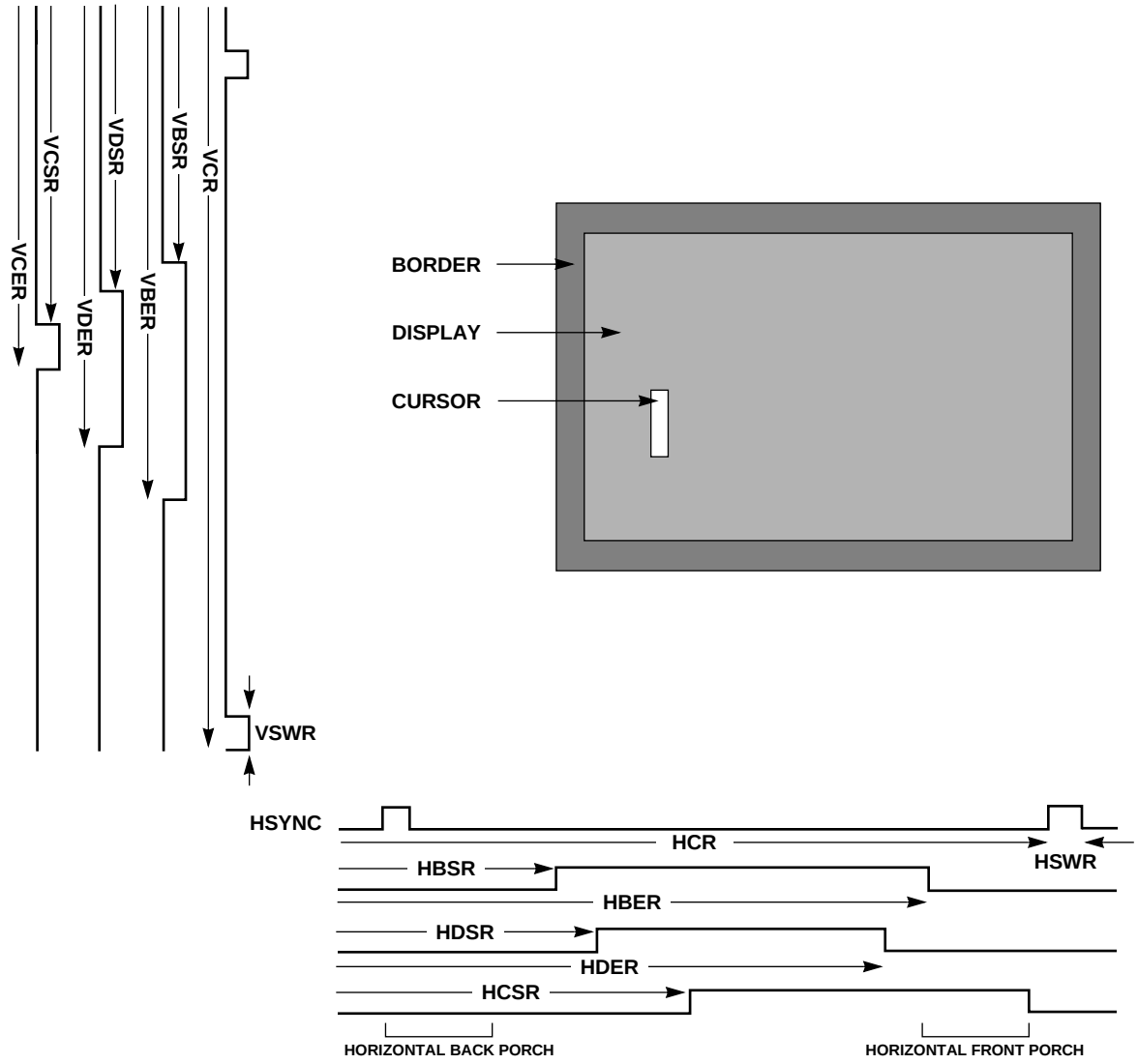


Figure 16-1. The Video and Sound Macrocell Display Format Definitions

The register allocation is shown in Table 16-1. An 'x' denotes the actual data field, and any unused bit should be programmed with a logic zero.

NOTE: Do not access any register at any location other than that shown as the actual register map is multiple-mapped.

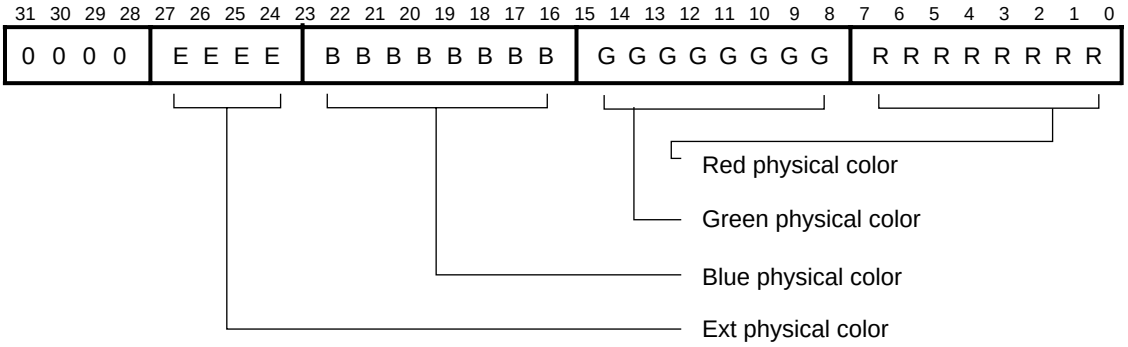
Table 16-1. The Video and Sound Macrocell Register Allocation

Address (Hex)	Register Name	Page
0xxxxxxx	Video Palette	142
100000xx	Video Palette Address	142
20000000	Reserved	
300000xx	LCD Offset 0	143
310000xx	LCD Offset 1	143
4xxxxxxx	Border Color	144
5xxxxxxx	Cursor Palette Logical Color 1	144
6xxxxxxx	Cursor Palette Logical Color 2	144
7xxxxxxx	Cursor Palette Logical Color 3	144
8000xxxx	Horizontal Cycle	145
8100xxxx	Horizontal Sync Width	145
8200xxxx	Horizontal Border Start	145
8300xxxx	Horizontal Display Start	146
8400xxxx	Horizontal Display End	146
8500xxxx	Horizontal Border End	146
8600xxxx	Horizontal Cursor Start	147
8700xxxx	Reserved	147
8800xxxx	Test	147

Address (Hex)	Register Name	Page
8C00xxxx	Test	147
9000xxxx	Vertical Cycle	147
9100xxxx	Vertical Sync Width	148
9200xxxx	Vertical Border Start	148
9300xxxx	Vertical Display Start	148
9400xxxx	Vertical Display End	149
9500xxxx	Vertical Border End	149
9600xxxx	Vertical Cursor Start	149
9700xxxx	Vertical Cursor End	150
9800xxxx	Test	150
9A00xxxx	Test	150
9C00xxxx	Test	150
B00000x	Sound Frequency	154
B10000x	Sound Control	155
C00xxxxx	External	151
D000xxxx	Frequency Synthesis	152
E00xxxxx	Control	153
F000xxxx	Data Control	154

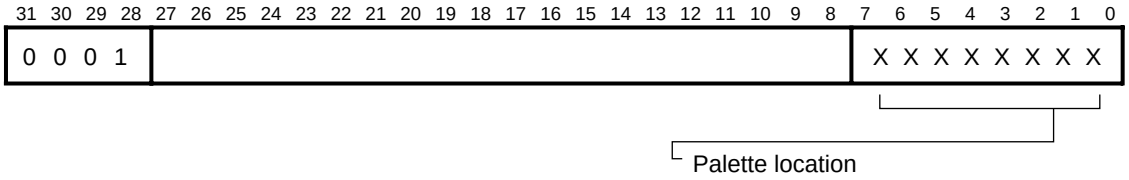
The External, Control, Sound Control, and Data Control registers all contain bits that are not initialized at power up. These registers must be programmed before the video and sound macrocell can operate properly.

16.2 Video Palette: Address 0x0



All entries of the video palette are written at address 0. To write any or all of the palette locations, the address pointer must first be written, as described below. The palette is programmed with a 28-bit word representing the physical data field.

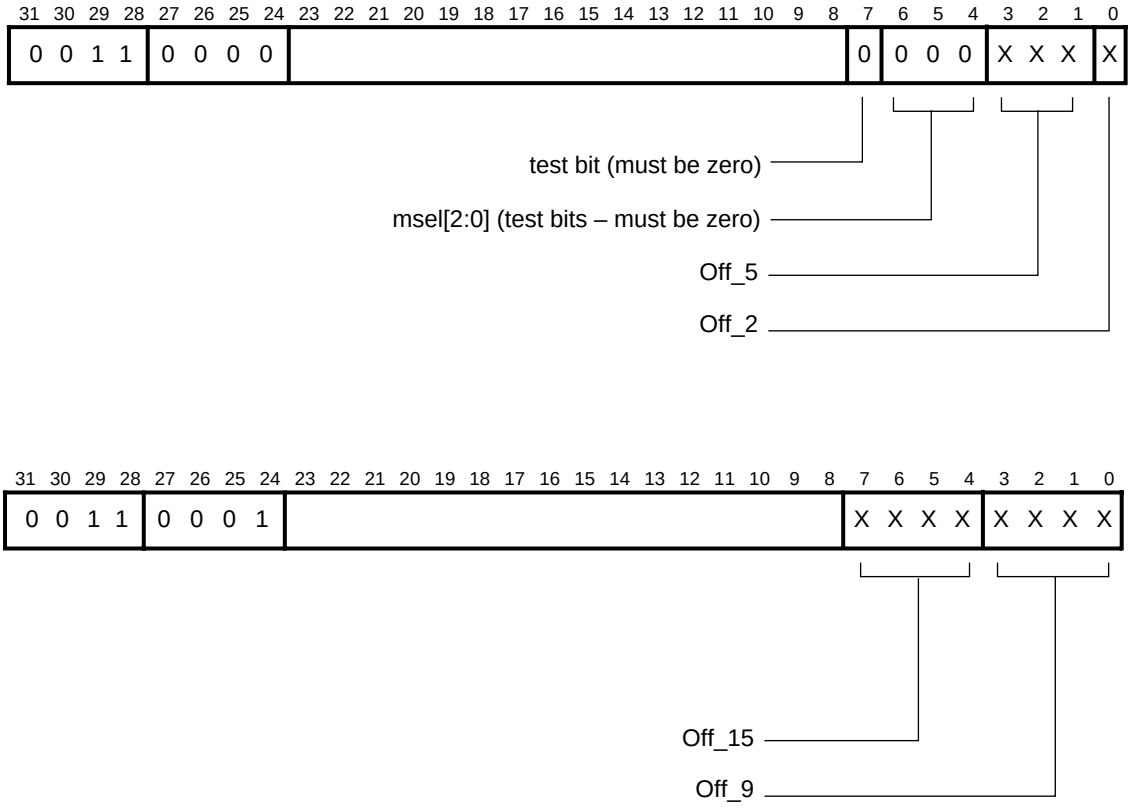
16.3 Video Palette Address Pointer: Address 0x1



The address pointer is programmed at address 1, and it can be programmed to any value from 0 to 255. The first write to the palette then occurs at this location, and the address pointer post-increments so that the next palette write occurs to the following location. The counter wraps around from 255 to 0.

Once the address pointer is written, any number of palette locations can be programmed, and the pointer can be reprogrammed at any time if only part of the whole palette is to be updated.

16.4 LCD Offset Registers: Addresses 0x30 and 0x31



These two 8-bit registers define the offsets required for driving a dual panel LCD screen. Register 0 defines the offsets for the five and two frame duty cycle gray scales, as well as reset and test mode bits. Register 1 defines the offsets for the nine- and fifteen-frame, duty-cycle grayscales.

The registers values are dependent upon the size of the LCD screen to be driven, and are calculated as shown in Equation 16-1.

$$\begin{aligned} Off\_15 &= (3xL + 8) \bmod 15 \\ Off\_9 &= (7xL + 4) \bmod 9 \\ Off\_5 &= (1xL + 3) \bmod 5 \\ Off\_2 &= 0 \end{aligned}$$

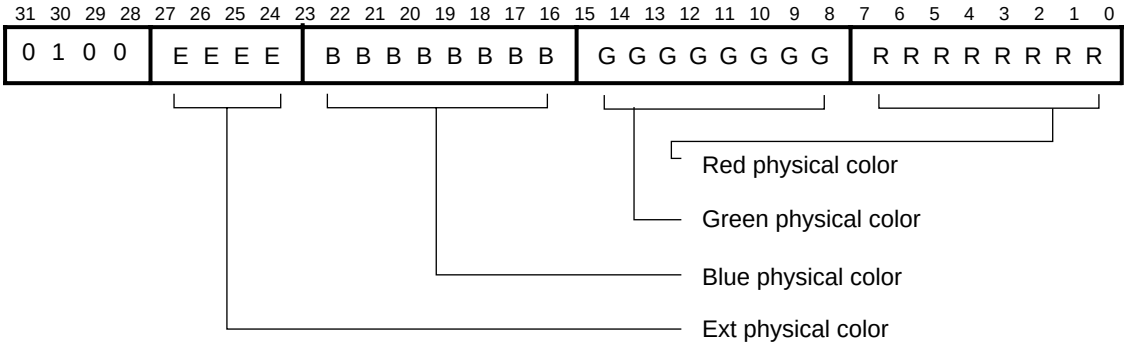
Equation 16-1

where,
L is the number of lines in the upper panel of the dual-panel LCD screen.

Bits 7:4 of register 0 are only used in test mode, and must all be set to ‘0’ in normal operation.

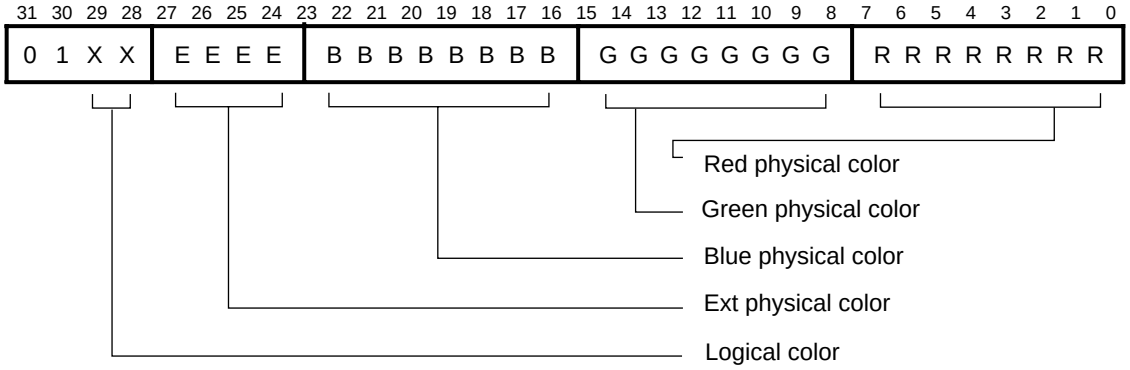
msel[2:0] are test bits and must be programmed low.

16.5 Border Color Register: Address 0x4



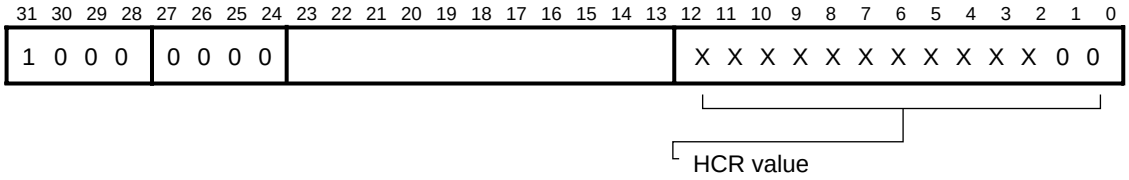
This register defines the physical border color, and is programmed with a 28-bit word. Note that this register is programmed directly, independent of the value of the video palette address pointer.

16.6 Cursor Palette: Addresses 0x5–0x7



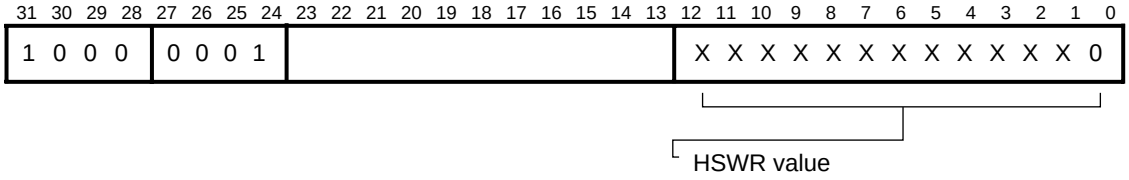
These three registers are programmed with the physical color of the three logical cursor colors. Note that cursor logical color 00 is defined as being transparent (that is, no cursor display), and its location is used for the Border Color register above.

16.7 Horizontal Cycle Register (HCR): Address 0x80



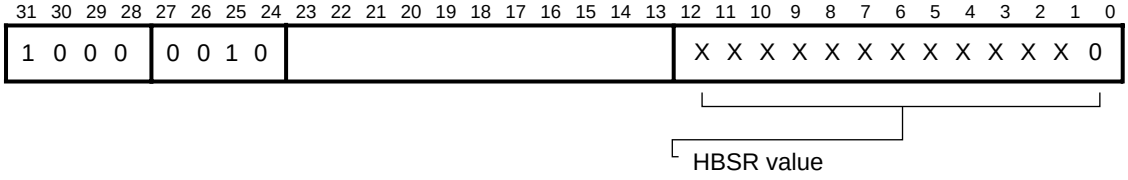
This register defines the period, in pixels, of the horizontal scan that is, display time + retrace time. This is a 14-bit register of which the bottom 2 bits must be programmed to ‘0’. If N pixels are required in the horizontal scan period, then value (N – 8) should be programmed into the HCR (N must be a multiple of 4).

16.8 Horizontal Sync Width Register (HSWR): Address 0x81



This register defines the period, in pixels, of the HSYNC pulse. This is a 14-bit register of which the bottom bit must be programmed to ‘0’. If N pixels are required in the HSYNC pulse, then value (N – 8) should be programmed into the HSWR (N must be a multiple of 2).

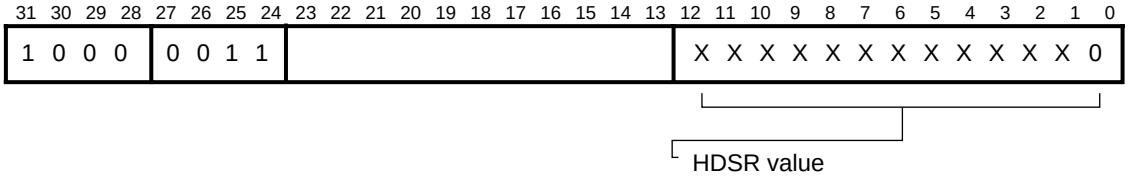
16.9 Horizontal Border Start Register (HBSR): Address 0x82



This register defines the time, in pixels, from the start of the HSYNC pulse to the start of the border display. This is a 14-bit register of which the bottom bit must be programmed to ‘0’. If N pixels are required in this time, then program the value (N – 12) into the HBSR (N must be a multiple of 2).

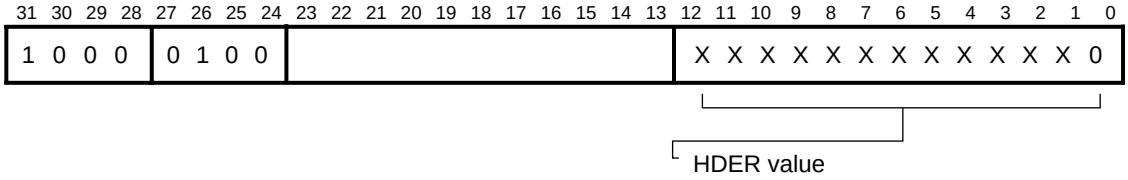
NOTE: This register must always be programmed, even when a border is not required. If a border is not required, then the value in the HBSR must be set to start the border in the same place as the display starts (that is, $N_{HBSR} = N_{HDSR)}$.

16.10 Horizontal Display Start Register (HDSR): Address 0x83



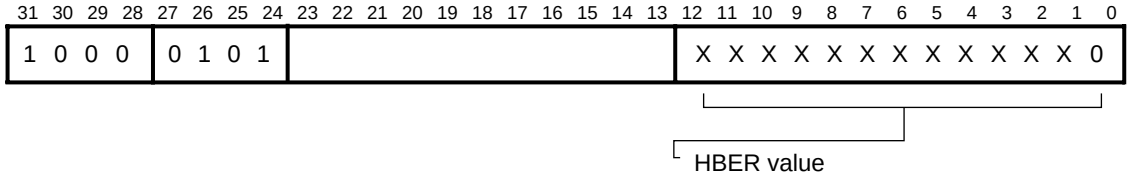
This register defines the time, in pixels, from the start of the HSYNC pulse to the start of the video display. This is a 14-bit register of which the bottom bit must be programmed to '0'. If N pixels are required in this time, then program the value (N – 18) into the HBSR (N must be a multiple of 2).

16.11 Horizontal Display End Register (HDER): Address 0x84



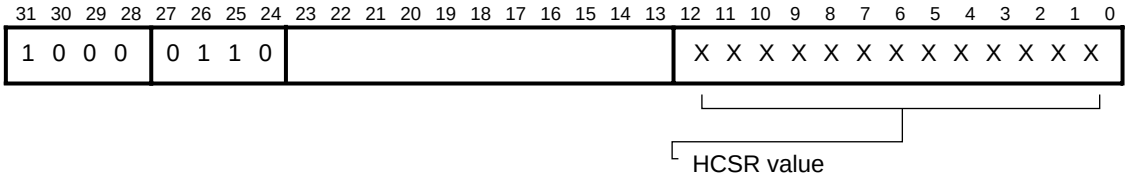
This register defines the time, in pixels, from the start of the HSYNC pulse to the end of the video display (that is, the first pixel that is not display). This is a 14-bit register of which the bottom bit must be programmed to 0. If N pixels are required in this time, then program the value (N – 18) into the HBER (N must be a multiple of 2).

16.12 Horizontal Border End Register (HBER): Address 0x85



This register defines the time, in pixels, from the start of the HSYNC pulse to the end of the border display (that is, the first pixel that is not border). This is a 14-bit register of which the bottom bit must be programmed to '0'. If N pixels are required in this time, then program the value (N – 12) into the HBER (N must be a multiple of 2.) Again, if no border is required, this register must still be programmed so that $N_{HBER} = N_{HDER}$.

16.13 Horizontal Cursor Start Register (HCSR): Address 0x86



This register defines the time, in pixels, from the start of the HSYNC pulse to the start of the cursor display. This is a 14-bit register where all bits can be programmed. If N pixels are required in this time, then program the value (N – 17) into the HCSR. The cursor can thus be programmed to start on any pixel. In HiRes mode, the cursor can still only be programmed to start on a normal pixel boundary. However, because the resolution of the cursor can be defined to a μ-pixel, by using different cursor images it is possible to position the cursor to any μ-pixel.

NOTE: Only the cursor start position needs to be defined, as the cursor is automatically disabled after 32 pixels in normal mode or 16 pixels in HiRes mode. If a smaller cursor is required, program the remaining bits in the cursor pattern to logical color '00' (transparent).

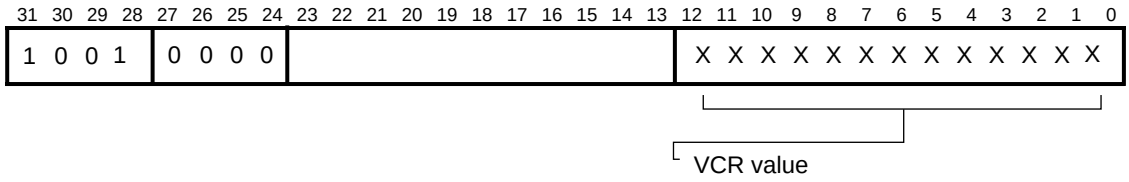
16.14 Horizontal Interlace Register (HIR): Address 0x87

Address 87H is reserved. Do not program this register.

16.15 Horizontal Test Registers: Addresses 0x88 and 0x8H

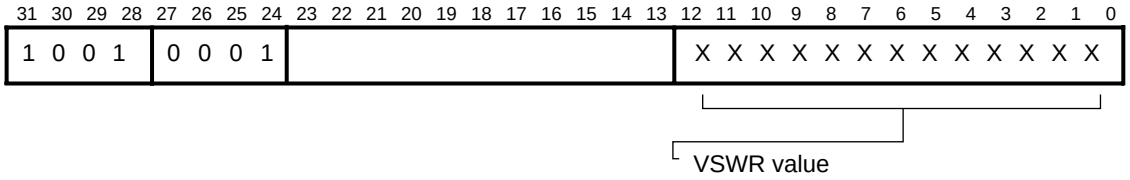
These two registers are provided for test purposes only. Neither of these registers are intended for use during normal operation.

16.16 Vertical Cycle Register (VCR): Address 0x90



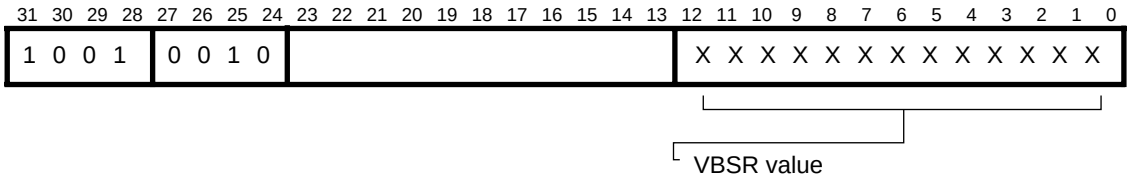
This 13-bit register defines the period, in units of a raster, of the vertical scan (that is, display time + FLYBACK time). If N rasters are required in a complete frame, then value (N – 2) should be programmed into the VCR. If an interlaced display is selected, (N – 3) ÷ 2 must be programmed into the VCR (N must be odd.) Here N is still the number of rasters in a complete frame, *not* a field.

16.17 Vertical Sync Width Register (VSWR): Address 0x91



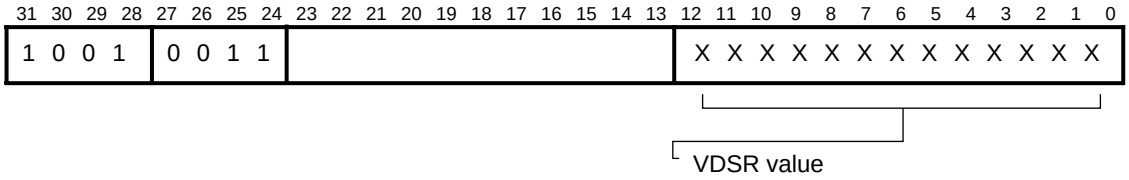
This 13-bit register defines the width, in units of a raster, of the VSYNC pulse. If N rasters are required in the VSYNC pulse, then program the value (N – 2) into the VSWR. The minimum value allowed for N is 2.

16.18 Vertical Border Start Register (VBSR): Address 0x92



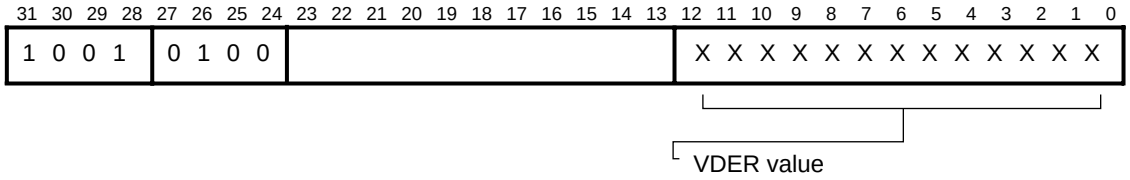
This 13-bit register defines the time, in units of a raster, from the start of the VSYNC pulse to the start of the border display. If N rasters are required in this time, then program the value (N – 1) into the VBSR. If no border is required, this register must still be programmed, in this case to the same value as the VDSR.

16.19 Vertical Display Start Register (VDSR): Address 0x93



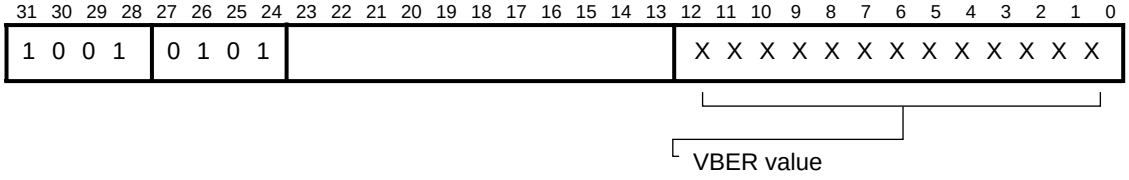
This 13-bit register defines the time, in units of a raster, from the start of the VSYNC pulse to the start of the video display. If N rasters are required in this time, then program the value (N – 1) into the VDSR.

16.20 Vertical Display End Register (VDER): Address 0x94



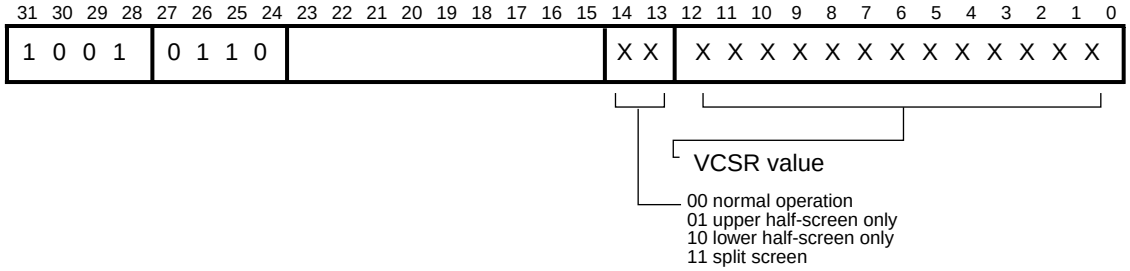
This 13-bit register defines the time, in units of a raster, from the start of the VSYNC pulse to the end of the video display. (that is, the first raster where the display is *not* present). If N rasters are required in this time, then program the value (N – 1) into the VDER.

16.21 Vertical Border End Register (VBER): Address 0x95



This 13-bit register defines the time, in units of a raster, from the start of the VSYNC pulse to the end of the border display (that is, the first raster where the border is not present). If N rasters are required in this time, then program the value (N – 1) into the VBER. If no border is required, then this register must be programmed to the same value as the VDER.

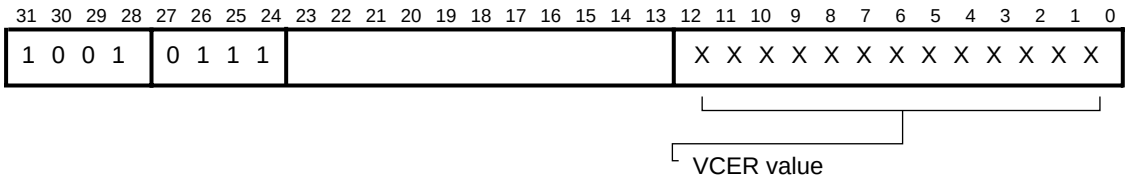
16.22 Vertical Cursor Start Register (VCSR): Address 0x96



This is a 15-bit register. The lower 13 bits define the time, in units of a raster, from the start of the VSYNC pulse to the start of the cursor display. If N rasters are required in this time, then program the value (N – 1) into the VCSR. The upper 2 bits control the display of the cursor in duplex LCD mode. Program these bits to '0' in all other modes.

When the upper 2 bits are programmed to '11' (split screen) the meaning of VCSR and VCER are altered as follows: The cursor is displayed in the lower half-screen only from the value of VDSR to the value of VCSR, and again in the upper half-screen only from the value of VCER to the value of VDER. This allows a cursor to be positioned across the boundary of the upper and lower half-screens of an LCD.

16.23 Vertical Cursor End Register (VCER): Address 0x97

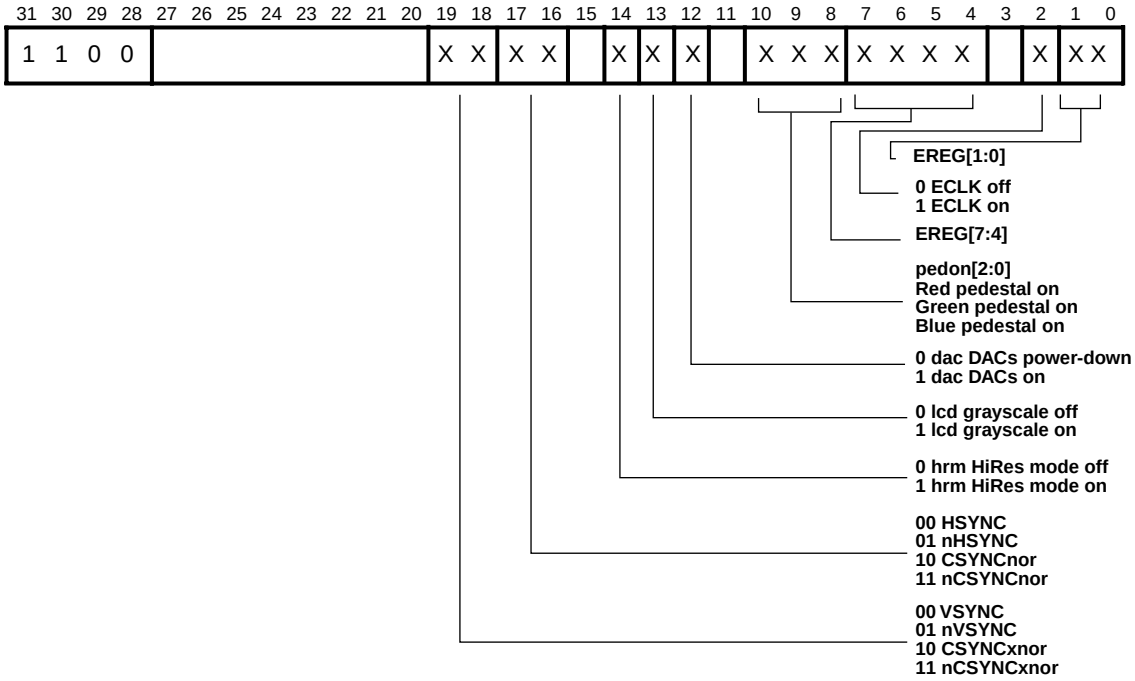


This 13-bit register defines the time, in units of a raster, from the start of the VSYNC pulse to the end of the cursor display (that is, the first raster where the cursor is not present). If N rasters are required in this time, then program the value (N – 1) into the VCER.

16.24 Vertical Test Registers: Addresses 0x98, 0x9A and 0x9C

These registers are provided for test purposes only. None are intended for use during normal operation.

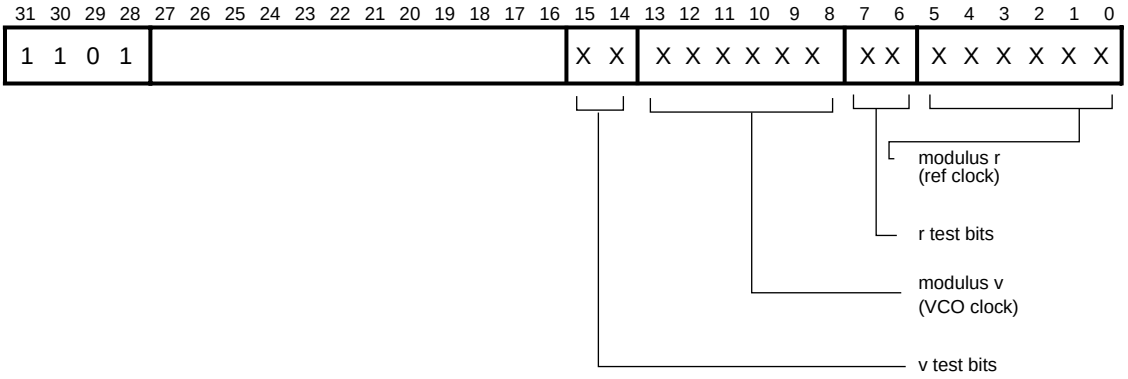
16.25 External Register (EREG): Address 0xC



This register contains the control bits for the external functions of the video and sound macrocell. In particular, it controls the DACs, the configuration of the External port, ED[7:0], and the configuration of the sync lines. EREG[1:0] are mapped internally to drive esel[1:0] by CL-PS7500FE. EREG[7:4] are exported from the chip on ED[7:4] if EREG[1:0] = 3. Refer to Chapter 12. The use of EREG[10:8] (pedon[2:0]) and DAC is defined in Chapter 12. The use of EREG[13] (lcd) and EREG[14] (hrm) are defined in Chapter 12.

CL-PS7500FE can export a variety of sync configurations on the pins HSYNC and VSYNC, as specified by the bits 16–17 and 18–19, respectively. For further explanation see Chapter 12.

16.26 Frequency Synthesizer Register (FSYNREG): Address 0xD

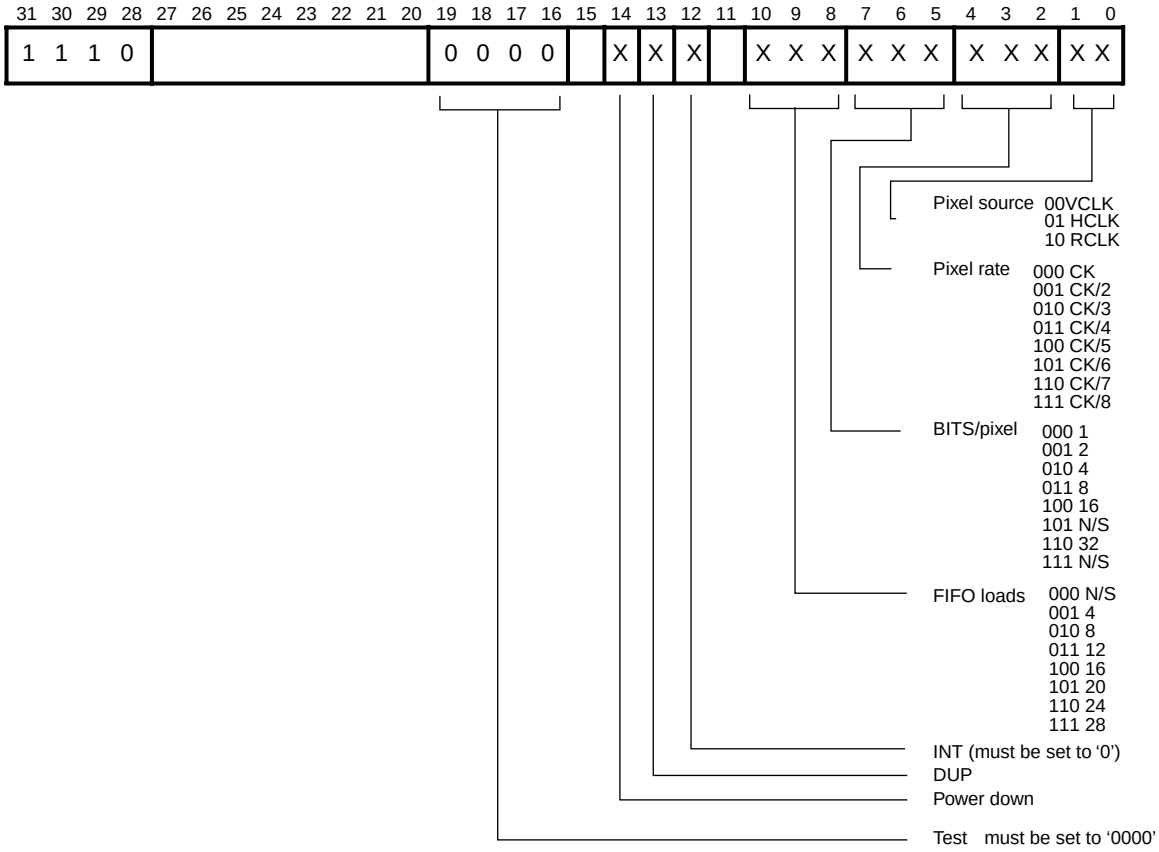


The CL-PS7500FE can drive a VCO to provide a suitable input frequency for the pixel clock derived from a reference clock. This is achieved by dividing the reference clock by modulus r, and the VCO clock by modulus v, then comparing the resultant frequencies. Refer to Chapter 12 for a more detailed explanation. The two moduli, r and v, are each 6-bit values, and are programmed in this register. Each counter has 2 associated test bits that are normally programmed to '0'.

Setting bit 6	forces the phase comparator high, which drives PCOMP high.
Setting bit 7	clears the r-modulus counter.
Setting bit 14	forces the phase comparator low, which drives PCOMP low.
Setting bit 15	clears the v-modulus counter.

To reduce power consumption, program this register with large values when the frequency synthesizer is not in use. In particular, do not set bits 6 and 14 at the same time. To get a modulus of r, program the value (r – 1) into FSYNREG. Do the same for the v modulus.

16.27 Control Register (CONREG): Address 0xE



This main control register determines the basic operation of the CL-PS7500FE. The pixel clock source, the pixel rate, the number of bits/pixel, the control of the video FIFO, and the data format are programmed in this register. In addition, there is a 4-bit test register that must be programmed to '0' for normal operation.

NOTE: The INT bit must always be set to '0'.

The pixel clock (PIXCLK) is selected from one of three sources, corresponding to the respective input pins, and the selected clock is then fed through a prescalar as defined by the 3 bits CONREG[4:2]. The output of this prescalar is the actual pixel clock. See Chapter 12 for more details.

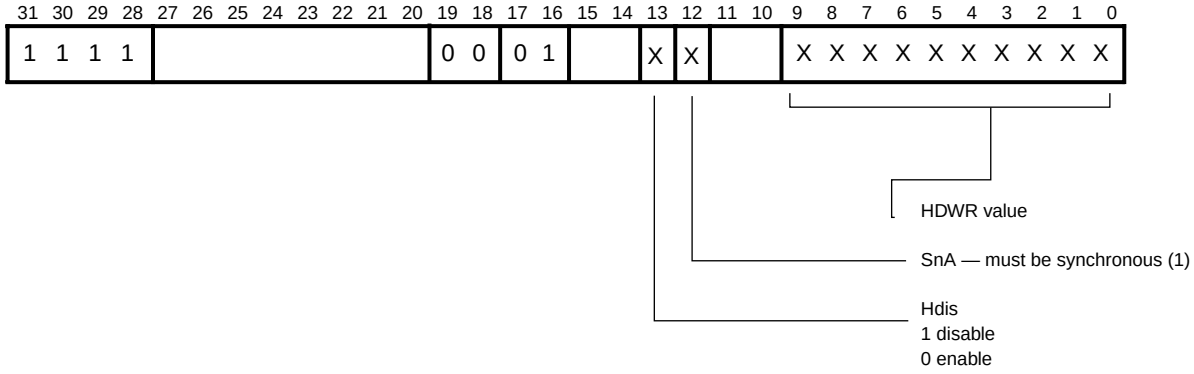
The video FIFO can be programmed to have any number of qwords loaded into it at the start of display. The value chosen should take into account the bandwidth of the display, as well as the latency of the DMA subsystem. Refer to Chapter 15 before programming these values. When CONREG[13] (DUP) is set, the display for dual-panel LCDs is configured. This is further described in Chapter 12.

NOTE: After a reset, CONREG must be the first register programmed. CONREG[14] must then be immediately be programmed low. The test registers bits (19:16) should also be programmed low, as any other setting inhibits normal operation.

The video macrocell uses dynamic logic structures for maximum performance. When CONREG[14] is set high, the main video data path is set into a state where it does not consume static current. This must be done before the CL-PS7500FE is set into STOP mode.

16.28 Data Control Register (DCTL): Address 0xF

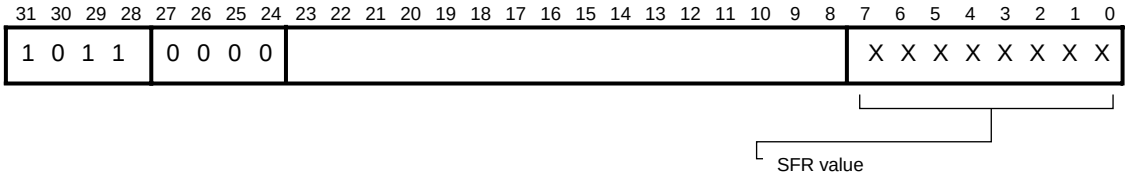
The horizontal display width is also defined in this register, and should be programmed to be the number of words of data in a displayed raster. It must be programmed in most configurations of the device, as it inhibits a DMA request near the end of a raster, when there are enough words in the video FIFO for that raster. The request is uninhibited after the HSYNC at the start of the next raster. When driving a dual-panel LCD screen, this register must be programmed with twice the number of words in a displayed raster. DCTL[14] (Hdis) must normally be programmed to '0'. If DCTL[14] is programmed to one, the inhibition of DMA requests is disabled.



NOTE: Bits 19:16 *must* be set to '0001' (binary).

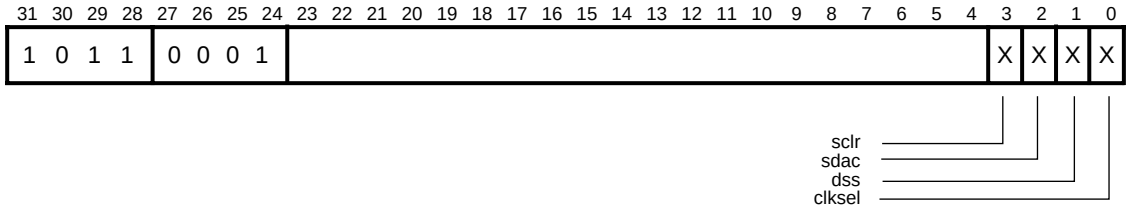
16.29 Sound Frequency Register (SFR): Address 0xB0

This 8-bit register specifies the byte sample rate of the sound data. It is defined in units of 1 µs. See Chapter 13 for more details.



If a sample rate of N µs is required, N – 2 should be programmed into SFR. N can have any value between 3 and 256.

16.30 Sound Control Register (SCTL): Address 0xB1



This 4-bit register defines various control bits for the sound system.

Bit 3: sclr	This bit must always be programmed low.
Bit 2: sdac	This bit must be written as ‘0’.
Bit 1: dss	This bit selects Serial Sound mode.
Bit 0: clksel	This bit selects the clock used in the sound system. When this bit is high, the internal CL-PS7500FE 32-MHz I/O reference clock is used; when low the optional sound clock is used.

17. FPA COPROCESSOR MACROCELL

The FPA is a floating-point accelerator for the ARM family of CPUs. The FPA is designed to maximize the performance/power, performance/cost, and performance/die size ratios while still providing a balanced floating-point versus integer performance for ARM-based systems.

Typical performance, in the range 3–8 Mflops, is expected at a clock frequency of 40 MHz; actual performance is dependent upon:

- the precision selected
- the system configuration
- the degree that the floating-point code is scheduled and otherwise optimized

The FPA in the CL-PS7500FE is an on-chip floating-point coprocessor connected to the ARM processor core. It is a fully static design and its low power consumption, especially when in Standby mode, makes it eminently suitable for portable and other power- and cost-sensitive applications. When used in conjunction with its support code, the FPA fully implements the IEEE Standard for binary floating-point arithmetic (ANSI/IEEE Std. 754-1985).

The design of the FPA is based on an 81-bit internal datapath, with autonomous load/store and arithmetic units that can operate concurrently. Single, double, and extended precision IEEE formats are all supported. The FPA achieves its high performance while remaining a low-cost and low-power solution, by employing RISC and other advanced design techniques. It is interfaced to the ARM CPU over a simple, high-performance coprocessor bus. The ARM instruction pipeline is mirrored on the FPA so that floating-point instructions can be executed directly with minimal communication overhead. Pipelining, concurrent execution units, and speculative execution are all employed to improve performance without having a great impact on power consumption.

A RISC approach has been taken in selecting between those floating-point instructions that are candidates for implementation in the FPA and those handled by software support. The FPA instruction repertoire includes only the basic operations plus compare, absolute value, round to integral value, and floating-point to integer and integer to floating-point conversions. In addition, only normalized operands and zeros are handled in hardware; operations on denormalized numbers, infinities, and NaNs are handled by the support code. Only the inexact exception is dealt with by hardware; all other exceptions cause the software support code to be called, whether or not the associated trap is enabled. This approach has helped to minimize the die size while having a negligible effect on performance in most applications.

17.1 FPA Functional Blocks

FPA consists of five main functional blocks described in the following sections.

- Coprocessor interface
- Instruction issuer
- Load-store unit
- Register bank
- ALU

17.1.1 Coprocessor Interface

This block is responsible for arbitrating instructions with the CPU and relating to the load-store unit when to go ahead with data transfers.

Like ARM integer instructions, all ARM floating-point instructions are conditional, obviating the need for branches for many common constructs. If a failed condition causes an instruction already issued to the load-store unit or ALU to be skipped, that instruction is cancelled and any results calculated thus far are discarded.

The same mechanism is used to cancel prefetched instructions if a branch is taken or if the ARM CPU gets interrupted before an FPA instruction has been arbitrated.

17.1.2 Instruction Issuer

The instruction issuer is responsible for examining the incoming instruction stream and deciding if any instructions are candidates for issuing to either the load-store unit or the ALU.

Instructions can be selected from the fetch, decode or execute stages of the ARM pipeline follower. Data anti-dependency hazards (write-after-write and write-after-read) are dealt with by this unit by preventing issuance until the hazard is cleared.

Instructions are issued strictly in order and only one can be issued per cycle.

17.1.3 Load-Store Unit

The load-store unit does the formatting and conversion necessary when moving data between the 32-bit ARM data bus and the 81-bit internal register format. It is also responsible for checking all input operands and flagging any that are not normalized numbers or zero.

Most subsequent operations on flagged data cause the instruction to be passed to software which will then emulate the instruction. All internal operations are performed to the internal 81-bit format.

17.1.4 Register Bank

The register bank contains eight 81-bit dual read-access, dual write-access registers.

Data dependency hazards (read-after-write) are handled by the register control logic; read requests from either unit are stalled until the hazard is cleared.

There is also a 33-bit temporary register, used by the FIX, FLT, and compare instructions to transfer intermediate results between the load-store unit and the ALU.

The register bank also contains logic for register-forwarding, allowing the result of one calculation to be used directly as the source for the next.

17.1.5 Arithmetic Unit

The arithmetic unit has a four-stage pipeline (Prepare, Calculate, Align, and Round) and can speculatively execute instructions up to, but not including, register writeback. Writeback can only occur once the instruction has been arbitrated with the ARM CPU.

An unusual feature of the pipeline is that each of the pipeline stages is offset by one half-cycle from the previous stage, allowing some instructions to traverse the pipeline in 2 cycles.

The calculate stage includes a 67-bit adder, iterative array multiplier and divide unit. Fast barrel shifters are used for pre-alignment and post-normalization.

Arithmetic operations are normally performed asynchronously to the ARM instruction stream so that an instruction is arbitrated with the CPU before the FPA has detected whether an exception will occur. Arithmetic exceptions are therefore normally imprecise. If precise exceptions are required (for example, in debugging), a mode bit (the SO bit in the FPSR) can be set. This forces arbitration to be delayed until the arithmetic operation completes, at the expense of a reduction in performance.

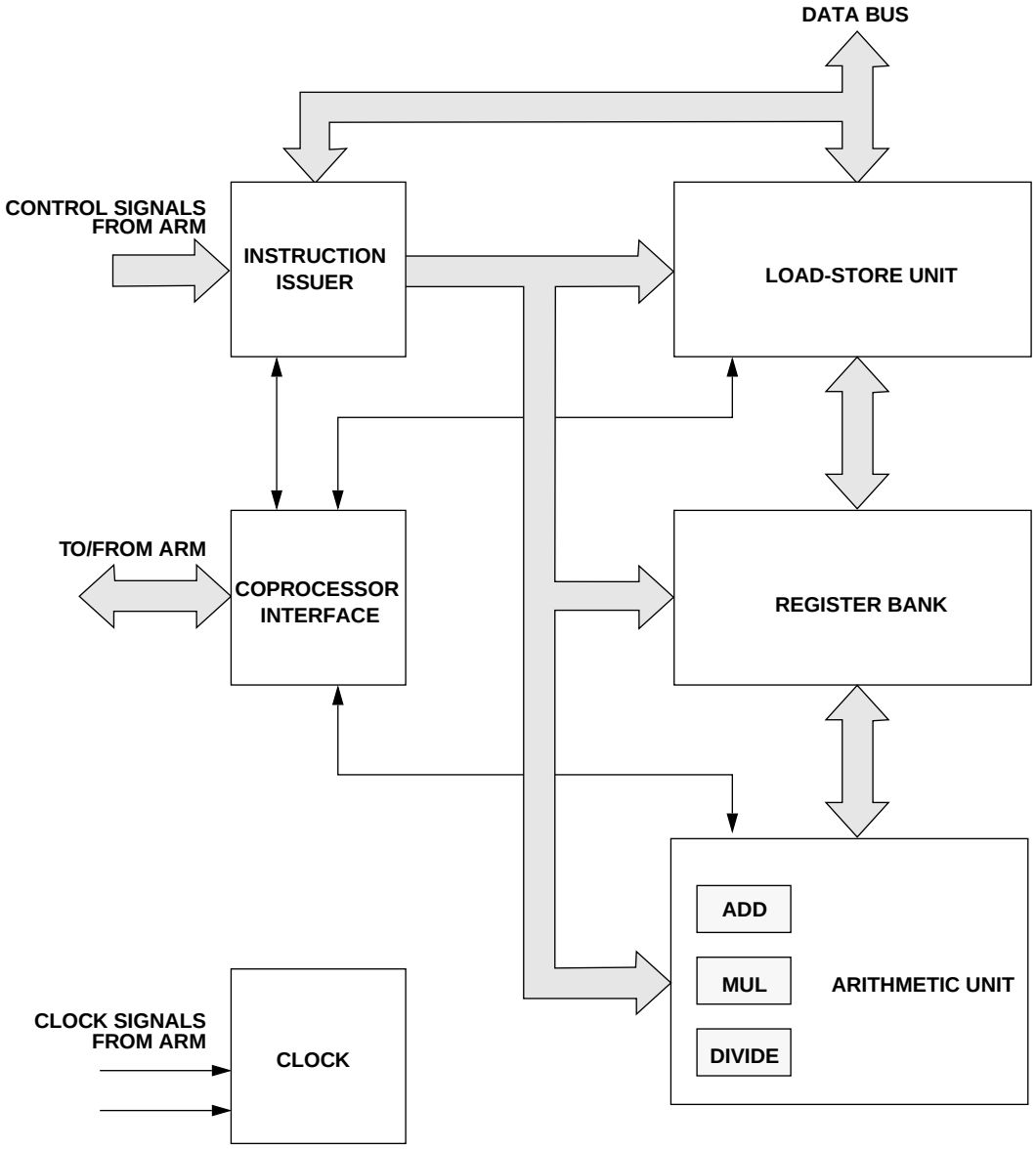


Figure 17-1. FPA Block Diagram

18. FLOATING-POINT COPROCESSOR PROGRAMMER'S MODEL

18.1 Overview

The ARM IEEE floating-point system has:

- 8 high-precision floating point registers, F0 to F7
- a working precision of 80 bits, comprising:
  - 64-bit mantissa
  - a 15-bit exponent
  - a sign bit

18.1.1 Floating-Point Status Register

There is a floating-point status register (FPSR) which, like ARM's PSR, holds all the necessary status and control information for the floating point system that an application should be able to access. It holds flags which indicate various error conditions, such as overflow and division by zero. Each flag has a corresponding trap enable bit that can be used to enable or disable a trap associated with the error condition. Bits in the FPSR allow a client to distinguish different implementations of the floating-point system and to enable or disable special features of the system.

18.1.2 Floating-Point Control Register

The FPA also contains a floating-point control register (FPCR). This is used to communicate status and control information between the FPA and the FPA support code.

NOTE: The definition of the FPCR may be different for other implementations of the ARM IEEE floating point system; the FPCR may not even exist in some implementations. Software outside the floating-point system should therefore not use the FPCR directly.

18.2 Floating-Point Operation

All basic floating-point instructions operate as though the result were computed to infinite precision and then rounded to the length and in the way specified by the instruction. The rounding is selectable from:

- Round to nearest
- Round to +infinity (P)
- Round to -infinity (M)
- Round to zero (Z)

The default is **round to nearest**: as required by the IEEE, this rounds to **nearest even** for the tie case. If one of the other rounding modes is required it must be given in the instruction.

The floating-point system architecture is a load/store architecture (like the ARM CPU); the data-processing operations only refer to floating-point registers. Values can be stored into ARM memory in one of five formats (only four are visible at any one time since P and EP are mutually exclusive):

- IEEE Single Precision (S)
- IEEE Double Precision (D)
- IEEE Double Extended Precision (E)

- Packed Decimal (P)
- Expanded Packed Decimal (EP)

If it is required to preserve register contents exactly (including signalling NaNs), the LFM and SFM instructions should be used. Note however that LFM and SFM should only be used for register preservation within programs and not for data which is to be transferred between programs and/or systems. The format of data stored using SFM is implementation-dependent and can generally only be restored by an LFM instruction from the same implementation.

Floating-point systems may be built from software only, hardware only, or some combination of software and hardware and the results look the same to the programmer. However, the supervising operating system needs to be aware the implementation in use to extract the best performance.

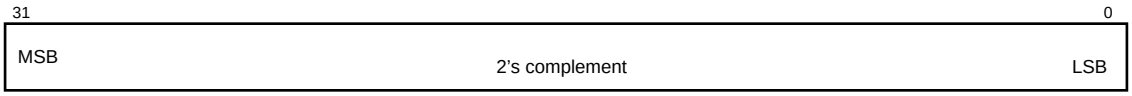
Similarly, compilers can be tuned to generate bunched FP instructions for the FPE and dispersed FP instructions for the FPA to improve overall performance. The manner that exceptions are signalled is at the discretion of the surrounding operating system.

NOTE: In the case of the FPA system, an exception caused by a floating-point data operation or a FLT may be asynchronous (due to the nature of the ARM coprocessor interface.) Such an exception is raised some time after the instruction has started, by which time the ARM may have executed a number of instructions following the one that has failed. This means that the exact address of the instruction that caused the exception may not be identifiable. However, all the information about the exception that the IEEE Standard recommends is available.

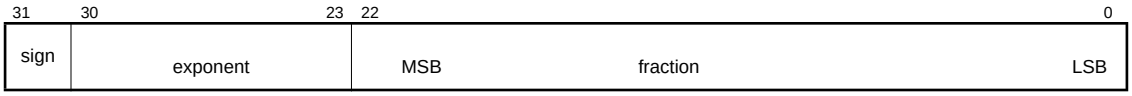
Furthermore, in the FPA a fully synchronous, but slow mode of operation is available that allows the address of the faulting instruction to be determined; this is described in bit 10 SO – Select Synchronous Operation of FPA on page 9-9.

18.3 ARM Integer and Floating-Point Number Formats

18.3.1 Integer



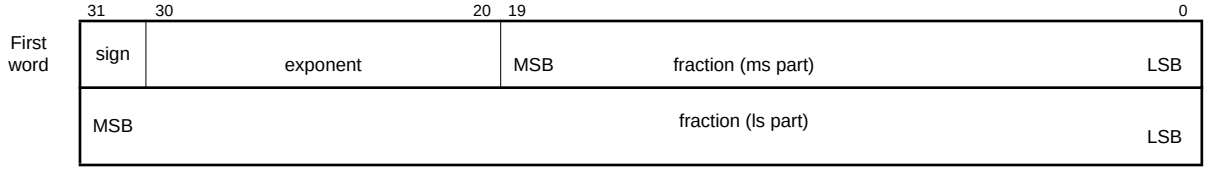
18.3.2 IEEE Single Precision (S)



127 Normalized number exponent bias

126 Denormalized number exponent bias

18.3.3 IEEE Double Precision (D)



1023 Normalized number exponent bias

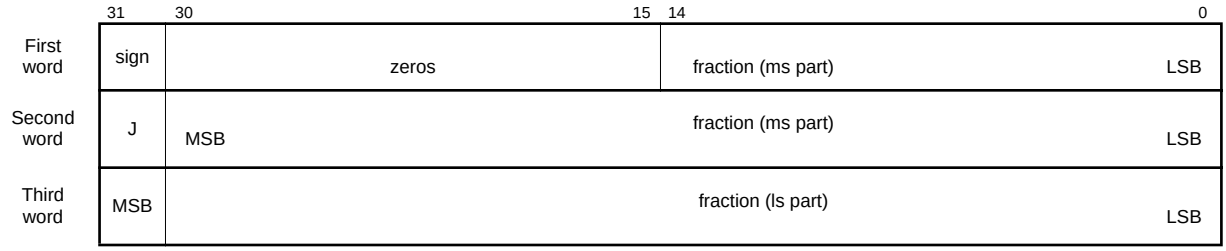
1022 Denormalized number exponent bias

Single and Double Values

Table 18-1. Single and Double Values

	Sign	Exponent	Fraction	Value represented
Quiet NaN	x	maximum	1xxxxxxxxx	IEEE Quiet NaN
Signalling NaN	x	maximum	0 non-zero	IEEE Signalling NaN
Infinity	sign	maximum	0000000000	$(-1)^{sign} \times$ infinity
Zero	sign	0	0000000000	$(-1)^{sign} \times 0$
Denormalized no	sign	0	non-zero	$(-1)^{sign} \times 0.fraction \times 2^{-(denorm.\ bias)}$
Normalized no.	sign	not 0 and not maximum	xxxxxxxxxx	$(-1)^{sign} \times 1.fraction \times 2^{(exponent\ -\ norm.\ bias)}$

18.3.4 IEEE Extended Double Precision (E)



J is the bit to the left of the binary point

16383 normalized and denormalized number exponent bias

Extended Values

Table 18-2. Extended Values

	Sign	Exponent	J	Fraction	Value represented
Quiet NaN	x	maximum	x	1xxxxxxxxx	IEEE Quiet NaN
Signalling NaN	x	maximum	x	0 non-zero	IEEE Signalling NaN
Infinity	sign	maximum	0	0000000000	$(-1)^{sign} \times$ infinity
Zero	sign	0	0	0000000000	$(-1)^{sign} \times 0$
Denormalized no	sign	0	0	non-zero	$(-1)^{sign} \times 0.fraction \times 2^{-(denorm.bias)}$
Normalized no.	sign	not max	1	xxxxxxxxxx	$(-1)^{sign} \times 1.fraction \times 2^{(exponent\ -\ norm.bias)}$
** Illegal value	x	not 0 and not max	0	xxxxxxxxxx	
** Illegal value	x	maximum	1	0000000000	

NOTE: In general, illegal values must not be used, although specific floating-point implementations may use these bit patterns for internal purposes.

18.3.5 Packed Decimal (P)

	31							0
First word	sign	e3	e2	e1	e0	d18	d17	d16
Second word	d15	d14	d13	d12	d11	d10	d9	d8
Third word	d7	d6	d5	d4	d3	d2	d1	d0

- the value is ± d × 10^(± e)
- d18 and e3 are the most significant digits of d and e respectively
- sign contains both the number's sign (bit 31) and the exponent's sign (bit 30). The other bits (29,28) are 0
- the value of d is arranged with the decimal point between d18 and d17, and is normalized so that for an ordinary number $1 \leq d18 \leq 9$
- the guaranteed ranges for d and e are 17 and 3 digits respectively: e3 and d0, d1 may always be zero in a particular system.
- the result is undefined if any of the packed digits is hexadecimal A through F

Packed Decimal Values

Table 18-3. Packed Decimal Values

	Sign (top bit)	Sign (next bit)	Exponent	Digit values
Quiet NaN	x	x	FFFF	d18 > 7, rest non-zero
Signalling NaN	x	x	FFFF	d18 < 8, rest non-zero
± Infinity	0,1	x	FFFF	all 0
± Zero	0,1	0	0000	all 0
Number	0,1	0,1	0000-9999	1-9.999999999999999999

All other combinations are undefined.

18.3.6 Expanded Packed Decimal (EP)

	31							0
First word	sign	e6	e5	e4	e3	e12	e1	e0
Second word	d23	d22	d21	d20	d19	d18	d17	d16
Third word	d15	d14	d13	d12	d11	d10	d9	d8
	d7	d6	d5	d4	d3	d2	d1	d0

- Value is ± d × 10^(± e).
- d23 and e6 are the most significant digits of d and e respectively.
- Sign contains both the number's sign (bit 31) and the exponent's sign (bit 30). The other bits (29,28) are 0.

- The value of d is arranged with the decimal point between d23 and d22, and is normalized so that for an ordinary number $1 \leq d23 \leq 9$.
- The guaranteed ranges for d and e are 21 and 4 digits respectively: e6, e5, e4 and d2, d1, d0 may always be zero in a particular system.
- The result is undefined if any of the packed digits is hexadecimal A through F.

Expanded Packed Decimal Values

Table 18-4. Expanded Packed Decimal Values

	Sign (top bit)	Sign (next bit)	Exponent	Digit values
Quiet NaN	x	x	FFFFFFF	d23 > 7, rest non-zero
Signalling NaN	x	x	FFFFFFF	d23 < 8, rest non-zero
+/- Infinity	0,1	x	FFFFFFF	all 0
+/- Zero	0,1	0	0000000	all 0
Number	0,1	0,1	0-9999999	1-9.99999999999999999999

All other combinations are undefined.

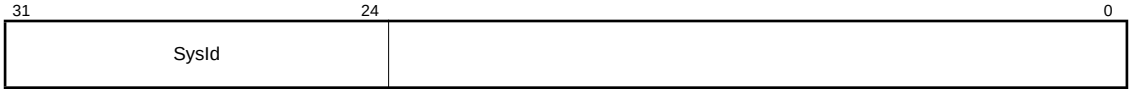
18.4 The Floating-Point Status Register (FPSR)

The floating-point status register (FPSR) consists of:

- a system ID byte
- an exception trap enable byte
- a system control byte
- a cumulative exception flags byte

NOTE: The FPSR is not cleared on reset. It is typically cleared by the support code using an appropriate WFS.

18.4.1 System ID Byte



The 8-bit SysId allows a user or operating system to distinguish which floating-point system is in use. The top bit (bit 31) is:

set	for HARDWARE (i.e. fast) systems
clear	for SOFTWARE (i.e. slow) systems

NOTE: The SysId is read-only.

List of System IDs

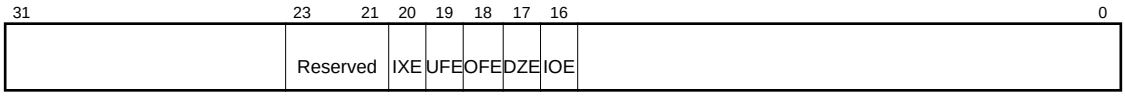
The following system IDs are defined:

Floating-point Emulator	01 (HEX) (Software only)
FPA System	81 (HEX)

The following system IDs are also defined for backwards compatibility:

00(HEX)	for pre-FPA software systems
80(HEX)	for pre-FPA hardware systems

18.4.2 Exception Trap Enable Byte

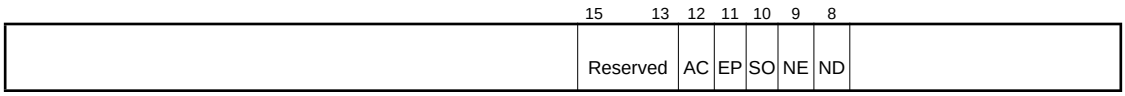


Each bit of the exception trap enable byte corresponds to one type of floating-point exception. The exception types (IX,UF,OF,DZ,IO) are described below.

A bit in the cumulative exception flags byte is set as a result of executing a floating-point instruction only if the corresponding bit *is not* set in the exception trap enable byte; if the corresponding bit in the exception trap enable byte *is* set, an exception trap will be taken instead of setting the exception flag. The trap handler code can then set the relevant cumulative exception bit if desired.

Normally, reserved FPSR bits should not be altered by user code. However, they can be initialized to zero.

18.4.3 System Control Byte



These control bits determine which features of the floating-point system are in use. Because these control bits are in the FPSR, their state will be preserved across context switches, allowing different processes to use different features if necessary. The following five control bits are defined for the FPA system:

Bit 8 ND - No Denormalized Numbers Bit

If this bit is set, the software forces all denormalized numbers to zero to reduce lengthy execution times when dealing with denormalized numbers. (Also known as abrupt underflow or flush to zero.) This mode is not IEEE-compatible but may be required by some programs for performance reasons. If this bit is clear, then denormalized numbers will be handled in the normal IEEE-conformant way.

Bit 9 NE - NaN Exception Bit

When this bit is clear, extended format is regarded as an internal format for conversions of signalling NaNs: only conversions between single and double-precision will produce an invalid operation exception because of a signalling NaN operand. This is required for compatibility with old programs which use STFE and LDFE to preserve register contents. When the NE bit is set, all conversions between single, double and extended precision will produce an invalid operation exception if the operand is a signalling NaN.

Bit 10 SO - Select Synchronous Operation of FPA

If this bit is set, all floating-point instructions will execute synchronously and ARM will be made to busy-wait until the instruction has completed. This will allow precise exceptions to be reported but at the expense of increased execution time. If this bit is clear, the class of floating-point instructions that can execute asynchronously to ARM will do so. Exceptions that occur as a result of these instructions may then be imprecise.

Bit 11 EP - Use Expanded Packed Decimal Format

If this bit is set, the expanded (four word) format will be used for Packed Decimal numbers. Use of this expanded format allows conversion from extended precision to packed decimal and back again to be carried out without loss of accuracy. If this bit is clear, standard (three word) format is used for Packed Decimal numbers.

Bit 12 AC - Use Alternative definition for C-flag on compare operations

If this bit is set, the ARM C-flag has the following interpretation after a compare:

```
C: Greater Than or Equal or Unordered
```

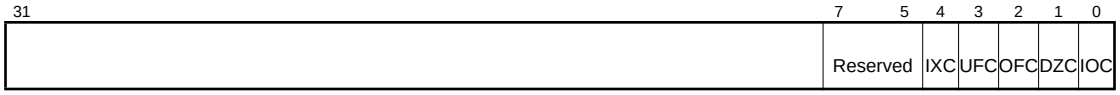
This interpretation of the C-flag allows more of the IEEE predicates to be tested by means of single ARM conditional instructions than is possible using the original interpretation of the C-flag as shown below.

If this bit is clear, the ARM C-flag has the following interpretation after a compare:

```
C: Greater Than or Equal
```

Normally, reserved FPSR bits should not be altered by user code. However, they may be initialized to zero.

18.4.4 Cumulative Exception Flags Byte



Whenever an exception condition arises and the corresponding trap enable bit is not set, the appropriate cumulative exception flag in bits 0–4 are set to ‘1’. If the relevant trap enable bit is set, an exception is delivered to the user's program in a manner specific to the operating system.

NOTE: In the case of underflow, the state of the trap enable bit determines under which conditions the underflow exception will arise.

These flags can only be cleared by a WFS instruction.

Normally, reserved FPSR bits should not be altered by user code. However, they may be initialized to zero.

IO -– Invalid Operation

The invalid operation exception arises when an operand is invalid for the operation to be performed. The result (if the trap is not enabled) is a quiet NaN.

Invalid operations are:

- Any operation on a signalling NaN, except an LDF, LFM or SFM, or an MVF, MNF, ABS or STF without change of precision.
- Magnitude subtraction of infinities, for example, + infinity + –infinity.
- Multiplication of 0 by an infinity.

- Division of 0/0 or infinity/infinity.
- x REM y where x is infinity or y is 0.
- Square root of any number less than zero (but SQT(–0) is –0).
- Conversion to integer when overflow, infinity or NaN make it impossible. If overflow makes a conversion to integer impossible, the largest positive or negative integer is produced (depending on the sign of the operand) and Invalid Operation is signalled.
- CMFE, CNFE when at least one operand is a NaN.

DZ – Division by Zero

The division-by-zero exception occurs if the divisor is zero and the dividend a finite, non-zero number. A correctly-signed infinity is returned if the trap is disabled.

OF – Overflow

The OFC flag is set whenever the destination format's largest number is exceeded in magnitude by what would have been the rounded result if the exponent range were unbounded. The untrapped result returned is either:

- the correctly signed infinity
- the format's largest finite number

depending on the rounding mode.

UF - Underflow

Two correlated events contribute to underflow:

1) Tininess
The creation of a tiny non-zero result smaller in magnitude than the format's smallest normalized number.

2) Loss of accuracy
A loss of accuracy due to denormalization that *may* be greater than would be caused by rounding alone.

If the underflow trap enable bit is set, the underflow exception occurs when tininess is detected, regardless of loss of accuracy. If the trap is disabled, then tininess and loss of accuracy must both be detected for the underflow flag to be set (in which case inexact will also be signalled).

IX – Inexact

The inexact exception occurs if:

- the rounded result of an operation is not exact (different from the value computable with infinite precision)
- overflow has occurred while the OFE trap was disabled
- underflow has occurred while the UFE trap was disabled.

OFE or UFE traps take precedence over IXE.

18.5 The Floating-Point Control Register (FPCR)

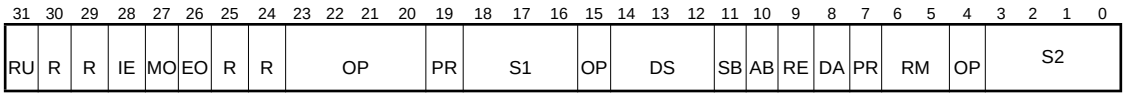
The Floating-Point Control register (FPCR) is an implementation-specific register: it may not exist in some versions of the ARM floating-point system and, when it does exist, it may contain different information for different versions of the system.

When present, it is used for internal communication within the floating point system and, in particular, to allow software and hardware components of the system to communicate with each other.

Use of the WFC and RFC instructions outside the floating-point system itself is strongly discouraged. In the case of User mode programs, it is actually prohibited: the WFC and RFC instructions will trap if executed in User mode.

The FPCR within the CL-PS7500FE has an FPCR. It enables and disables the device and communicates information about instructions the hardware cannot handle to the support code.

The FPA FPCR bit allocation is as follows:



31	RU	Rounded-up bit
30		Reserved
29		Reserved
28	IE	Inexact bit
27	MO	Mantissa overflow
26	EO	Exponent overflow
25		Reserved
24		Reserved
23:20	OP	AU operation code
19, 7	PR	AU precision
18:16	S1	AU source register 1
15	OP	AU operation code
14:12	DS	AU destination register
11	SB	Store bounce: decode (R14) to get opcode
10	AB	Arithmetic bounce: opcode supplied in rest of word
9	RE	Rounding Exception: Arithmetic bounce occurred during rounding stage and destination register was written
8	DA	Disable FPA
6:5	RM	AU rounding mode
4	OP	AU operation code
3:0	S2	AU source register 2 (bit 3 set denotes a constant)

All defined bits are cleared on reset, except bits 8, 10, and 11 (DA, AB, and SB) which are set.

Apart from by using the WFC instruction, the AB bit can only be set by the arithmetic unit and the SB bit can only be set by the load-store unit.

Only the arithmetic unit can write bits 31, 28:26, 23:12, 9, 7:0 of the FPCR.

The behavior of the FPCR when the RFC and WFC instructions are executed is as follows:

- A read of the FPCR by RFC clears the SB, AB and DA bits of the FPCR, and leaves the other bits of the FPCR unchanged.
- A write of the FPCR by WFC writes the SB, AB, & DA bits of the FPCR, and leaves the other bits of the FPCR unchanged.

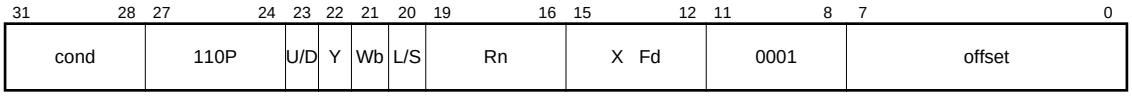
NOTE: This information about the FPCR in the FPA is only supplied to aid with modifications to the FPA support code. Using it for any other purpose is likely to lead to compatibility problems and is strongly discouraged.

19. FLOATING-POINT INSTRUCTION SET

NOTE: Not all of the instructions detailed in this chapter are implemented in hardware on the FPA; the remainder are supported by software emulation. Details of the generic ARM coprocessor instruction set can be found in Chapter 4.

19.1 Coprocessor Data Transfer

19.1.1 LDF/STF — Load and Store Floating



Load or store the high-precision value from or to memory, using one of the five memory formats.

On store, the value is rounded using the *round to nearest* rounding method to the destination precision, or is precise if the destination has sufficient precision. Thus, other rounding methods may be used by having applied a suitable floating-point data operation at some time before the store; this does not compromise the requirement of *rounding once only* since no additional rounding error is introduced by the store instruction.

Cond	condition field
P	pre/post-indexing bit:
	0 post-indexing
	1 pre-indexing
U/D	up/down bit
	0 down
	1 up
Y	transfer length (see below)
Wb	write-back bit
L/S	load/store bit
	0 store to memory
	1 load from memory
Rn	base register
X	transfer length (see below)
Fd	floating-point register number
offset	unsigned 8-bit immediate offset

The length field is encoded into bits 22 and 15 as shown in Table 19-1.

Table 19-1. Length Field

Precision		Bit 22	Bit 15	FPSR.EP	Data Format Size	Note
Single	S	0	0	x	1 memory word	
Double	D	0	1	x	2 memory word	
Extended	E	1	0	x	3 memory word	
Packed decimal	P	1	1	0	3 memory word	a
Expanded packed decimal	EP	1	1	1	4 memory word	a

a LDFP and STFP are deprecated instructions and are intended for backwards compatibility only. These functions should be implemented by appropriate calls to a C library.

The offset in bits [7:0] is specified in words and is added to (U/D = 1) or subtracted from (U/D = 0) a base register (Rn), either before (P = 1) or after (P = 0) the base is used as the transfer address. The modified base value may be written back into the base register (Wb = 1) or the old value of the base may be preserved (Wb = 0).

NOTE: Post-indexed addressing modes require explicit setting of the Wb bit, unlike LDR and STR that always write-back when post-indexed. The value of the base register, modified by the offset in a pre-indexed instruction, is used as the address for the transfer of the first word. The second word (if more than one is transferred) goes to or comes from an address one word (4 bytes) higher than the first transfer, and the address increments by one word for each subsequent transfer.

19.1.2 Assembler Syntax

```
<LDF|STF>{cond}<S|D|E|P> Fd,[Rn]
             [Rn,<#expression>]{!}
             [Rn],<#expression>
```

Pre-indexed Addressing Specification

`[Rn]`	offset of zero
`[Rn, #<expression>]{!}`	offset of <expression> bytes
`{!}`	Write back the base register (set the Wb bit) if ! is present.

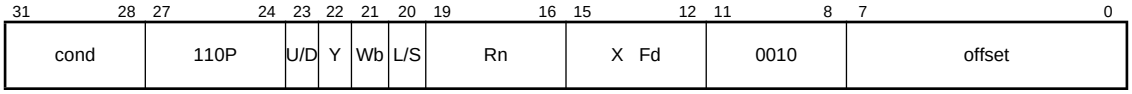
NOTE: If Rn is R15, writeback should not be specified.

Post-indexed Addressing Specification

`[Rn],#<expression>`	offset of <expression> bytes

NOTE: The assembler automatically sets the Wb bit in this case. R15 should not be used as the base register where post-indexed addressing is used. The `<expression>` must be divisible by 4 and be in the range –1020 to 1020.

19.1.3 Load and Store Multiple Floating Instructions



The Load/Store Multiple Floating instructions allow between one and four floating-point registers to be transferred from/to memory in a single operation. These operations allow groups of registers to be saved and restored efficiently (for example, across context switches).

Cond	Condition field
P	Pre/post-indexing bit:
	0 post-indexing
	1 pre-indexing
U/D	Up/down bit:
	0 down
	1 up
Y	Register count (see below)
Wb	Write-back bit
L/S	Load/store bit
	0 store to memory
	1 load from memory
Rn	Base register
X	Register count (see below)
Fd	Floating-point register number offset – unsigned 8-bit immediate offset

The values are transferred as three words of data for each register (the data format used is not defined (and may change in future implementations) and the only legal operation that can be performed on this data is to load it back into the FPA using the same implementation's LFM instruction. The data stored in memory by an SFM instruction should not be used or modified by any user process.

NOTE: Coprocessor number 2 (bits 11:8 in the instruction field) rather than the usual FPA coprocessor number of 1 must be used for these instructions.

The offset in bits [7:0] is specified in words and is added to (U/D = 1) or subtracted from (U/D = 0) a base register (Rn), either before (P = 1) or after (P = 0) the base is used as the transfer address. The modified base value may be written back into the base register (Wb = 1) or the old value of the base may be preserved (Wb = 0). Note that post-indexed addressing modes require explicit setting of the Wb bit, unlike LDR and STR that always write-back when post-indexed. The value of the base register, modified by the offset in a pre-indexed instruction, is used as the address for the transfer of the first word. The second word goes to or comes from an address one word (4 bytes) higher than the first transfer, and the address increments by one word for each subsequent transfer.

19.1.4 Assembler Syntax — Form 1

```
<LFM|SFM>{cond} Fd,<count>,[Rn]
                [Rn, #<expression>]{!}
                [Rn],#<expression>
```

The first register to transfer is specified as Fd.

The number of registers to transfer is specified in the <count> field and is encoded in bit 22 and bit 15 as shown in Table 19-2.

Table 19-2. Count Field

Bit 22	Bit 15	No. of Registers to Transfer
0	1	1
1	0	2
1	1	3
0	0	4

Registers are always transferred in ascending order and wrap around at register F7. For example:

```
SFM F6,4,[R0]
```

transfers F6, F7, F0, and F1 to memory starting at the address contained in register R0.

Pre-Indexed Addressing Specification

`[Rn]`	offset of zero
`[Rn, #<expression>]{!}`	offset of <expression> bytes
`{!}`	Write back the base register (set the Wb bit) if ! is present.

NOTE: If Rn is R15, writeback should not be specified.

Post-Indexed Addressing Specification

`[Rn],#<expression>`	offset of <expression> bytes

NOTE: The assembler automatically sets the Wb bit in this case. R15 should not be used as the base register where post-indexed addressing is used.

The `<expression>` *must be divisible by 4 and be in the range –1020 to 1020.*

19.1.5 Assembler Syntax — Form 2

```
<LFM|SFM>{cond}<FD,EA> Fd,<count>,[Rn]{!}
```

This form of the instruction is intended for stacking type operations on the floating-point registers. Table 19-3 shows how the assembler mnemonics translate into bits in the instruction.

Table 19-3. Assembler Mnemonics

Name	Stack	L Bit	P Bit	U Bit
Post-increment load	LFMFD	1	0	1
Pre-decrement load	LFMEA	1	1	0
Post-increment store	SFMEA	0	0	1
Pre-decrement store	SFMFD	0	1	0

FD and EA define pre/post indexing and the up/down bit by reference to the form of stack required. The F and E refer to a full or empty stack (that is, if a pre-index has to be done (full) before storing to the stack).

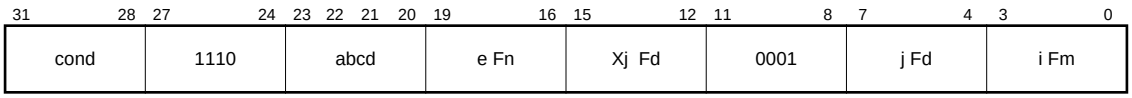
The A and D refer to whether the stack is ascending or descending. If ascending, an SFM goes up and LFM down; if descending, vice-versa.

NOTE: Only EA and FD are permitted: the LFM/SFM instructions are not capable of supporting empty descending or full ascending stacks.

`{!}` Write back the base register (set the Wb bit) if ! is present.

If Rn is R15, writeback should not be specified.

19.2 Coprocessor Data Operations



where:

abcd	opcode
j	dyadic/monadic:
	0 dyadic
	1 monadic
ef	destination size
gh	rounding mode
i	constant /Fm

19.2.1 Dyadic Operations

```
<ADF|SUF|RSF|MUF|OVF|RDF|>{cond}<S|D|E>{P|M|Z} Fd, Fn, <Fm|#value>
<FML|FDV|FRD|RMF>
```

19.2.2 Monadic Operations

```
<ABS|URD|NRM|MVF|MNF|SQT|RND>{cond}<S|D|E>{P|M|Z} Fd, Fn, <Fm|#value>
```

19.2.3 Library Calls

It is recommended that the following floating-point operators are implemented with calls to an appropriate library (for example, the C library):

- Power
- Reverse power
- Polar angle
- Logarithm base 10
- Logarithm base 0
- Exponent
- Sine
- Cosine
- Tangent
- Arc sine
- Arc cosine
- Arc tangent

19.2.4 Backwards Compatibility

For backwards compatibility with existing floating-point code, the following floating-point mnemonics are defined in the ARM floating-point instruction set. These opcodes are treated by the FPA as undefined instructions, and must be handled by support code, less efficient than using library calls.

```
<POW|RPW|POL>
<LOG|LGN|EXP|SIN|COS|TAN|ASN|ACS|ATN>
```

abcdj	Mnemonic	Description	Operation	Note
00000	ADF	Add	Fd := Fn + Fm	
00010	MUF	Multiply	Fd := Fn × Fm	
00100	SUF	Subtract	Fd := Fn – Fm	
00110	RSF	Reverse Subtract	Fd := Fm – Fn	
01000	DVF	Divide	Fd := Fn ÷ Fm	
01010	RDF	Reverse Divide	Fd := Fm ÷ Fn	
01100	POW	Power	Fd := Fn raised to the power of Fm	1
01110	RPW	Reverse Power	Fd := Fm raised to the power of Fn	1
10000	RMF	Remainder	Fd := IEEE remainder of Fn / Fm	
10010	FML	Fast Multiply	Fd := Fn × Fm	

abcdj	Mnemonic	Description	Operation	Note
10100	FDV	Fast Divide	Fd := Fn ÷ Fm	
10110	FRD	Fast Reverse Divide	Fd := Fm ÷ Fn	
11000	POL	Polar angle (ArcTan2)	Fd := polar angle of (Fn, Fm)	1
11010	---	trap: undefined instruction		
11100	---	trap: undefined instruction		
11110	---	trap: undefined instruction		
00001	MVF	Move	Fd := Fm	
00011	MNF	Move Negated	Fd := - Fm	
00101	ABS	Absolute value	Fd := ABS (Fm)	
00111	RND	Round to integral value	Fd := integer value of Fm	
01001	SQT	Square root	Fd := square root of Fm	
01011	LOG	Logarithm to base 10	Fd := $\log_{10}$ of Fm	1
01101	LGN	Logarithm to base e	Fd := $\log_{e}$ of Fm	1
01111	EXP	Exponent	Fd := e ** Fm	1
10001	SIN	Sine	Fd := sine of Fm	1
10011	COS	Cosine	Fd := cosine of Fm	1
10101	TAN	Tangent	Fd := tangent of Fm	1
10111	ASN	Arc Sine	Fd := arcsine of Fm	1
11001	ACS	Arc Cosine	Fd := arccosine of Fm	1
11011	ATN	Arc Tangent	Fd := arctangent of Fm	1
11101	URD	Unnormalized Round	Fd := integer value of Fm, possibly in abnormal form	
11111	NRM	Normalize	Fd := normalized form of Fm	

ef	Suffix	Destination Rounding Precision	Note
00	S	IEEE Single precision	2
01	D	IEEE Double precision	2
10	E	IEEE Double Extended precision	2
11		trap: undefined instruction	

gh	suffix	Rounding Mode
00		Round to Nearest (default)
01	P	Round towards Plus Infinity
10	M	Round towards Minus Infinity
11	Z	Round towards Zero

NOTES:

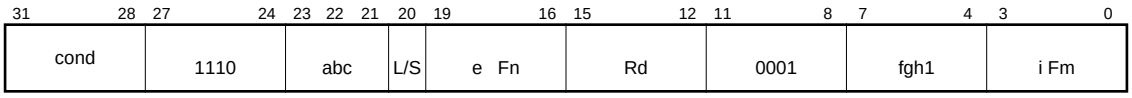
1) Deprecated instruction: included for backwards compatibility only.
2) The precision must be specified; there is no default.
3) These are specified when i = 1.

i Fm	Value Assigned	Note
1000	0.0	3
1001	1.0	3
1010	2.0	3
1011	3.0	3
1100	4.0	3
1101	5.0	3
1110	0.5	3
1111	10.0	3

NOTES:

1) FML, FRD, and FDV are only defined to work with single precision operands. It is not guaranteed that any particular implementation executes the ‘fast’ instructions any quicker than their respective ‘normal’ versions (MUF, DVF, and RDF).
2) Directed rounding is done only at the last stage of a SIN, COS, and so on; the intermediate calculations to compute the value are done with round-to-nearest using the full working precision.
3) The URD instruction performs the IEEE round-to-integer-value operation, but can leave its result in an abnormal unnormalized form. The NRM instruction converts this abnormal result into a proper floating-point value.
4) Direct use of the result of a URD instruction by any instruction other than NRM can produce unexpected results and should therefore not be done. However, there is an exception to this rule, where a URD result may safely be preserved and restored by STFE/LDFE or SFM/LFM before being processed by NRM. So there is no need, for instance, to disable interrupts around a URD/NRM instruction sequence.
5) Similarly, the NRM instruction should only be used on an URD result. Again, use of it on other values can produce unexpected results.

19.3 Coprocessor Register Transfer




```
FLT{cond}<S|D|E>{P|M|Z}  Fn, Rd
FIX{cond}{P|M|Z}         Rd, Fm
<WFS|RFS|WFC|RFC>{cond}  Rd
```

When L/S is:

1	the transfer is *to* an ARM register
0	the transfer is *from* an ARM register

abc L/S	Mnemonic	Description	Operation	Note
0000	FLT	Convert integer to floating point	Fn : = Rd	
0001	FIX	Convert floating point to integer	Rd : = Fm	
0010	WFS	Write Floating Point Status register	FPSR : = Rd	
0011	RFS	Read Floating Point Status register	Rd : = FPSR	
0100	WFC	Write Floating Point Control register	FPCR: = Rd	1
0101	RFC	Read Floating Point Control register	Rd : = FPCR	1
011x		trap: undefined instruction		
1000		trap: undefined instruction		
1010		trap: undefined instruction		
1100		trap: undefined instruction		
1110		trap: undefined instruction		

NOTES:

1) Supervisor-only instructions.

Definition of the efgh Bits

The definition of the efgh bits is instruction-dependent:

FLT

ef	destination size (Section 19.2)
gh	rounding mode (Section 19.2)

FIX

ef	these bits are reserved and should be zero.
gh	rounding mode (Section 19.2)

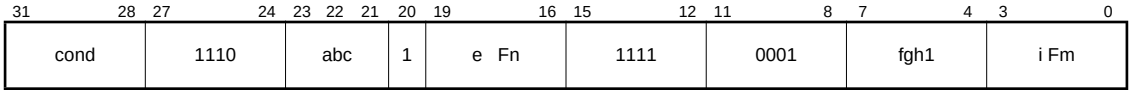
WFS,RFS,WFC, and RFC

efgh these bits are reserved and should be zero.

Constants

Constants cannot be specified in the Fm field for the FIX instruction, as there is no point FIXing a known value into an ARM integer register; it would be quicker to use a MOV instruction.

19.3.1 Compare Operations



NOTE: These are special cases of the general CPRT instruction, with Rd = 15 and L/S = 1.

```
<CMF|CNF|CMFE|CNFE>{cond}  Fn, Fm
```

abc operation

i constant ROM/Fm (see Section 19.2)

efgh are reserved and should be '0'

abc	Mnemonic	Description	Operation
100	CMF	Compare floating	compare Fn with Fm
101	CNF	Compare negated floating	compare Fn with -Fm
110	CMFE	Compare floating with exception	compare Fn with Fm
111	CNFE	Compare negated floating with exception	compare Fn with -Fm

Compares

Compares are provided with and without the exception that could arise if the numbers are unordered. When testing IEEE predicates, the CMF instruction should be used to test for equality (that is, when a BEQ or BNE is used afterwards) or to test for unorderedness (in the V flag). The CMFE instruction should be used for all other tests (BGT, BGE, BLT, and BLE afterwards). CMFE produces an exception if the numbers are unordered (whenever at least one operand is a NaN). CMF only produces an exception when at least one operand is a signalling NaN.

The ARM flags N, Z, C, and V refer to Table 19-4 after compares.

Table 19-4. Flag Settings When the AC Bit in the FPSR is Clear

Flag	Description	Example
N	Less Than	Fn less than Fm (or -Fm)
Z	Equal	
C	Greater Than or Equal	Fn greater than or equal to Fm
V	Unordered	

NOTE: That when two numbers are not equal N and C are not necessarily opposites: if the result is unordered they are both false.

Table 19-5. Flag Settings When the AC Bit in FPSR is Set

Flag	Description
N	Less than
Z	Equal
C	Greater than or equal or unordered
V	Unordered

NOTE: In this case, N and C are necessarily opposites.

19.4 FPA Instruction Set

The FPA and support software together implement the ARM floating point instruction set as defined in the previous section. The FPA itself implements a subset of the instruction set as defined in Table 19-8.

The FPA does not however, execute arithmetic instructions in Section 19.2 if one or more of the operands has one of the following exceptional values (also known as *uncommon values*):

- Infinity
- NaN
- Denormalized
- Illegal extended precision bit patterns

In this case the instruction is 'bounced' to the software support code for emulation.

19.4.1 Infinities and NaNs

Infinities and NaNs should occur very rarely in normal code. Although not common, there are a few 'normal' programs that frequently underflow and produce denormalized numbers. Handling of denormalized operands in software can cause performance degradation. If necessary, this performance degradation can be minimized by setting a bit in the Status register that disables support for denormalized numbers.

19.4.2 Exceptional Conditions

Certain other exceptional conditions that arise during an operation causes the FPA to transfer that operation to the support code. These conditions include all cases of the following IEEE exceptions:

- Invalid operation
- Division by zero
- Overflow
- Underflow
- Inexact

If the Inexact condition is detected, operation is only transferred to the support code if the Inexact trap enable bit is set in the Floating Point Status register. Some other rare cases (such as, mantissa overflow that occurs during the rounding stage of a Store Floating instruction) that do not in fact produce an IEEE exception also traps to the support software.

Table 19-6. Instructions Implemented in FPA

Mnemonic	Instructions	IEEE Required
LDF(S/D/E)	Load (Single/Double/Extended)	✓
STF(S/D/E)	Store (Single/Double/Extended)	✓
ADF	Add	✓
SUF	Subtract	✓
RSF	Reverse Subtract	
MUF	Multiply	✓
DVF	Divide	✓
RDF	Reverse Divide	
FML	Fast Multiply	
FDV	Fast Divide	
FRD	Fast Reverse Divide	
ABS	Absolute	
URD	Round to Integral Value, possibly producing abnormal value	
NRM	Normalize result of URD	
MVF	Move	✓
MNF	Move Negated	
FLT	Integer to floating point conversion	✓
FIX	Floating point to integer conversion	✓
WFS	Write Floating Point Status	✓
RFS	Read Floating Point Status	✓
WFC	Write Floating Point Control	
RFC	Read Floating Point Control	
CMF	Compare Floating	✓
CNF	Compare Negated Floating	
CMFE	Compare Floating with Exception	✓
CNFE	Compare Negated Floating with Exception	
LFM	Load Floating Multiple (new to FPA)	
SFM	Store Floating Multiple (new to FPA)	

Table 19-7. Instructions Supported by Software Support Code (FPASC)

Mnemonic	Instructions	IEEE Required
SQT	Square root	✓
RMF	Remainder	✓
RND	Round to integral value	✓

19.5 Floating-point Support Code

Software support for the FPA includes the FPA support code (FPASC) and a software-only floating-point emulator (FPE).

The FPA system and the FPE produce identical results; both systems are fully IEEE-conformant. Both systems seamlessly implement the ARM floating-point instruction set.

The purpose of the FPASC is to:

1) Emulate in software those instructions rejected by the FPA because they involve uncommon values.

2) Provide support for exception conditions reported by the FPA.

3) Emulate in software those instructions in the floating point instruction set that are not implemented in the FPA (see list above).

4) Emulate in software any instructions that are included for backwards compatibility only; see Section 19.2.4.

19.5.1 IEEE Standard Conformance

The full name of the IEEE Floating Point Standard is as follows:

"IEEE Standard for Binary Floating Point Arithmetic - ANSI/IEEE Std 754-1985"

This is referred to as the IEEE standard or merely as IEEE in this data book.

NOTE: The FPA hardware on its own is not IEEE-conformant.

Support software (the FPASC - FPA Support Code) is required to:

1) Implement the IEEE-required operations not provided by the FPA.

2) Handle operations on uncommon values bounced by the FPA.

3) Provide exception trap-handling capability.

19.6 Instruction Cycle Timing

The following table shows the number of cycles that FPA takes in executing each instruction. Two numbers are given:

- The instruction latency
- The maximum instruction throughput

Table 19-8. FPA Cycle Time

Instruction	Precision	No. Registers	Throughput	Latency	Note
LDF/STF	S		2	3	
LDF/STF	D		3	4	
LDF/STF	E		4	5	
LFM/SFM		1	4	5	
LFM/SFM		2	7	8	
LFM/SFM		3	10	11	
LFM/SFM		4	13	14	
MVF/MNF/ABS	S/D/E		1	2	1
ADF/SUF/RSF/URD/NRM	S/D/E		2	4	
MUF	S/D/E		8	9	
FML	S/D/E		5	6	
DVF/RDF/FDV/FRD	S		30	31	2
DVF/RDF/FDV/FRD	D		58	59	2
DVF/RDF/FDV/FRD	E		70	71	2
FLT	S/D/E		6	8	
FIX			8	9	
CMF/CMFE/CNF/CNFE			5	6	
RFS/RFC			3	4	3
WFS/WFC			3	3	

NOTES:

1) Cannot be sustained for more than 2 cycles out of every 3 cycles.
2) May be less if the division comes out exactly, causing *early termination* of the division algorithm (minimum of 6 cycles throughput, 7 cycles latency).
3) May be 2 or 3 cycles, depending on the previous instruction.

Throughput

Throughput is the number of cycles between the start of an instruction and the start of a succeeding instruction of the same type, both instructions occurring in a long sequence of instructions of the same type; repeated use of the same register may only occur in a sequence of length greater than or equal to 8.

Latency

Latency is the number of cycles between the start of instruction execution and its completion. The number of cycles taken by a sequence of floating point instructions, each depends on the result of the preceding instruction in the sequence, can generally be found by adding the latencies of the individual instructions. There can be minor discrepancies from this rule for particular sequences.

The exact definition is dependent on the type of instruction being executed:

Arithmetic instructions	From register read to register write.
LDF, LFM, FLT	From start of instruction arbitration to register write.
STF, SFM, CMF, FIX	From register read to start of next instruction arbitration.
WFS, WFC	From start of instruction arbitration until the next instruction would be deemed to start by these rules.
RFS, RFC	From the time that the previous instruction would be deemed to end by these rules to the start of the next instruction arbitration.

NOTE: Speculative execution, concurrent execution between arithmetic and load/store instructions and concurrent execution between ARM integer instruction and FPA instructions can significantly reduce the effective timings shown.

19.6.1 Instruction Classification

Instructions can be classified into arithmetic, load/store, and joint instructions:

Arithmetic	Those instructions that execute completely within the arithmetic unit. These include all the hardware-implemented coprocessor data operations (see Section 19.2).
Load/store	Those instructions that execute completely within the load/store unit. These include LDF, STF, LFM and SFM.

Joint Arithmetic and Load/Store Instructions

FIX, CMF,CNF,CMFE,CNFE	Arithmetic followed by load/store.
FLT	Load/store followed by arithmetic.
WFS,RFS,WFC,RFC	Occupy both arithmetic and load/store units, since the arithmetic unit must be empty before any of these instructions may be executed.

19.6.2 Performance Tuning

The FPA is capable of executing load/store and arithmetic instructions concurrently and is also capable of executing instructions speculatively (that is, before they have been committed to execution by the ARM CPU). Both of these features can be exploited to maximize the performance of the FPA. The code fragment shown below is an example of how to achieve this:

```
1   SFM  F0,4,[R0],#48
2   DVFS F0,F1,#3
3   SFM  F4,4,[R0],#48
4   MOV  R1,R2
5   MOV  R3,R4
```

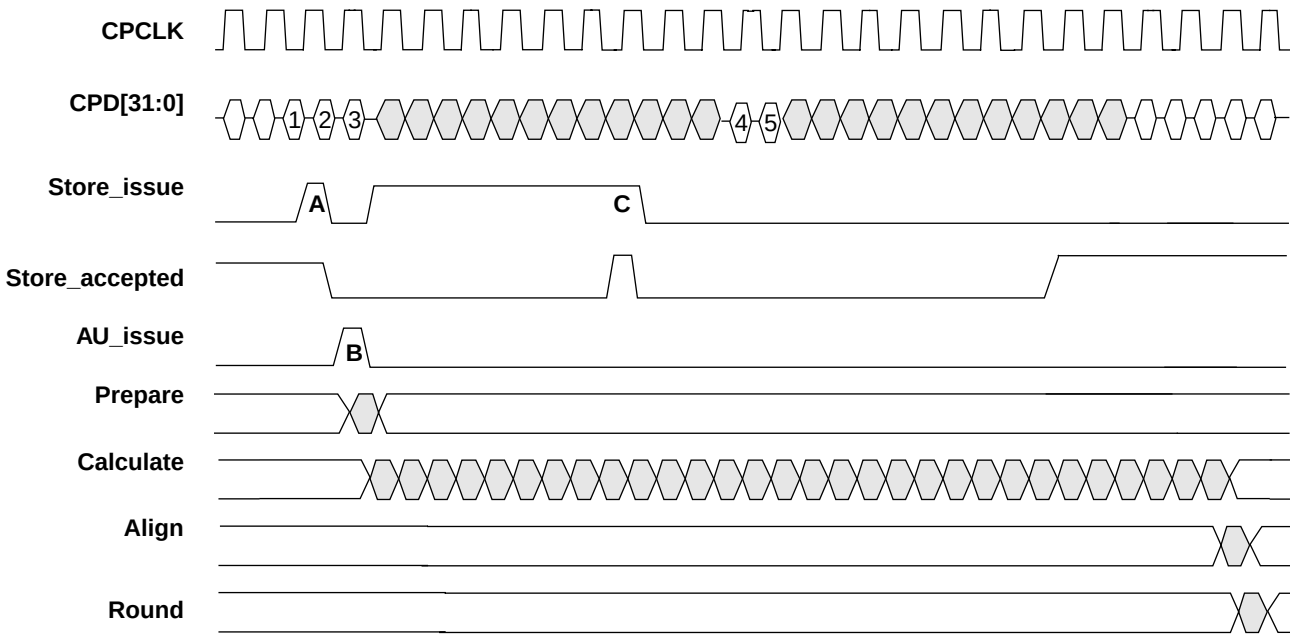


Figure 19-1. Performance Tuning

The labels 1, 2, 3, 4, and 5 indicate the cycles when these instructions are fetched on the CPD[31:0] bus, while A, B, and C indicate the cycles when the floating-point instructions are issued to their respective units in the FPA.

The first store multiple instruction (1) is issued (A) to the load/store unit, resulting in 12 words of data being transferred on CPD[31:0] as shown by the shaded boxes on the timing diagram. Meanwhile, the divide instruction (2) is issued (B) to the arithmetic unit (AU) that then begins execution speculatively; its progress through the Prepare, Calculate, Align and Round stages of the AU pipeline is shown by the shaded boxes on the timing diagram.

The second SFM instruction (3) is issued (C) to the load/store unit as soon as it is ready. This second SFM then executes while the AU is still busy on the divide instruction; the second set of shaded boxes on the CPD[31:0] bus indicates the 12 words of data being transferred for the second SFM instruction. This example shows how the divide instruction's execution time can effectively be hidden by other instructions.

NOTE: The concurrence between ARM integer unit execution and FPA execution can also be exploited. Contact ARM Ltd. for further details on optimizing floating-point code for the FPA.

20. BUS INTERFACE

20.1 Bus Arbitration

Arbitration for the main CL-PS7500FE data bus is carried out with the priorities shown below:

1) Video/cursor DMA

2) Sound DMA

3) DRAM refresh

4) ARM processor memory cycles

As the CL-PS7500FE contains a cached processor, ARM internal cycles can continue while DMA is in progress, but the CPU stalls when it suffers a cache miss and requires to fill a cache line from memory.

Once an external memory cycle has started, DMA has to wait until it is completed. The exception is for I/O reads or writes and SUSPEND mode, where the write data is latched internally at the start of the cycle, then DMA requests can be serviced even though the I/O access or SUSPEND mode is under way. The end of an I/O access is held up until the current DMA access is complete. I/O read data is latched internally when available, and is not enabled onto the CL-PS7500FE data bus until any DMA transfers have completed.

20.2 Bus Cycle Types

There are a large number of different types of cycle that use of the CL-PS7500FE data bus. Except for DMA accesses, the cycle type is decoded according to the address put out by the ARM processor macrocell, and the detailed timing is controlled by the relevant section of the I/O or memory controller subsystem.

The ARM processor supports two basic types of external cycle:

- Non-sequential consists of an idle cycle followed by a memory cycle
- Sequential consists simply of a memory cycle

The idle cycle allows the memory and I/O controller subsystems time to prepare for a new cycle type. These two cycles are used as the basic building block for the more complex I/O and memory access cycle timings generated by the CL-PS7500FE. ARM processor external cycles are clocked by the internal MEMCLK signal generated by the CL-PS7500FE memory controller according to the type of cycle.

Only the latched version of the ARM processor's address is exported from the ARM processor, and this can only change immediately after the falling edge of the internal MEMCLK signal that clocks the ARM for external accesses. The timing diagrams in Chapter 22 include MEMCLK as a reference as it indicates the end of a particular cycle. To save power, the CL-PS7500FE internal data bus is not always exported during internal register programming.

The ARM processor requests an external memory access for a number of reasons:

- A cache line fetch always consists of memory reads from four sequential addresses.
- A Level 1 translation fetch consists of a read from memory followed by the address translation so that the next address output by the ARM is the translated physical address as generated from the read-back section descriptor.
- A Level 2 translation fetch is always preceded by a Level 1 fetch, and returns the page table entry, then used to create the physical address for the next cycle.

External buffered and unbuffered write cycles occur with indistinguishable bus timing. When the ARM needs to read from a location and the data is not in the cache or is uncacheable (for example, for I/O), an external read access is performed.

20.3 Video DMA Bandwidth

The maximum video DMA bandwidth depends on the MEMCLK frequency and the DRAM width (16 or 32 bit), but can be calculated as follows. The length of the non-sequential cycle at the start of a DRAM read varies. Assuming DRAMCR[5] is low:

- In Page mode, each non-sequential cycle takes 5 cycles
- In EDO mode, each non-sequential cycle takes 6 cycles

This increases by one if DRAMCR[5] is high, and again by one or two to preserve RAS precharge times, depending on whether the access just finished was to the same bank as the current one and if DRAMCR[6] is also set.

Assuming Fast Page mode without further non-sequential delays, each qword DMA requires 5 + 2 + 2 + 2 = 11 MEMCLK cycles to complete. It is possible for DMA requests for the video to be serviced sequentially such that the second and subsequent qword DMA bursts take only 2 + 2 + 2 + 2 = 8 MEMCLK cycles each. However, all accesses is broken up at page boundaries (every 256 words). So every 64 DMA bursts, three extra MEMCLK periods are required.

Therefore, at a 32 MHz MEMCLK with a 32-bit-wide DRAM, 64 qwords would be transferred approximately every 16 µs. The maximum theoretical DMA bandwidth is thus 63.6 Mbytes/sec. If a greater video DMA bandwidth is required, a higher MEMCLK frequency must be used. In an actual system, the average bandwidth does not achieve this theoretical maximum.

20.4 Video DMA Latency

DMA latency is defined as the time from the generation of the internal request for more data from the video FIFO in the video macrocell, to the time that the first word of DMA data is clocked into the video macrocell.

There are several possible limiting factors that can determine the worst-case DMA latency. This depends on the CL-PS7500FE memory system configuration. There are three possible limiting cases:

1) Internal register programming cycles.

2) Burst mode ROM accesses, or very long non-sequential ROM accesses.

3) DRAM accesses in 16-bit mode.

The following assumes that the internal MEMRFCK frequency is equal to the MEMCLK frequency (that is, the prescalars are set to divide-by-one). The above cases determine the maximum period before arbitration for DMA occurs in different systems. In addition to the latency resulting from these sequences, the worst-case DMA latency has a possible 5.5 MEMCLK cycles factor for synchronization, such that the synchronized request arrives just too late to be arbitrated for, and CL-PS7500FE commits to a memory cycle. The 5.5 MEMCLK cycles also includes the ARM processor idle cycle where the arbitration (that was just missed) occurs.

From the clock edge where arbitration finally occurs, to the time the first word of DMA data is clocked into the video macrocell, is 5.5 MEMCLK cycles or 7.5 MEMCLK cycles if the preceding access was to DRAM in the same bank. These values assume DRAMCR[7:5] are all set high (that is, EDO memory).

Internal register programming bursts can occur in blocks of up to four before re-arbitration occurs; this takes 16 MEMCLK cycles. Burst mode ROM cycles are re-arbitrated after every four, so are sequential DRAM accesses. Successive non-sequential accesses always allows DMA onto the bus. For this reason, it is unlikely that these accesses are the cause of the worst-case DMA latency. However, it is possible to use the ROM interface in half-speed mode, with the slowest ROM timing and a 16-bit-wide ROM, in which case an access could take 28 internal MEMRFCK cycles. Under these circumstances the ROM interface could be the limiting factor.

To determine the limiting factor in a system, calculate the number of cycles required for a worst case ROM access. The number of cycles for each programmed value in ROMCR is shown below:

For a non-sequential access, programming ROMCR[2:0] to:

'000' – 7 cycles	
'001' – 6 cycles	For all:
'010' – 5 cycles	Multiply by 2 if 16-bit mode set
'011' – 4 cycles	Multiply by 2 if half-speed bit set
'100' – 3 cycles	
'101' – 2 cycles	

If the burst bits (ROMCR[4:3]) are programmed to a value other than '00', then the total worst-case number of cycles is one multiplied by the non-sequential number above, plus three multiplied by the burst number from the following:

'01' – 4 cycles	For all:
'10' – 3 cycles	Multiply by 2 if 16-bit mode set
'11' – 2 cycles	Multiply by 2 if half-speed bit set

At this point calculate the number of cycles required for a worst-case DRAM access. This can be the only limiting factor when 16-bit-wide DRAM is used, and in this case the delay is:

$$9 + (2 \times 7) = 23 \text{ cycles} \qquad \textbf{Equation 20-1}$$

As described above, the worst-case delay for four sequential internal register programming cycles is 16 cycles. Therefore, the worst-case delay is caused by internal register access cycles, ROM or DRAM according to the worst-case results of the above calculations.

DMA can continue over the top of I/O accesses, so these do not feature in the options for worst case delay. So for a system that is limited by internal register access cycles, the worst case latency is:

$$3.5 + 2 + 16 + 5.5 = 27 \text{ MEMCLK cycles} \qquad \textbf{Equation 20-2}$$

So if MEMCLK is running at 32 MHz, the total worst-case DMA latency is 0.84 μs.

As another example, consider a ROM interface where the non-sequential access time is programmed at 7 cycles and the sequential access is programmed to 4 cycles using 16-bit-wide ROM. The total latency is:

$$3.5 + 2 + 14 + 8 + 8 + 8 + 5.5 = 49 \text{ MEMCLK cycles} \quad \textbf{Equation 20-3}$$

At 32 MHz this corresponds to 1.5 µs.

21. CLOCKS, POWER SAVING, AND RESET

21.1 Clock Control

CL-PS7500FE has a clocking scheme designed to allow maximum flexibility for the system designer. There are three main clock inputs:

- CPUCLK — CPU clock, used to generate the ARM processor's FCLK
- MEMCLK — Memory subsystem clock, used to generate the memory system clock, and the ARM processor MCLK
- I_OCLK — I/O system clock, this is fixed at 32 MHz and generates all the fixed-frequency I/O clocks and refresh rates.

21.1.1 Video and Sound Subsystem Clocks

The video subsystem has two separate external clock inputs and includes a phase-locked loop to enable the control of an external VCO.

The pixel clock source can be VCLKI (selected by using an external VCO), HCLK, driven directly in from the HCLK pin, or IOCK32 (also referred to as RCLK), which is the internal I/O subsystem clock and is generated directly from the main I_OCLK input pin as described below. The sound subsystem can be clocked either from IOCK32 generated internally from I_OCLK, or by using an externally generated clock connected to the SCLK pin.

Selection between these various clock sources is described in the video and sound sub-systems section of this data sheet.

21.1.2 I/O Clock Outputs

Four fixed frequency I/O clocks are output by the CL-PS7500FE, all divided down from the fixed frequency input I_OCLK, set to 32 MHz in Divide-by-1 mode. These are:

- CLK16 (16 MHz)
- REF8M (8 MHz)
- CLK8 (An inverted version of REF8M)
- CLK2 (2 MHz)

21.1.3 Synchronous/Asynchronous Mode for the ARM Processor

The ARM processor macrocell can be configured for synchronous or asynchronous mode, under the control of the SnA pin. Synchronous mode can only be used within the CL-PS7500FE if the internal ARM processor clocks, FCLK and MCLK, are of exactly the same fundamental frequency and timing, and in fact when SnA is set high, MEMCLK is routed straight through to drive both FCLK and MCLK, with a suitable delay to ensure the required phase relationship between FCLK and MCLK is held correctly, i.e. CPUCLK is ignored when SnA = 1. If the FCLK frequency is required to be different from the MEMCLK frequency, the SnA pin must be held low, and a suitable frequency applied to CPUCLK.

21.1.4 Clock Prescalars

7	6	5	4	3	2	1	0
X	X	X	X	X	C	M	I

Each of the three main clock inputs CPUCLK, I_OCLK and MEMCLK has a selectable divide by 2 prescalar available within CL-PS7500FE to enable a guaranteed 50:50 mark-space ratio internal clock to be produced using a higher frequency external oscillator. The internal clocks, referred to elsewhere in this data book, are called FCLK, IOCK32, and MEMRFCK, respectively.

At power on reset, all the prescalars is set to divide by 2. Prescaling is controlled by the CLKCTL register at address 0x0320003C, and there is one bit to enable or disable each divide by 2 prescalar as required:

C CPUCLK divide control

M MEMCLK divide control

I I_OCLK divide control

Write bit[2]

- 0 FCLK $\times$ 2 = CPUCLK
- 1 FCLK = CPUCLK

bit[1]

- 0 MEMRFCK $\times$ 2 = MEMCLK
- 1 MEMRFCK = MEMCLK

bit[0]

- 0 IOCK32 $\times$ 2 = I_OCLK
- 1 IOCK32 = I_OCLK

Read return above value

Power on reset

set all to ‘0’, that is, divide by 2 clocks

21.1.5 Clocking Schemes

The simplest mode of operation of the CL-PS7500FE has all three of the main clocks driven by a single 32-MHz oscillator, with the prescalars set to Divide-by-1 mode. However, it is possible to increase the speed of the memory and CPU clocks, noting that if this causes CPUCLK and MEMCLK frequencies to be different, the SnA input must be set low for asynchronous operation. The I_OCLK frequency must remain at 32 MHz (or 64 MHz if the divide by 2 prescalars are enabled).

NOTE: Nearly all timings in this datasheet assume that both I_OCLK and MEMCLK are running at 32 MHz (or 64 MHz with the divide by 2 prescalars on).

Increasing the memory clock frequency allows the system designer to take advantage of faster DRAM memory. The CL-PS7500FE includes full synchronization at the interface between the memory and I/O sub-systems to ensure safe operation under asynchronous conditions.

21.2 Power Management

The CL-PS7500FE includes power management circuitry that greatly enhances its suitability for battery powered portable applications where power consumption is of paramount importance. There are three power management modes:

NORMAL	the default operating condition where all clocks are running and the chip is functioning normally.
SUSPEND	the clocks to the CPU (FCLK and MCLK) are stopped, but all other parts of the chip remain active so DMA can continue and the display can continue to be refreshed. It s also possible to stop some of the external I/O clock outputs to save more power if this can be done safely without causing problems for I/O peripherals connected to these clocks.
STOP	allows all the clocks to the CL-PS7500FE to be stopped, and the whole device then draws only leakage currents if all required registers are appropriately programmed. Outputs are provided from the CL-PS7500FE to enable the oscillator(s) to be powered down, and circuitry to allow the oscillator(s) to cleanly restart using an external RC delay before the clocks inside the CL-PS7500FE are reenabled. Before STOP mode is entered, a number of registers need to be programmed appropriately in the video sub-system, and further details of the full sequence of events required to make most effective use of the power management features can be found in Section 21.2.2.

21.2.1 SUSPEND Mode

Entry into SUSPEND mode is achieved by writing to the register location 0x0320001C. Any value can be used, but the value written to bit 0 determines whether the external I/O output clocks CLK16, CLK8, REF8M and CLK2 are stopped. DMA can continue unaffected, allowing the display and DRAM data to remain refreshed.

Exit from SUSPEND mode is achieved by a falling edge on either of the asynchronous input event pins, nEVENT1 and nEVENT2, or by any enabled interrupt source generating a FIQ or IRQ interrupt for the ARM processor. The assertion of nRESET also causes exit from SUSPEND mode. It is important that the interrupt mask and enable registers are programmed appropriately before SUSPEND mode is entered if it is intended that an interrupt source be used to terminate the power saving mode.

The CPU merely sees SUSPEND mode as a write to a location in the memory and I/O register area. It is unaware of the duration of this write, as both MCLK and FCLK are frozen, and it is a fully static device. The careful use of SUSPEND mode when no CPU operations are required has a significant effect on the device‘s average power consumption. It could be used, for example, between key presses while waiting for more user input. The keyboard controller is still clocked during SUSPEND mode and so is able to generate interrupts that cause the termination of the write cycle and then cause the CPU to take the interrupt exception.

Details of the SUSMODE register (address 0x0320001C) are shown below:

7	6	5	4	3	2	1	0
X	X	X	X	X	X	X	S

S	SUSPEND mode control of external I/O clocks
Write	turn off external I/O clocks when in this mode
	0 turn off
	1 don't turn off
	Enter Suspend mode with MCLK,FCLK,I/O clocks and some internal clocks stopped. DMA continues and instruction completes on either wake-up event, nIRQ or nFIQ.
Read	return above value
Reset	set to zero

21.2.2 STOP Mode

Entry into STOP mode is achieved by writing to the register location 0x0320002C. Any value can be written to the register to enter STOP mode, but the value written appears on the external data bus of the CL-PS7500FE while the chip is in STOP mode. It is therefore recommended that the value 0xFFFFFFFF be written to this register as this indicates that both D[31:0] and LA[28:0] are driven high during STOP mode.

It is very important that all DMA activity is stopped, that all I/O activity is completed, and that the video subsystem is powered down correctly before the STOP mode register is written to. The OSCPOWER output is controlled by the power management circuitry, and is forced low a short time after the write cycle begins. This output can be used to disable the external oscillator(s).

Exit from STOP mode can only be achieved by the use of the asynchronous wake-up event pins nEVENT1 and nEVENT2. When either of these is forced low, a sequence of events are triggered that cause the oscillator(s) to be restarted cleanly.

During STOP mode, a zero is driven out from the OSCDELAY pin, ensuring that an external capacitor forming part of an RC network attached to the OSCDELAY pin remains discharged. As soon as a wake up event occurs the OSCPOWER pin is set high again, and the open drain OSCDELAY pin is allowed to float and becomes an input.

At this point, the external capacitor starts to charge, until the schmitt threshold of the OSCDELAY input is exceeded. From this point, a further two rising edges must be seen on the input clock from the oscillator before the clock is allowed through to the internal CL-PS7500FE circuitry. The component values used in the RC circuit should be chosen to ensure that the oscillator has sufficient time to stabilize before the OSCDELAY input is triggered.

As the video subsystem is inherently dynamic for performance reasons, it is necessary to set it into a special Power down mode before STOP mode is entered. To do this, the video Ext register should be programmed with the data 0xC0000000, the Video Control register should be programmed with the data 0xE00040xx (the last byte depends on the clock source and configuration), and the Sound Control register should be programmed with the data 0xB1000000 (if the sound system is configured for use with the SCLK pin as the clock source. If the sound system is being clocked from the CL-PS7500FE's internal 32-MHz I/O clock, then the register should be programmed with the value 0xB1000001). These actions disable the video data path and ensure the entire macrocell is forced into a static state. To ensure that the

comparators in the A-to-D converters do not consume current, they should be shut down by programming the value 0x00 to the ATODICR register at location 0x032000E0.

CL-PS7500FE includes support for self refresh DRAM, and it is intended that this feature should be used during STOP mode to ensure that DRAM contents are preserved. This DRAM mode is activated by allowing direct software control of the nCAS and nRAS output pins. The SELFREF register (0x032000D4) can be used to directly force the nRAS and nCAS output pins according to the protocol required for a particular DRAM to enter self-refresh mode. This programming must be performed by code executing from ROM.

In STOP mode CL-PS7500FE consumes leakage currents only, and can be held indefinitely without corruption of the internal registers, CPU cache, etc.

21.3 Reset

The CL-PS7500FE has three pins associated with reset. The nPOR pin is intended for use with an external RC delay to generate a power-on-reset pulse when the chip is switched on. The nRESET pin is an open drain I/O pin, intended to generate a 'soft' reset. Both nPOR and nRESET are active low schmitt inputs. The active high RESET pin is a clean reset output, created from the synchronized version of the nRESET input, and is also forced high during nPOR.

A low state on the nPOR input sets the POR bit in the IRQA status register. This bit can later be examined to show that the reset that occurred was an nPOR type rather than nRESET. The POR bit in the IRQA status register is not reset until the POR clear bit in the IRQA request register is written to. nPOR also causes the prescalars on the clock inputs to be set to divide by 2. The nPOR input is passed through a pulse stretcher that ensures that even a short pulse on the input guarantees a full reset of the whole of CL-PS7500FE. During nPOR reset, nCAS is forced low throughout and the nRAS outputs are changed. While nPOR is low, nRESET and ID (both open drain pins) are held low, and an incrementing address value is output on the LA address bus.

A low state on the nRESET input is used to generate a 'soft' reset. This does not set any interrupt flags, and the nRESET low state must exist for longer than 1us to guarantee that it is seen, as it is passed through a synchronizer before being used by the internal circuitry. At the start of the nRESET active period, the whole CL-PS7500FE (including the DRAM refresh state machine and counter) is reset for 1-µs, and for the remaining duration of the nRESET pulse, DRAM refresh takes place at the highest selectable rate. During nRESET, the ARM processor outputs an incrementing address on the LA bus.

22. ELECTRICAL SPECIFICATIONS

22.1 Absolute Maximum Ratings

Specification	Maximum Rating
Ambient temperature while operating (T_A)	0°C to 70°C
Storage temperature	–65°C to 150°C
Voltage on any pin	V_{SS} – 0.5 V to V_{DD} + 0.5 V
Operating power dissipation	2 W (watts)
Power supply voltage	7.0 V
Injection current (latch-up testing)	100 mA

NOTES:

1) System components must be operated within the limits of the absolute maximum ratings. If system components are run at conditions at or outside these limits, the system components can be permanently damaged.
2) Functional operation at or outside any of the conditions indicated in the absolute maximum ratings is not implied.
3) Exposure to absolute maximum rating conditions for extended periods can affect system reliability.

22.2 DC Specifications — Digital Values

In the table below, V_{DD} = 5.0 ± 0.25 V and T_A = 0°C to 70°C, unless otherwise specified.

Symbol	Parameter	40 MHz		56 MHz		Units	Note
		MIN	MAX	MIN	MAX		
V_{DD}	Power Supply Voltage (TTL)	4.75	5.25	4.75	5.25	V	
V_{IHC}	Input High Voltage (CMOS)	0.8 V_{DD}	V_{DD} + 5% V_{DD}	0.8 V_{DD}	V_{DD} + 5% V_{DD}	V	a,b
V_{ILC}	Input Low Voltage (CMOS)	–0.5	0.2 V_{DD}	–0.5	0.2 V_{DD}	V	a, b
V_{ILT}	Input Low Voltage (TTL)	0	0.8	0	0.8	V	a, c
V_{IHT}	Input High Voltage (TTL)	2.2	V_{DD} + 5% V_{DD}	2.2	V_{DD} + 5% V_{DD}	V	a, c
V_{ILS}	Input low voltage (SCHMITT)	0	0.8	0	0.8	V	a, d
V_{IHS}	Input high voltage (SCHMITT)	2.0	V_{DD} + 5% V_{DD}	2.0	V_{DD} + 5% V_{DD}	V	a, d
V_{OHC}	Output high voltage (CMOS)	V_{DD} – 0.8		V_{DD} – 0.8		V	a, e,f
V_{OLC}	Output low voltage (CMOS)		V_{SS} + 0.8		V_{SS} + 0.8	V	a, c, f
I_{IL}	Input low current		–10		–10	µA	
I_{IH}	Input high current		10		10	µA	
I_{OZ}	Output leakage current	–10	10	–10	10	µA	c
C_{IN}	Input capacitance		10		10	pF	d
I_{OH1}	x1 Output high current (Vout = V_{DD} – 0.8 V)		4		4	mA	
I_{OL1}	x1 Output low current (Vout = V_{DD} + 0.8 V)		–4		–4	mA	
I_{OH2}	x2 Output high current (Vout = V_{DD} – 0.8 V)		10		10	mA	
I_{OL2}	x2 Output low current (Vout = V_{DD} + 0.8 V)		–10		–10	mA	
I_{OH3}	x3 Output high current (Vout = V_{DD} – 0.8 V)		20		20	mA	
I_{OL3}	x3 Output low current (Vout = V_{DD} + 0.8 V)		–20		–20	mA	
V_{IHST}	IS input rising voltage threshold		3.53		3.53	V	d
V_{ILST}	IS input falling voltage threshold		1.05		1.05	V	d
ESD	HBM ESD		2		2	kV	g
I_{CC}	At maximum frequency		150		180	mA	

[a] Voltages measured with respect to V_{SS}.

[b] IC – CMOS inputs.

[c] IT – TTL inputs (includes BTZ, TOD, and IT pin types).

[d] IS – CMOS Schmitt inputs (includes ICS and CSOD pin types).

[e] OCZ – Output, CMOS levels, tristateable (includes OCZ, BTZ, TOD, and CSOD pin types).

[f] With 1×, 2×, 3× lead currents.

[g] This does not apply to the video and sound analog pins: VIREF, ROUT, GOUT, and BOUT.

22.3 Derating

The AC timings included with each timing diagram include only the intrinsic delay through the output pads. In order to calculate actual delays when designing the CL-PS7500FE into a system, it is necessary to add the load-dependent element of the output pad delay.

The output pads of CL-PS7500FE are CMOS drivers which exhibit a propagation delay that increases linearly with the increasing capacitance. An *Output derating* figure is provided for each of the three types of output pads, showing the increase in output delay with increasing load capacitance.

Details about which driver is used for which output can be found in Chapter 2.

Derating figures are quoted for rising and falling edges.

Table 22-1. CL-PS7500FE Pad Derating

Label	Pad Type	Rising	Falling	Units
x1	Low drive capability pad	0.18	0.16	ns/pF
x2	Medium drive capability pad	0.061	0.046	ns/pF
x3	High drive capability pad	0.029	0.019	ns/pF

22.4 AC Parameters — List of Timing Figures

22.5 System Reset Timing

Table 22-2. Reset Timing

Symbol	Parameter	40 MHz		56 MHz		Units
		MIN	MAX	MIN	MAX	
t_{pr}	Time for which nPOR must be held low to guarantee a reset	20		20		ns
t_{pre} [a]	Length of internal reset	2	4	2	4	µs
t_{nr} [b, c]	Time for which nRESET must be held low to guarantee reset	2		2		µs
t_{re} [c]	Length of internal reset	2		2		µs
t_{res}	nRESET setup to I_OCLK rising	0		0		ns
t_{reh}	nRESET hold from I_OCLK rising	30		30		ns

[a] t_{pre} = 2 µs if I_OCLK is 64 MHz. t_{pre} is 32 MHz; this reset forces Divide-by-2 mode on the clock inputs.

[b] DMA or writes from the ARM processor prevent nRESET having any effect for their duration. Thus the 'soft' reset cannot break write cycles or cause partial DRAM refresh.

[c] Assuming IOCK32 is 32 MHz.

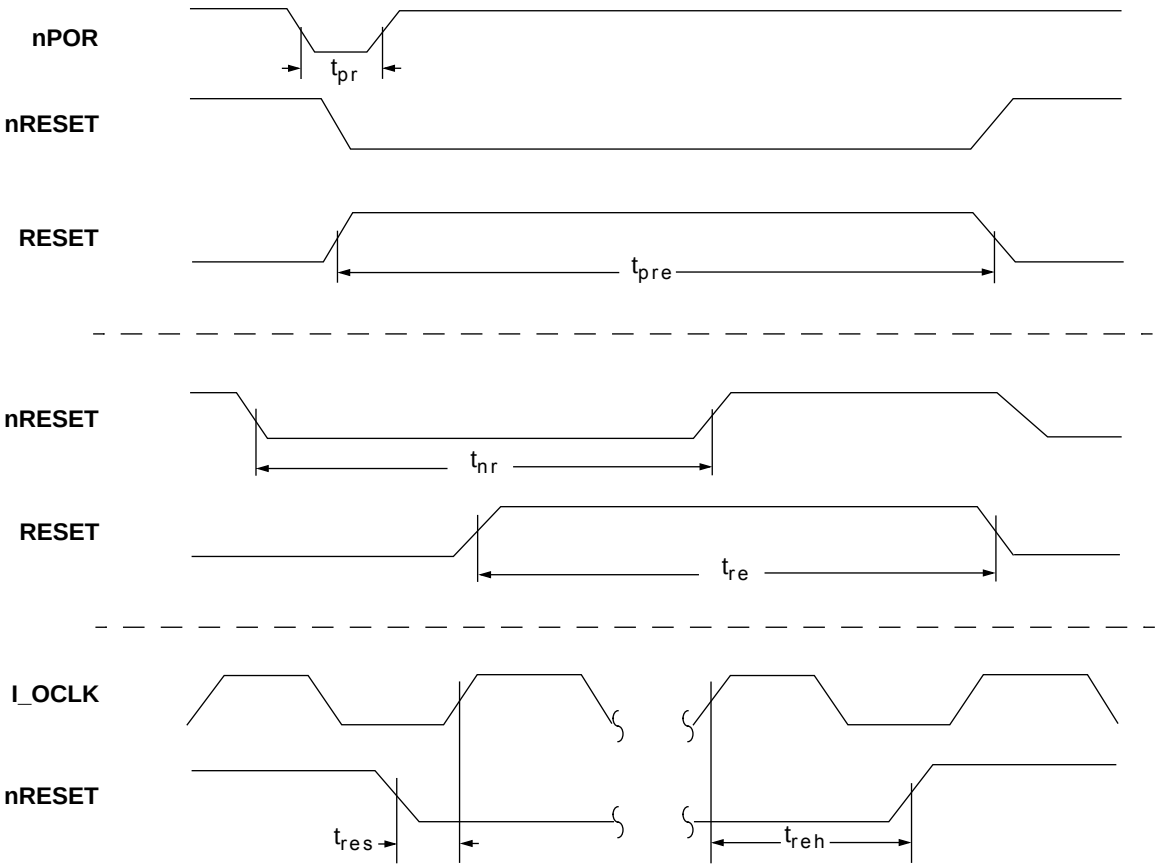


Figure 22-1. System Reset Timing

22.6 Memory Subsystems

Table 22-3. ROM Timing

Symbol	Parameter	40 MHz		56 MHz		Units
		MIN	MAX	MIN	MAX	
t_{la}	MEMCLK rising to LA[28:0] changing		23		23	ns
t_{ds_rom}	Data setup to MEMCLK rising	0		0		
t_{rcsl}	MEMCLK rising to nROMCS falling		18		18	ns
t_{rcsh}	MEMCLK rising to nROMCS rising		20		20	ns
t_{dh_rom}	DATA hold from MEMCLK rising	12		12		ns
t_{rda1}	MEMCLK rising to write DATA valid		22		22	ns
t_{rda2}	MEMCLK rising to write DATA valid		33		33	ns
t_{rda3}	MEMCLK rising to write DATA valid		16		16	ns
t_{rdah}	Write DATA hold after MEMCLK rising	11		11		ns
t_{roel}	MEMCLK rising to nIOR (nOE) falling		14		14	ns
t_{roeh}	MEMCLK rising to nIOR (nOE) rising		14		14	ns
t_{rwel}	MEMCLK rising to nIOR (nOE) falling		14		14	ns
t_{rweh}	MEMCLK rising to nIOR (nOE) rising		13		13	ns

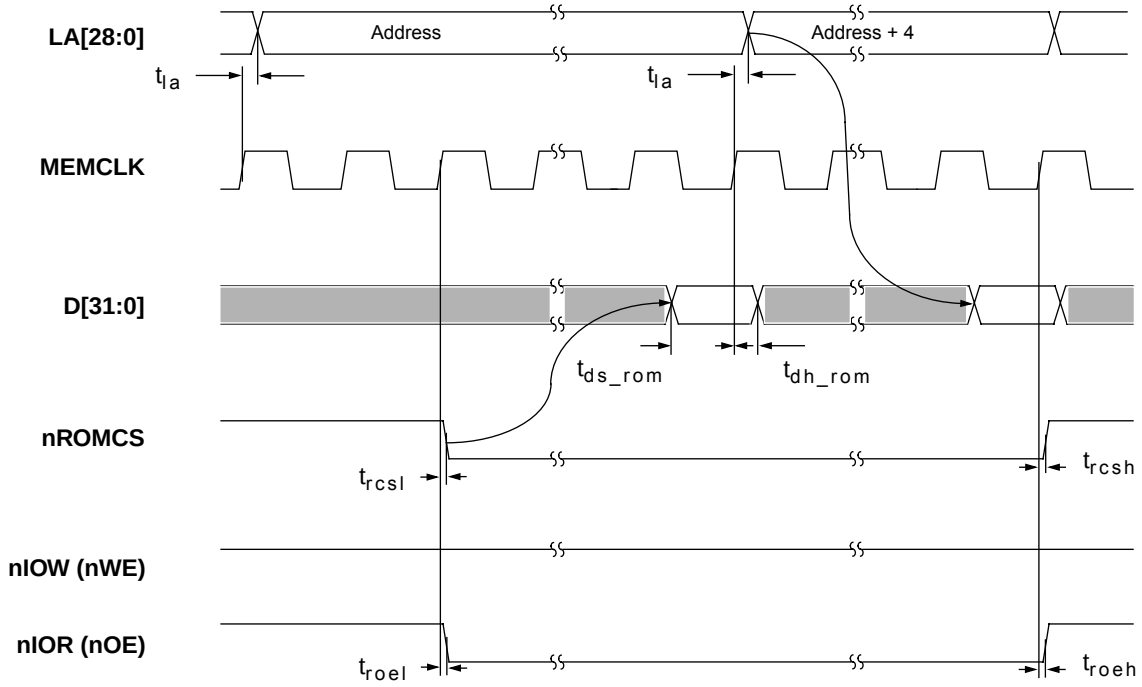


Figure 22-2. ROM Read Access Timing without Burst Mode (32 Bit)

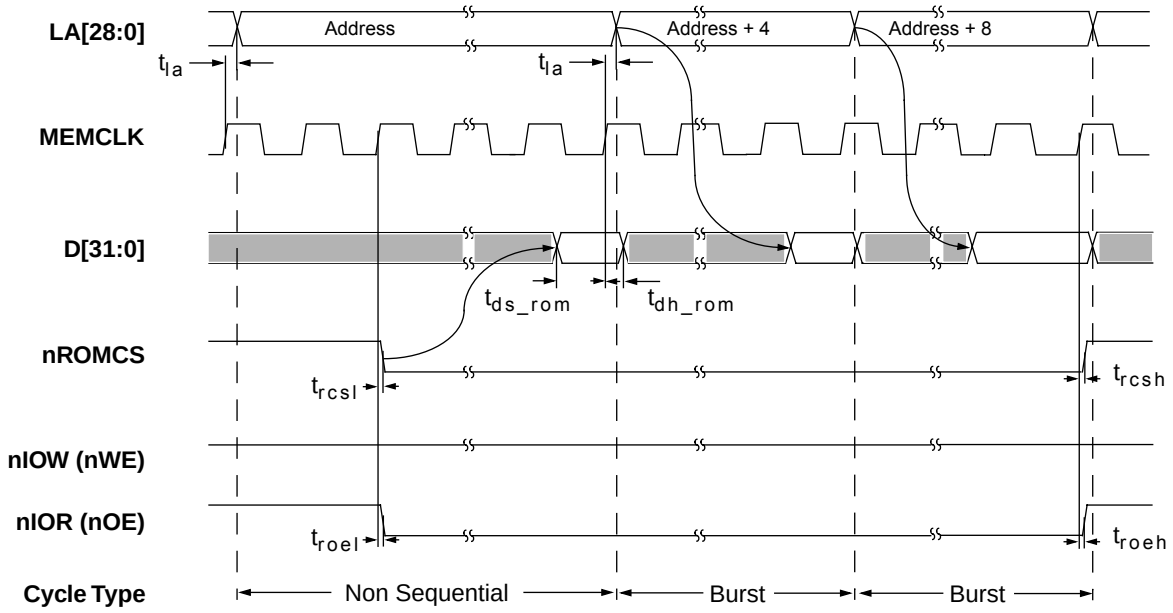


Figure 22-3. ROM Read Access Timing, Burst Mode (32 Bit)

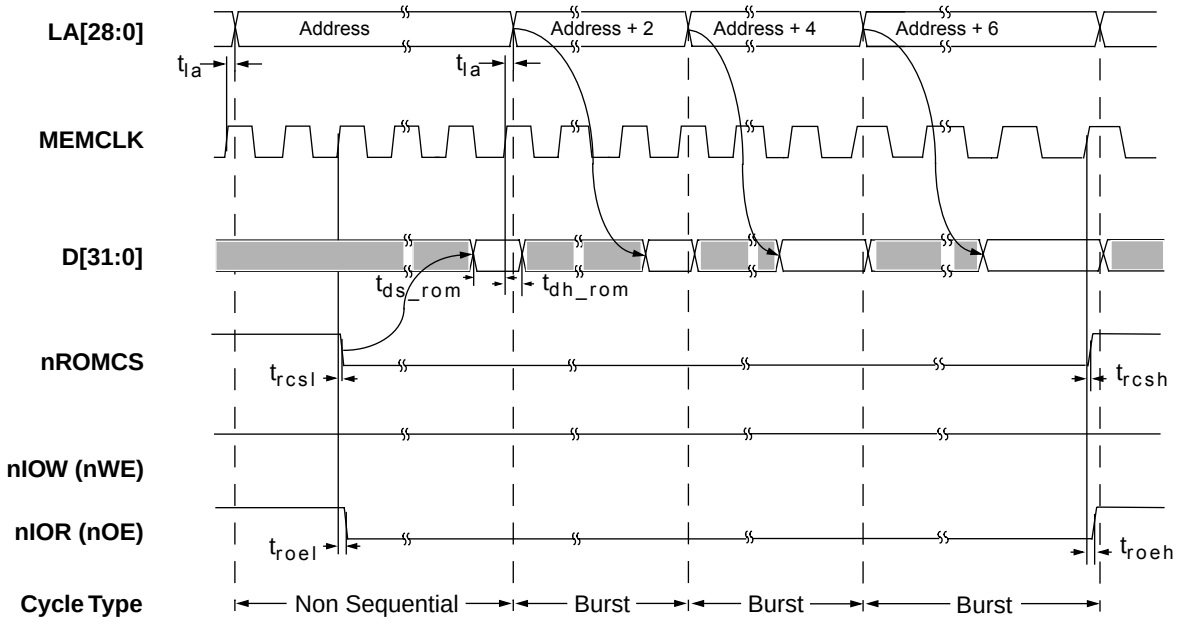


Figure 22-4. ROM Read Access Timing, Burst Mode (16 Bit)

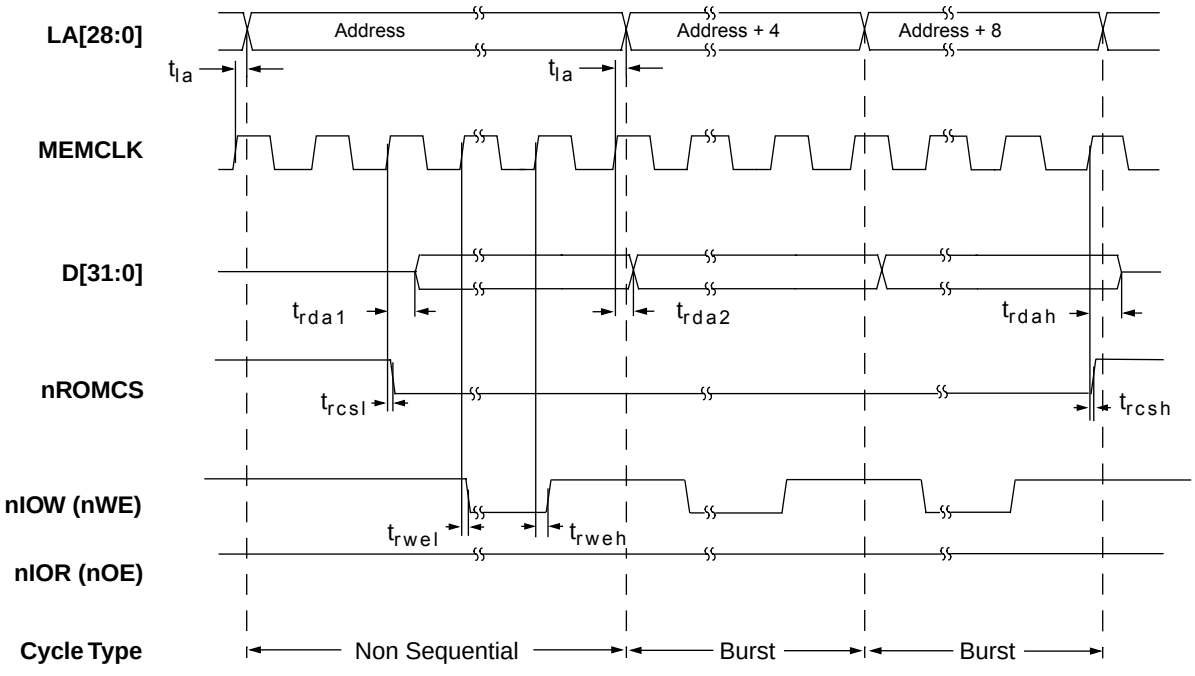


Figure 22-5. ROM Write Access Timing, Burst Mode (32 Bit)

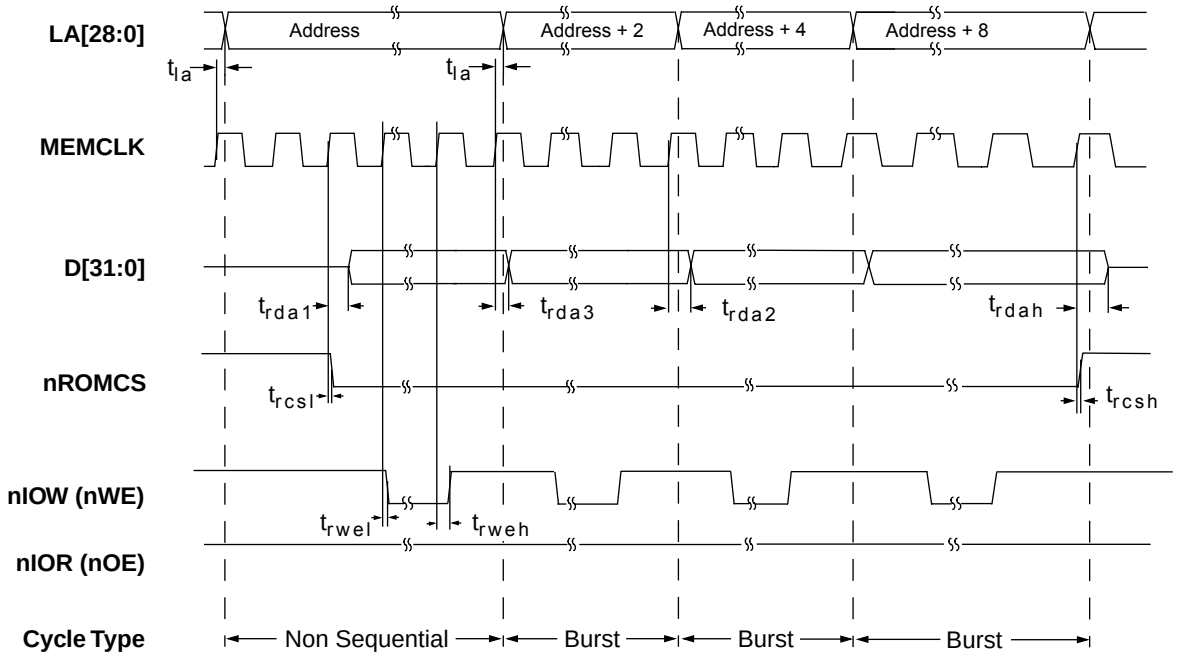


Figure 22-6. ROM Write Access Timing, Burst Mode (16 Bit)

Table 22-4. DRAM Timing

Symbol	Parameter	40 MHz		56 MHz		Units
		MIN	MAX	MIN	MAX	
t_{casl}	MEMCLK rising to nCAS[3:0] falling		12		12	ns
t_{cash}	MEMCLK rising to nCAS[3:0] rising		11		11	ns
t_{ds_dram}	Read DATA setup to MEMCLK rising	–5		–5		ns
t_{dh_dram}	Read DATA hold from MEMCLK rising	16		16		ns
t_{cac_fp} [a]	nCAS falling to data latched	21		21		ns
t_{cac_edo} [b]	nCAS falling to data latched	25		25		ns
t_{da1}	MEMCLK rising to write DATA valid		23		23	ns
t_{da2}	MEMCLK rising to write DATA valid		33		33	ns
t_{da3}	MEMCLK falling to write DATA valid	1.0t – 6	15	1.0t – 6	15	ns
t_{wdh}	Write DATA hold from MEMCLK rising	9		9		ns
t_{rash}	MEMCLK rising to nRAS[x] rising		11		11	ns
t_{rasl}	MEMCLK rising to nRAS[x] falling		13		13	ns
t_{ra1} [c]	MEMCLK rising to RA[11:0] valid (row address)		36		36	ns
t_{ra2} [d]	MEMCLK rising to RA[11:0] valid (row address)		23		23	ns
t_{ca1}	MEMCLK rising to RA[11:0] valid (column address)		15		15	ns
t_{ca2}	MEMCLK falling to RA[11:0] valid (column address)		14		14	ns
t_{cah}	Column address, RA[11:0] valid from MEMCLK rising	12		12		ns
t_{nwel}	MEMCLK rising to nWE falling		12		12	ns
t_{nweh} [e]	MEMCLK rising to nWE rising	10		10		ns
t_{cas2l}	MEMCLK falling to nCAS[3:0] falling		12		12	ns
t_{cas2h}	MEMCLK falling to nCAS[3:0] rising		12		12	ns
t_{rp} [f]	nRAS precharge time	3		3		MEMCLKs

[a] Minimum nCAS access time for Fast Page mode DRAM across all conditions, with nCAS loading of ≤ 100 pF, when MEMCLK = 32 MHz.

[b] Minimum nCAS access time for EDO DRAM across all conditions, with nCAS loading of 100 pF or less.

[c] CPU accesses.

[d] DMA accesses.

[e] nWE rising does not change while external nCAS signals are still low.

[f] The minimum nRAS precharge time can be extended to four cycles by setting bit 6 of the DRAMCTL register.

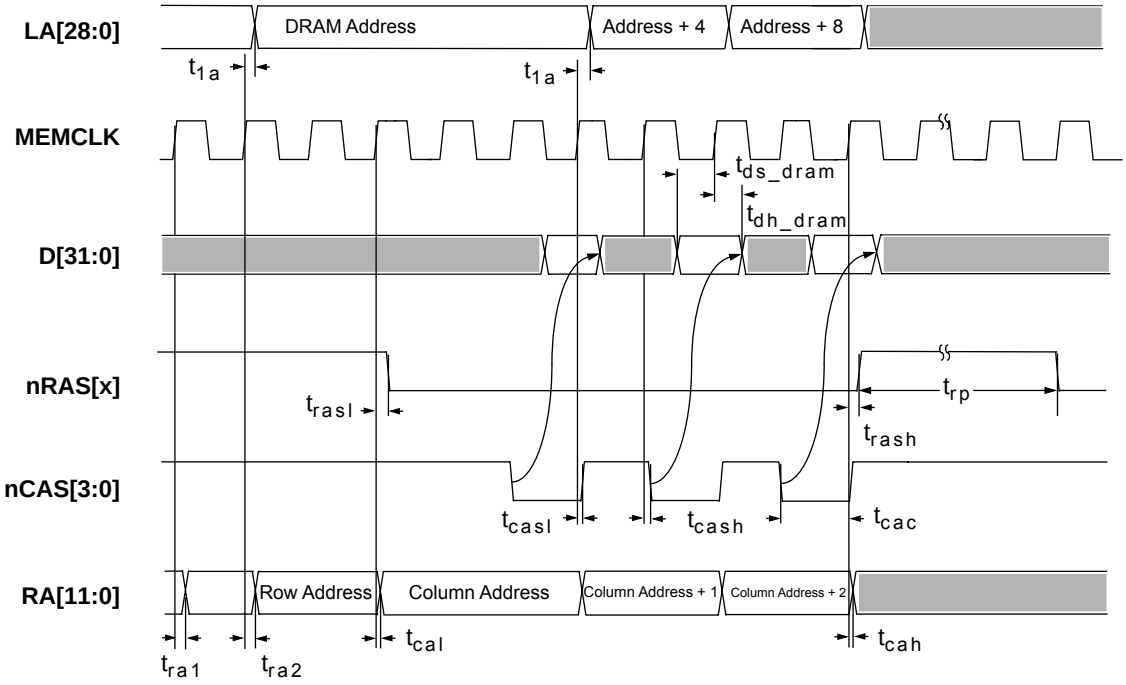


Figure 22-7. Fast Page Mode DRAM Read Timing (32 Bit)

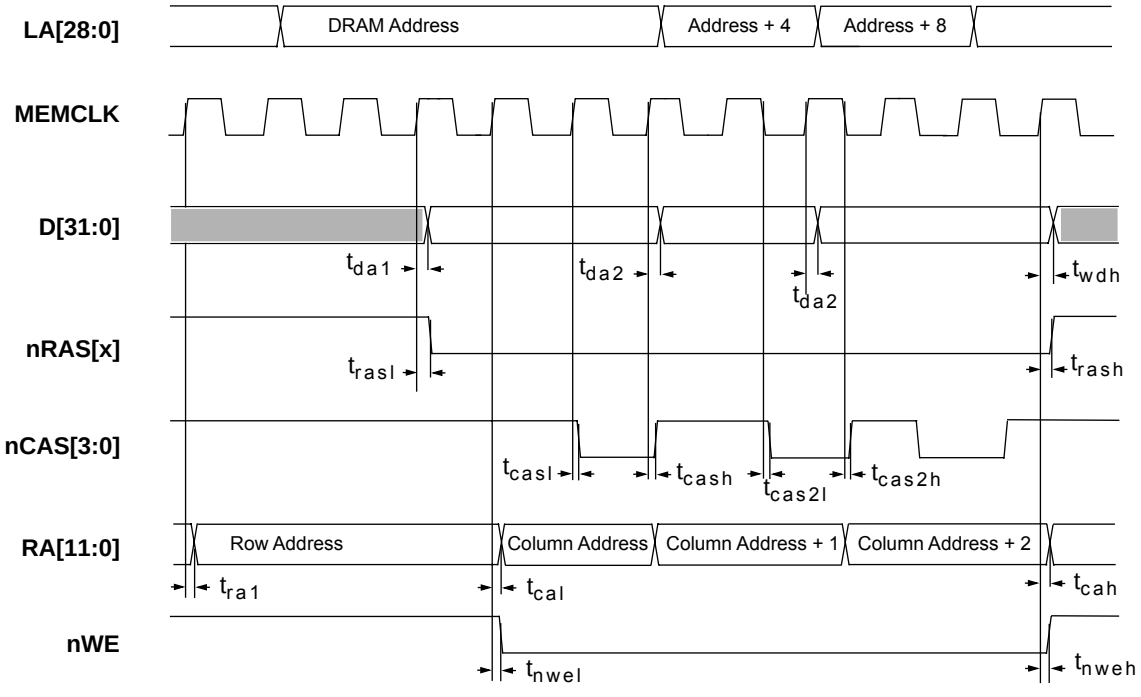


Figure 22-8. Fast Page Mode DRAM Write Timing (32 Bit)

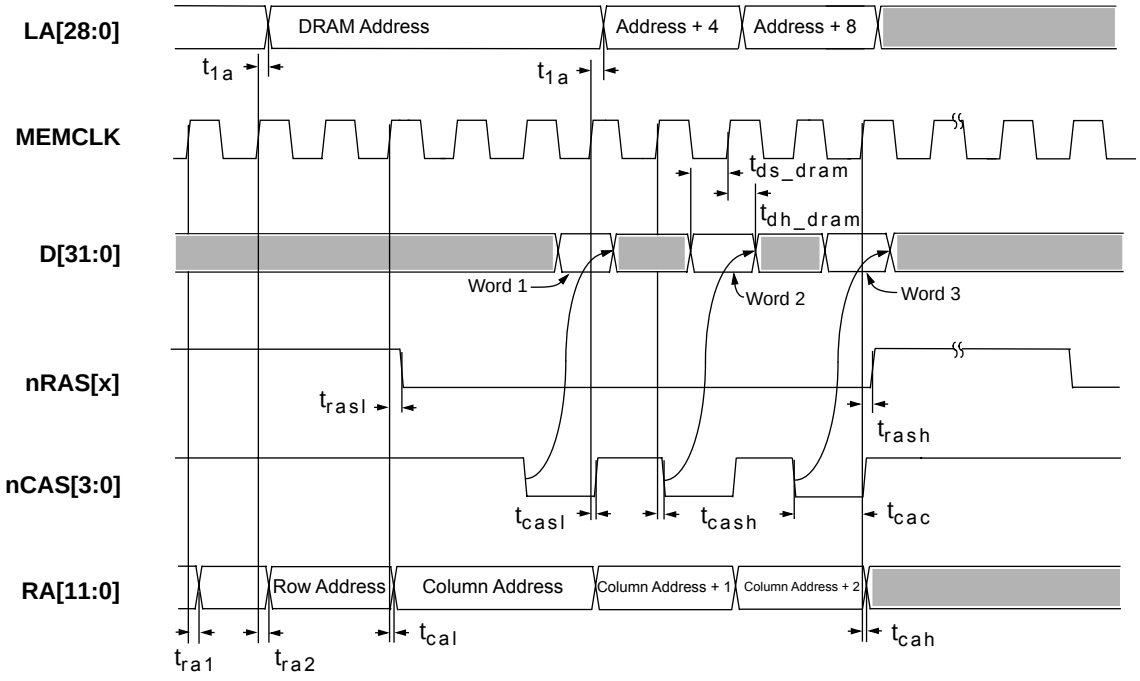


Figure 22-9. EDO DRAM Read Timing (32 Bit)

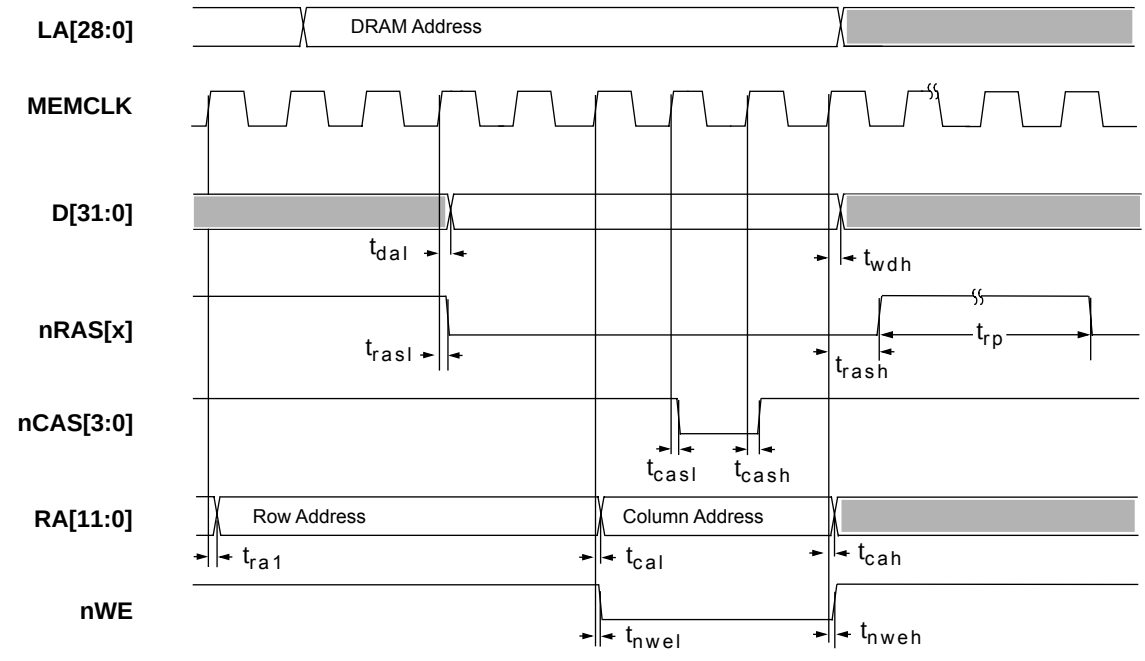


Figure 22-10. EDO DRAM Write Timing — Single Word

Table 22-5. DRAM Memory Refresh Timing

Symbol	Parameter	40 MHz		56 MHz		Units
		MIN	MAX	MIN	MAX	
t_{rref1}	MEMCLK rising to nRAS		12		12	ns
t_{rref2}	MEMCLK falling to nRAS		11		11	ns
t_{crefl}	MEMCLK rising to nCAS[3:0] falling		16		16	ns
t_{crefh}	MEMCLK rising to nCAS[3:0] rising		16		16	ns
t_{rarf}	MEMCLK rising to RA[11:0] changing		22		22	ns

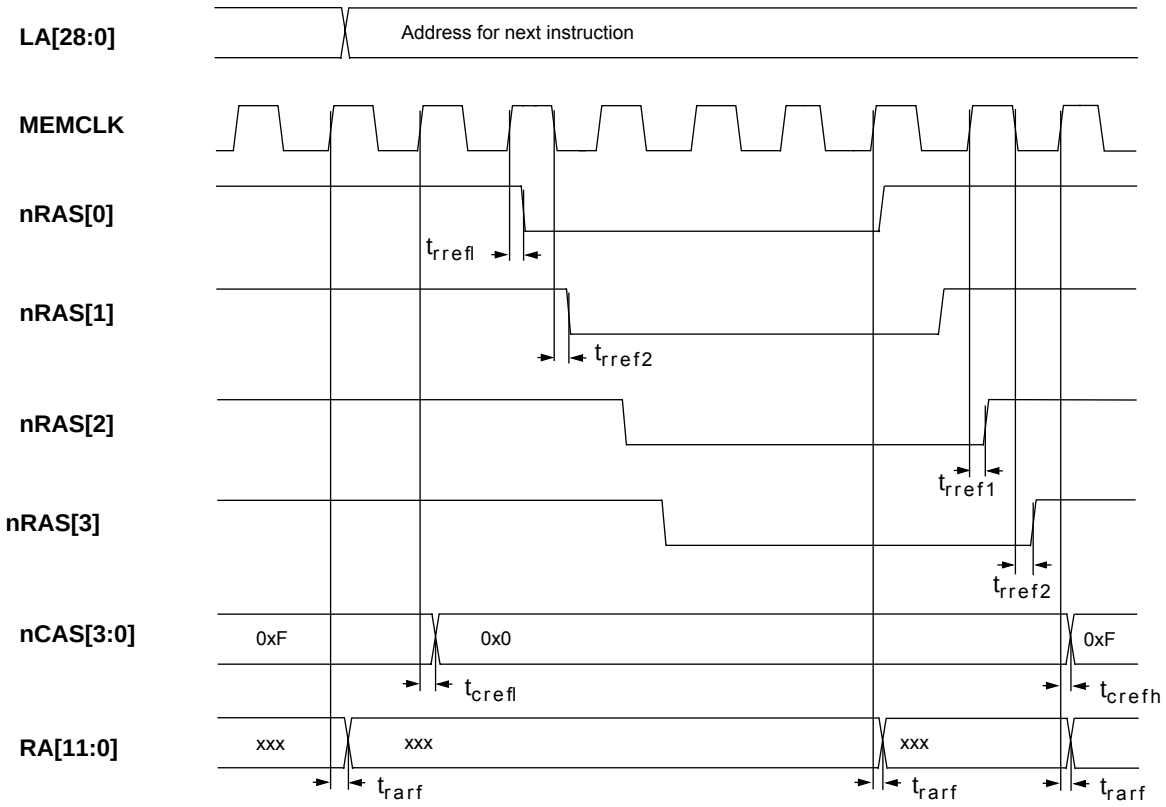


Figure 22-11. DRAM Memory Refresh Timing

22.7 I/O Subsystems

Table 22-6. Simple 8-MHz I/O Timing

Symbol	Parameter	40 MHz		56 MHz		Units
		MIN	MAX	MIN	MAX	
t_{clk8l}	I_OCLK rising to CLK8 falling		14		14	ns
t_{clk8h}	I_OCLK rising to CLK8 rising		14		14	ns
t_{clk2l}	I_OCLK rising to CLK2 falling		16		16	ns
t_{clk2h}	I_OCLK rising to CLK2 rising		16		16	ns
t_{csl_sio}	I_OCLK rising to nSIOCS1/nSIOCS2 falling		19		19	ns
t_{csh_sio}	I_OCLK rising to nSIOCS1/nSIOCS2 rising		20		20	ns
t_{bd1} [a]	I_OCLK rising to BD write data valid	0	102	0	102	ns
t_{bd1s} [b]	I_OCLK rising to BD write data valid (SYNC cycles)	0	476	0	476	ns
t_{bd2} [c, d]	I_OCLK rising to BD write data valid	133	152	133	152	ns
t_{bd2} [c, e]	I_OCLK rising to BD write data valid	149	168	149	168	ns
t_{bdh}	DATA hold from I_OCLK rising	10		10		ns
t_{bds}	DATA setup to I_OCLK rising	0		0		
t_{iornwh}	I_OCLK falling to IORNW rising		13		13	ns
t_{iornwl}	I_OCLK rising to IORNW falling		16		16	ns
t_{niorl}	I_OCLK rising to nIOR falling		16		16	ns
t_{niorh}	I_OCLK rising to nIOR rising		16		16	ns
t_{niowl}	I_OCLK rising to nIOW falling		17		17	ns
t_{niowh}	I_OCLK rising to nIOW rising		16		16	ns
t_{add1} [f]	LA[28:0] changing after I_CLK rising before start	0	143	0	143	ns
t_{add1s} [g]	LA[28:0] changing after I_OCLK rising before start (SYNC cycle)	0	518	0	518	ns
t_{add2} [d, h]	LA[28:0] changing after I_OCLK rising after end	74	89	74	89	ns
t_{add2} [e, h]	LA[28:0] changing after I_OCLK rising after end	90	105	90	105	ns

[a] Synchronization penalty is between 0 and 3 I_OCLK cycles.
[b] Synchronization penalty is between 0 and 15 I_OCLK cycles.
[c] Delay includes 4 MEMCLK cycles.
[d] Timings refer to the case where ASTCR bit = 0.
[e] Timings refer to the worst case where ASTCR bit = 1 (see Appendix C).
[f] Synchronization penalty is between 0 and 4 I_OCLK cycles.
[g] Synchronization penalty is between 0 and 16 I_OCLK cycles.
[h] Delay includes 2 MEMCLK cycles.

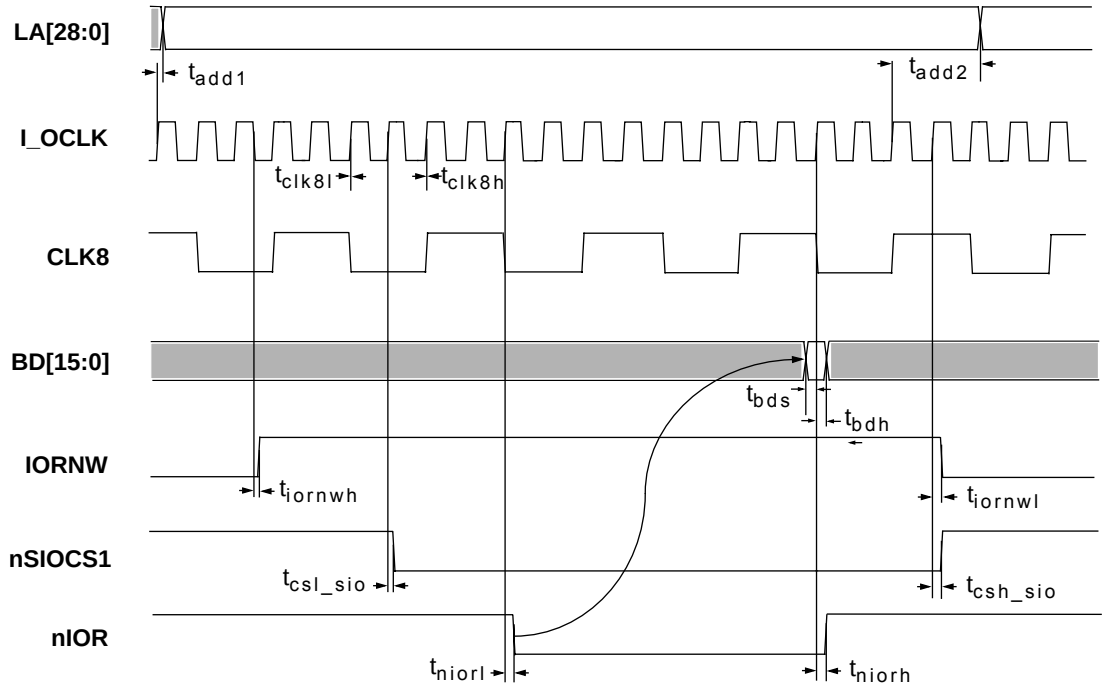


Figure 22-12. 8-MHz Simple I/O Read Timing

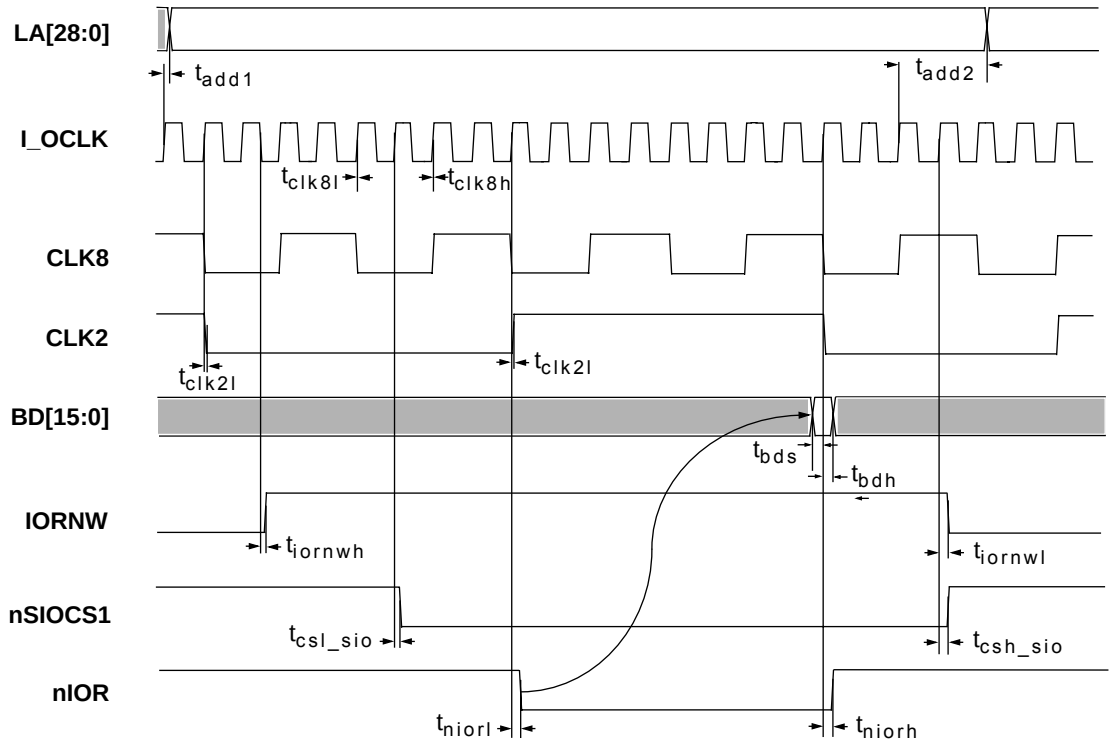


Figure 22-13. 'Sync' 8-MHz Simple I/O Read Cycle Timing

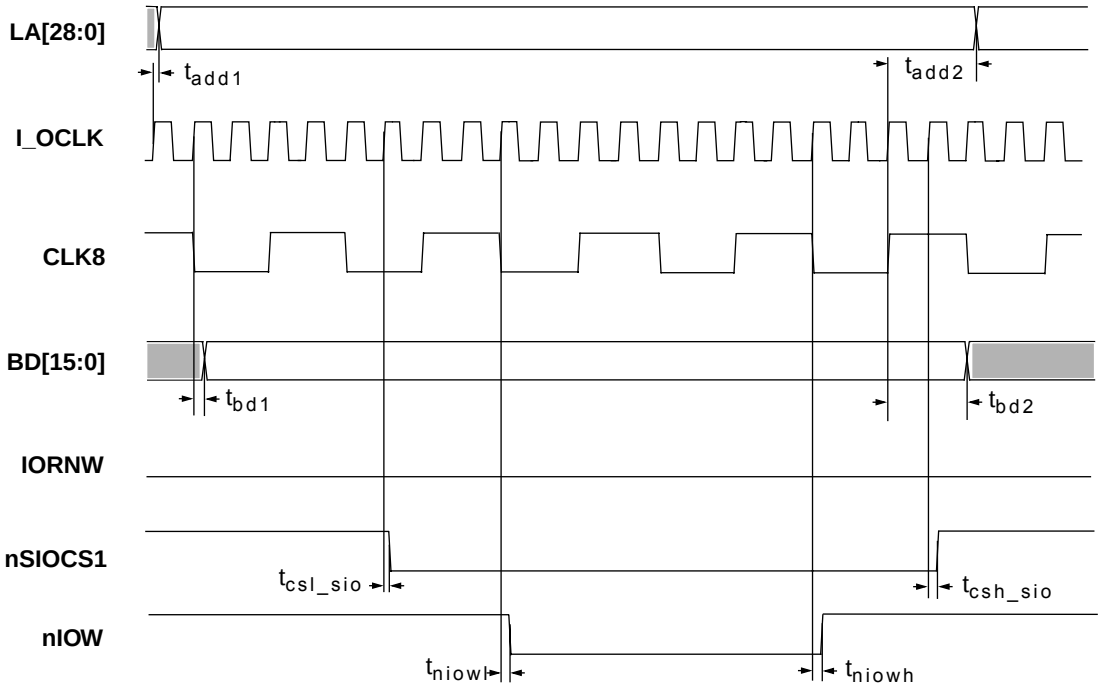


Figure 22-14. 8-MHz Simple I/O Write Timing

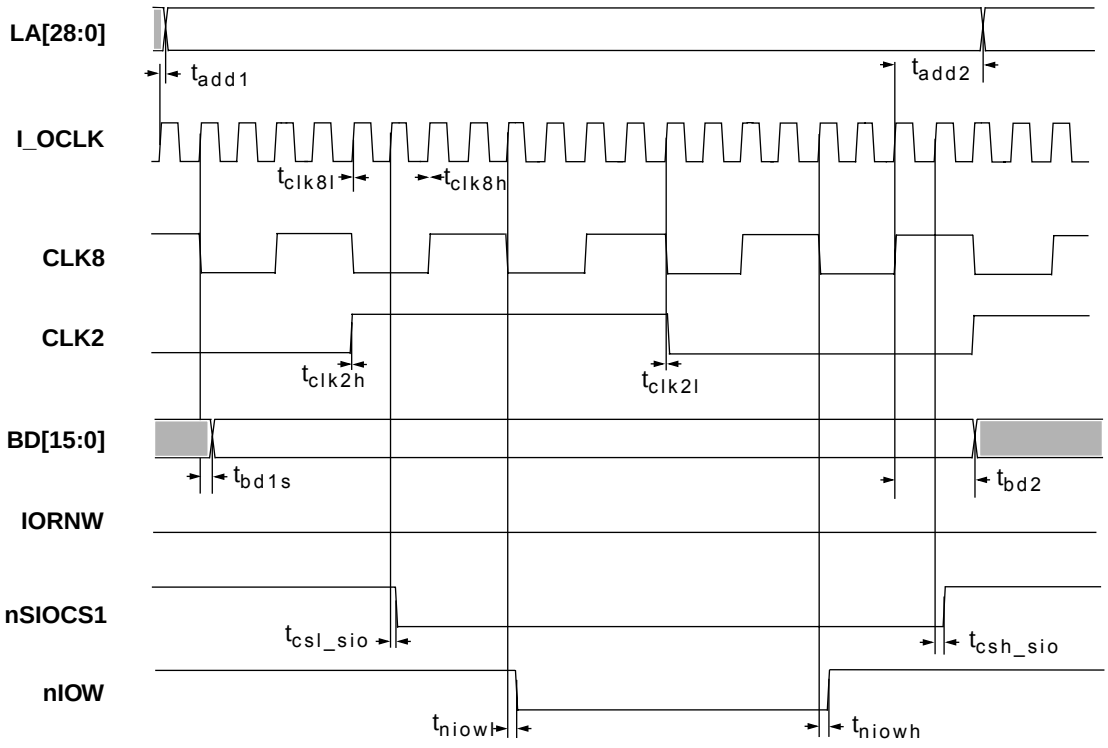


Figure 22-15. 'Sync' 8-MHz Simple I/O Write Cycle Timing

Table 22-7. Module 8-MHz I/O Timing

Symbol	Parameter	40 MHz		56 MHz		Units
		MIN	MAX	MIN	MAX	
t_{bds1}	DATA setup to nBLI falling	0		0		
t_{bdh1}	DATA hold from nBLI falling	2		2		ns
t_{csl_ms}	I_OCLK falling to nMSCS falling		18		18	ns
t_{csh_ms}	I_OCLK falling to nMSCS rising		22		22	ns
t_{iornwh}	I_OCLK falling to IORNW rising		14		14	ns
t_{iornwl}	I_OCLK falling to IORNW falling		14		14	ns
t_{bd1} [a]	I_OCLK rising to BD write data valid	0	102	0	102	ns
t_{bd2} [b, c]	I_OCLK rising to BD write data valid	133	150	133	150	ns
t_{bd2}	I_OCLK rising to BD write data valid	164	181	164	181	ns
t_{niorql}	I_OCLK rising to nIORQ falling		15		15	ns
t_{niorqh}	I_OCLK rising to nIORQ rising		15		15	ns
t_{r8ml}	I_OCLK rising to REF8M falling		13		13	ns
t_{r8mh}	I_OCLK rising to REF8M rising		12		12	ns
t_{gts}	setup of nIOGT to I_OCLK rising	0		0		
t_{gth}	hold of nIOGT to I_OCLK rising	5		5		
t_{add1} [d]	LA[28:0] changing after I_OCLK rising before start	0	143	0	143	ns
t_{add2} [e, c]	LA[28:0] changing after I_OCLK rising after end	74	89	74	89	ns
t_{add2} [e, f]	LA[28:0] changing after I_OCLK rising after end	105	120	105	120	ns

[a] Synchronization penalty is between 0 and 3 I_OCLK cycles.
[b] Delay includes 4 MEMCLK cycles.
[c] Timings refer to the case where ASTCR bit = 0.
[d] Synchronization penalty is between 0 and 4 I_OCLK cycles.
[e] Delay includes 2 MEMCLK cycles.
[f] Timings refer to the worst case where ASTCR bit = 1 (see Appendix C).

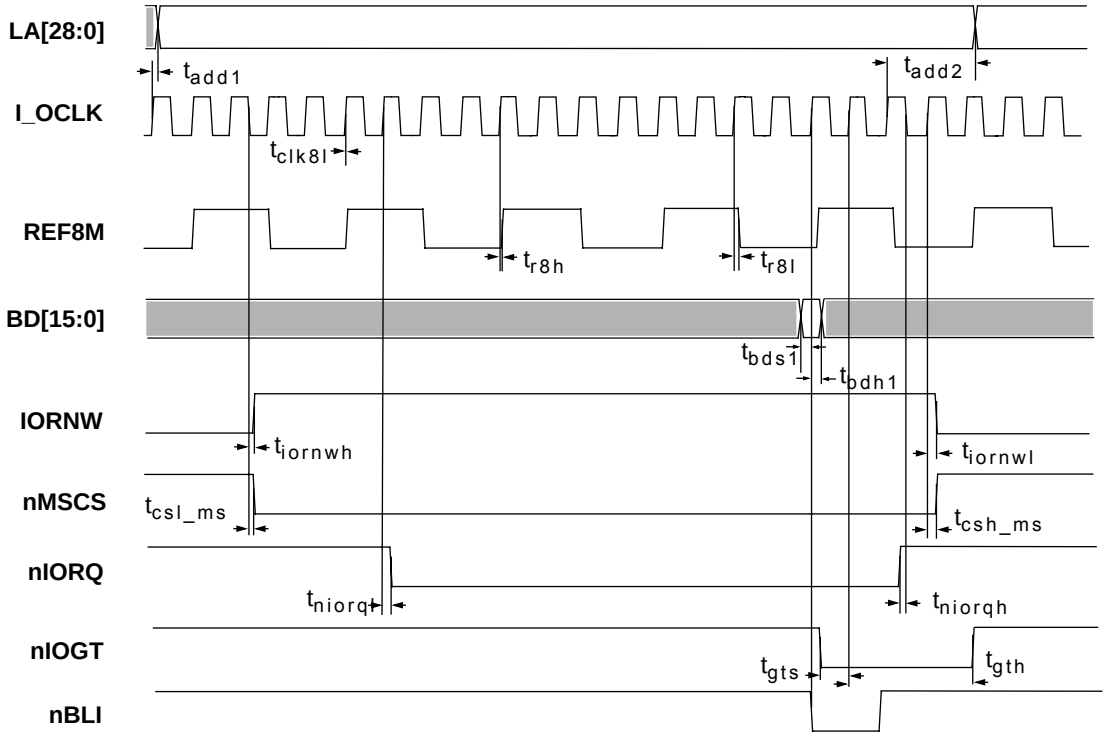


Figure 22-16. 8-MHz Module I/O Read Timing

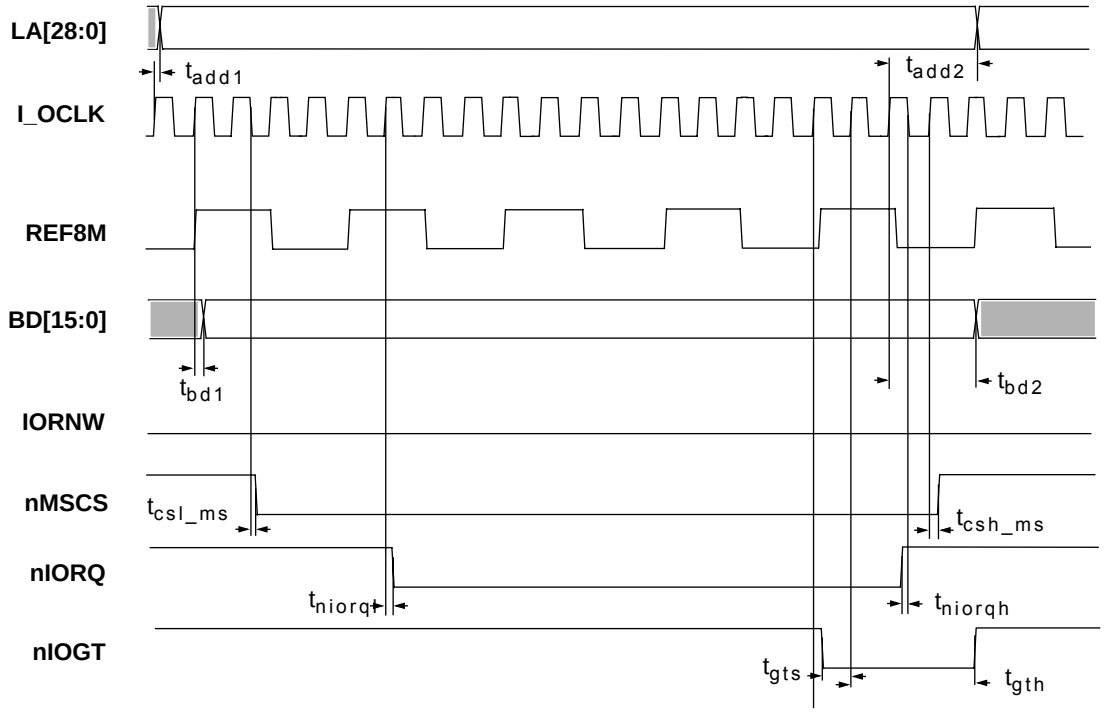


Figure 22-17. 'Sync' 8-MHz Module I/O Write Cycle Timing

Table 22-8. 16-MHz I/O Timing

Symbol	Parameter	40 MHz		56 MHz		Units
		MIN	MAX	MIN	MAX	
t_{nmxl}	nXIPMUX16 falling to upper data output on BD[15:0]		6		6	ns
t_{nmxh}	nXIPMUX16 rising to lower data output on BD[15:0]		5		5	ns
t_{xls}	DATA setup to nXIPLATCH falling	1		1		ns
t_{xlh}	DATA hold from nXIPLATCH falling	2		2		ns
t_{c16l}	I_OCLK rising to CLK16 falling		12		12	ns
t_{c16h}	I_OCLK rising to CLK16 rising		12		12	ns
t_{bdh}	DATA hold from I_OCLK rising	10		10		ns
t_{bds}	DATA setup to I_OCLK rising	0		0		
t_{iornwh}	I_OCLK falling to IORNW rising		13		14	ns
t_{iornwl}	I_OCLK rising to IORNW falling		16		16	ns
t_{csl_pc} [a]	I_OCLK rising to PC I/O chip select falling		20		20	ns
t_{csh_pc} [a]	I_OCLK rising to PC I/O chip select rising		22		22	ns
t_{rds}	READY setup to I_OCLK rising	0		0		
t_{rdh}	READY hold from I_OCLK rising	8		8		ns
t_{bd2} [b, c]	I_OCLK rising to BD write data valid	133	150	133	150	ns
t_{bd2} [b, d]	I_OCLK rising to BD write data valid	164	181	164	181	ns
t_{bd3} [e]	I_OCLK rising to BD write data valid	0	40	0	40	ns
t_{niorl}	I_OCLK rising to nIOR falling		16		16	ns
t_{niorh}	I_OCLK rising to nIOR rising		16		16	ns
t_{noh1}	I_OCLK rising to nBLO rising, read		18		18	ns
t_{nol1}	I_OCLK rising to nBLO falling, read		20		20	ns
t_{noh2}	MEMCLK rising to nBLO rising, write		22		22	ns
t_{nol2}	MEMCLK rising to nBLO falling, write		19		19	ns
t_{nwbeh}	I_OCLK falling to nWBE rising		20		20	ns
t_{nwbel}	I_OCLK rising to nWBE falling		17		17	ns
t_{rbel}	MEMCLK rising to nRBE falling		16		16	ns
t_{rbeh}	MEMCLK rising to nRBE rising		21		21	ns
t_{niowl}	I_OCLK rising to nIOW falling		17		17	ns
t_{niowh}	I_OCLK rising to nIOW rising		16		16	ns
t_{du}	MEMCLK rising to D[31:16] valid		35		35	ns
t_{add3} [f]	LA[28:0] changing after I_OCLK rising before start	0	82	0	82	ns
t_{duh}	MEMCLK rising to D[31:16] invalid	10		10		ns
t_{add2} [c, g]	LA[28:0] changing after I_OCLK rising at end	74	89	74	89	ns
t_{add2} [d, g]	LA[28:0] changing after I_OCLK rising at end	105	120	105	120	ns

[a] Timing is for all PC style I/O chip selects: nCCS, nCDACK, nPCCS1, nPCCS2, nEASCS, and TC.
[b] Delay includes four MEMCLK cycles.
[c] Timings refer to where ASTCR bit = 0.
[d] Timings refer to where ASTCR bit = 1.
[e] Synchronization penalty is between 0 and 1 I_OCLK cycles.
[f] Synchronization penalty is between 0 and 2 I_OCLK cycles.
[g] Delay includes two MEMCLK cycles.

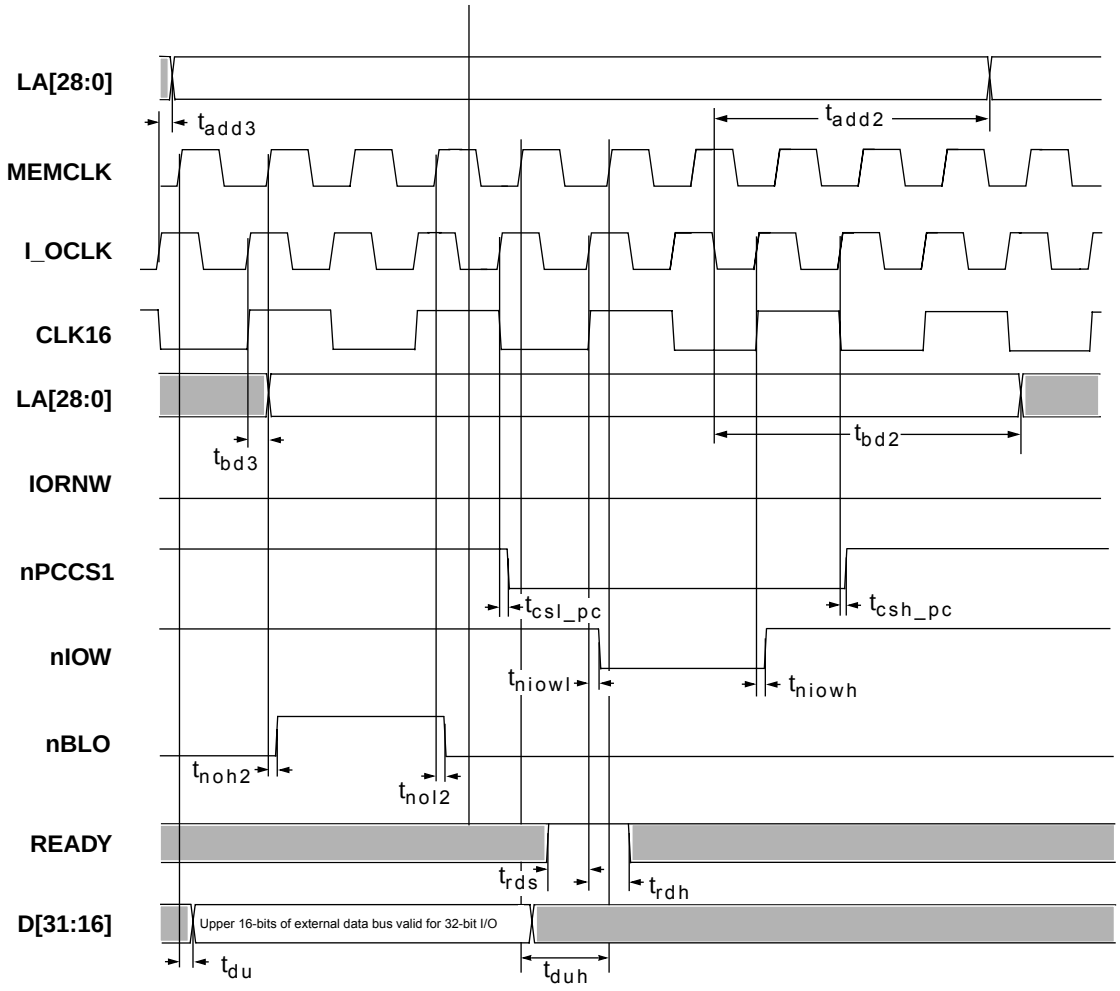


Figure 22-18. 16-MHz Type D I/O Write Cycle Timing

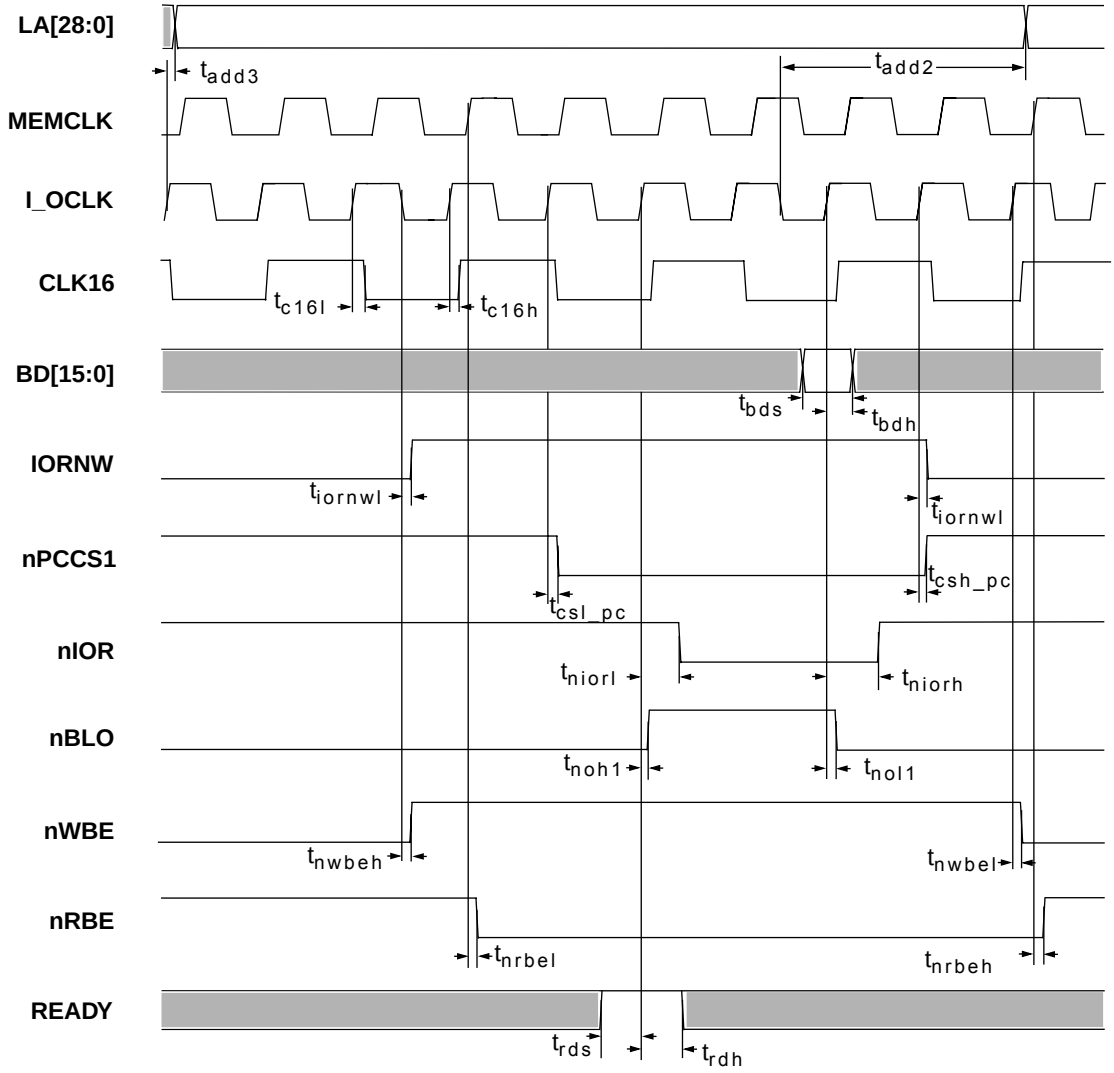


Figure 22-19. 16-MHz Type D I/O Read Cycle Timing

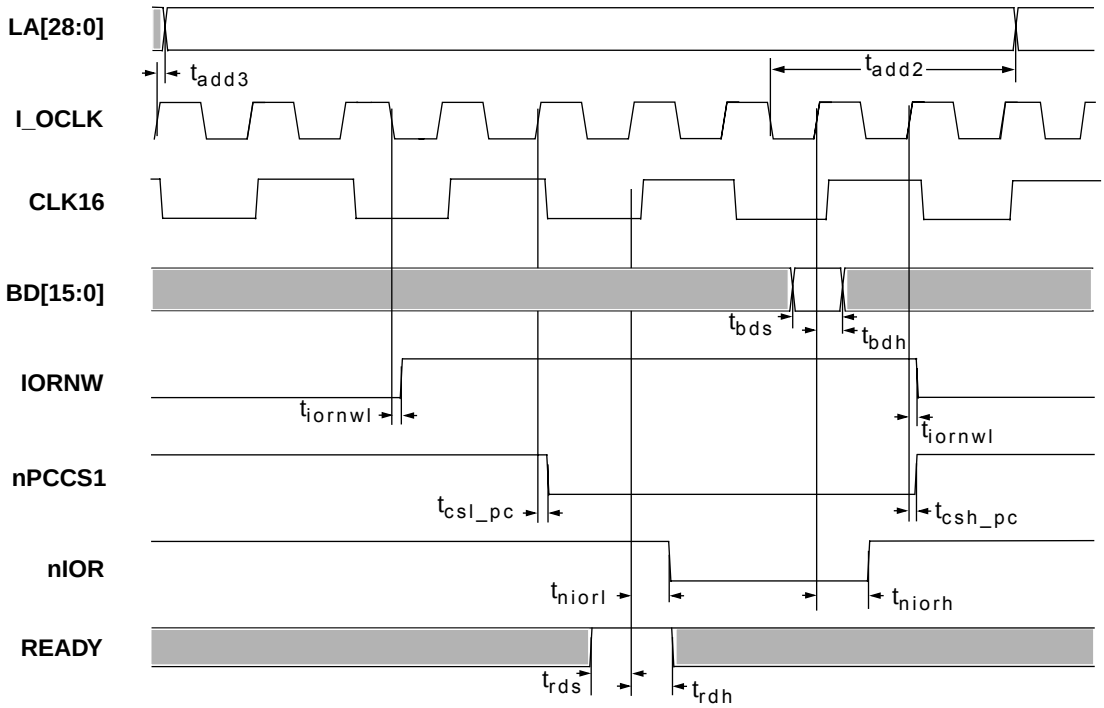


Figure 22-20. 16-MHz Type C, B, and A I/O Read Cycle Timing

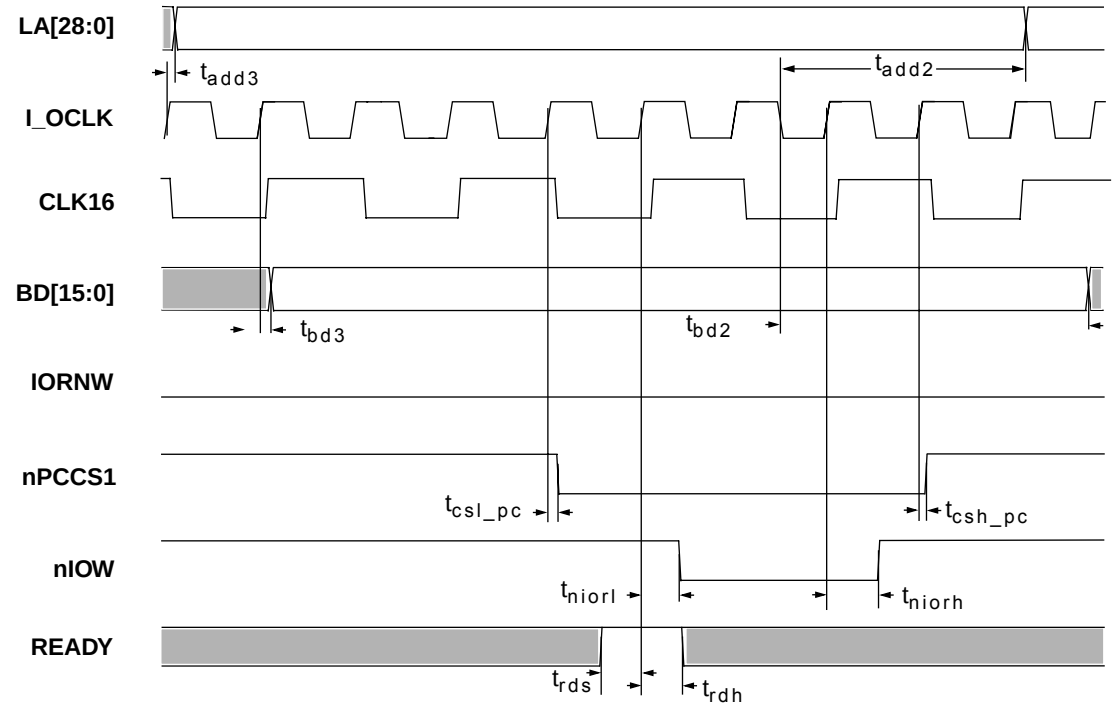


Figure 22-21. 16-MHz Type C, B, and A I/O Write Cycle Timing

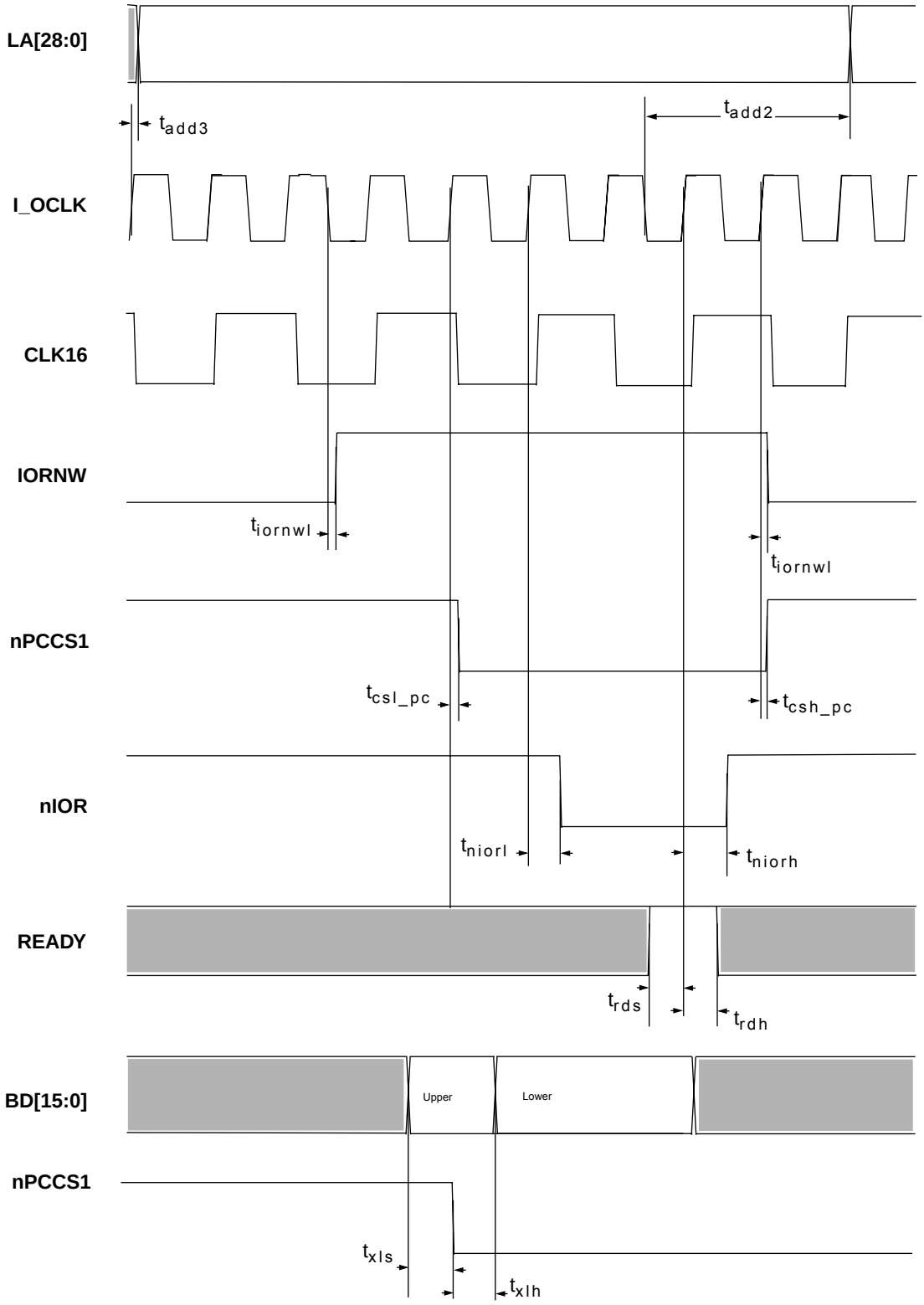


Figure 22-22. 16-MHz Type B I/O Read Cycle Timing with PCMCIA

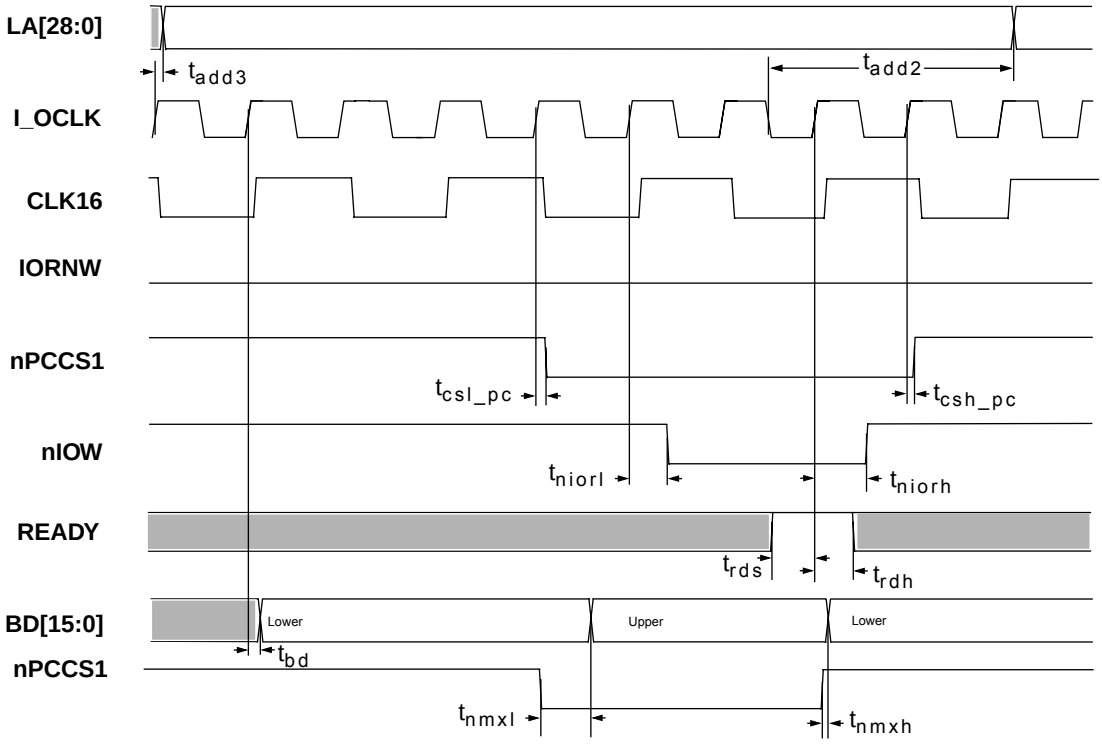
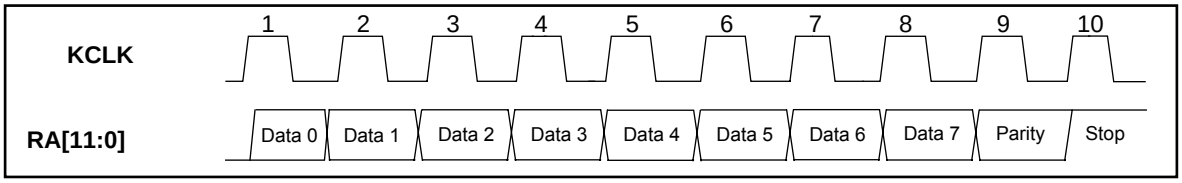


Figure 22-23. 16-MHz Type B /O Write Cycle Timing with PCMCIA

Table 22-9. Keyboard/Mouse Timing

Symbol	Parameter	40 MHz		56 MHz		Units
		MIN	MAX	MIN	MAX	
t_{kclk}	Keyboard clock period	1	100	1	100	µs
t_{kckl}	Keyboard clock low time	0.5	50	0.5	50	µs
t_{kckh}	Keyboard clock high time	0.5	50	0.5	50	µs
t_{dhi}	Hold on DATA from CLK rising for receive	1	$t_{kclk} - 1$	1	$t_{kclk} - 1$	µs
t_{dsi}	Setup on DATA to CLK falling for receive		$t_{kclk} - 1$		$t_{kclk} - 1$	µs
t_{dso}	Setup on DATA to CLK rising for transmit	$t_{kclk} - 1$	t_{kclk}	$t_{kclk} - 1$	t_{kclk}	µs
t_{dho}	Hold on DATA from CLK falling for transmit	0	1	0	1	µs
t_{ki}	Time for which CLK is held low to request a send	63.5	64.5	63.5	64.5	µs
t_{krg}	Clock low from CL-PS7500FE to clock low from the keyboard for a request to send	1		1		µs
t_{ksb}	Clock low to DATA low hold time for a request to send	1		1		µs



Keyboard/Mouse Controller Receive Protocol

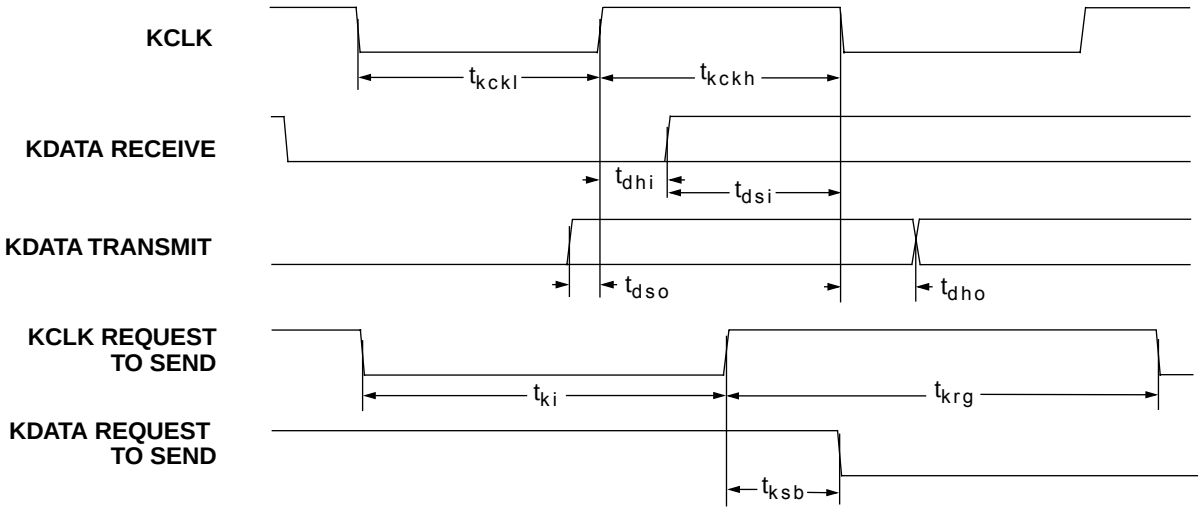


Figure 22-24. Keyboard/Mouse Timing

Table 22-10. Serial Sound Output Timing

Symbol	Parameter	40 MHz		56 MHz		Units
		MIN	MAX	MIN	MAX	
t_{sdo}	SDCLK falling to SDO valid (normal format)	0	5	0	5	ns
t_{sdoj}	SDCLK falling to SDO valid (Japanese format)	0	5	0	5	ns

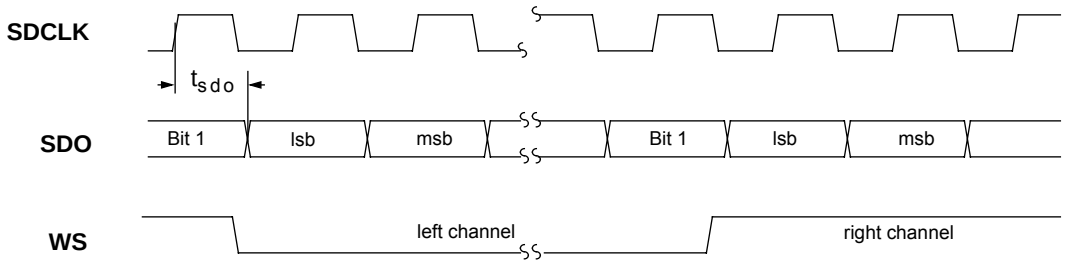


Figure 22-25. Serial Sound Output Timing (Normal Format)

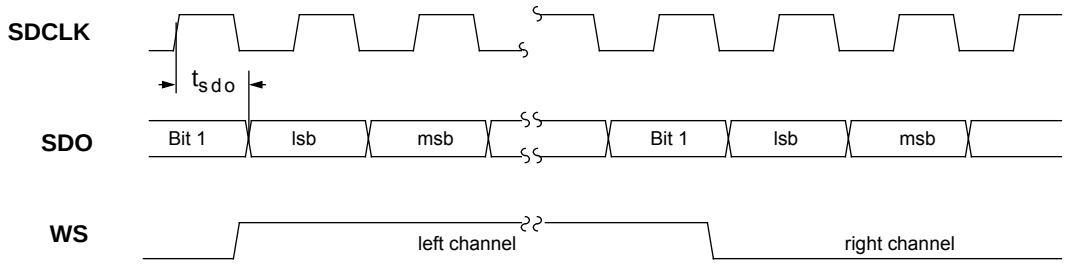


Figure 22-26. Serial Sound Output Timing (Japanese Format)

22.8 System Timing (Clocks)

Table 22-11. System Timing (Clocks)

Symbol	Parameter	40 MHz		56 MHz		Units
		MIN	MAX	MIN	MAX	
t_{cpck1l}[a]	CPUCLK low time	11.25		8		ns
t_{cpck1h}[a]	CPUCLK high time	11.25		8		ns
t_{mck1l}[a]	MEMCLK low time	7		7		ns
t_{mck1h}[a]	MEMCLK high time	7		7		ns
t_{iock1l}[a, b]	I_OCLK low time	14		14		ns
t_{iock1h}[a, b]	I_OCLK high time	14		14		ns
t_{cpck2l}[c]	CPUCLK low time	6.25		6.25		ns
t_{cpck2h}[c]	CPUCLK high time	6.25		6.25		ns
t_{mck2l}[c]	MEMCLK low time	5		5		ns
t_{cpck2h}[c]	MEMCLK high time	5		5		ns
t_{iock2l}[c, d]	I_OCLK low time		7.8		7.8	ns
t_{iock2h}[c, d]	I_OCLK high time		7.8		7.8	ns
t_{vckl}	VCLKI low time	4		4		ns
t_{vckh}	VCLKI high time	4		4		ns
t_{hckl}	HCLK low time	4		4		ns
t_{hckh}	HCLK high time	4		4		ns
t_{ed}[e]	ECLK to ED delay	5	7	5	7	ns
t_{lcded}	ECLK to ED delay (LCD mode)	$t_{lcded} \div 4 + 5$	$t_{lcded} \div 4 + 7$			ns

[a] Divide-by-1 prescalar selected.

[b] I_OCLK = 32 MHz in Divide-by-1 mode.

[c] Divide-by-2 prescalar selected.

[d] I_OCLK = 64 MHz in Divide-by-2 mode.

[e] ECLK mark-space ratio is not always 1:1, depending on the pixel clock divide.

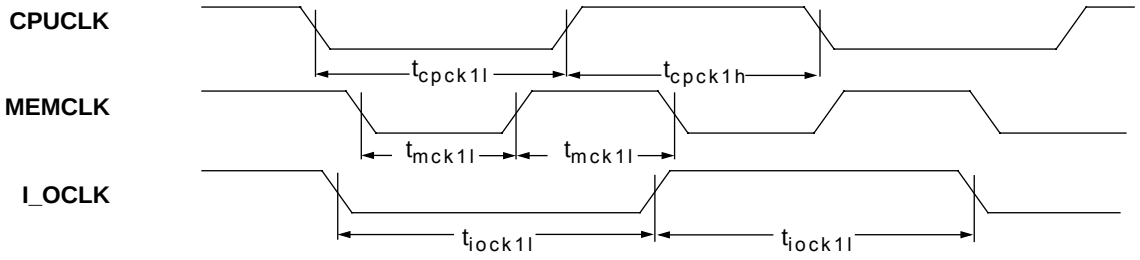


Figure 22-27. Clock Timing with Divide-by-1 Prescalars Selected

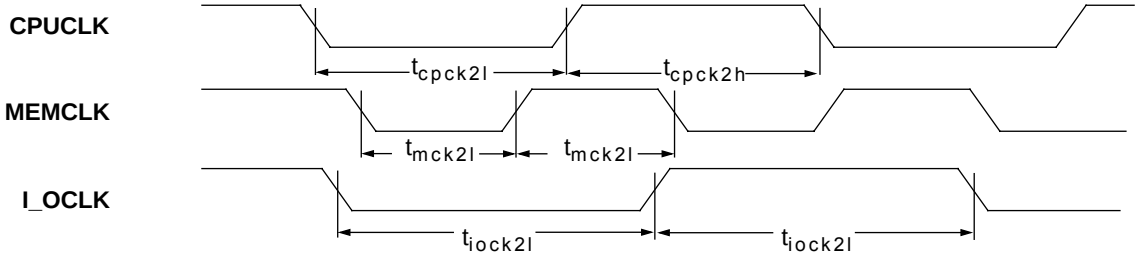


Figure 22-28. Clock Timing with Divide-by-2 Prescalars Selected

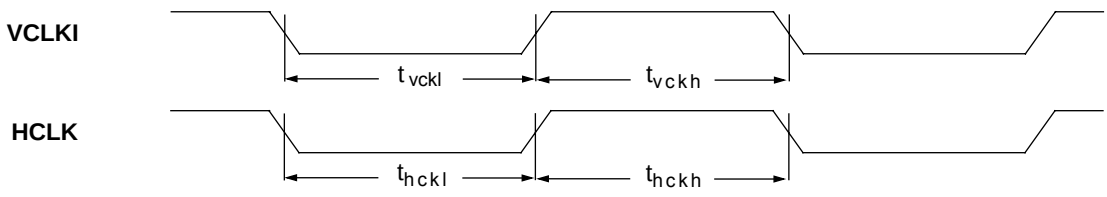


Figure 22-29. Video Clock Timing

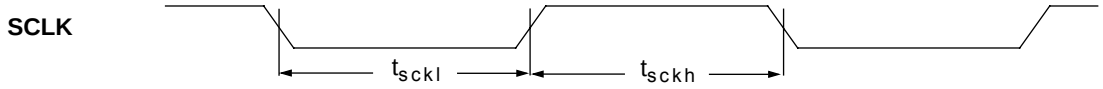


Figure 22-30. Sound Clock Timing

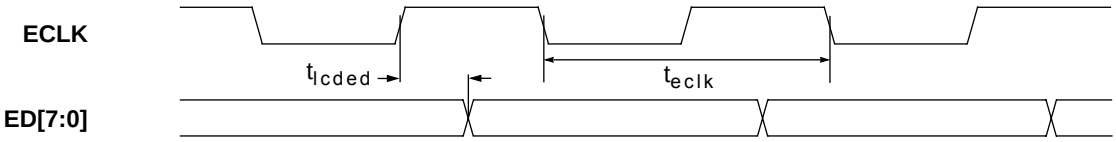


Figure 22-31. Timing Relationship Between ECLK and ED in LCD Grayscale Mode

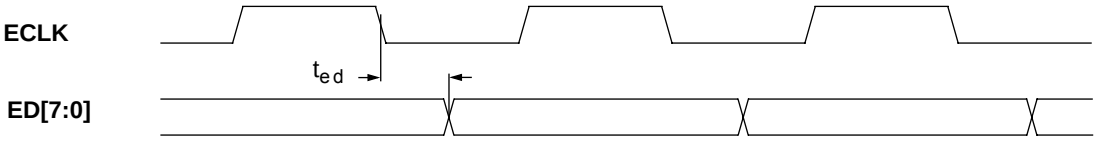
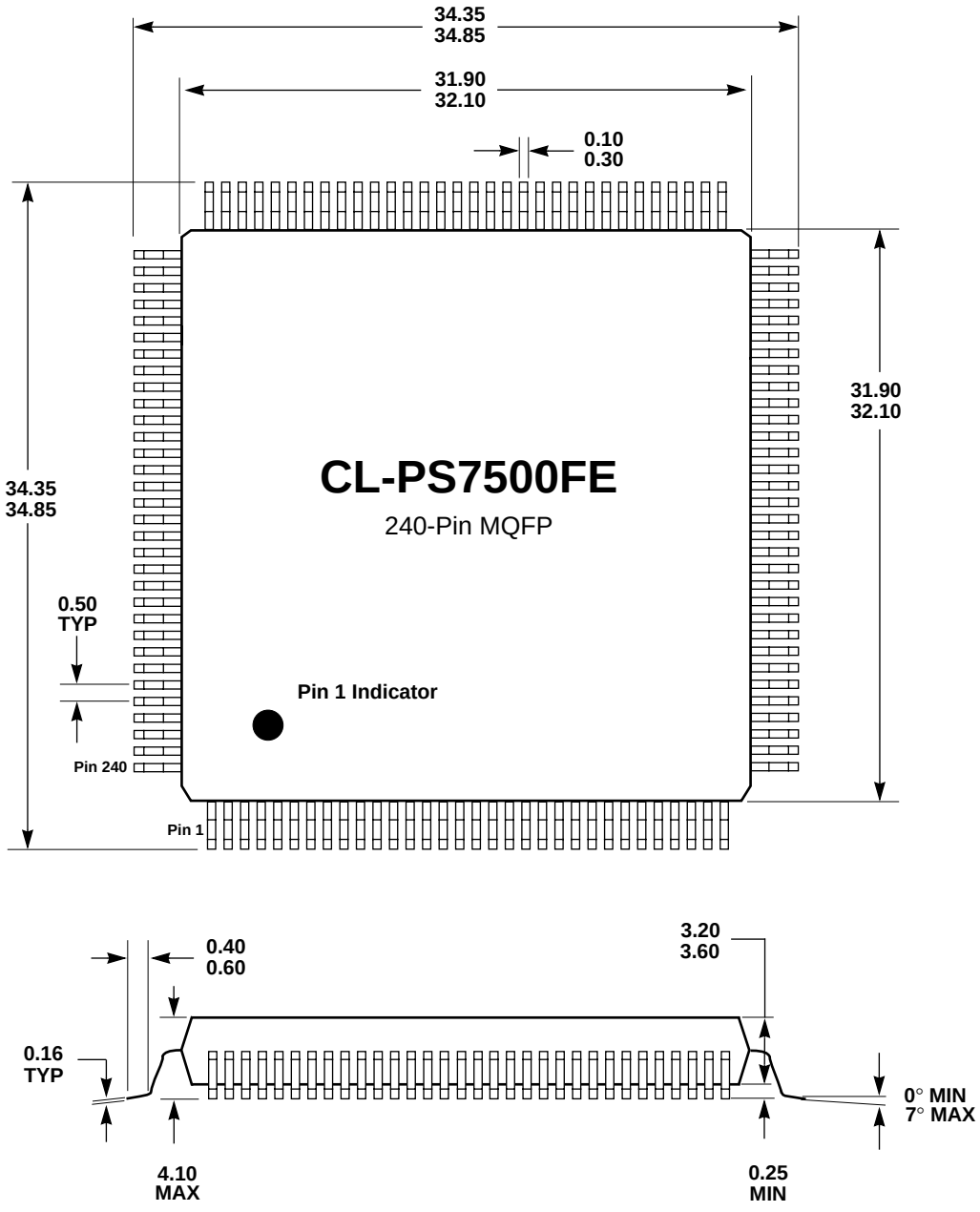


Figure 22-32. Timing Relationship Between ECLK and ED in All Other Modes

23. PACKAGE

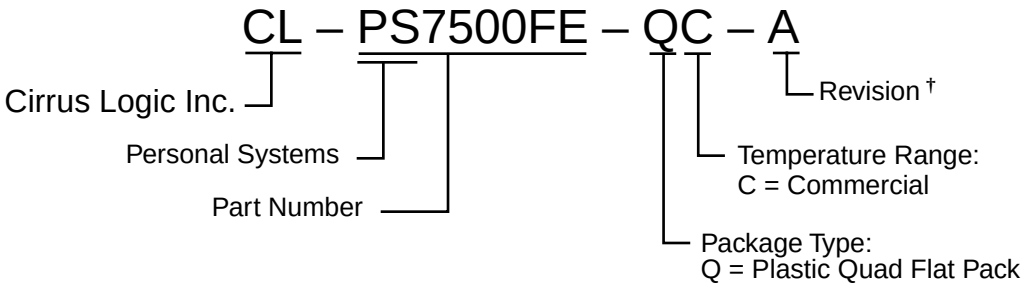
23.1 240-Pin MQFP Package Example



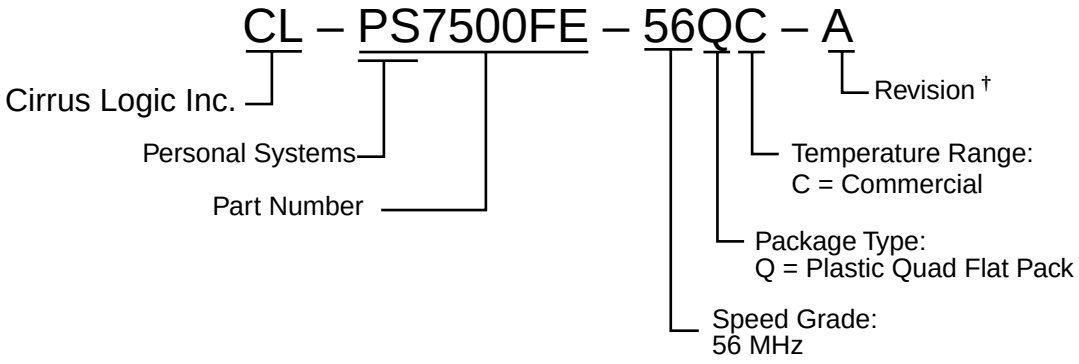
NOTES:

1) Dimensions are in millimeters, and controlling dimension is millimeter.
2) Drawing above does not reflect exact package pin count.
3) Before beginning any new design with this device, please contact Cirrus Logic for the latest package information.

24. ORDERING INFORMATION EXAMPLE



† Contact Cirrus Logic for up-to-date information on revisions.



† Contact Cirrus Logic for up-to-date information on revisions.

Appendix A

Initialization and Boot Sequence

1. Introduction

CL-PS7500FE is designed to operate with 16- or 32-bit-wide memory systems. To avoid a hardware selection mechanism, the CL-PS7500FE is designed to always power-up with ROMCR0[6] set to '1', then the device expects to receive the first instructions from a 16-bit-wide ROM bank. For a system that is actually using 16-bit-wide ROM, no special action is required. For a system that uses 32-bit-wide ROM, a software solution is required for the device to boot successfully.

A sample method of programming the first locations of ROM to successfully boot the device is described in Section 1. This example assumes that the reset vector is located at physical memory address zero.

2. Sample Boot Sequence

The processor starts executing code from physical address 0. As CL-PS7500FE is initially configured to operate with a 16-bit-wide ROM, it fetches the lower half-word of the first instruction from the lower 16 bits of address 0; the upper half-word of the instruction is fetched from the lower 16 bits of address 4.

If these first two locations are programmed with instructions to load the PC with the reset and undefined instruction vectors, then the combination of the lower half-words from the first and second location always create an instruction with a 'never-true' condition code, therefore execution drops through to the next instruction. This is true for all LDR PC instructions in the exception table. Table A-1 is the exception table occupying the first eight locations in ROM. This vector table resides at physical address 0.

Table A-1. Vector Table

Address	Instruction
0	LDR PC, RESET_VEC
4	LDR PC, UNDEF_VEC
8	LDR PC, SWI_VEC
C	LDR PC, PREF_VEC
10	LDR PC, DATA_VEC
14	LDR PC, RES_VEC
18	LDR PC, IRQ_VEC
1C	LDR PC, FIQ_VEC

Immediately after the table, the CL-PS7500FE is in 32-bit mode. The eight locations, from address 20 to 3C, must be programmed with eight half-words in the lower sixteen bits of each location. These form the four required 32-bit instructions when read in pairs by the CL-PS7500FE. The upper 16 bits of each location are ignored by the CL-PS7500FE while still in 16-bit mode.

The four instructions program ROMCR0 into 32-bit mode and cause program execution to jump back to the reset vector at physical address zero. This now be executes correctly. The MOV PC – #0 instruction, that actually causes execution to jump back to zero, *must* be prefetched in 16-bit mode even though it occurs after the ROMCR0 is reprogrammed. Table A-2 shows the data required at memory locations 0x20 to 0x3C to implement this scheme.

Table A-2. Instructions for Programming the ROM Register

Data	Address	Instruction	Notes
0x0000B632	20		
0x0000E3A0	24	MOV R11, 0x03200000	Point at register base.
0x00000000	28		
0x0000E3A0	2C	MOV R0, #&0	32b, slow, 218.75 µs, no burst.
0x00000080	30		
0x0000E5CB	34	STRB R0, [R11,0x80]	Program ROMCR0 and switch mode.
0x0000F000	38		
0x0000E3A0	3C	MOV PC, #0	Jump to zero.

The boot code above is a general example to set the ROM interface to use the slowest access timing This ensures it works with all systems. To speed execution, program the ROM control registers with the fastest parameters of the interface. On power up, the default state of the CLKCTL register is for the CPUCLK, MEMCLK, and I_OCLK external clock inputs to be divided by 2. If appropriate, program these clocks to divide-by-1. This also speeds execution.

3. Other Methods

The above method is an example of how the CL-PS7500FE can be booted from a system using 32-bit-wide ROM. There are other methods to do this that may be more appropriate for the required application. The main advantage of the above method is that it allows the exception vector table to reside at physical address 0. If this is not a requirement the instructions that reprogram ROMCR0 can reside from location 0 up, and the vector table can be mapped into DRAM by the operating system software.

Appendix B

Dual-Panel Liquid Crystal Displays

1. Programming the Video Subsystem

EREG[13] (lcd) must be programmed to '1' for normal LCD operation. CONREG[13] (address 0xE00xxxxx) must be programmed to '1'. This is the dup bit to set Duplex mode.

Video data is simultaneously channelled to the top and bottom half-screens. The first qword received from memory is interpreted for the first part of the first raster in the top half-screen; the second qword is for the identical part of the lower half-screen. CL-PS7500FE handles the sequencing of DMA data so the video buffer can still be programmed as if there were only one panel.

Program VCSR[14:13] (address 0x9600xxxx) as:

Bit		Description
14	13	
0	0	Dual Panel mode not activated
0	1	Cursor in upper half-screen
1	0	Cursor in lower half-screen
1	1	Cursor straddles both half-screens

Normally VCSR defines the number of rasters from VSYNC to the start of the cursor, and VCER defines the number of rasters from VSYNC to the end of the cursor display. See Chapter 16 for programming details.

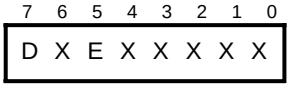
Split-screen Operation

For split-screen operation, the programming of VCER and VCSR is the same as for a single-panel LCD when the cursor is in the top or bottom half-screen. When the cursor straddles the boundary, the values of these two registers have a different meaning. The value in the VCSR is the number of rasters from the top of the lower panel to the end of the cursor image and VCER is programmed with the number of rasters from the top of the display to the start of the cursor image in the upper panel.

The cursor is displayed in the lower half-screen from the value of VDSR to VCSR and in the upper half-screen from the value of VCER to VDER. Essentially, the start register is effectively defining the 'end' of the cursor in the bottom half-screen and the end register is defining the 'start' of the cursor in the top half-screen. This is because the top of the lower half-screen is written to before the bottom of the upper half-screen.

2. Configuring DMA within the CL-PS7500FE

The video and sound macrocell must first be programmed to drive dual-panel LCDs as previously described. When this is done, the macrocell always makes qword DMA requests in pairs. The CL-PS7500FE is then set into Dual-panel mode by programming the 'dup' bit (VIDCR[7], address 0x1E0) to '1'. The eight bits of the VIDCR are now allocated as:



X = Undefined

E = Enable

D = Duplex LCD

To enable two parallel data streams to be output by the video and sound macrocell to the two panels of the LCD when Duplex mode is enabled, the CL-PS7500FE DMAs two qwords from memory, offset by half the size of the video buffer. Since all DMA is qword-only, the auto-increment of the DMA address is now always 0x10.

The VIDSTART and VIDEND registers are programmed normally (that is, for a single panel) with the addresses of the first and last qwords in memory. Program the VIDINITA register with the address of the first qword to display on the upper panel of the LCD; program the VIDINITB register with the address of the first qword to display on the lower panel of the LCD. The difference between the two addresses is half the number of bytes in the video buffer.

It is possible for VIDINITA to point to an address in the lower half of the buffer, in which case set VIDINITB to point to an address in the top half of the buffer, offset again by half the buffer size.

If either of the INIT register values are equal to the end register, bit 30 of the relevant INIT register must be set high for correct operation (the 'last' bit).

NOTE: Never program both of the 'last' bits simultaneously high.

3. Cursor

To ensure a smooth transition of the cursor across the dual-panel boundary, four images of the cursor must be stored in memory. The CL-PS7500FE DMA registers must only be programmed with qword-aligned addresses. Since the cursor is always 32-pixels wide at 2 bpp, the address of data corresponding to a particular row of the cursor can be aligned with a 2-word boundary. Arrange the four images as two pairs of contiguous images of the cursor. Only alternate rows of each cursor image start on qword boundaries.

For reasons previously stated, the two pairs of images are offset so that the first has all the odd rows starting on qword boundaries; the second has all even rows starting on qword boundaries. This means that CL-PS7500FE can address any row of the cursor using only qword-aligned DMA pointers.

Normally, only the first image is used. However, when the cursor is straddling the split-screen boundary, VCSR and VCER are programmed as described above and the cursor INIT register must be set to point to the location corresponding to the position of the row of the cursor that appears at the top of the lower section of the screen. In conjunction with the different meaning of the vertical cursor position registers in the video and sound macrocell, this enables a smooth transition across the split-screen boundary.

4. Video Frame Buffer Restrictions

For dual-panel LCDs to be driven correctly, it is necessary for the video frame buffer to contain an even number of qwords and be aligned to a qword boundary; the cursor buffer must also be aligned to a qword boundary.

Appendix C

Using ASTCR at High MEMCLK Frequencies

1. Using ASTCR

Whenever the ARM processor performs a memory cycle, it is clocked by MCLK derived from MEMCLK. The I/O controller inside CL-PS7500FE is clocked by derivatives of I_OCLK. Thus, when the ARM processor performs a read from or a write to an area of I/O space, some synchronization must occur.

The CL-PS7500FE bus controller decodes the address of the ARM processor access and, if it recognizes it as an I/O access, must send an I/O cycle request signal to the I/O controller. This is synchronized to the internal I/O clock, IOCK32. The I/O controller then performs the necessary cycle, asserting one (or more) of the I/O chip select signals, (for example, nCCS).

When the I/O controller can determine that the I/O cycle is about to finish, it asserts an I/O grant signal synchronized back to the internal memory clock, MEMRFCK. The bus controller then terminates the cycle by creating a falling edge on MCLK that clocks the ARM processor.

The address from the ARM processor is latched when MCLK is low so that it is held stable throughout I/O cycles (as well as ROM). It is important that MCLK does not fall too quickly after the end of the I/O chip select. This could cause the address to change too quickly, violating the required hold time. The CL-PS7500FE is designed to support an MEMCLK running at a frequency much higher than I_OCLK. The I/O grant generated by the I/O controller is synchronized quickly back to MEMRFCK and the address changes sooner after the end of the I/O chip select. To ensure the address hold time is maintained, the I/O controller must delay the point where it generates the I/O grant.

Using ASTCR, bit 0x032000CC allows the address hold time to be maintained when the MEMCLK frequency is greater than the I_OCLK frequency, at the same time not imposing greater-than-necessary wait states when MEMCLK is the same, or lower frequency than I_OCLK.

For a given system, fix the I_OCLK frequency at 32 MHz, while the MEMCLK frequency is fixed according to the speed of DRAM used. The amount of hold time required between the end of the I/O chip select and the latched address change is determined, then ASTCR is set based on the following details.

2. ASTCR I/O Cycle Type and Hold Times

NOTE: This assumes Divide-by-1 mode for the clocks, MEMCLK and I_OCLK.

When ASTCR is low (reset value):

I/O cycle type	Minimum hold time
Simple I/O	2 MEMCLK periods – 1 I_OCLK period
Module I/O	2 MEMCLK periods – 1.5 I_OCLK periods
PC style I/O	2 MEMCLK periods – 1.5 I_OCLK periods

When ASTCR is high:

I/O cycle type	Minimum hold time
Simple I/O	2 MEMCLK periods – 0.5 I_OCLK periods
Module I/O	2 MEMCLK periods – 0.5 I_OCLK periods
PC style I/O	2 MEMCLK periods – 0.5 I_OCLK periods

3. Example

In a system with:

- **I_OCLK** = 32 MHz
- **MEMCLK** = 40 MHz

the minimum hold time for a PC-style access is:

- 3.125 ns, if ASTCR = 0
- 34.375 ns, if ASTCR = 1

In addition there is a small amount of extra hold time, due to the delay from the internal memory clock, to the latch enable signal for the address.

NOTE: These times refer to the signals changes at the pad on the inside of the CL-PS7500FE. The relative capacitive loading of the latched address and I/O chip select determines the overall timing.

Appendix D

Expanding PC-Style I/O to 32 Bits

1. 32-Bit I/O

The CL-PS7500FE provides 16-bit I/O accesses as standard using the BD[15:0] port for all I/O types. The PC-style I/O access, however, can be extended to allow full 32-bit access without any loss in access speed by adding external 16-bit transceivers. The CL-PS7500FE provides all control signals required to support these external devices.

During PC-style I/O write cycles, the I/O controller routes the lower, 16-bit half-word from the ARM processor data bus onto BD[15:0] and drives the upper, 16-bit half-word onto D[31:16].

During read cycles, the ARM processor data bus is driven from two sources:

- the lower half-word from the data latched from BD[15:0]
- the upper half-word from D[31:16]

If the external devices that provide the upper half-word of data are not present or the I/O peripheral does not support more than 16 bits, the software must ignore the upper half-word read back into the ARM processor registers.

Figure D-1 on page 235 is an example of the system connections required to provide a full 32-bit I/O interface.

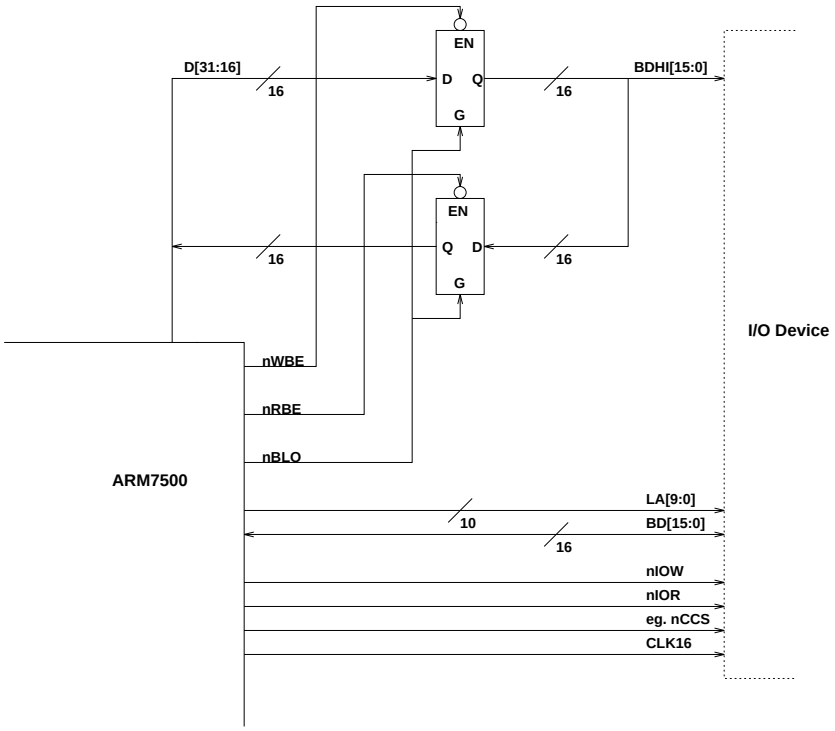


Figure D-1. 32-bit I/O Interface

The write and read path each contain a 16-bit latch with tristate output enable control:

- The write latch latches data from D[31:16] when nBLO is high and drive the latched data onto the expanded I/O bus, BDHI[15:0], when nWBE is active-low.
- The read latch should latch data from BDHI[15:0] when nBLO is high and drive the latched data onto D[31:16], when nRBE is active-low.

NOTE: Like the BD[15:0] bus, the write enable nWBE remains active-low by default. It is deasserted only during the read cycles, thus the I/O device must not attempt to drive BD[15:0] or BDHI[15:0], except when a read cycle occurs.

Appendix E

CL-PS7500FE Video Clock Sources

1. Introduction

To facilitate the high-resolution screen modes that the CL-PS7500FE can produce, apply a suitable high-frequency clock. As the screen mode changes, the pixel rate must also change. This can be done:

- through the various clock inputs
- by the on-chip *prescaler*
- using an external *voltage controlled oscillator*, combined with the on-chip phase comparator, to form a PLL.

It is intended that most systems be built with a PLL system. The required circuitry is simple and allows a high degree of flexibility. The advantages are that all the necessary clock frequencies can be derived from one circuit, eliminating the need for multiple on-board crystals and clock-switching circuitry.

2. Clock Sources

CL-PS7500FE has three primary inputs for its pixel clock:

- HCLK
- VCLKI
- RCLK (this is the internal 32-MHz IOCK32 derived from I_OCLK)

VCLKI and the internal I_OCLK32 (derived from I_OCLK) drive the phase comparator, and HCLK only provides the highest-frequency clock – if this frequency is above the maximum VCO frequency.

The pixel clock source is selected by programming bits 0 and 1 of the control register. The pixel clock selected can then be passed through a prescaler to divide the clock by between 1 and 8. This is done by programming CONREG[4:2]. See Chapter 16.

SCLCK

In addition to the pixel clock inputs, there is one other clock input, SCLK.

The sound system can be clocked from the internal 32-MHz IOCK32 or a 16-MHz SCLK (there is a divide-by-2 in the sound system). The digital sound system can run at a different frequency (low MHz range); this must be applied directly to SCLK.

NOTE: Any unused clock pin should be tied low.

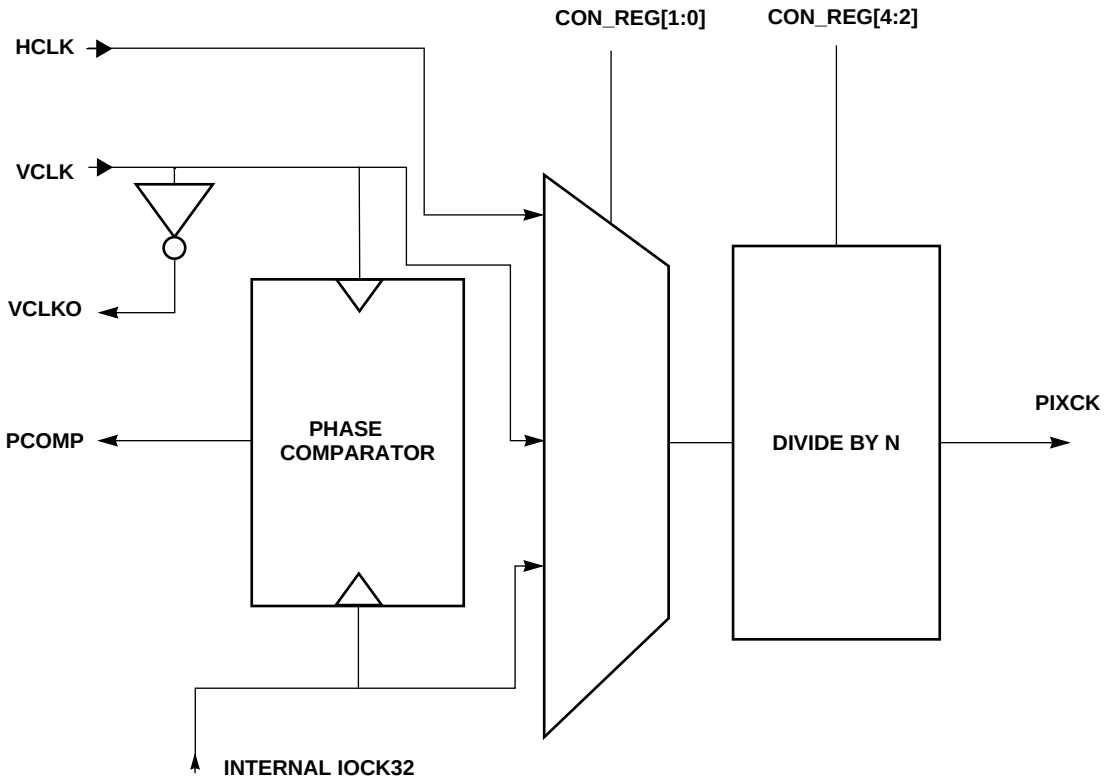


Figure E-1. Video and Sound Macrocell Internal Clock System

3. Using the Phase Comparator

The video and sound macrocell contains a phase comparator that, in conjunction with an external VCO, can build a PLL.

The phase comparator contains:

- two counters
- a phase detector

The counters are pre-loadable down counters, one clocked from the internal IOCK32 signal derived from I_OCLK, and the other clocked from VCLKI. The moduli of the counters is programmed in the Frequency Synthesizer register.

In the Frequency Synthesizer register, the test bits are:

bit 6	force PCOMP high and driven
bit 7	clear r-modulus counter
bit 14	force PCOMP low and driven
bit 15	clear v-modulus counter

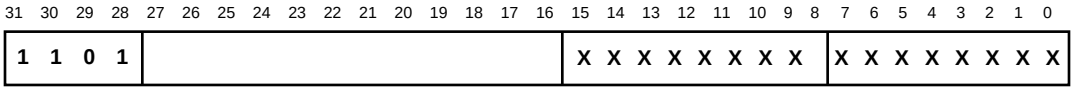


Figure E-2. Frequency Synthesizer Register

These bits are only programmed during test and at reset (see Section 4 on page 240).

The internal IOCK32 signal derived from the I_OCLK input, provides a reference clock that is recommended to be 32 MHz. The VCLKI input is driven from the output of the VCO, then this is selected as the pixel clock.

The VCO is driven by the CL-PS7500FE PCOMP output, for most of the time is at the tristate value. When the VCO frequency must be increased, PCOMP goes high, and vice-versa when the frequency must be decreased.

The PCOMP output requires a filter before applying to the VCO. The user must select the filter and VCO. A very simple and effective system can be built using an 74AC04 inverter pack, and a simple LC filter. The filtered VCO output controls the operating voltage of the 74AC04 device. This system is shown in Figure E-3, and has wide range of frequencies (LF to hundreds of MHz).

Since the output of this VCO is AC-coupled, VCLKI must be biased at the mid-voltage point. To do this, connect a large resistor between VCLKI and VCLKO (VCLKO is the inversion of VCLKI).

NOTE: Low-power systems may require more complex circuitry to avoid DC paths during SUSPEND or STOP modes.

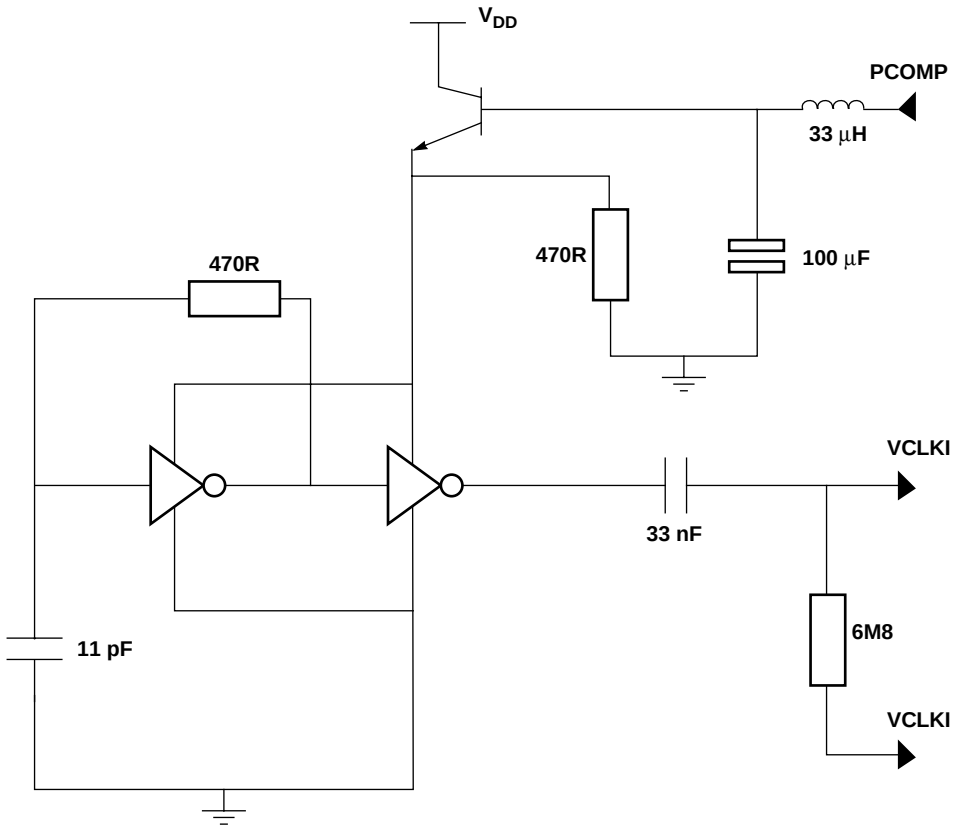


Figure E-3. Suggested VCO/PLL Circuit

The actual frequency of the VCO is determined by the ratio of the v-modulus to the r-modulus as shown in Equation E-1.

$$F_{VCO} = F_{REF} \times \frac{vmodulus}{rmodulus}$$

Equation E-1

NOTE: For a modulus of r, r – 1 is programmed, and likewise for the *vmodulus*.

Table E-1 shows a list of useful frequencies with corresponding values of r and v moduli, assuming a reference frequency of 32 MHz. There are many values of r and v that return the same ratio. The lower the value, the more frequently the output of the VCO is updated, so choose the r and v values to suit the response of the filter.

Table E-1. Synthesized VCO Frequency Settings

r Modulus	v Modulus	VCO Frequency (MHz)
8	2	8.0
16	6	12.0
4	2	16.0
8	6	24.0
2	2	32.0
8	9	36.0
16	35	70.0
4	15	120.0

4. Phase Comparator Reset

The phase comparator and VCO form a closed-loop feedback system that can become unstable. If the system powers up in the state where the PCOMP output is trying to drive the VCO output higher, it can quickly reach a frequency where the phase comparator cannot resolve making recovery is impossible. To avoid this, carefully apply the following reset procedure.

4.1 Reset Procedure

The test bits in the Frequency Synthesizer register can force the phase comparator output either high or low. Soon after power-up, program this register with:

- bits 15, 14, and 7 high
- bit 6 low

The r and v moduli can have anything programmed into them, but r must be greater than v. This operation forces the VCO frequency to decrease.

Program the real pixel rate in two steps:

1) Program the values of the r and v moduli, but leave the test bits in the initialization state.

2) Clear all test bits.

The VCO then ramps up to its operating frequency. Subsequently, a change of frequency can be achieved simply by reprogramming the r and v moduli.

Appendix F

CL-PS7500FE Test Modes

1. Introduction

The CL-PS7500FE has a pin, nTEST, combined with the nINT8, nINT3, and nINT6 pins to set the device into various test modes. Most of these are intended for use only during production testing to allow the individual macrocells within the CL-PS7500FE to be tested directly from the external pins using a MUX-isolation scheme.

2. Test Modes Description

When the nTEST pin is high, the CL-PS7500FE is in normal operating mode irrespective of the states of nINT8, nINT3, and nINT6. However, when nTEST is set low, the chip is set into one of five possible test modes dependent on the state of the three inputs nINT8, nINT3, and nINT6. Four of these test modes are reserved for use on the tester.

However there is one test mode that, when selected, causes all the CL-PS7500FE outputs to be tristate. This test mode is accessed by setting nTEST = 0, nINT8 = 0, nINT3 = 1, and nINT6 = 1.

IMPORTANT: Select no other combinations.

Notes

Appendix G

CL-PS7500FE Pinout Differences from the ARM7500

1. Introduction

Table G-1 documents the pinout differences between the CL-PS7500FE and the ARM7500 processors.

Table G-1. CL-PS7500FE and ARM7500 Pin Differences

Pin Number	ARM7500 Pin Name	CL-PS7500FE Pin Name
61	VDD_SOUND	VDD
62	LP	VDD
63	RP	VSS
64	RM	VSS
65	LM	VDD_CORE
66	VSS_SOUND	VSS
67	SIREF	VSS_CORE
68	SDO_MUTE	SDO (Name change only)
71	WS_LNR	WS (Name change only)

Notes

INDEX

W

Notes

Direct Sales Offices

Domestic

N. CALIFORNIA
Fremont
TEL: 510/623-8300
FAX: 510/252-6020

S. CALIFORNIA
Westlake Village
TEL: 805/371-5860
FAX: 805/371-5861

NORTHWESTERN AREA
Portland, OR
TEL: 503/620-5547
FAX: 503/620-5665

SOUTH CENTRAL AREA
Austin, TX
TEL: 512/255-0080
FAX: 512/255-0733

Irving, TX
TEL: 972/252-6698
FAX: 972/252-5681

Houston, TX
TEL: 281/257-2525
FAX: 281/257-2555

NORTHEASTERN AREA
Andover, MA
TEL: 978/794-9992
FAX: 978/794-9998

SOUTHEASTERN AREA
Raleigh, NC
TEL: 919/859-5210
FAX: 919/859-5334

Boca Raton, FL
TEL: 561/241-2364
FAX: 561/241-7990

International

CHINA
Beijing
TEL: 86/10-6428-0783
FAX: 86/10-6428-0786

FRANCE
Paris
TEL: 33/1-48-12-2812
FAX: 33/1-48-12-2810

GERMANY
Herrsching
TEL: 49/81-52-92460
FAX: 49/81-52-924699

HONG KONG
Tsimshatsui
TEL: 852/2376-0801
FAX: 852/2375-1202

ITALY
Milan
TEL: 39/2-3360-5458
FAX: 39/2-3360-5426

JAPAN
Tokyo
TEL: 81/3-3340-9111
FAX: 81/3-3340-9120

KOREA
Seoul
TEL: 82/2-565-8561
FAX: 82/2-565-8565

SINGAPORE
TEL: 65/743-4111
FAX: 65/742-4111

TAIWAN
Taipei
TEL: 886/2-2718-4533
FAX: 886/2-2718-4526

UNITED KINGDOM
London, England
TEL: 44/1727-872424
FAX: 44/1727-875919

High-Value Systems in Silicon

Cirrus Logic is a premier supplier of system-level integrated circuits that demand high-performance mixed-signal processing. Our company's software-rich 'systems in silicon' add high value to major brands worldwide. We apply our system expertise to enable high-volume applications in data storage, networking, and multimedia for both computing and consumer electronics markets as well as ultra-high-precision data acquisition applications for industrial automation and instrumentation markets.

Cirrus Logic's manufacturing strategy ensures maximum product quality and availability, as well as access to world-class processing technologies through joint ventures with IBM® and Lucent Technologies®.

Contact one of our systems and applications specialists to see how your company can benefit from the high value Cirrus Logic adds to its customers' products.

Copyright © 1997 Cirrus Logic, Inc. All rights reserved.

Advance product information describes products that are in development and subject to developmental changes. Cirrus Logic, Inc. has made best efforts to ensure that the information contained in this document is accurate and reliable. However, the information is subject to change without notice. No responsibility is assumed by Cirrus Logic, Inc. for the use of this information, nor for infringements of patents or other rights of third parties. This document is the property of Cirrus Logic, Inc. and implies no license under patents, copyrights, or trade secrets. No part of this publication may be copied, reproduced, stored in a retrieval system or transmitted in any form or by any means (electronic, mechanical, photographic, or otherwise) or used as the basis for manufacture or sale of any items without the prior written consent of Cirrus Logic, Inc. Cirrus, Cirrus Logic, AccuPak, Alpine, Clear3D, Crystal, CrystalClear, CrystalWare, DirectVPM, DIVA, FastEn, FastPath, FasText, FeatureChips, FilterJet, Get into it, Good Data, IntelliFilter, Laguna, Laguna3D, LagunaTV, Matterhorn, MediaDAC, MediaMax, Mojave, MotionVideo, MVA, SimulSCAN, S/LA, SmartAnalog, SMASH, SofTarget, SoundFusion, Stargate, Systems in Silicon, TextureJet, True-D, TVTap, UXART, VisualMedia, VPM, V-Port, V-Port Manager, Voyager, WavePort, and WebSet are trademarks of Cirrus Logic, Inc., which may be registered in some jurisdictions. Other trademarks in this document belong to their respective companies. CRUS and Cirrus Logic International, Ltd. are trade names of Cirrus Logic, Inc.

Cirrus Logic, Inc.
3100 West Warren Ave., Fremont, CA 94538
TEL: 510/623-8300 FAX: 510/252-6020

Publications Ordering: 800/359-6414 (USA) or 510/249-4200
Worldwide Web: http://www.cirrus.com
447500-001