16 Memory and I/O Programmers' Model

This chapter details the programmable registers for the memory and I/O subsystem.

Memory and I/O Programmers’ Model

16.1 Introduction

The ARM7500FE contains over 100 programmable registers (in addition to those in the ARM processor, the FPA coprocessor and the 256 video palette entries), which are grouped into three sets. Those inside the ARM processor are described fully in Chapters 3 to 7 and those inside the FPA coprocessor in Chapters 8 to 10.
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 the video and sound chapters. The remaining ARM7500FE 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 ARM7500FE registers which do not form part of the ARM processor, the FPA coprocessor, or the video and sound macrocell are described in the following section.

16.2 Summary of Registers

All addresses are in hex and are relative to the base address 0x03200000.
In the following table:

✓	means can write or read
✗	means do not write or read

Name	Address	Size	Read	Write	Function
IOCR	00	8	✓	✓	I/O control
KBDDAT	04	8	✓	✓	Keyboard data
KBDCR	08	8	✓	✓	Keyboard control
IOLINES	0C	8	✓	✓	General-purpose I/O lines
IRQSTA	10	8	✓	✗	IRQA status
IRQRQA	14	8	✓	✓	IRQA request/clear
IRQMSKA	18	8	✓	✓	IRQA mask
SUSMODE	1C	8	✓	SUSPEND	Enter SUSPEND mode
IRQSTB	20	8	✓	✗	IRQB status
IRQRQB	24	8	✓	✗	IRQB request
IRQNSKB	28	8	✓	✓	IRQB mask
STOPMODE	2C	8	✗	STOP	Enter STOP mode
FIQST	30	8	✓	✗	FIQ status

Table 16-1: ARM7500FE registers

Name	Address	Size	Read	Write	Function
FIQRQ	34	8	✓	✗	FIQ request
FIQMSK	38	8	✓	✓	FIQ mask
CLKCTL	3C	8	✓	✓	Clock divider control
T0LOW	40	8	✓	✓	Timer 0 LOW bits
T0HIGH	44	8	✓	✓	Timer 0 HIGH bits
T0GO	48	8	✗	GO	Timer 0 go command
T0LAT	4C	8	✗	LATCH	Timer 0 latch command
T1LOW	50	8	✓	✓	Timer 1 LOW bits
T1HIGH	54	8	✓	✓	Timer 1 HIGH bits
T1GO	58	8	✗	GO	Timer 1 go command
T1LAT	5C	8	✗	LATCH	Timer 1 latch command
IRQSTC	60	8	✓	✗	IRQC status
IRQRQC	64	8	✓	✗	IRQC request
IRQMSKC	68	8	✓	✓	IRQC mask
VIDMUX	6C	8	✓	✓	LCD and IIS control bits
IRQSTD	70	8	✓	✗	IRQD status
IRQRQD	74	8	✓	✗	IRQD request
IRQMSKD	78	8	✓	✓	IRQD mask
ROMCR0	80	8	✓	✓	ROM control bank 0
ROMCR1	84	8	✓	✓	ROM control bank 1
REFCR	8C	8	✓	✓	Refresh period
ID0	94	8	✓	✗	Chip ID number LOW byte
ID1	98	8	✓	✗	Chip ID number HIGH byte
VERSION	9C	8	✓	✗	Chip version number
MSEDAT	A8	8	✓	✓	Mouse data

Table 16-1: ARM7500FE registers (Continued)

Name	Address	Size	Read	Write	Function
MSECR	AC	8	✓	✓	Mouse control
IOTCR	C4	8	✓	✓	I/O timing control register
ECTCR	C8	8	✓	✓	Expansion card timing control register
ASTCR	CC	8	✓	✓	Asynchronous I/O timing control
DRAMCTL	D0	8	✓	✓	DRAM control
SELFREF	D4	8	✓	✓	Force CAS/RAS lines LOW individually for self refresh
ATODICR	E0	8	✓	✓	A to D interrupt control register
ATODSR	E4	8	✓	✗	A to D status register
ATODCC	E8	8	✓	✓	A to D convertor control register
ATODCNT1	EC	16	✓	✗	A to D counter 1
ATODCNT2	F0	16	✓	✗	A to D counter 2
ATODCNT3	F4	16	✓	✗	A to D counter 3
ATODCNT4	F8	16	✓	✗	A to D counter 4
SD0CURA	180	32	✓	✓	Sound DMA 0 CurA
SD0ENDA	184	32	✓	✓	Sound DMA 0 EndA
SD0CURB	188	32	✓	✓	Sound DMA 0 CurB
SD0ENDB	18C	32	✓	✓	Sound DMA 0 EndB
SD0CR	190	8	✓	✓	Sound DMA control
SD0ST	194	8	✓	✗	Sound DMA Status
CURSCUR	1C0	32	✓	✓	Cursor DMA current
CURSINIT	1C4	32	✓	✓	Cursor DMA Init
VIDCURB	1C8	32	✓	✓	Duplex LCD current register B
VIDCURA	1D0	32	✓	✓	Video DMA current A
VIDEND	1D4	32	✓	✓	Video DMA End

Table 16-1: ARM7500FE registers (Continued)

Name	Address	Size	Read	Write	Function
VIDSTART	1D8	32	✓	✓	Video DMA start
VIDINITA	1DC	32	✓	✓	Video DMA Init
VIDCR	1E0	8	✓	✓	Video cursor DMA control
VIDINITB	1E8	32	✓	✓	Duplex LCD init register B
DMAST	1F0	8	✓	✗	DMA interrupt status
DMARQ	1F4	8	✓	✗	DMA interrupt request
DMASK	1F8	8	✓	✓	DMA interrupt mask

Table 16-1: ARM7500FE registers (Continued)

16.3 Register Description

16.3.1 IOCR (0x00) - I/O control

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

This register is used to control 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 would be important for genlocking as FLYBACK will provide 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 should only be done during the FLYBACK period, when this bit has been 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)

16.3.2 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

16.3.3 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

16.3.4 IOLINES (0x0C) - IOP[7:0] port control

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

This register is the control for the 8-bit I/O port included in the ARM7500FE. 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

16.3.5 IRQSTA (0x10) - IRQ A interrupts status

7	6	5	4	3	2	1	0
1	T	U	R	F	N	0	P

This is the first of four sets of IRQ interrupt control, masking and status registers in ARM7500FE. Not all the bits in each register are used. Note that this status register contains a bit (7) which is always active, and this can be used to force an interrupt from software by programming the corresponding bit in the IRQA mask register HIGH.

1	always active bit
T	2MHz timer 1, rising edge triggered
U	2MHz 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 zero power on reset sets bit[4] to 1 push button reset maintains the current bit[4] value

16.3.6 IRQRQA (0x14) - IRQ A interrupts request/clear

7	6	5	4	3	2	1	0
1	T	U	R	F	N	X	P

1	always active bit	
T	2MHz timer 1, rising edge triggered	
U	2MHz 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	don't clear interrupt
	1	clear interrupt
Read	requests, as status, but bitwise ANDed with mask	

16.3.7 IRQMSKA (0x18) - IRQ A interrupts mask

7	6	5	4	3	2	1	0
1	T	U	R	F	N	0	P

1	always active bit	
T	2MHz timer 1, rising edge triggered	
U	2MHz 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	don't form part of nIRQ
	1	form part of nIRQ
Read	value set by write	
Reset	set all zeros (none affect nIRQ)	

16.3.8 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 ARM7500FE into SUSPEND mode. Only one bit (0) is used, and writing to this bit will cause SUSPEND mode to be entered.
The value written to bit 0 determines whether the external I/O clocks, normally output from the chip, are also disabled during SUSPEND mode. The value programmed will depend on the nature of the peripherals being driven by those clocks.

S	SUSPEND mode control of external I/O clocks. Enter SUSPEND mode with MCLK,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	don't turn off
Read	return above value	
Reset	set to zero	

16.3.9 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

16.3.10 IRQRQB (0x24) - IRQ B interrupts request

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	request, status bitwise ANDed with mask

16.3.11 IRQMSKB (0x28) - IRQ B interrupts mask

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	set mask for each interrupt source: 0 don't form part of nIRQ 1 form part of nIRQ
Read	value set by write
Reset	set all zeros (none affect nIRQ)

16.3.12 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 will be permanently forced out on the main data bus during STOP mode, and for most systems it will be 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

16.3.13 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

16.3.14 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 ANDed with mask

16.3.15 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 don't form part of nFIQ
	1 form part of nFIQ
Read	value set by write
Reset	set all zeros (none affect nFIQ)

16.3.16 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 will be reset such that all three main clocks have a divide by 2 prescale at the inputs to the chip. This register will sometimes need to be reprogrammed as part of the initial tasks of the operating system, to set the prescalers 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 x 2 = **CPUCLK**
	1 FCLK = **CPUCLK**
	bit[1]
	0 MEMRFCK x 2 = **MEMCLK**
	1 MEMRFCK = **MEMCLK**
	bit[0]
	0 IOCK32 x 2 = **I_OCLK**
	1 IOCK32 = **I_OCLK**
Read	return above value

Power On Reset only

set all to zero, i.e. divide by 2 clocks
Push button reset does not affect this register

16.3.17 T0LOW (0x40) - timer 0 LOW bits

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

There are eight registers associated with the two 16-bit timers in ARM7500FE.

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

16.3.18 T0HIGH (0x44) - timer 0 HIGH bits

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

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

16.3.19 T0GO (0x48) - timer 0 Go command

Write	load counter with HIGH and LOW latch values and start decrementing (value ignored)
Read	ignored

16.3.20 T0LAT (0x4C) - timer 0 Latch command

Write	latch timer value in HIGH and LOW count latches (value ignored)
Read	ignored

16.3.21 T1LOW (0x50) - timer 1 LOW bits

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

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

16.3.22 T1HIGH (0x54) - timer 1 HIGH bits

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

H	HIGH byte of timer
Write	set HIGH byte latch value which is loaded into timer when it reaches end count
Read	read value of HIGH count latched by the 'Latch' command T1LAT

16.3.23 T1GO (0x58) - timer 1 Go command

Write	load counter with HIGH and LOW latch values and start decrementing (value ignored)
Read	ignored

16.3.24 T1LAT (0x5C) - timer 1 Latch command

Write	latch timer value in HIGH and LOW count latches (value ignored)
Read	ignored

16.3.25 IRQSTC (0x60) - IRQ C interrupts status

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

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

16.3.26 IRQRQC (0x64) - IRQ C interrupts request

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

I	**IOP[7:0]** pins, active LOW
Write	ignored
Read	request, status bitwise ANDed with mask

16.3.27 IRQMSKC (0x68) - IRQ C interrupts mask

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

I	**IOP[7:0]** pins, active LOW	
Write	set mask for each interrupt source	
	0	don't form part of nIRQ
	1	form part of nIRQ
Read	value set by write	
Reset	set all zeros (none affect nIRQ)	

16.3.28 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 Sound Features chapter.
Bit 0	controls the color LCD multiplexer which is 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 zero (normal)	

16.3.29 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

16.3.30 IRQRQD (0x74) - IRQ D interrupts request

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

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 ANDed with mask

16.3.31 IRQMSKD (0x78) - IRQ D interrupts mask

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

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 don't form part of nIRQ 1 form part of nIRQ
Read	value set by write
Reset	set all zeros (none affect nIRQ)

16.3.32 ROMCR0,1 (0x80,0x84) - ROM control

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

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, ARM7500FE doubles the length of all the timings and will allow 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) which (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 ARM7500FE.

Bit 7 allows writes to occur to this address space; the data will be 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, ie. double the above delays when H=0. Normally, the H bit should be programmed to 1 (normal speed)	
S	16/32-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, i.e. the 16-bit, slowest access time, to ensure all systems can be booted from reset.	

16.3.33 REFCR (0x8C) - refresh period

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

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	16us
	0010	32us
	0100	64us
	1000	128us
	all others are undefined	
Read	return the above values	
Reset	set to 0001 (fastest available refresh rate)	

16.3.34 ID0 (0x94) - chip ID number LOW byte

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

The ID registers and the version register read back the ARM7500FE ID and version numbers. These registers are read only and must NOT be written to, as they are used to set the ARM7500FE into special modes during production test.

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

16.3.35 ID1 (0x98) - chip ID number HIGH byte

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

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

16.3.36 VERSION (0x9C) - chip version number

Write	ignored
Read	chip version number byte

16.3.37 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.

16.3.38 MSECR (0xAC) - mouse control

As KBDCR register.

16.3.39 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	**NPCCS1/2** 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	

16.3.40 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	

16.3.41 ASTCR (0xCC) - I/O asynchronous timing control

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

This register is used where I/O is being used with a very fast memory system clock. Normally it will always be programmed to zero to give the minimum delay for these cycles; however, in some configurations it may be necessary to program the register bit to one 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

16.3.42 DRAMCTL (0xD0) - DRAM control

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

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 non-sequential 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	16/32-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)	

16.3.43 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 DRAM, so that before the ARM7500FE 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 NCAS's LOW	
R	force NRAS's 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)	

16.3.44 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 such that various combination of interrupts from the channels can be used to generate an interrupt request in the IRQD interrupt request register. It should be noted that the logical OR of all four basic enables is used to power 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] is used to power up all the comparators. Thus they reset to the powered-up state.*

16.3.45 ATODSR (0xE4) - A TO D status

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

This register shows which of the A TO D channels have been 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 ANDed with the interrupt enables from the interrupt control register.

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
	bit [3:0]	
	0	not stopped
	1	stopped
Reset	set all zero (not requesting or stopped)	

16.3.46 ATODCC (0xE8) - A to D convertor control

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

The lower 4 bits of this register directly reset each of the four counters, so that they can be set back to zero before a new analog to digital conversion cycle takes place. The counter will start counting as soon as the relevant clear bit is set back to zero. The discharge transistor controls causes the channel comparator input to be pulled firmly down to Vss, thus discharging an external capacitor and ensuring zero volts across the capacitor until the discharge bit is programmed LOW again.
With the system connected as it is expected to be used, the external capacitor will begin charging as soon as the discharge bit is reset, so it is expected that the discharge bit would be reset at the same time as the counter clear bit for that channel is re-enabled.

D[3:0]	discharge transistor control for channels 4 to 1	
C[3:0]	clear counter for channels 4 to 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)	

16.3.47 ATODCNT1 (0xEC) - A to D counter 1

Write	ignored
Read	returns 16-bit counter value

16.3.48 ATODCNT2 (0xF0) - A to D counter 2

Write	ignored
Read	returns 16-bit counter value

16.3.49 ATODCNT3 (0xF4) - A to D counter 3

Write	ignored
Read	returns 16-bit counter value

16.3.50 ATODCNT4 (0xF8) - A to D counter 4

Write	ignored
Read	returns 16-bit counter value

16.3.51 SDCURA (0x180) - sound DMA current A

31 29	28 12	11 4	3 0
X X X	P P P P P P P P P P P P P P P P P	F F F F F F F F	0 0 0 0

The operation of the sound DMA channel is described in the Memory Subsystems chapter. 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 zero

16.3.52 SDENDA (0x184) - sound DMA end A

31	30	29 — 12	11 — 4	3 — 0
S	L	X X X X X X X X X X X X X X X X X X	E E E E E E E E	0 0 0 0

This register should be programmed with the offset within the page of the final quad word. Bit 30 should always be programmed to zero unless the channel is being initialized for a single transfer in which case it must be programmed HIGH.

S	stop bit
L	last bit
E	end[11:0]
Write	bit[31] stop bit:
	0 don't stop after reaching End
	1 stop after reaching End
	bit[30] last bit
	0 not last transfer
	1 last quad word 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 zero

16.3.53 SDCURB (0x188) - sound DMA current B

31 — 29	28 — 12	11 — 4	3 — 0
X X X	P P P P P P P P P P P P P P P P P	F F F F F F F F	0 0 0 0

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 zero

16.3.54 SDENDB (0x18C) - sound DMA end B

31	30	29 … 12	11 … 4	3 … 0
S	L	X X X X X X X X X X X X X X X X X X	E E E E E E E E	0 0 0 0

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	don't stop after reaching end
		1	stop after reaching end
	bit[30] last bit		
		0	not last transfer
		1	last quad word 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 zero		

16.3.55 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	don't 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 zero		
	bit[6] always reads zero		

	bit[5] enable
	0 disabled
	1 enabled
	bits[4:0] read as 10000 (binary), historically signifying a quadword transfer
Reset	enable set to zero

16.3.56 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 used to control 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)

16.3.57 CURSCUR (0x1C0) - cursor DMA current

31		29	28																								4	3			0
X	X	X	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	0	0	0	0

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 quad word 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 zero

16.3.58 CURSINIT (0x1C4) - cursor DMA init

31 29	28 4	3 0
X X X	I I I I I I I I I I I I I I I I I I I I I I I I I	0 0 0 0

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 zero

16.3.59 VIDCURB (0x1C8) - duplex LCD video DMA current B

31 29	28 4	3 0
X X X	C C C C C C C C C C C C C C C C C C C C C C C C C	0 0 0 0

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.

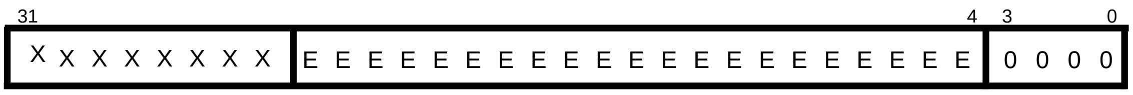
C	current fetch location B
Write	bits[31:29] unused
	bits[28:4] video current B DMA fetch location
	bits[3:0] ignored
Read	bits[31:29] undefined
	bits[28:4] video current B DMA fetch location
	bits[3:0] always zero

16.3.60 VIDCURA (0x1D0) - video DMA current A

31 29	28 4	3 0
X X X	C C C C C C C C C C C C C C C C C C C C C C C C C	0 0 0 0

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 zero

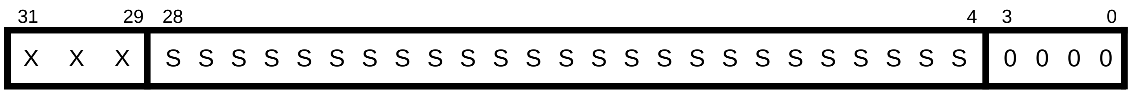
16.3.61 VIDEND (0x1D4) - video DMA end



The video END register should be loaded with the address of the final quadword 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 zero

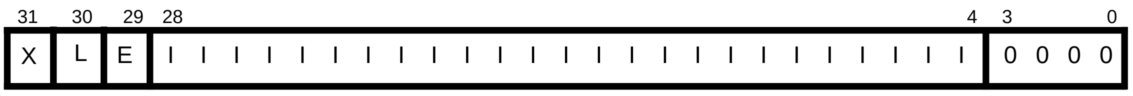
16.3.62 VIDSTART (0x1D8) - video DMA start



The video start register should be loaded with the location of the first quadword at the start of the video frame buffer. All the DMA control registers can only be loaded with quadword-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 zero

16.3.63 VIDINITA (0x1DC) - video DMA init A



For normal CRT displays and single panel LCD data only the 'A' registers are used. The init register should be loaded with the address within the frame buffer of the first quad word to be displayed in the first raster at the top of the screen. In the case of dual panel displays, this register should be loaded with the address of the first quadword in the frame buffer to be displayed at the top left of the upper panel.

The last bit (30) should only be set if the init A register has been 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] zero		
	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 zero		

16.3.64 VIDCR (0x1E0) - video DMA control

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

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

Note: *For driving normal CRT displays, bit 7 should be set to zero.*

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 read back one, DRAM mode		
	bits[4:0] read as 10000 (binary), historically meaning quadword transfer		
Reset	set to zero (disabled, normal mode)		

16.3.65 VIDINITB (0x1E8) - duplex LCD video DMA init B

31	30	29	28 4	3 0
X	L	E	I I I I I I I I I I I I I I I I I I I I I I I I I	0 0 0 0

For normal CRT displays and single panel LCD data only the 'A' registers are used, and this register should be programmed with all zeros. In the case of dual panel displays, this register should be loaded with the address of the first quadword in the frame buffer to be displayed at the top left of the lower panel. The last bit (30) should only be set if the init B register has been 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] zero		
	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 zero		

16.3.66 DMAST/DMARQ/DMAMSK (0x1F0,0x1F4,0x1F8) - DMA interrupt control

These 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 ANDed 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	don't affect nIRQ
	1	affect nIRQ
Read	mask	
	bits[7:5,3:0] unused	
	bit[4] read value written above	

17 Memory Subsystems

This chapter describes the ROM and DRAM interfaces, and the DMA channels.

Memory Subsystems

17.1 ROM Interface

The ARM7500FE 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, which can be externally split to give separate chip selects for two 16MB 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 which causes all ROM timings to take twice as many **MEMCLK** cycles, when the bit is set to zero.

The ROM interface of ARM7500FE can also support write cycles with the generation of an output enable and a write enable. The feature is disabled on reset such that write cycles will not:

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

When the 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 the 32 or 16-bit wide devices are not handled directly. External logic can be used to decode address bits **LA[1:0]** and the write enable to 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 32MHz, the access time for a non-sequential cycle can be varied from 220ns to 62.5ns in steps of 31.25ns. For burst mode cycles, **LA[3:2]** of the latched address from ARM7500FE are incremented to allow up to four sequential reads. The access time for burst mode cycles can be programmed from 125ns down to 62.5ns, again in steps of 31.25ns.

Note: *Due to the timing of the write enable, the smallest cycle length for a write cycle is 3 MEMCLK cycles, ie. 93.75ns.*

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

Support for 16-bit wide ROMs is provided via a programmable bit in each of the ROM control registers. If a 16-bit wide device is selected, then two memory system cycles will be required to fetch the full 32-bit word required by the ARM. If burst mode is disabled for that bank, then ARM7500FE will perform 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 will be a standard non-sequential cycle, but the second will be 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:

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

Contents

When this is read, the ARM will see:

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

17.1.1 ROM bank configuration and timing

There are two identical registers which 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, which 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 an ARM7500FE 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, i.e. 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, ie. 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 will not be active during accesses to ROM space.

17.1.2 Timing examples

Note: *All diagrams assume divide by 1 mode for **MEMCLK**.*

Figure 17-1: ROM read access timing without burst mode (32-bit mode) shows the timing of non-sequential and sequential 32-bit ROM accesses without burst mode.

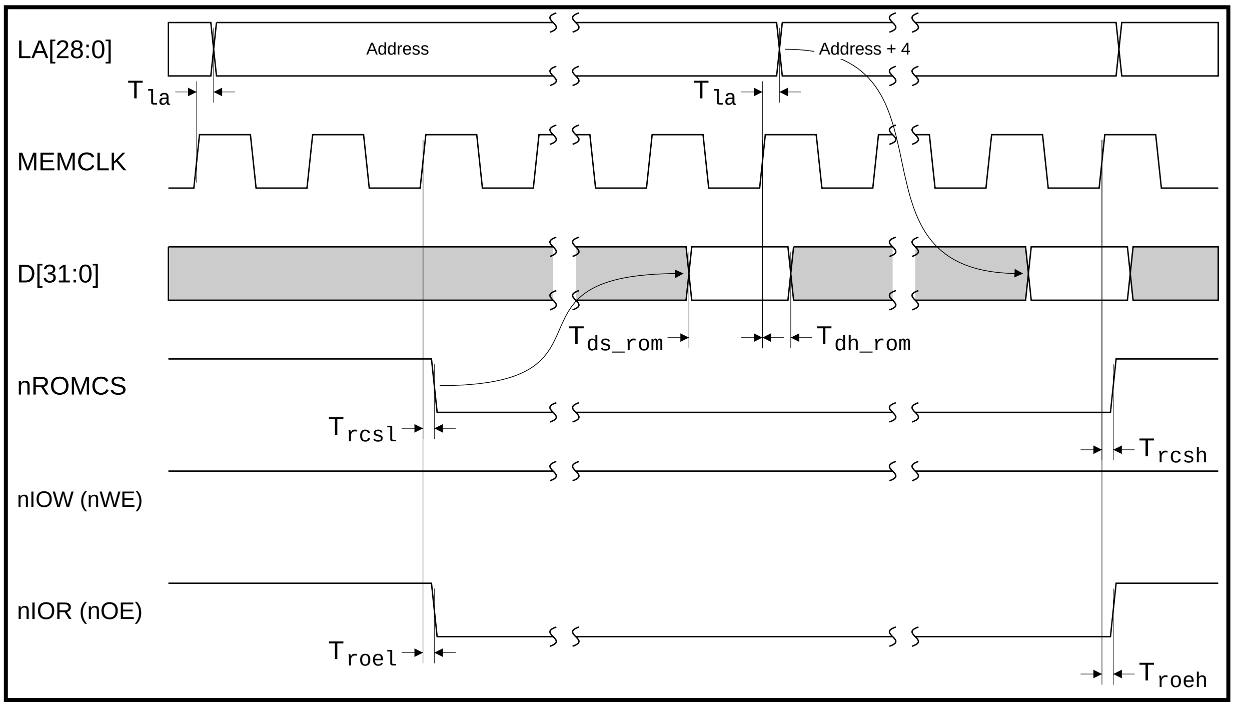


Figure 17-1: ROM read access timing without burst mode (32-bit mode)

Figure 17-2: ROM read access timing—burst mode (32-bit) shows the timing of non-sequential and sequential 32-bit ROM accesses with burst mode.

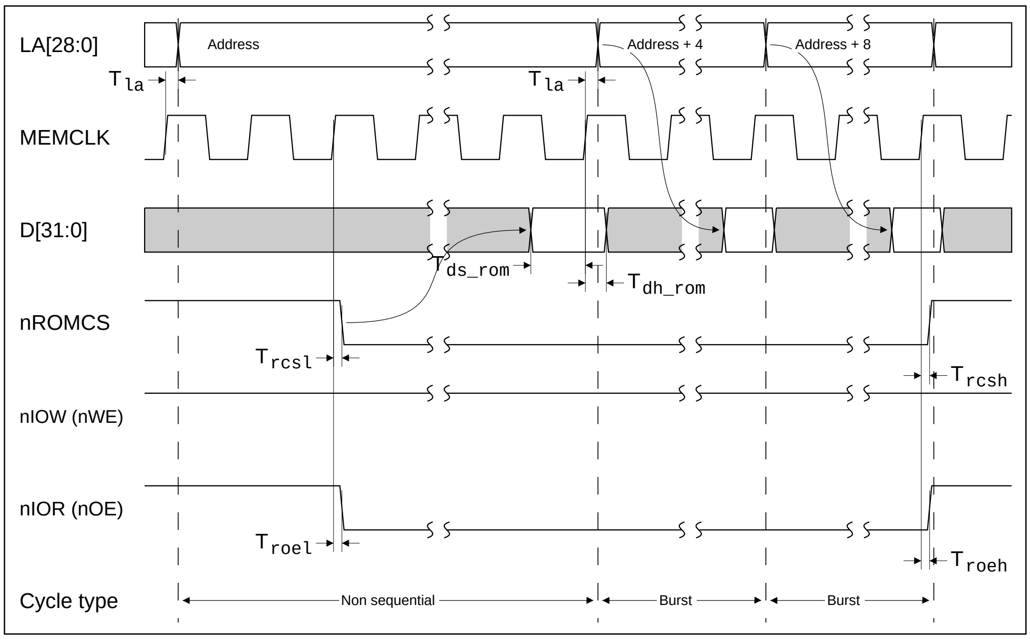


Figure 17-2: ROM read access timing—burst mode (32-bit)

Figure 17-3: ROM read access timing with burst mode—16-bit mode shows the timing of non-sequential and sequential 16-bit ROM accesses with burst mode.

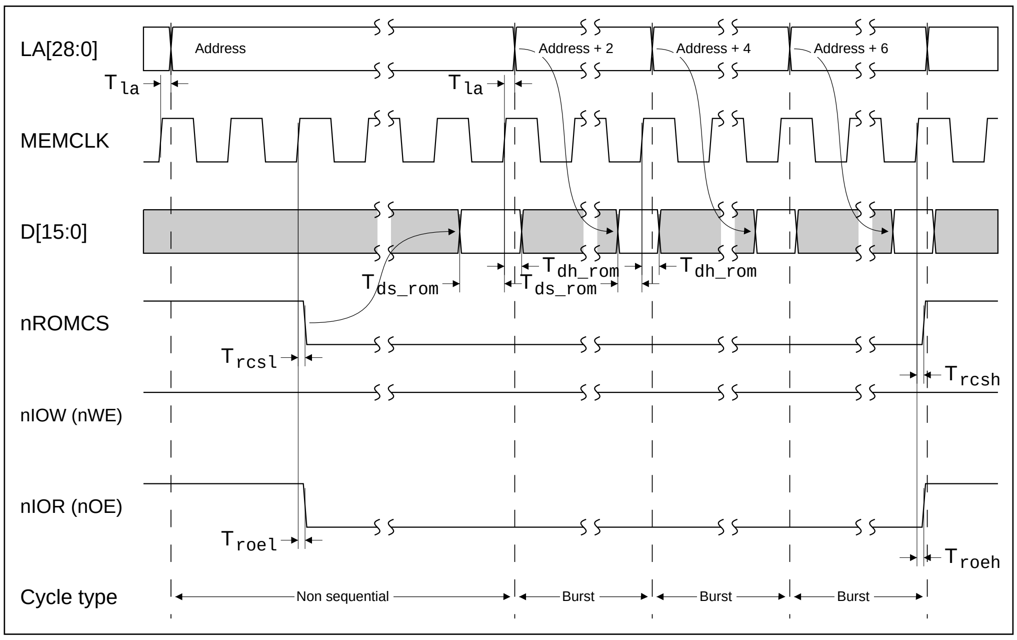


Figure 17-3: ROM read access timing with burst mode—16-bit mode

Figure 17-4: ROM write access with burst mode — (32-bit) on page 17-6 shows the timing of non-sequential and sequential 32-bit ROM write cycles with burst mode.

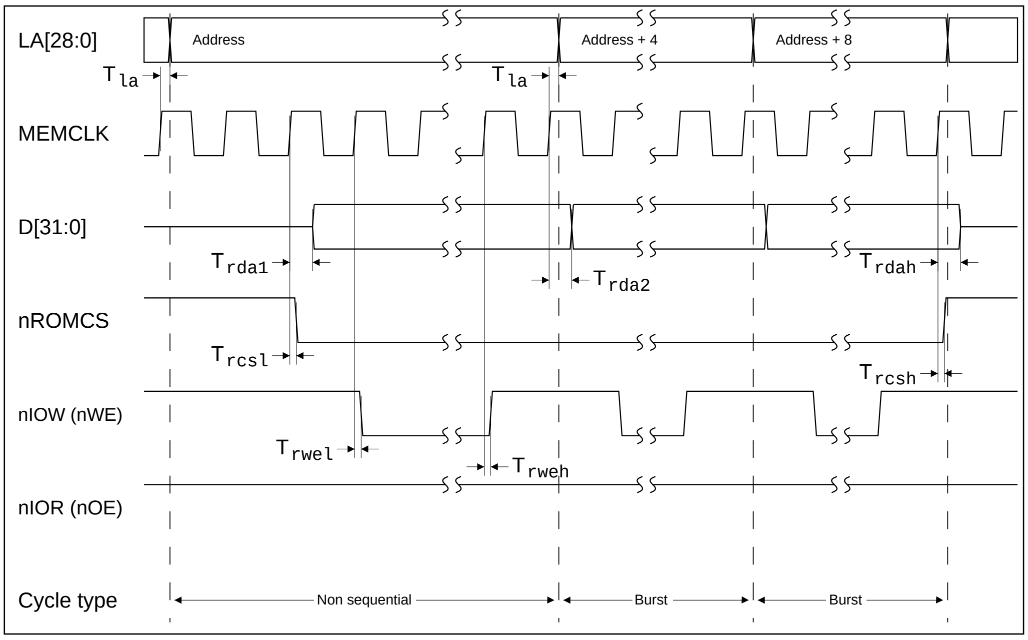


Figure 17-4: ROM write access with burst mode — (32-bit)

Figure 17-5: ROM write access with burst mode — (16-bit) shows a write cycle for a 16-bit ROM.

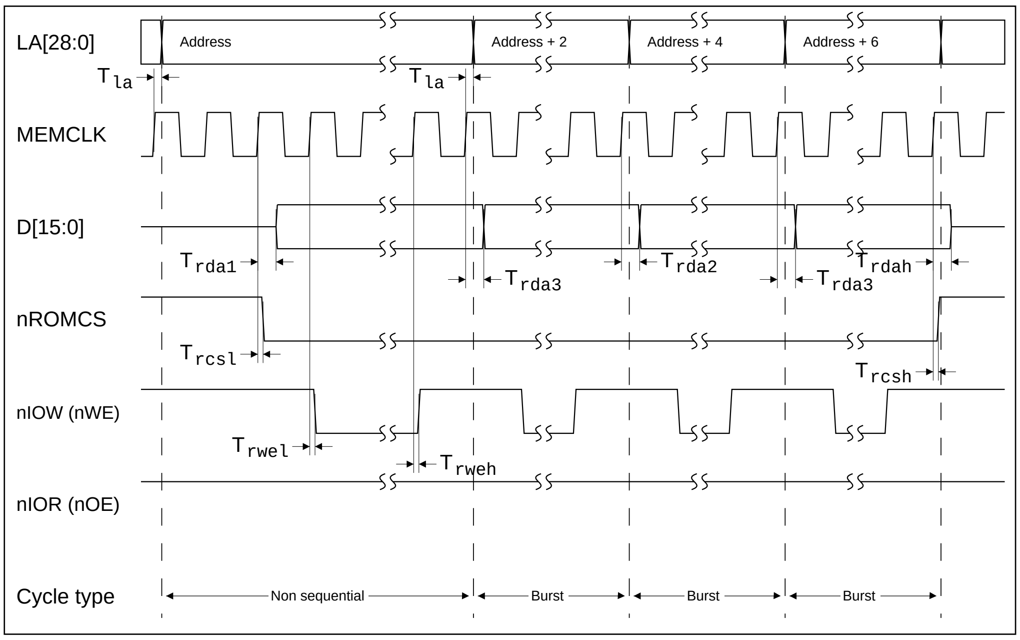


Figure 17-5: ROM write access with burst mode — (16-bit)

Symbol	Parameters	Min	Max	Units
Tla	**MEMCLK** rising to **LA[28:0]** changing		22	ns
Tds_rom	DATA setup to **MEMCLK** rising edge	0		ns
Trcsl	**MEMCLK** rising to **nROMCS** falling		14	ns
Trcsh	**MEMCLK** rising to **nROMCS** rising		14	ns
Tdh_rom	DATA hold from **MEMCLK** rising edge	12		ns
Trda1	**MEMCLK** rising to write DATA valid		15	ns
Trda2	**MEMCLK** rising to write DATA valid		33	ns
Trda3	**MEMCLK** rising to write DATA valid		16	ns
Trdah	Write DATA hold time after **MEMCLK** rising	11		ns
Troel	**MEMCLK** rising to **nIOR** (nOE) falling		14	ns
Troeh	**MEMCLK** rising to **nIOR** (nOE) rising		14	ns
Trwel	**MEMCLK** rising to **nIOW** (nWE) falling		14	ns
Trweh	**MEMCLK** rising to **nIOW** (nWE) rising		13	ns

Table 17-1: ARM7500FE ROM timing

Note: *The output delays above only include the intrinsic delay of the output pad driver. See section 22.5 De-rating* on page 22-6 *to calculate the final delay dependent upon the expected output load.*

17.2 DRAM Interface

The DRAM interface can directly drive four banks of DRAM to give a maximum of 64MB 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, which means that the DRAM address space will be non-contiguous if the full 64MB 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 ARM7500FE 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 via 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 precharge requirements. Page mode sequential cycles require two **MEMCLK** cycles.

17.2.1 DRAM control registers

There are three registers associated with DRAM control:

DRAMCTL 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 3 remaining bits:

- one selects EDO memory support
- one inserts an extra wait state between falling **nRAS** and falling **nCAS** on non-sequential cycles to preserve Trac
- the final bit selects between 3 and 4 MEMCLK cycles of minimum **nRAS[x]** precharge time, Trp

SELFREF allows direct forcing of the **nRAS** and **nCAS** outputs. The default state of each of these bits is zero, which allows 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 128µs to 16µs.

17.2.2 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 in turn.

17.2.3 Selection between 16 and 32-bit DRAM

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

The DRAMCTL register at address 0x032000D0 allows the width of each of the four DRAM banks to be defined for ARM7500FE. 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	16/32-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)	

17.2.4 EDO and timing mode selection

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

The DRAMCTL register at address 0x032000D0 also controls EDO mode and some other timing features. On reset all these bits are set low, ie. inactive. In many systems after reset these register bits will have to be programmed correctly before the DRAM is used to ensure reliable operation.

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 non-sequential cycles
	1	3 **MEMCLK** cycles RAS to CAS delay on non-sequential cycles
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)	

In order to take advantage of the faster page mode accesses provided by EDO DRAMs, the memory clock frequency should be increased accordingly. For example, a system using 80ns Fast Page DRAMs will need a memory clock in the region of 32MHz, whereas one using 80ns EDO DRAMs could use a memory clock of around 50MHz. This would improve the asymptotic DRAM bandwidth from 64MB/s to 100MB/s for a 32-bit wide system.

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

The P bit controls the guaranteed minimum RAS precharge time. The minimum time from rising **nRAS[x]** at the end of one access to the next falling **nRAS[y]** (different bank) will be 2 **MEMCLK** cycles. If a new non-sequential access to the same bank occurs, then with P=0 there will be 3 **MEMCLK** cycles of **nRAS[x]** high and with P=1 there will be 4 **MEMCLK** cycles of **nRAS[x]** high.

The R bit controls the number of ticks from the falling **nRAS** to the first falling **nCAS** at the start of non-sequential cycles (reads and writes). If R=0 then there will be 2 **MEMCLK** cycles between falling **nRAS** and **nCAS** and if R=1 then there will be 3 **MEMCLK** cycles. For reads this will ensure that the DRAM datasheet parameter Trac and Tcsh timings are not violated at faster memory clock frequencies. For writes this will ensure the Tcsh time is not violated at faster memory clock frequencies.

The E bit controls whether EDO DRAMS are being used. When E=0 then it is assumed fast page DRAMs are being used (or EDO with slow memory clock) and the data is internally latched at the end of the **nCAS** low time giving one **MEMCLK** for read access. When E=1 then it is assumed EDO DRAMs are being used and the data is internally latched 2 **MEMCLK** cycles after the falling **nCAS**. For both reads and writes the cycle will terminate with at least 1 **MEMCLK** where **nRAS** is still low but **nCAS** has returned high. This ensures that the DRAM datasheet parameter Tras, Trsh and Tral timings are met even for single non-sequential cycles.

17.2.5 DRAM interface timing specification

32-bit mode

In 32-bit mode, byte reads and writes have the same timing as word accesses, but only one **nCAS** output is selected according to the decode of bits 1 and 0 of the address

Note: All timing diagrams assume divide by 1 is selected for **MEMCLK**.

Figure 17-6: Fast page DRAM read timing (32-bit mode), shows the timing of non-sequential and sequential 32-bit DRAM read cycles.

Figure 17-7: Fast page DRAM write timing (32-bit mode) on page 17-12 shows the timing of both types of 32-bit DRAM write cycles.

Figure 17-8: EDO DRAM read timing (32-bit mode) on page 17-13 shows the timing of a multiple EDO read when bit 6 of DRAMCTL is set to extend the RAS to CAS delay.

Figure 17-9: Single word EDO DRAM write on page 17-13 shows the timing when bit 6 of DRAMCTL is set to extend the RAS to CAS delay.

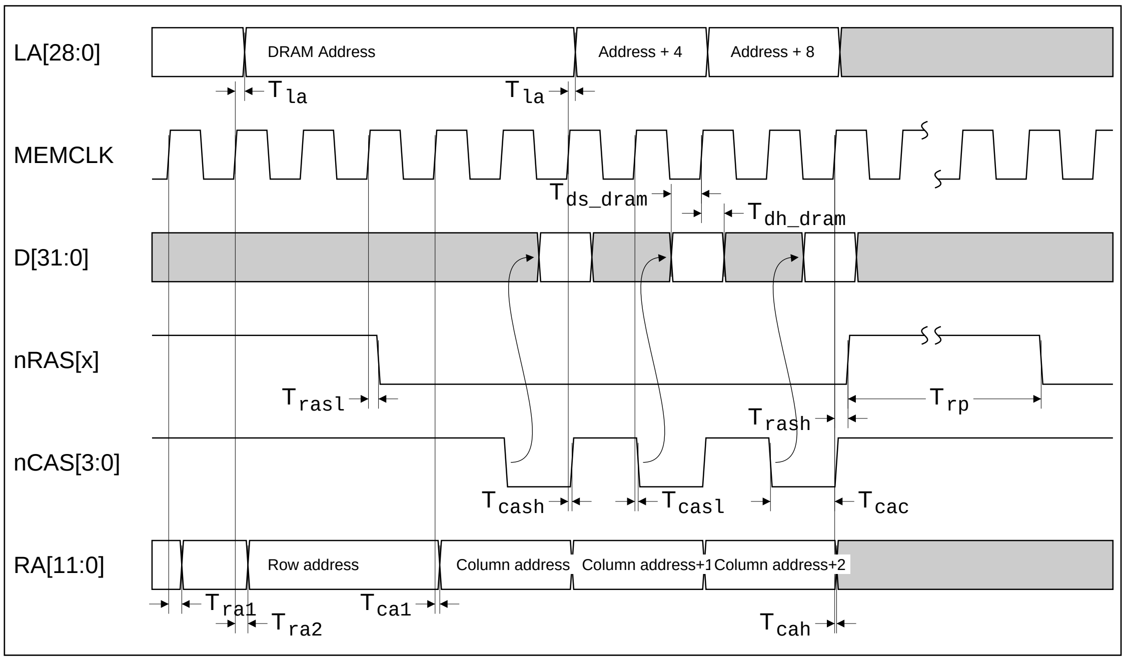


Figure 17-6: Fast page DRAM read timing (32-bit mode)

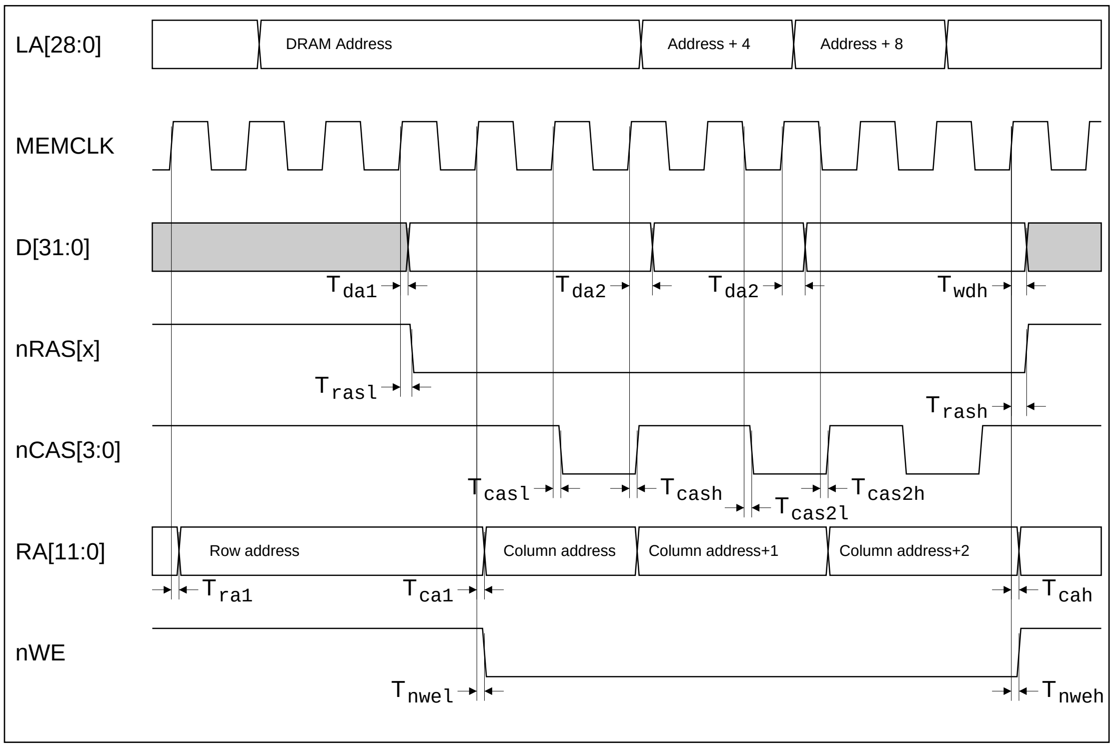


Figure 17-7: Fast page DRAM write timing (32-bit mode)

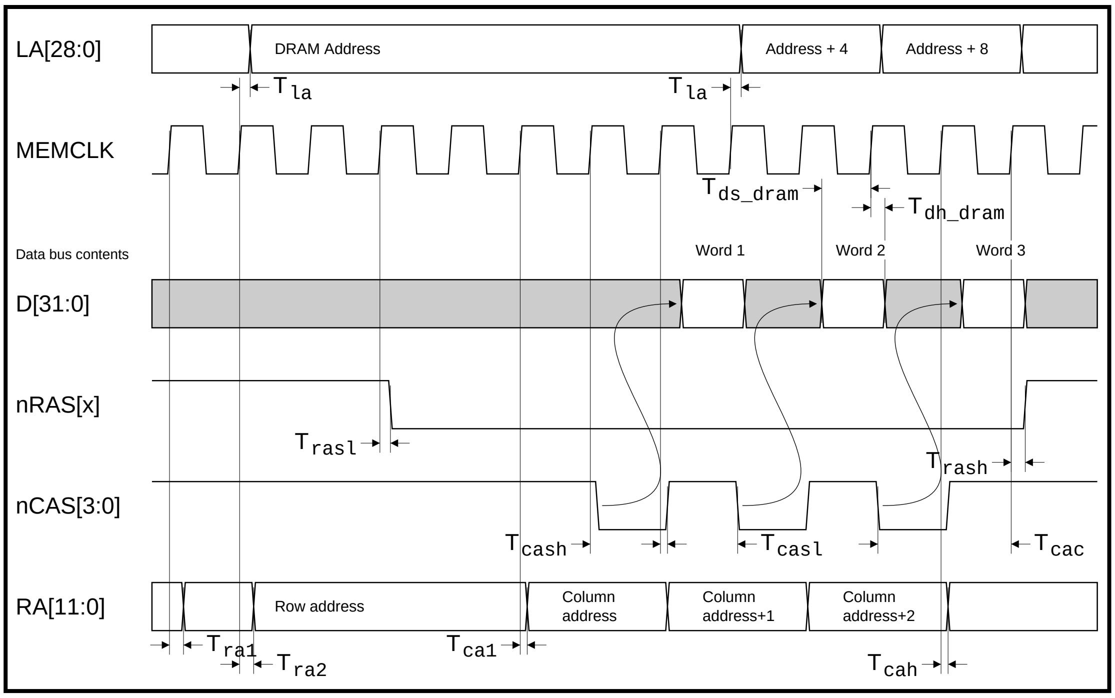


Figure 17-8: EDO DRAM read timing (32-bit mode)

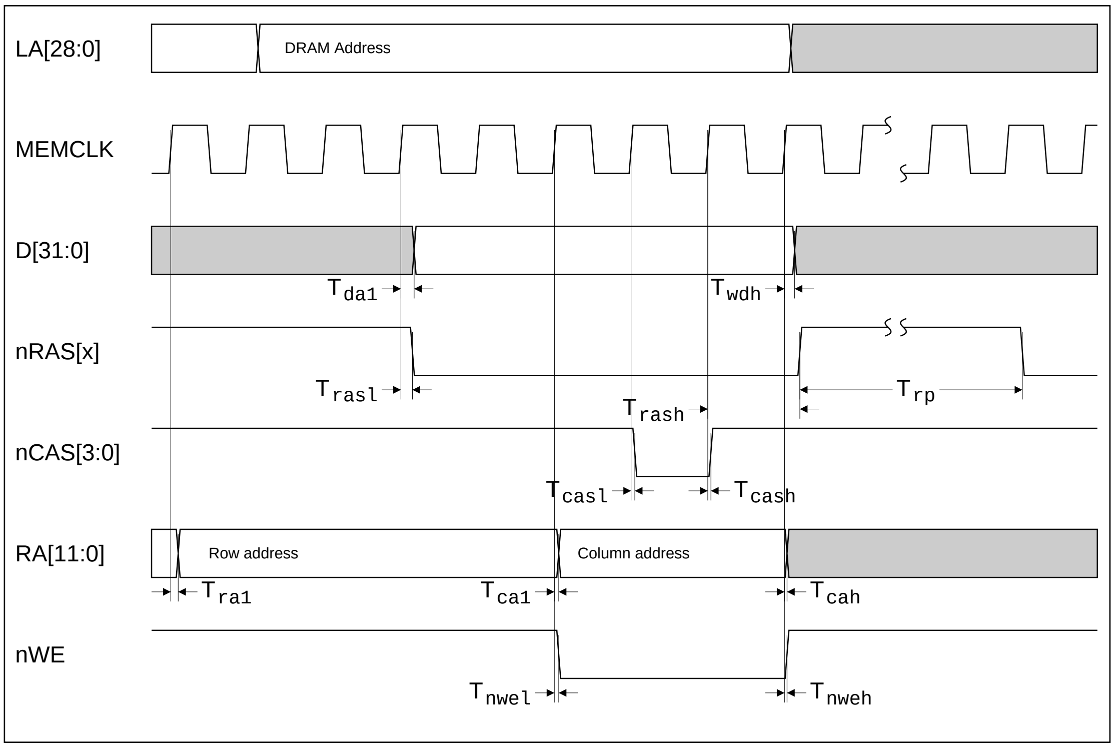


Figure 17-9: Single word EDO DRAM write

16-bit mode

In 16-bit mode ARM7500FE 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, which means that only 32MB of DRAM is supported per bank. Words are stored in DRAM with the upper half-word at the lower 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	0x10000000
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0x10000001

When this is read, the ARM will see:

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
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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. Byte reads and writes for 16-bit wide DRAM thus have the same timing as for the non-sequential 32-bit case as shown in Figures 14-4 and 14-5.

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 between 7 and 9 **MEMCLK** cycles, after which sequential accesses can continue until a page boundary is reached, taking 2 cycles for each half word.

Figure 17-10: Fast page DRAM read timing (16-bit mode) shows a 16-bit-mode read cycle.

Figure 17-11: Fast page DRAM write timing (16-bit mode) on page 17-15 shows a 16-bit mode write cycle.

Figure 17-12: EDO DRAM read timing (16-bit mode) on page 17-16 shows a multiple read from 16-bit wide EDO RAM.

Figure 17-13: EDO DRAM write timing (16-bit mode) on page 17-16 shows a 16-bit mode write, without bit [6] of DRAMCTL set.

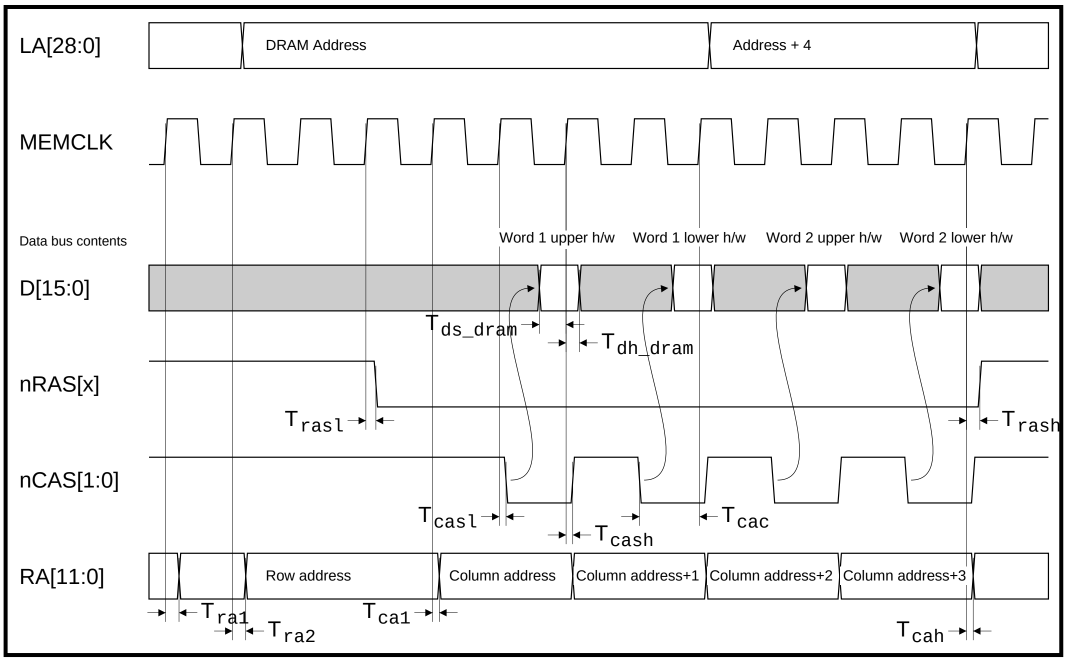


Figure 17-10: Fast page DRAM read timing (16-bit mode)

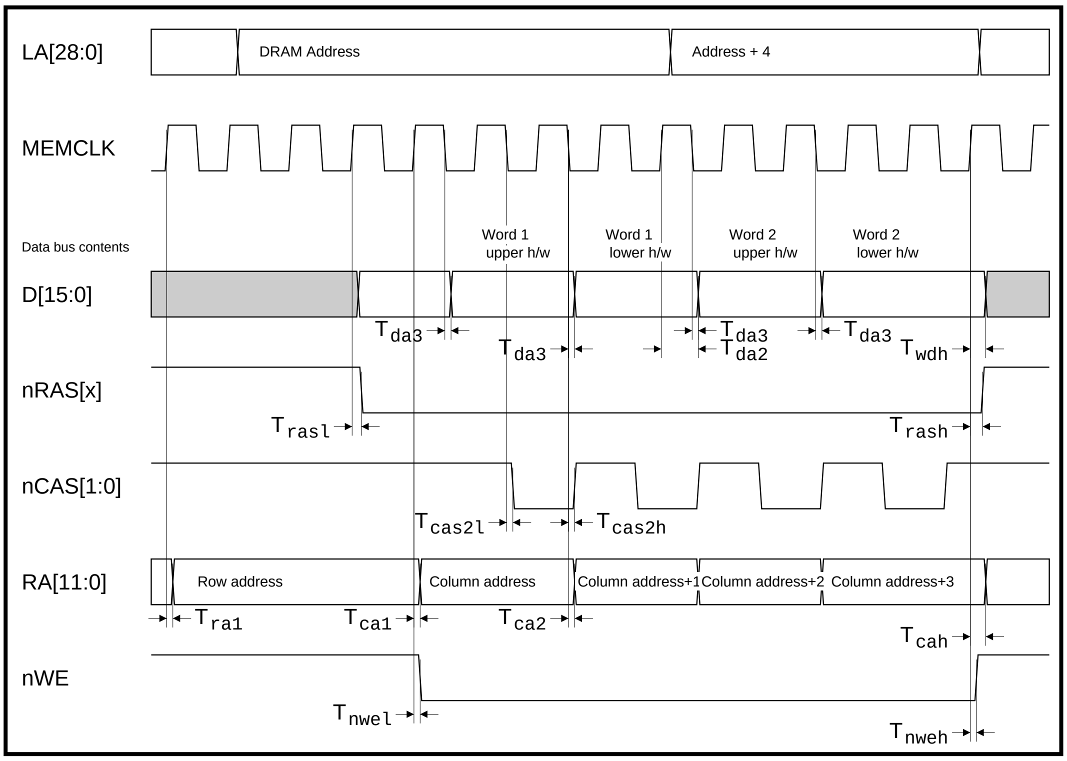


Figure 17-11: Fast page DRAM write timing (16-bit mode)

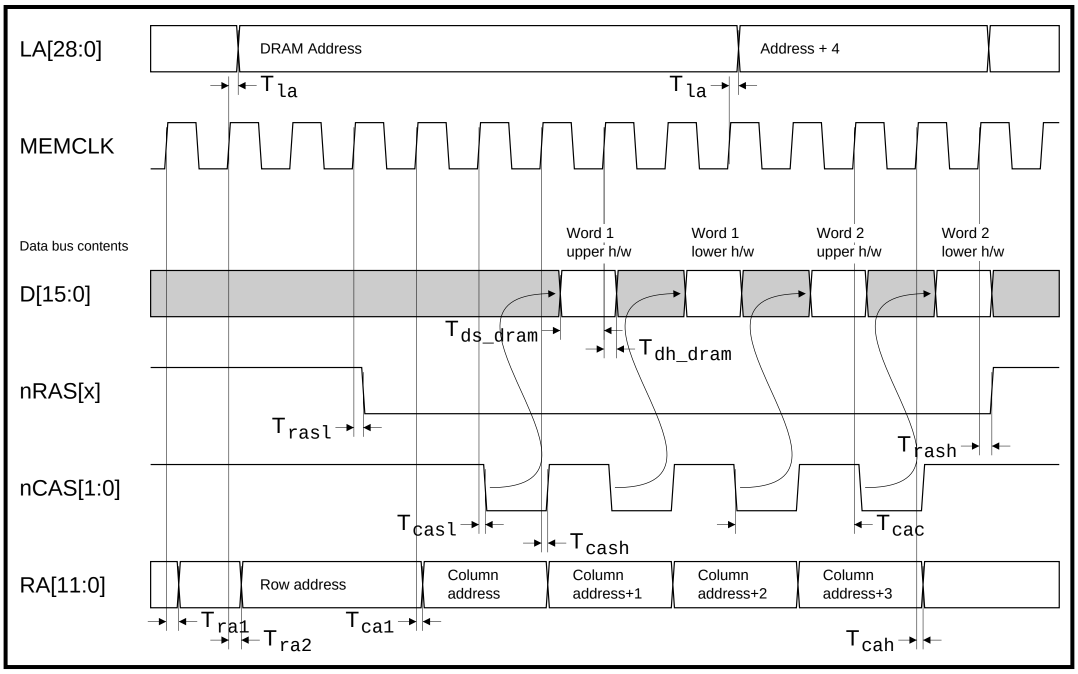


Figure 17-12: EDO DRAM read timing (16-bit mode)

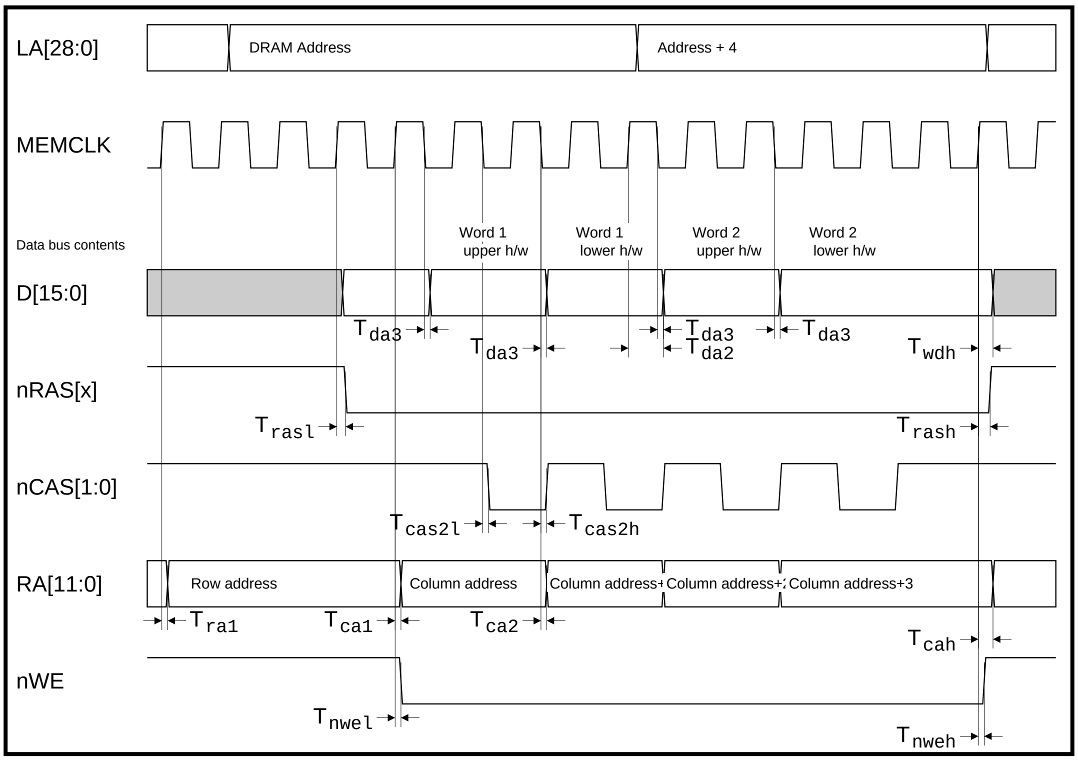


Figure 17-13: EDO DRAM write timing (16-bit mode)

Symbol	Parameters	Min	Max	Units	Note
Tcasl	**MEMCLK** rising to **Ncas[ ]** falling		12	ns	
Tcash	**MEMCLK** rising to **Ncas[ ]** rising		11	ns	
Tds_dram	read DATA setup to **MEMCLK** rising	-5		ns	
Tdh_dram	read DATA hold from **MEMCLK** rising	16		ns	
Tcac_fp	nCAS falling to data latched	21		ns	1
Tcac_edo	nCAS falling to data latched	25		ns	2
Tda1	**MEMCLK** rising to write DATA valid		14	ns	
Tda2	**MEMCLK** rising to write DATA valid		33	ns	
Tda3	**MEMCLK** falling to write DATA valid		15	ns	
Twdh	write DATA hold from **MEMCLK** rising	9		ns	
Trash	**MEMCLK** rising to **NRAS[ ]** rising		10	ns	
Trasl	**MEMCLK** rising to **NRAS[ ]** falling		13	ns	
Tra1	**MEMCLK** rising to **RA[ ]** valid (row address)		36	ns	3
Tra2	**MEMCLK** rising to **RA[ ]** valid (row address)		23	ns	4
Tca1	**MEMCLK** rising to **RA[ ]** valid (column address)		15	ns	
Tca2	as Tca1 but **MEMCLK** falling		14	ns	
Tcah	column address, **RA[ ]**, hold from **MEMCLK** rising	12		ns	
Tnwel	**MEMCLK** rising to **NWE** falling		12	ns	
Tnweh	**MEMCLK** rising to **NWE** rising	8		ns	5
Tcas2l	as Tcasl but **MEMCLK** falling		12	ns	
Tcas2h	as Tcash but **MEMCLK** falling		12	ns	
Trp	RAS precharge times	3		**MEMCLK** cycles	6

Table 17-2: ARM7500FE DRAM timing

Note: *The output delays above only include the intrinsic delay of the output pad driver. See section 22.5 De-rating* on page 22-6 *to calculate the final delay dependent upon the expected output load.*

In *Table 17-2: ARM7500FE DRAM timing* on page 17-17:

Note 1:	Minimum **nCAS** access time for Fast Page mode DRAM across all conditions with **nCAS** loading of 100pF or less, when **MEMCLK** = 32MHz.
Note 2:	Minimum **nCAS** access time for EDO DRAM across all conditions with **nCAS** loading of 100pF or less, when **MEMCLK** = 56MHz.
Note 3:	CPU accesses.
Note 4:	DMA accesses,
Note 5:	**nWE** rising will not change while external **nCAS** signals are still LOW.
Note 6:	The minimum RAS precharge time can be extended to 4 cycles by setting bit 6 of the DRAMCTL register.

17.2.6 DRAM refresh

DRAM refresh is controlled by a small state machine and counter within ARM7500FE. The refresh interval timer is clocked by a clock derived from the fixed frequency **I_OCLK**, and thus the refresh intervals will 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 (16µs), and the counter and state machine are clocked such that refresh continues even during reset.

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

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

The output states for DRAM refresh cycles are shown in *Figure 17-14: Refresh cycle timing* on page 17-19.

Note: This assumes divide-by-1 mode for **MEMCLK**.

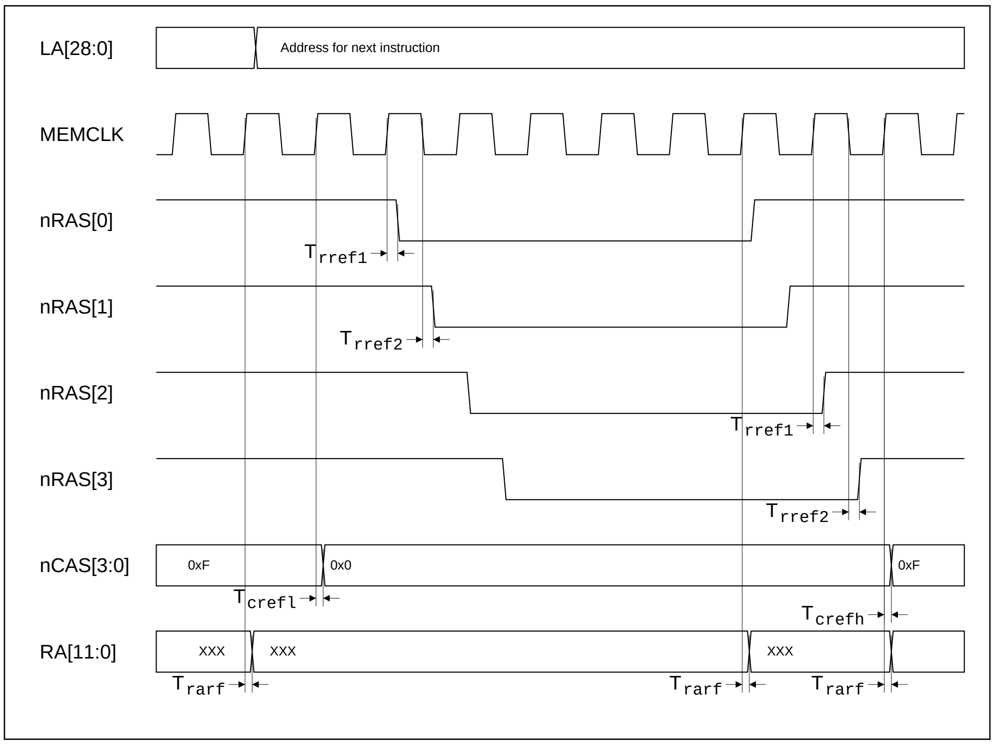


Figure 17-14: Refresh cycle timing

Symbol	Parameters	Min	Max	Units
Trref1	**MEMCLK** rising to **nRAS**		12	ns
Trref2	**MEMCLK** falling to **nRAS**		11	ns
Tcrefl	**MEMCLK** rising to **nCAS[3:0]** falling		16	ns
Tcrefh	**MEMCLK** rising to **nCAS[3:0]** rising		16	ns
Trarf	**MEMCLK** rising to **RA[11:0]** changing		22	ns

Table 17-3: ARM7500FE refresh cycle timing

Note: *The output delays above only include the intrinsic delay of the output pad driver. See section 22.5 De-rating* on page 22-6 *to calculate the final delay dependent upon the expected output load.*

17.2.7 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 ARM7500FE clocks have been stopped. All DMA must be stopped and the code which writes to this register must be executing from ROM.

C	force **nCAS**'s LOW	
R	force **nRAS**'s 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)	

17.2.8 Non-sequential access time and RAS precharge

At the end of one DRAM access, the earliest the next access may start is two memory clock cycles later. The new access must be to a different DRAM bank for this to be allowed. If the new access is to the same bank as the previous, to maintain the RAS precharge time (Trp), 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. By setting bit 7 of the DRAMCTL register high this can be increased to 4 **MEMCLK** cycles. These wait states will increase the access time of a non-sequential DRAM access by 1 or 2 cycles.

In order to meet some DRAM parameters, such as RAS access delay (Trac), at higher memory clock frequencies, bit 6 of the DRAMCTL register can be set. This will insert a wait state between the falling **nRAS** and the first falling **nCAS** of a non-sequential cycle.

Setting bit 5 of the DRAMCTL register delays the latching of data into ARM7500FE by one cycle to support EDO DRAM and so increases non-sequential access time by one cycle. It also keeps **nRAS** low for an extra cycle at the end of writes to meet some DRAM parameters at speeds associated with EDO.

The following table shows how to calculate the non-sequential DRAM access time:

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

Figure 17-15: Non-sequential DRAM access time

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

17.3 DMA Channels

The ARM7500FE supports video, cursor and sound DMA to enable direct transfer of quad words of data from DRAM to the video and sound processing interfaces. All DMA is in units of four words (quad words) and data can be read from any of the four banks of DRAM in either 16 or 32-bit mode. ARM7500FE contains a DMA Address Generator, which 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 which 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 from which the DMA will take place. 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 the programmer's model section of the datasheet.

17.3.1 Video DMA

The video DMA channel can be used in two modes. Duplex mode is used for fetching DMA data for use with a dual panel LCD display, and involves fetching a quad word of data for the top half of the display, followed by a quad word of data for the bottom half of the display, then the next quad word for the top half and so on. This is implemented using two parallel sets of registers which must be programmed accordingly.
A description of how to use the ARM7500FE 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 via bit 7 of the VIDCR register at location 0x032001E0. Bit 5 of the same register enables the video DMA channel. It should not be enabled until the other address registers have been 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 register, and can create an interrupt by programming the IRQA mask register appropriately.

There is a single VIDSTART register, which should be programmed with the location in memory of the first quad word 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 quad word in the frame buffer image.

For normal mode operation, the VIDINITA register should be programmed with the address in memory of the data which will be used to create 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 which are quad word aligned (ie. bits 0 to 3 are zero).

There is an extra bit (30) in the VIDINITA register, which 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 register can be written to, and the video DMA channel will become operational. The channel is then controlled by a video request signal from the video controller part of ARM7500FE. When a request for more video data arrives and the current bus cycle finishes, the bus controller will arbitrate in favor of the DMA (which has the highest priority on the bus) to fetch a quad word of data for the video sub system. Immediately after each DMA access, the address in the current register is incremented by 16 (one quad word) 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 will be the last one in the buffer, and after the next DMA, the current register will be reloaded from the VIDSTART register. During the FLYBACK period, the current register will be 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 ARM7500FE will toggle 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.

17.3.2 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 video DMA control register. The operation of the channel is the same for normal or duplex modes, but it is necessary to program the cursor differently depending on which mode is 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 registers in the video sub system. Each quadword fetch will result in two rasters' worth of cursor data being transferred, except in Hi-Res Mode (see *14.4 Hi-Res Support* on page 14-6). At the end of each fetch, the value in the CURSCUR register is increased by 16, to address the start of the next quadword. The value programmed into the CURSINIT register must be quadword-aligned.

17.3.3 Sound DMA

The Sound DMA channel provides data for the ARM7500FE 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 4KByte 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 quad word and must be from quad word 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, which must be programmed HIGH if the initial value in the current register is the same as the end register for that buffer, ie for a single transfer.

There is also an interrupt mask and status bit for the sound channel which allows the status of the sound DMA state machine to be monitored. The state machine will generate 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 ARM7500FE sound subsystem which asserts a request signal, and the bus controller will arbitrate in favour of the sound DMA when the current bus cycle has completed as long as there is not an outstanding video or cursor DMA request.

17.3.4 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 state 110, such that the Overrun and Int bits are set. The Overrun bit indicates when a channel has stopped because it has finished a transfer and the other pointer pair has not been programmed. The Int bit indicates when the channel is requesting an interrupt. The A/B bit indicates which pair of current/end pointers is in use.

The state machine diagram in the figure below 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 which causes the state transition. The “stop” bit in the end register is used to terminate 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 will be generated which 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 will be set, and DMA cannot continue until the pointers are reprogrammed.

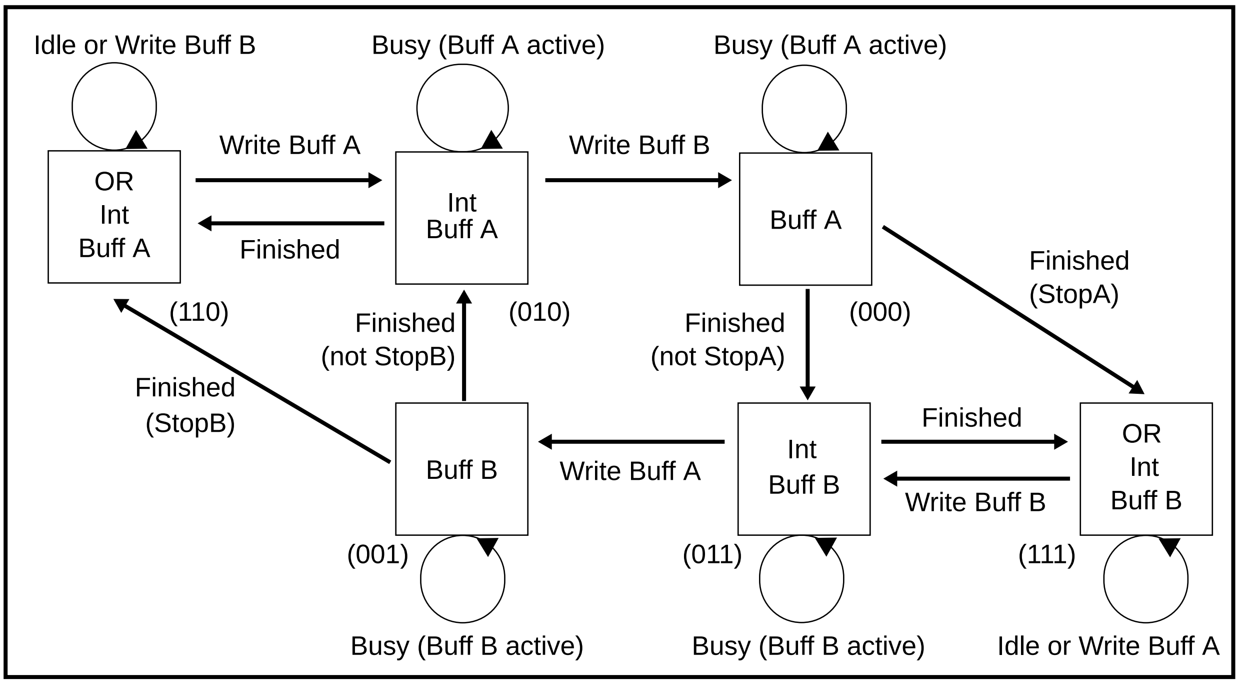


Figure 17-16: Hardware DMA state machine diagram