OMAP5910 Dual-Core Processor System DMA Controller Reference Guide

Literature Number: SPRU674
October 2003

Preface

Read This First

About This Manual

This document describes the system DMA controller for the OMAP5910 multimedia processor.

Notational Conventions

This document uses the following conventions.

❑ Hexadecimal numbers are shown with the suffix h. For example, the following number is 40 hexadecimal (decimal 64): 40h.

Related Documentation From Texas Instruments

The following documents describe the OMAP5910 device and related peripherals. Copies of these documents are available on the Internet at www.ti.com. *Tip:* Enter the literature number in the search box provided at www.ti.com.

OMAP5910 Dual-Core Processor MPU Subsystem Reference Guide (literature number SPRU671)

OMAP5910 Dual-Core Processor DSP Subsystem Reference Guide (literature number SPRU672)

OMAP5910 Dual-Core Processor Memory Interface Traffic Controller Reference Guide (literature number SPRU673)

OMAP5910 Dual-Core Processor System DMA Controller Reference Guide (literature number SPRU674)

OMAP5910 Dual-Core Processor LCD Controller Reference Guide (literature number SPRU675)

OMAP5910 Dual-Core Processor Universal Asynchronous Receiver/Transmitter (UART) Devices Reference Guide (literature number SPRU676)

OMAP5910 Dual-Core Processor Universal Serial Bus (USB) Reference Guide (literature number SPRU677)

OMAP5910 Dual-Core Processor Clock Generation and System Reset Management Reference Guide (literature number SPRU678)

OMAP5910 Dual-Core Processor General-Purpose Input/Output (GPIO) Reference Guide (literature number SPRU679)

OMAP5910 Dual-Core Processor MMC/SD Reference Guide (literature number SPRU680)

OMAP5910 Dual-Core Processor Inter-Integrated Circuit (I2C) Controller Reference Guide (literature number SPRU681)

OMAP5910 Dual-Core Processor Timer Reference Guide (literature number SPRU682)

OMAP5910 Dual-Core Processor Inter-Processor Communication Reference Guide (literature number SPRU683)

OMAP5910 Dual-Core Processor Camera Interface Reference Guide (literature number SPRU684)

OMAP5905 Dual-Core Processor Multichannel Serial Interface (MCSI) Reference Guide (literature number SPRU685)

OMAP5910 Dual-Core Processor Micro-Wire Interface Reference Guide (literature number SPRU686)

OMAP5910 Dual-Core Processor Real-Time Clock (RTC) Reference Guide (literature number SPRU687)

OMAP5910 Dual-Core Processor HDQ/1-Wire Interface Reference Guide (literature number SPRU688)

OMAP5910 Dual-Core Processor PWL, PWT, and LED Peripheral Reference Guide (literature number SPRU689)

OMAP5910 Dual-Core Processor Frame Adjustment Counter Reference Guide (literature number SPRU690)

OMAP5910 Dual-Core Processor Multichannel Buffered Serial Port (McBSP) Reference Guide (literature number SPRU708)

Trademarks

OMAP and the OMAP symbol are trademarks of Texas Instruments.

Contents

Figures

Tables

System DMA Controller

This document describes the system DMA controller for the OMAP5910 multimedia processor.

1 Introduction

The system direct memory access (DMA) controller transfers data between points in the memory space without intervention by the MPU. The DMA allows movements of data to and from internal memory, external memory, and peripherals to occur in the background of MPU operation. It is designed to off-load the block data transfer function from the MPU processor.

Figure 1 shows the OMAP5910 device with the DMA controller highlighted. Figure 2 shows the DMA controller in more detail.

Figure 1. Highlight of System DMA Controller

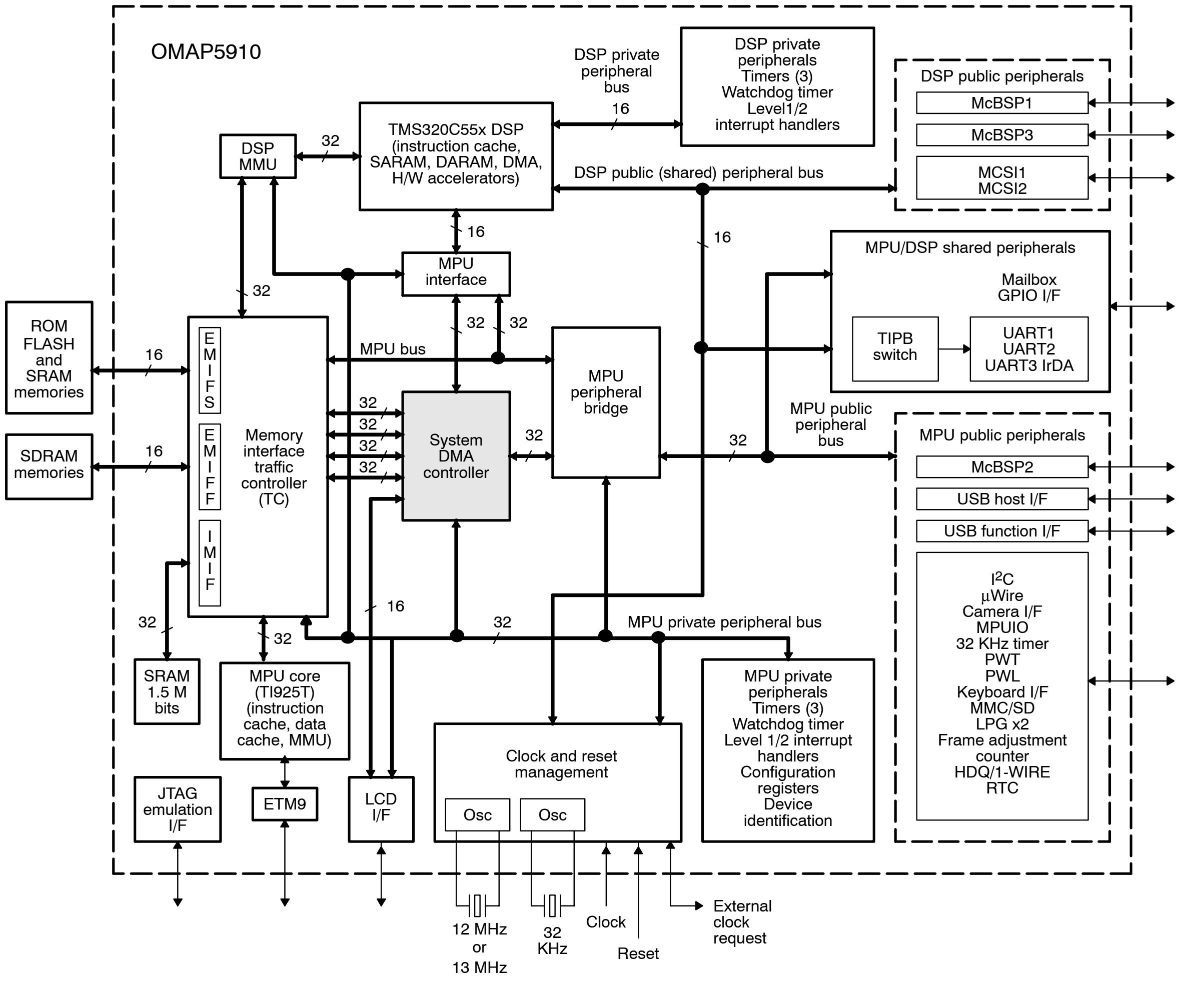
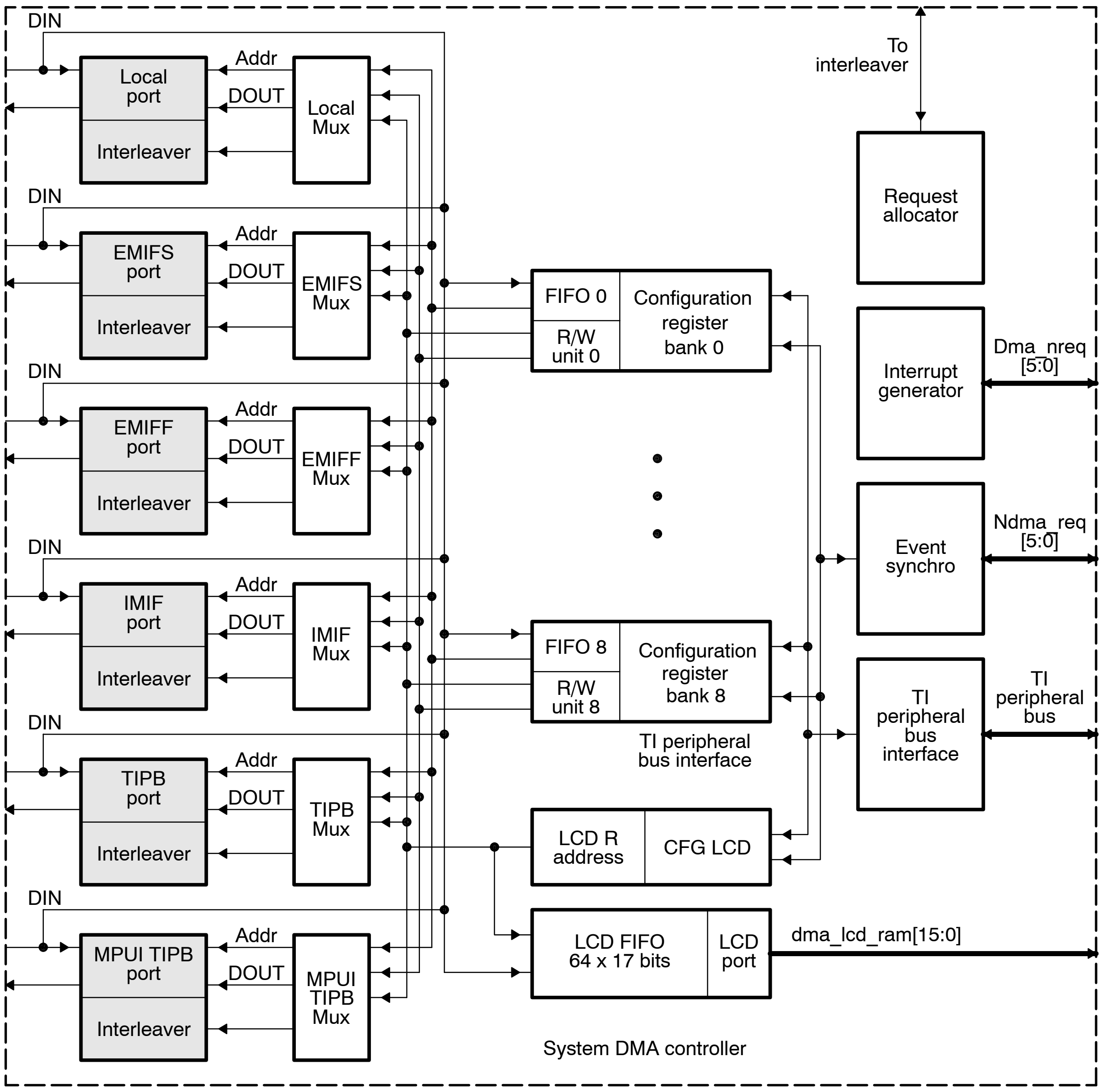

OMAP5910
DSP private peripheral bus
16
DSP private peripherals Timers (3) Watchdog timer Level1/2 interrupt handlers
DSP public peripherals
McBSP1
McBSP3
MCSI1 MCSI2
DSP MMU
32
TMS320C55x DSP (instruction cache, SARAM, DARAM, DMA, H/W accelerators)
DSP public (shared) peripheral bus
16
16
MPU interface
MPU/DSP shared peripherals
Mailbox GPIO I/F
TIPB switch
UART1 UART2 UART3 IrDA
32
32
32
ROM FLASH and SRAM memories
16
SDRAM memories
16
EMIFS
EMIFF
IMIF
Memory interface traffic controller (TC)
MPU bus
32
32
32
32
System DMA controller
32
MPU peripheral bridge
32
MPU public peripheral bus
MPU public peripherals
McBSP2
USB host I/F
USB function I/F
I2C µWire Camera I/F MPUIO 32 KHz timer PWT PWL Keyboard I/F MMC/SD LPG x2 Frame adjustment counter HDQ/1-WIRE RTC
32
16
32
32
MPU private peripheral bus
SRAM 1.5 M bits
MPU core (TI925T) (instruction cache, data cache, MMU)
MPU private peripherals Timers (3) Watchdog timer Level 1/2 interrupt handlers Configuration registers Device identification
Clock and reset management
Osc
Osc
JTAG emulation I/F
ETM9
LCD I/F
12 MHz or 13 MHz
32 KHz
Clock
Reset
External clock request

Figure 2. DMA Controller Block Diagram



Transfers are made through a physical channel that can be thought of as a pipe that connects a source and a destination for the duration of a transfer. Data flows through this pipe from the source to the destination. After the transfer is completed, the pipe (channel) can be used to perform other data transfers that involve the same or other sources and destinations.

The DMA channels are said to be physical because each of them is hardware-implemented. Each channel is controlled by a set of configuration registers, where the software sets up the transfer parameters such as length, source, and destination addresses. This set of registers is called the transfer descriptor or the transfer contexts. The transfer contexts are set up by the processor through the MPU's private TI peripheral bus (TIPB). Physical channel configuration registers are in the MPU's memory space.

All of the physical channels operate concurrently, which allows several data transfers to run in parallel, one in each channel. If several channels use the same DMA port, they are time-multiplexed by this port and can be priority scheduled through software parameters available to the user. Thus, the user is able to control the sharing of a port between channels by the assignment of priority levels.

The system DMA controller has nine independently programmable generic channels plus one channel that is specifically dedicated only to transfers from either the IMIF or the EMIFF ports to the LCD port. By dynamically assigning one of the nine physical channels to a pair of ports, data can be transferred to/from the designated ports.

The DMA interface with the source and destination of transfers is called a port. Each port can have its own protocol to communicate to the resource (memory or TIPB bridge) to which it is connected. All of the existing DMA ports used by the system DMA are described in detail later in this document.

The DMA has six general-purpose ports:

- External memory interface fast (EMIFF) port
- External memory interface slow (EMIFS) port
- Internal memory interface (IMIF) port
- Local bus interface (Local) port
- MPU interface (MPUI) port
- TIPB interface (TIPB) port

A seventh, special port is provided to interface with an LCD controller. This port, the LCD port, has a dedicated channel that connects either the IMIF or EMIFF port to the LCD controller. Unlike the other DMA ports, the LCD port can only receive data from the IMIF or EMIFF ports.

The system DMA controller is controlled by the MPU (via the TIPB). The DMA controller meets the high-rate-flow requirements of the multichannel applications used by wireless base stations.

The system DMA controller is designed for low-power operation. Its clock can be automatically disabled as required. This function is synchronous to the MPU TIPB and is entirely under hardware control. No specific programming is required.

The system DMA controller includes the following features.

- Nine general-purpose and one dedicated (LCD) DMA channels
- Software programmable DMA access priority-based on resource allocation versus processor (MPU or DSP) access. Through the assignment of priority levels for each channel, the user can determine how the ports are shared between channels.
- Concurrent DMA transfers capability
- Start of transfer on peripheral request or host request
- Byte alignment capability
  Byte (odd-address) alignment is supported within the DMA for general purpose non-burst transfers. However, odd-address alignment is not supported when the TIPB is both the source/destination. The starting address for the general-purpose channel must be 4 bytes aligned. The starting address for the LCD channel must be 16 bytes aligned.
- Byte packing/unpacking
- Byte transfer count
- Configurable indexes through memory for each channel source and destination address register. The address may remain constant or post-incremented.
- Access available to all of the memory range (physical memory mapping and I/O space)
- Seven ports available for different kinds of hardware resources. All data exchanges are done with a simple handshake mechanism with request, ready, and abort signals. All of the ports except the TIPB have burst access capability.

- EMIFS port
- EMIFF port
- IMIF port
- MPUI port
- TIPB port
- Local port
- LCD port

- Memory-to-memory transfer granularity of 8, 16, and 32 bits. Only the number of programmed bytes is transferred; that is, there are no trailing or dirty bytes at the end of a transfer.
- TIPB-to-memory transfer:
  - For peripherals whose FIFO size is programmable, e.g., a UART, 16 bytes, which is the same length as the DMA channel FIFO, is recommended for maximum performance.
  - If the peripheral FIFO size is not 16 bytes, use a general-purpose timer as a time-out counter to avoid the possibility that some data might remain in the channel FIFO at the end of the frame. When the MPU receives the interrupt, it must check the DMA channel status.

Additional functional features and limitations of the DMA controller include:

- General-purpose DMA channels can perform 4x32 bit bursts; this is the only burst mode supported by the non-LCD DMA channels. Bursts can only be performed on the IMIF, EMIFF, EMIFS, and local bus ports. The starting and ending addresses must be burst aligned, i.e., 4 x 32bits.
- All LCD DMA accesses are performed in bursts of 8 x16 bits. The LCD video buffer data must be a multiple of 16 bytes. The starting and ending addresses must be aligned on a 16-byte boundary.
- General Purpose channel priority on the external EMIFF/IMIF bus is software programmable.
- The LCD channel FIFO size is 64x17 bits. The LCD DMA channels support 8x16 bit bursts.
- The interface between the LCD controller and the DMA controller uses a different protocol from the other DMA channels. The LCD controller is the master of this interface, and it generates the addresses according to the FIFO status.
- The LCD controller allows the use of one or two video buffer(s) and is configurable.

❑ No bursts are supported on the TIPB and MPUI DMA interfaces.

❑ Burst transfers can be combined with single or double-indexed addressing modes, but the burst request is only issued if contiguous addresses can be ensured for the length of the burst. For example, if the element index contains a value that causes noncontiguous addresses between elements, then the system DMA logic does not generate burst requests.

Table 1 lists the possible data transfers, and Table 2 lists the possible transfer sizes and types.

Table 1. Possible Data Transfers

Destinations vs. Sources	**EMIFS Bus**	**EMIFF Bus**	**IMIF Bus**	**TIPB**	**MPUI**	**LCD**	**Local Bus**
EMIFS bus	Yes	Yes	Yes	Yes	Yes	No	Yes
EMIFF bus	Yes	Yes	Yes	Yes	Yes	Yes	Yes
IMIF bus	Yes	Yes	Yes	Yes	Yes	Yes	Yes
TIPB	Yes	Yes	Yes	Yes	Yes	No	Yes
MPUI	Yes	Yes	Yes	Yes	Yes	No	Yes
Local bus	Yes	Yes	Yes	Yes	Yes	No	Yes

Table 2. Possible Transfer Sizes and Types

Source	**Destination: 8-Bit TIPB**	**Destination: 8-Bit Non-TIPB**	**Destination: 16-Bit TIPB**	**Destination: 16-Bit Non-TIPB**	**Destination: 32-Bit TIPB**	**Destination: 32-Bit Non-TIPB**
8-bit TIPB	Valid only with no packing s8	Valid s8	Not allowed	Valid s8	Not allowed	Valid s8
8-bit non-TIPB	Valid only with no packing s8	Valid s8	Not allowed	Valid s8,s16	Not allowed	Valid s8,s16,s32
16-bit TIPB	Not allowed	Valid s16	Valid only with no packing s16	Valid s16	Not allowed	Valid s16
16-bit non-TIPB	Valid with s8 and no packing	Valid s8	Valid only with no packing s16	Valid s8,s16	Valid only with packing s32	Valid s8,s16,s32

Note: s8 = 8-bit scalar data type; s16 = 16-bit scalar data type; s32 = 32-bit scalar data type

Table 2. Possible Transfer Sizes and Types (Continued)

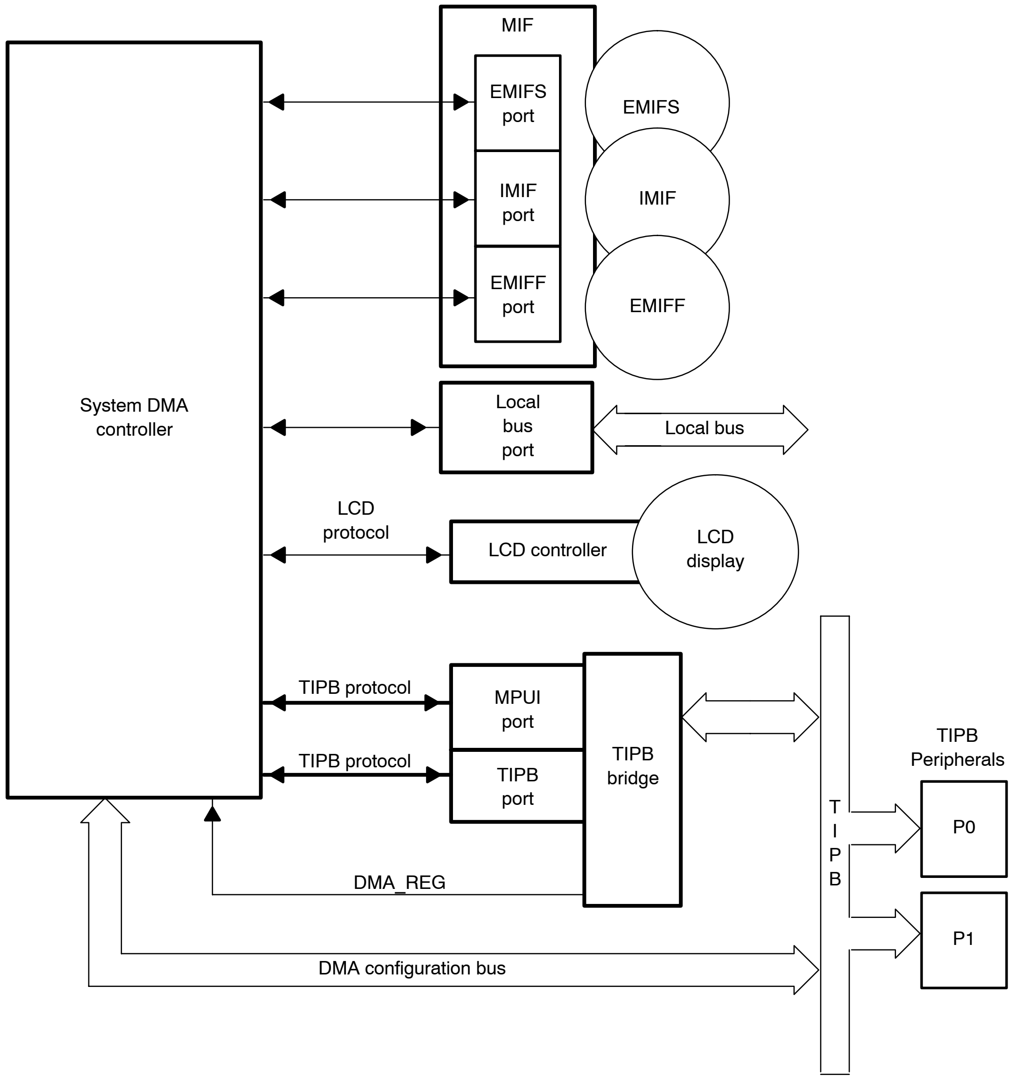
Source	Destination: 8-Bit TIPB	Destination: 8-Bit Non-TIPB	Destination: 16-Bit TIPB	Destination: 16-Bit Non-TIPB	Destination: 32-Bit TIPB	Destination: 32-Bit Non-TIPB
32-bit TIPB	Not allowed	Valid s32	Not allowed	Valid s32	Valid s32	Valid s32
32-bit non-TIPB	Valid with s8 and no packing	Valid s8	Valid only with no packing s16	Valid s8,s16	Valid S32	Valid s8,s16,s32

Note: s8 = 8-bit scalar data type; s16 = 16-bit scalar data type; s32 = 32-bit scalar data type

2 External Connections

The system DMA controller is interconnected with other OMAP5910 components as shown in Figure 3.

Figure 3. System DMA External Connections



3 Generic Channels

This section discusses the following generic channel topics:

- ❏ Transfers
- ❏ Addressing modes
- ❏ Data packing and bursting
- ❏ Data alignment
- ❏ Constraint on channel configuration parameters
- ❏ Endianism
- ❏ Interrupt generation
- ❏ Memory space protection

3.1 Transfers

3.1.1 Transfer Sources and Destination

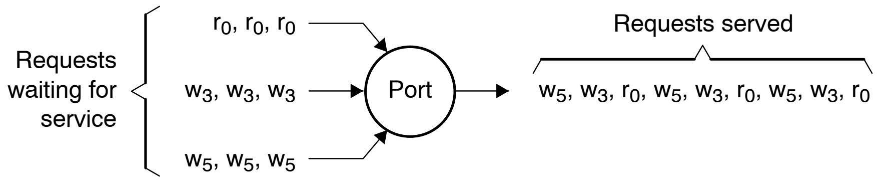
Each DMA channel can be configured individually. A port can be shared by several channel requests. Therefore, these requests are time-multiplexed by the port.

For example, in Figure 4, a DMA port must service requests from three DMA channels:

- ❏ Channel 0 as a source port (read requests, r_0)
- ❏ Channel 3 as a destination port (write requests, w_3)
- ❏ Channel 5 as a destination port (write requests. w_5)

Figure 4 shows how these requests are multiplexed in time by the port.

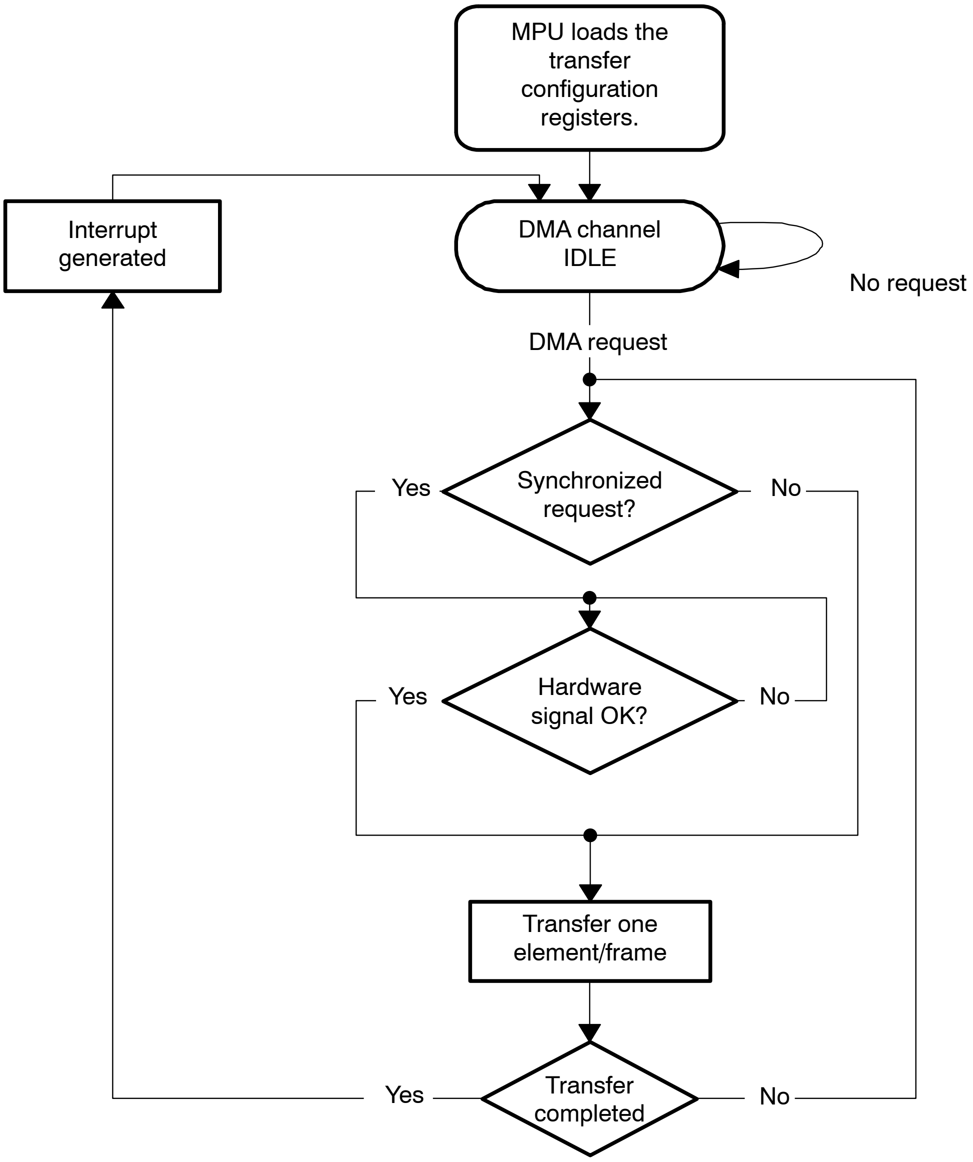
Figure 4. Time-Sharing on a DMA Port


Requests waiting for service
r_0, r_0, r_0
w_3, w_3, w_3
w_5, w_5, w_5
Port
Requests served
w_5, w_3, r_0, w_5, w_3, r_0, w_5, w_3, r_0

A system DMA port can handle 10 read requests and 10 write requests (all the possible requests in the DMA) simultaneously.

3.1.2 Transfer Control

Figure 5. Basic Flow of DMA Transfer



3.1.3 Transfer Start

There are two ways to start a DMA transfer:

❑ Software start (software request): After setting up the configuration registers of a DMA channel, the processor activates the transfer in this channel by writing the DMA_CCR EN bit of this channel. The transfer immediately starts.

- Hardware start (hardware request): The processor sets up a DMA channel and sets the transfer in this channel as a synchronized transfer (demand-driven by DMA request signal lines from the outside world). The channel then waits for a DMA request to start transferring data.

Note:

Be sure to enable the channel before the DMA request.

Each time a DMA request is made for this channel, an amount of data is transferred. This amount of data can be:

- An element

  A complete element is transferred in response to a DMA request.

- An entire frame

  A complete frame of several elements is transferred in response to a DMA request.

All of the transfers can be synchronized on DMA requests, regardless of their sources and destinations. DMA requests can come from DMA ports. Each channel can be triggered by one DMA request among 31. One DMA request can trigger several channels at the same time. The relevant bits are extracted from the DMA_CCR register.

3.1.4 Transfer Suspension

All of the DMA channels enter a suspended state if:

- The DMA idle input is active. The DMA suspends all of its transfers and its clock can be externally cut off for power-saving purposes.
- The DMA suspend input is active, and the FREE bit in the DMA_GCR register is equal to zero. The processor asserts this input, then it is halted by a breakpoint. When this occurs, the DMA suspends or continues its transfers, according to the value of the FREE bit. The DMA clock must be on when the DMA is suspended.

3.1.5 Autoinitialization

A DMA channel (synchronized or not) can operate in two modes.

- Single transfer mode

In this mode, a channel stops when the current transfer finishes.

- Autoinitialization mode

  In this mode, a channel loads a new configuration and automatically restarts a new transfer when the current one finishes.

  A DMA channel has two sets of configuration registers: programming and active. Only the programming set is accessible through the TIPB. When a channel is enabled for the first time or when a channel autoinitializes, the programming set is copied in the active set of registers. While the current transfer is running, the programming set to configure the next transfer can be programmed.

  This feature can be used in two ways:

  - Continuous operation

    The programming registers can be changed while the current configuration is being executed. The next transfer defined by the new configuration starts without stopping the DMA.

  - Repetitive operation

    Never modify the programming registers. The same context is always used.

  The programming set includes the following registers:

  - DMA_CSSA_L
  - DMA_CSSA_U
  - DMA_CDSA_L
  - DMA_CDSA_U
  - DMA_CEN
  - DMA_CFN
  - DMA_CFI
  - DMA_CEI

The following registers are part of the working set and are accessible. They always have impact on the current transfer.

- DMA_CSDP
- DMA_CCR
- DMA_CICR
- DMA_CSR

To avoid the reload of a configuration when the MPU programs the channel, the MPU can use the autoinitialization bits of the DMA_CCR register, which are described in Table 3.

Table 3. Autoinitialization Configuration Bits Summary

AUTOINIT	END_PROG	REPEAT	Autoinitialization Behavior
0	x	x	No autoinitialization
1	0	0	At the end of current transfer, channel waits until END_PROG = 1 to load the programming register set in its working register set.
1	1	0	At the end of current transfer, channel immediately loads the programming register set in its working register set.
1	x	1	

3.1.6 Priorities Between Channels

Each channel can be given a low- or high-priority level. When a DMA port receives requests from several channels, it looks at their priorities:

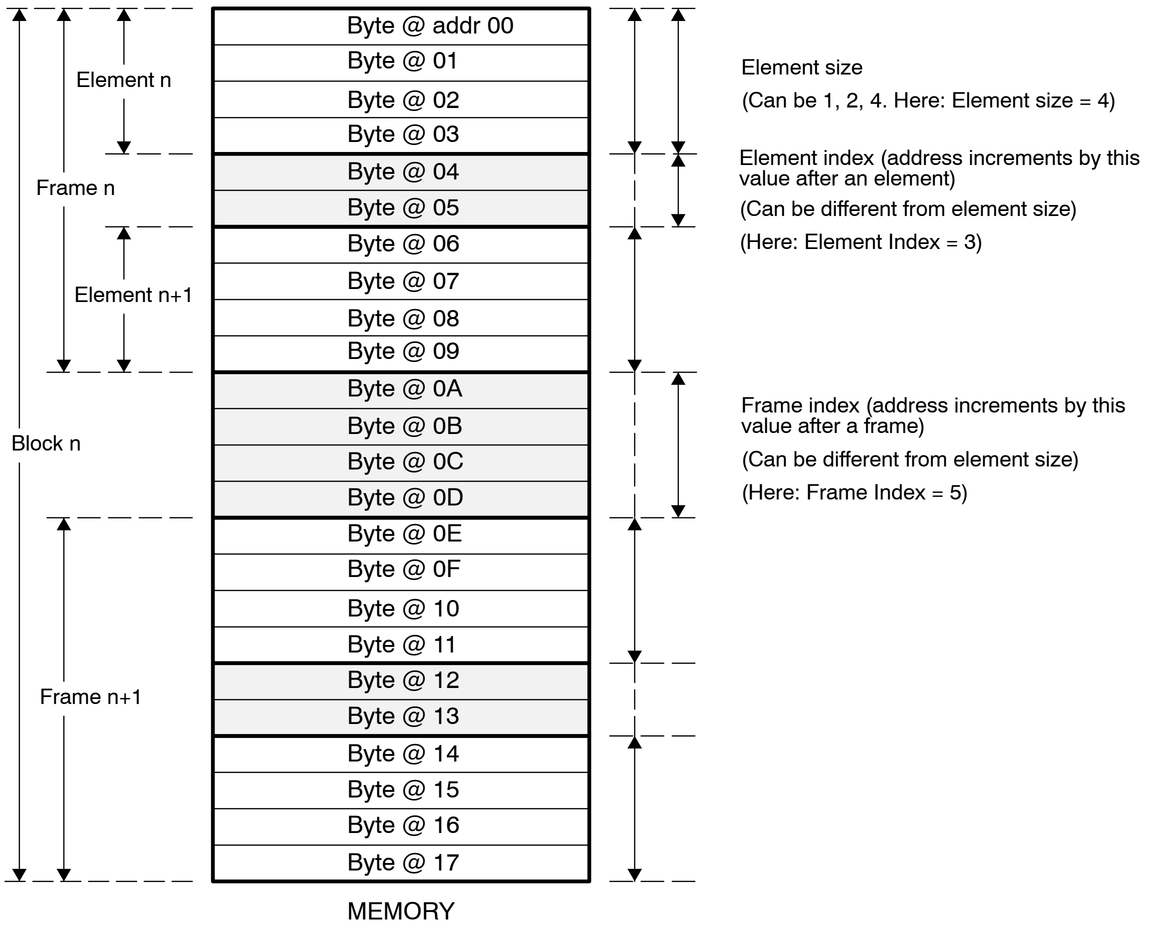
- Requests from high-priority channels are served first.
- Requests from low-priority channels are served only if there are no requests from high-priority channels on the port. This can occur if there are no high-priority channels activated, if the high-priority channels are stalled (by a slow source or destination), or if the high-priority channels are waiting for a synchronization event.
- Requests of the same priority level are served in a round-robin manner (time division multiplex).

3.1.7 Addressing Modes

Figure 6 provides an example of address index management. In this example, Element Size = 4, Element Index = 3, Frame Size = 2, and Frame Index = 5.

The system DMA has four addressing modes: constant, post-incremented, indexed, and double-indexed.

Figure 6. Memory Representation



The amount of data (block size) to transfer is programmed in bytes. This size can be odd or even. The starting address for a general-purpose burst transfer must be 4x32 bit aligned. The starting address can be either even or odd for a non-burst transfer. Furthermore, a DMA channel using bursting must be configured such that all addresses accessed by that channel are also 32-bit word aligned. The data block to transfer is split in frames and elements. The byte size of this data block is:

BS = FN x EN x ES

where:

BS is the block size in bytes.

FN is the number of frames in the block, $1 \le FN \le 65535$.

EN is the number of elements per frame, $1 \leq EN \leq 65535$.

ES is the number of bytes per element, $ES \in \{1, 2, 4\}$.

An element can be:

- ❑ 8-bit scalar data, s8
- ❑ 16-bit scalar data, s16
- ❑ 32-bit scalar data, s32

Types of data transferred in a channel include s8, s16, and s32. FN, EN, and ES (or data type) are extracted from the configuration registers of the channel.

To set up a channel for a transfer, the software must program two addressing modes:

- ❑ Source addressing mode
- ❑ Destination addressing mode

These modes work independently. For example, to transfer data from a TIPB serial port to internal memory, the source-addressing mode is constant (for example, when the read operation must be done at a unique register address) and the destination addressing mode is post-incremented.

The number of frames, the number of elements, and the element size must be the same for the source and destination. Each of the following algorithms describes address computation for each byte of the transfer.

3.1.8 Constant Addressing Mode

Address remains constant.

$a(i) = SA, 0 \leq i \leq BS - 1$

where:

a(i) is the address of the byte number i within the transfer.

SA is the starting address of the transfer.

BS is the block size in bytes.

3.1.9 Post-Incremented Addressing Mode

Address is always incremented by 1.

$a(0) = SA$

$a(i) = a(i - 1) + 1, 1 \leq i \leq BS - 1$

where:

$a(i)$ is the address of the byte number i within the transfer.

SA is the starting address of the transfer.

BS is the block size in bytes.

3.1.10 Single-Indexed Addressing Mode

Address is incremented by 1 if the end of the current element is not reached.

Address is incremented by an element index if the end of the current element is reached.

$a(0) = SA$

$a(i) = a(i - 1) + 1$ if $(i \bmod ES) \neq 0, 1 \leq i \leq BS - 1$

$a(i) = a(i - 1) + EI$ if $(i \bmod ES) = 0, 1 \leq i \leq BS - 1$

where:

$a(i)$ is the address of the byte number i within the transfer.

SA is the starting address of the transfer.

BS is the block size in bytes.

ES is the element size in bytes, $1 \leq ES \leq 2$.

EI is the element index in bytes, specified in a configuration register, $-32768 \leq EI \leq 32767$.

3.1.11 Double-Indexed Addressing Mode

Address is incremented by a frame index if the end of the current frame is reached.

Address is incremented by an element index if the end of the current element is reached and end of frame is not reached.

Address is incremented by one if the end of the current element and the end of current frame are not reached.

$a(0) = SA$

$a(i) = a(i - 1) + 1$ if $(i \bmod ES) \neq 0$ and $(i \bmod FS) \neq 0, 1 \leq i \leq BS - 1$

$a(i) = a(i - 1) + EI$ if $(i \bmod ES) = 0$ and $(i \bmod FS) \neq 0, 1 \leq i \leq BS - 1$

$a(i) = a(i - 1) + FI$ if $(i \bmod FS) = 0, 1 \le i \le BS - 1$

where:

a(i) is the address of the byte number i within the transfer.

SA is the starting address of the transfer.

BS is the block size in bytes.

ES is the element size in bytes.

EI is the element index in bytes, specified in a configuration register, $-32768 \le EI \le 32767$.

FS is the frame size in bytes, FS = ES x EN.

FI is the frame index in bytes, specified in a configuration register, $-32768 \le FI \le 32767$

3.1.12 Data Packing, Splitting and Bursting

A DMA channel has the capacity to:

- ❑ Pack several consecutive byte accesses in a single word16 (16-bit word), word32 (32-bit word), burst4 (burst of four 32-bit words), or burst8 (burst of eight words) access (only the LCD channel can perform burst8 accesses). This increases the transfer rate. For a channel, the decision to pack or burst accesses to its source port is made by the source address unit and depends on source port access capability. The decision to pack and/or burst accesses to its destination port is made by the destination address unit and depends on destination port access capability. Packing and bursting are performed only if the software allows it via proper configuration of the DST_PACK, SRC_PACK, DST_BURST_EN, and SRC_BURST_EN fields in the appropriate DMA_CSDP register.
- ❑ Split a single word transfer into several byte accesses. This is done if the DMA port data size is less than the size (or does not match the type) of the data moved.

Table 4 summarizes the possible transfer configurations and shows the cases where packing and splitting are performed.

Table 4. Packing and Splitting Summary

Data Type	Port Access Capability	Packing/Splitting
s8	8	–
	16	pack 2 x s8 => 16
	32	pack 4 x s8 => 32
s16	8	split s16 => 2 x 8
	16	–
	32	pack 2x s16 => 32
s32	8	split s32 => 4 x 8
	16	split s32 => 2 x 16
	32	–

To compute the type of an access (8-, 16-, 32-bit or burst) and decide whether or not to pack consecutive accesses, an address unit checks:

- ❑ Its related DMA port capabilities
  - ■ Can the port perform byte, 16-bit, 32-bit accesses?
  - ■ Can the port perform burst accesses?
- ❑ What is allowed by the software in the configuration registers:
  - ■ Is packing allowed (DST_PACK or SRC_PACK set)?
  - ■ Is bursting allowed (DST_BURST_EN or SRC_BURST_EN set)?
- ❑ The last bits of the address:
  - ■ Is the address even or odd?
  - ■ Is the address word16, word32, burst4, burst8 aligned?
- ❑ The number of elements remaining in the frame.

When the type of access is determined, the current byte address can be incremented by 1, 2, or 4 to reach the next memory space location to access. Then, the DMA port checks the channel FIFO to see if there is enough data (write access) or enough space (read access) in the FIFO before issuing the access.

Not every DMA port has the capability to support every data type (s8, s16, s32) and transfer size. Therefore, software must carefully set up all DMA transfers according to the constrains detailed in Table 2.

A transfer to/from a DMA port with 32-bit only access capability must be set up as follows:

- ❑ Element size is a multiple of four.
- ❑ Starting address is aligned on a 32-bit word.
- ❑ If frame index is used, it must always produce addresses aligned on a 32-bit word.
- ❑ If element index is used, it must always produce addresses aligned on a 32-bit word.

A transfer to/from a DMA port with 16-bit only access capability must be set up as follows:

- ❑ Element size is a multiple of two.
- ❑ Starting address is aligned on a 16-bit word.
- ❑ If frame index is used, it must always produce addresses aligned on a 16-bit word.
- ❑ If element index is used, it must always produce addresses aligned on a 16-bit word.
- ❑ Example 1 provides an example of packing and splitting.

Example 1. Packing 2 x s16 => 32

- ❑ A channel is set up for a transfer with the following parameters for its source:
  - ■ Number of frames in the block: FN = 2
  - ■ Number of elements per frame: EN = 5
  - ■ Type of data is s16
  - ■ Frame index in bytes: FI = 13
  - ■ Element index in bytes: EI = 1
  - ■ Source starting address: SA = 2
  - ■ The source port is a 32-bit port with byte word16 and word32 access capability.
  - ■ Bursts are disabled.

The memory block to transfer is as identified in Table 5 (element i, j is the element number j of frame i).

Table 5. Data Block to Transfer

Address	Byte 0	Byte 1	Byte 2	Byte 3
0			Element 1, 1	
4	Element 1, 2		Element 1, 3	
8	Element 1, 4		Element 1, 5	
12				
16				
20				
24	Element 2, 1		Element 2, 2	
28	Element 2, 3		Element 2, 4	
32	Element 2, 5			
36				
40				

The computed addresses and access types are as identified in Table 6.

Table 6. Address and Access Types

Clock Cycle	Frame Number j	Element Number i	Address	Access
0	1	1	2	16 bits
1	1	2	4	32 bits
2	1	4	8	32 bits
3	2	1	24	32 bits
4	2	3	28	32 bits
5	2	5	32	16 bits
6		End of transfer		

3.1.13 Data/Address Alignment

During a transfer, all the addresses computed by the DMA must be aligned on the type of data transferred:

- ❑ If the data type is s8 (8 bits scalar data), addresses can have any value.
- ❑ If the data type is s16 (16 bits scalar data), addresses must be aligned on 16-bit word boundary (the least significant bit of the address is always 0).
- ❑ If the data type is s32 (32 bits scalar data), addresses must be aligned on 32-bit word boundary (the two least significant bits of the address are always 00).
- ❑ If bursting is enabled, addresses must be aligned on a four-word burst boundary (128 bits) regardless of data type (the four least-significant bits of the address are always 0000).

When using the indexed addressing modes (element index and/or frame index), all the addresses computed must be aligned on the data type.

CAUTION

Failure to adhere to these address alignment requirements could yield unexpected results. In the case of the four-word bursting alignment, failure to properly align the addresses could result in a lockup condition on the DMA channel. To accomplish proper alignment, programming of the starting address, block size, frame size, and all indexes must be such that the address of every DMA access is properly aligned (for a burst, this would mean the first access of the burst).

3.1.14 Constraint on Channel Configuration Parameters

Verifying this constraint ensures correct DMA operations when transferring data between ports with different access capabilities as follows:

SA modulo ES = 0,

Where SA is source address and ES is the element size.

3.1.15 Interrupt Generation

Each DMA physical channel can generate an interrupt to the processor to reflect the transfer status. Each DMA physical channel has a dedicated interrupt line to the processor. All the DMA interrupts are level interrupts.

For every DMA logical channel, the following interrupt sources can be programmed:

- End of block: The last byte of the transfer has been written in the destination location.
- End of frame: The last byte of the current frame has been written in the destination location.
- Half of frame: The middle byte of the current frame has been written in the destination location.
- Start of last frame: The first word of the last frame has been written in the destination location.
- DMA request collision: A new DMA request occurred before the end of service of the previous one.
- Time-out: An access has been timed out.

  To prevent a definitive lock by a channel on a memory location or peripheral, all the DMA ports to memory/peripheral requests are monitored by a time-out counter in the following sequence:

  1) When the request is sent by the DMA to transfer data in a channel, a time-out counter is triggered. The time-out counter does not start until the data transfer is started. Receive requests are time-multiplexed.
  2) The request is acknowledged, and the time-out counter is stopped.
  3) If the time-out counter reaches its threshold before the request is acknowledged, the request is discarded and an error is reported in the DMA channel by setting the relevant bit in DMA_CSR (channel status register) and sending an interrupt to the processor. The channel is stopped.

  The time-out information is generated in the resources accessed by the DMA:

  - System IMIF/local bus port: Time-out is signaled by the IMIF or the local bus.
  - System TI peripheral bus port: Time-out is signaled by the TIPB bridge.

- System EMIFS/EMIFF port: Time-out is signaled by the EMIFS/EMIFF (or traffic controller).

The system DMA has nine physical channels; each has the capability to generate interrupts. The DMA has seven interrupt lines, some of which are shared by two physical channels. Each of these seven interrupt lines is routed as an interrupt input on the MPU level2 interrupt handler.

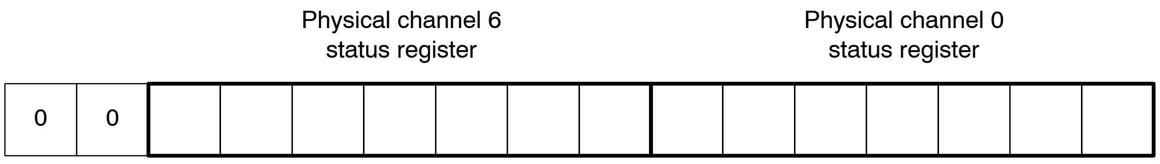
When an interrupt is issued by a physical channel, its status register (DMA_CSR) is set to record the interrupt cause. The processor interrupt service routine (ISR) can read this channel status register to find the source of the interrupt. The status bits are automatically cleared after they are read. One read in the status register clears all the status bits.

- Interrupt line 0 (MPU level2 IRQ19) is shared by channel 0 and 6.
- Interrupt line 1 (MPU level2 IRQ20) is shared by channel 1 and 7.
- Interrupt line 2 (MPU level2 IRQ21) is shared by channel 2 and 8.
- Interrupt line 3 (MPU level2 IRQ22) is dedicated to channel 3.
- Interrupt line 4 (MPU level2 IRQ23) is dedicated to channel 4.
- Interrupt line 5 (MPU level2 IRQ24) is dedicated to channel 5.
- Interrupt line 6 (MPU level2 IRQ25) is dedicated to the LCD channel.

If simultaneous events occur in two physical channels that share the same interrupt line, only one interrupt is generated, and all the relevant status bits are set.

Each physical channel has a 7-bit status register. If two physical channels share one interrupt line, the MPU can distinguish the interrupt source by reading the DMA_CSR register of either channel. Figure 7 shows the data read format for two shared physical channels.

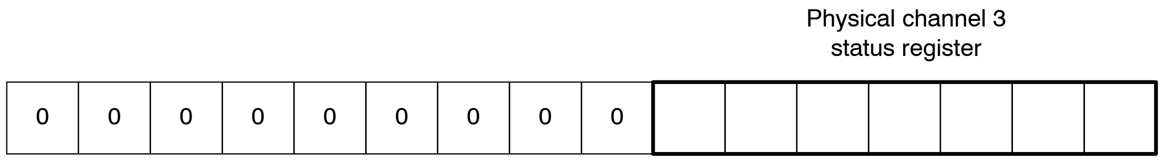
Figure 7. Data Read Format—Two Shared Physical Channels


Physical channel 6 status register
Physical channel 0 status register
0
0

This unique status is accessible either at channel 6 DMA_CSR, or at channel 0 DMA_CSR. Any MPU read (at channel 0 DMA_CSR address or at channel 6 DMA_CSR address) clears all status for the two channels.

When an interrupt is dedicated to one physical channel, the MPU can read the status from this channel in one TIPB access. Figure 8 shows the data read format for one physical channel.

Figure 8. Data Read Format—One Physical Channel



3.1.16 Memory Space Protection

Starting a DMA transfer requires two addresses:

- ❑ A source address within the source port memory space.
- ❑ A destination address within the destination port memory space.

Specifying an address outside the port memory space will cause unexpected results, e.g., an EMIFF source port with an EMIFS starting address specified.

CAUTION

It is the programmer's responsibility to ensure coherency between the source port and source starting address, and between the destination port and destination starting address.

4 LCD Dedicated Channel

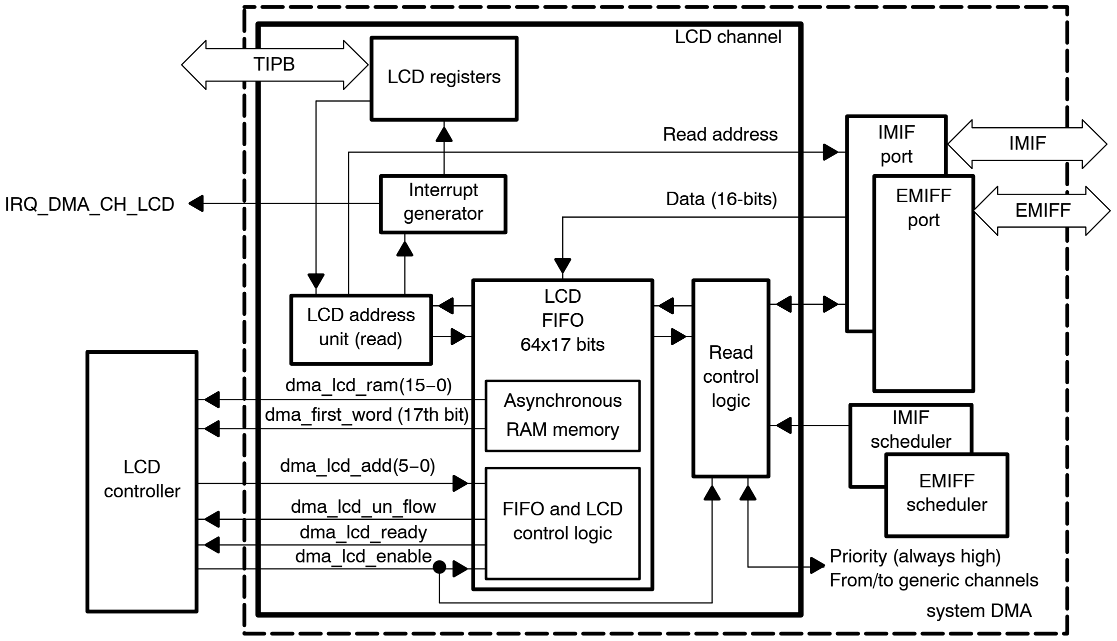
The LCD channel transfers 16-bit data to the LCD controller from a video frame buffer stored in memory.

4.1 Functional Description

The memory source for the LCD dedicated transfer can be either IMIF or EMIFF. These transfers can be arranged to one or two frames but they are always done in frame addressing mode. There is no capability for various addressing modes as in the other channels.

The dual-frame mode allows concurrent transfer and image processing (reload of one frame while another is being processed).

Figure 9. LCD Channel



To support the two frames, the LCD channel loads the top address of the second frame buffer after the first frame buffer has been fully transferred. The LCD channel sends the read request on the relevant port according to the LCD_SOURCE bit.

The LCD_SOURCE bit is read from the DMA_LCD_CTRL register. The LCD channel contains a 64 by 17-bit words FIFO. One bit is used to synchronize the FIFO with the LCD controller and 16 bits are available for user data. To ensure correct throughput from the DMA to the LCD FIFO, the maximum burst length is set to eight. Only near frame boundaries, in case of nonmultiple frames, are single transfers started; otherwise, all requests to the source memory are in 8x16 burst requests. The LCD channel priority bit is fixed high (hard-coded LCD channel constant parameter).

The configuration registers contain top and bottom address registers for the two frame buffers and a control register that manages the operation mode (dual or single), the enable bits for the interrupts, and the source port for the next transfer. It then returns information by setting the status bits in the same control register. An interrupt can be sent at the end of the transfer of each frame. This interrupt line is connected to the global nIRQ line of the DMA.

4.2 Addressing Units

Unlike the general purpose DMA channels, the LCD channel does not have a destination address register because the LCD channel’s destination is fixed to the FIFO of the LCD controller.

4.2.1 LCD Addressing

At the beginning of LCD operation, the LCD channel gives a starting address, which is conventionally called the top address of the current frame buffer. This address is sent to the relevant memory port via the scheduler. LCD DMA requests to the memory port are time-multiplexed along with requests from generic channels, as described in section 1.3.1, *Transfers*.

```
a(0) = SA

if(((a(next) == BF2) & DFM) ||

  ((a(next) = = BF1) & !DFM))

    a(next) = TF1

elseif ((a(next) = = BF1) & DFM)

    a(next) = TF2

else

    a(next) = a(current +2)
```

where:

a(0) is the first address to be computed.

SA is the starting address of the transfer, which is always the top address given for the first frame buffer.

a(current) is the current address of the byte number within the transfer.

a(next) is the next address to be computed.

BF1 is bottom address for frame 1.

BF2 is bottom address for frame 2.

TF1 is top address for frame 1.

TF2 is top address for frame 2.

DFM is the dual-frame mode.

In other words, the next address is always the current address + 2 unless the frame boundaries (inclusive) have been reached (address is a byte address; it is necessary to increment by two, because all LCD transfers are 16 bits). In this case, the next address computed is the top address for the frame 1 if in single frame mode; otherwise, the top address for frame 2 is loaded.

4.3 LCD Channel Usage Restrictions

4.3.1 Exclusive Frames

The hardware design does not support inclusion of a frame buffer into another frame buffer; that is, the starting and ending address of each buffer must represent two different physical parts into the memory. In dual-frame mode the top address for the second frame must be greater (and not equal) than the bottom address of the first frame.

4.3.2 Both Frames Must Belong to the Same Port (EMIFF or IMIF)

In case of dual-frame operation it is not possible to have one frame read from one source port and one frame read from a second source port. For changing from one source port (EMIFF or IMIF) to another, the LCDEN bit of the LCD control register, *LCD Controller Registers*) must be cleared to 0 and all pending LCD interrupts processed.

4.3.3 LCD Registers Must Remain Steady From One Transfer to Another

Changing LCD registers is prohibited until a transfer has fully completed. No separate programming registers exist in the LCD channel as in generic channels. To update registers, the LCDEN bit should be cleared to 0 and all pending LCD interrupts processed.

4.3.4 FIFO Out of Data (Bandwidth Break)

If a time-out or bus error occurs during a LCD transfer, the transfer is suspended, and the IRQ_LCD_CTRL interrupt is generated. The DMA LCD Control register (DMA_LCD_CTRL) is available to determine the cause of the interrupt. The transfer can be resumed by clearing the LCDEN bit field in the LCD Control Register (LCDControl) once, and setting it.

4.4 LCD Transfer Examples

4.4.1 EMIFF to LCD, One Frame

Figure 10 shows a transfer for a video frame located in EMIFF to the LCD controller. The size for the LCD display is 6x16 pixels with 16 bits per pixel.

So the length of the video frame is:

6 x16 x 2 (in bytes) + 32 bytes for the palette = 224 bytes

If the video frame starts at address 0x100B00000, the bottom address of the video frame is 0x100B00DE.

Registers settings are shown in Table 8.

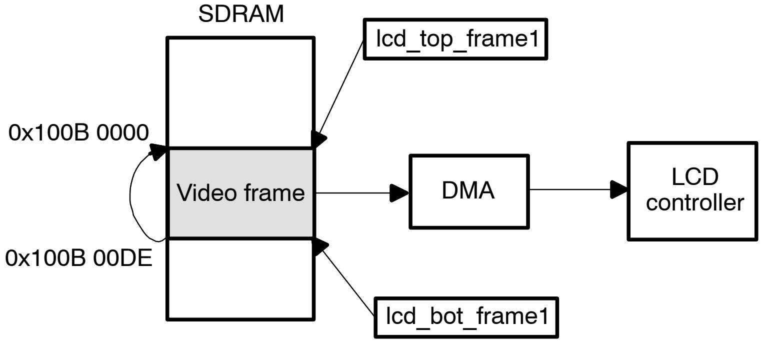
Table 7. EMIFF to LCD Register Settings—One Frame

DMA_LCD_CTRL	Register Settings
Frame_mode	0 (one frame)
Frame_it_ie	1
Bus_error_ie	1
Lcd_source	0 (SDRAM)
DMA_LCD_TOP_F1_U	0x100B
DMA_LCD_TOP_F1_L	0x0000
DMA_LCD_BOT_F1_U	0x100B
DMA_LCD_BOT_F1_L	0x00DE
DMA_LCD_TOP_F2_U	irrelevant
DMA_LCD_TOP_F2_L	irrelevant
DMA_LCD_BOT_F2_U	irrelevant
DMA_LCD_BOT_F2_L	irrelevant

The transfer starts when the enable (hardware) signal from the LCD controller is asserted high.

The transfer runs, and an interruption is generated at the end of the frame.

Figure 10. LCD One Frame Mode Transfer Scheme



When an interrupt occurs, read the DMA_LCD_CTRL register to find the source of the interrupt.

If DMA_LCD_CTRL(3) = 1, end frame 1 interrupt.

If end of frame is reached, the DMA restarts at the top of the frame. Reset DMA_LCD_CTRL(3) and wait for another interrupt.

4.4.2 IMIF to LCD, Two Frames

Figure 11 shows a transfer from two video frames located in IMIF to the LCD controller. The size for the LCD display is 6 x 16 pixels with 16 bits per pixel. So the length of one video frame is:

6 x 16 x 2 (in bytes) + 32 bytes for the palette = 224 bytes

If the video frame 1 starts at address 0x200B0000, the bottom address of the video frame is 0x200B00DE. If the video frame 2 starts at address 0x200C0000, the bottom address of the video frame is 0x200C00DE.

Registers settings are shown in Table 8.

Table 8. IMIF LCD Register Settings—Two Frames

DMA_LCD_CTRL	Register Settings
Frame_mode	1 (two frame)
Frame_it_ie	1
Bus_error_ie	1
Lcd_source	1 (IMIF)
DMA_LCD_TOP_F1_U	0x200B

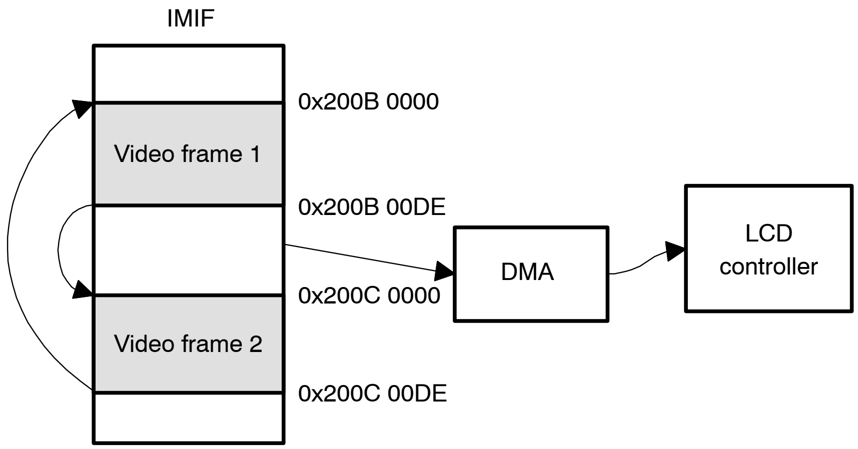
Table 8. IMIF LCD Register Settings—Two Frames (Continued)

DMA_LCD_CTRL	**Register Settings**
DMA_LCD_TOP_F1_L	0x0000
DMA_LCD_BOT_F1_U	0x200B
DMA_LCD_BOT_F1_L	0x00DE
DMA_LCD_TOP_F2_U	0x200C
DMA_LCD_TOP_F2_L	0x0000
DMA_LCD_BOT_F2_U	0x200C
DMA_LCD_BOT_F2_L	0x00DE

The transfer starts when the enable (hardware) signal from the LCD controller is asserted high.

The transfer runs, and the interrupts are generated at the ends of frames 1 and 2.

Figure 11. LCD Dual-Frame Mode Transfer Scheme



When an interrupt occurs, read the DMA_LCD_CTRL register to find the source of the interrupt.

If DMA_LCD_CTRL(3) = 1, end frame 1 interrupt.

If DMA_LCD_CTRL(4) = 1, end frame 2 interrupt.

When the bottom of frame 1 is reached, the DMA loads the address for the top of frame 2. When the bottom of frame 2 is reached, the DMA loads the address for the top of frame 1.

Reset DMA_LCD_CTRL3 and 4 and wait for another interrupt.

5 System DMA Request Mapping

Table 9 shows the system DMA request mapping for the OMAP5910 device, which is set by the SYNC bits of the DMA channel control register (DMA_CCR) for each channel.

Table 9. System DMA Request Mapping

MPU System DMA Requests	DMA_CCR SYNC	MPU System DMA
No Synchronization	00000	NA
MCSI1 TX	00001	DMA_REQ_01
MCSI1 RX	00010	DMA_REQ_02
I^{2}C RX	00011	DMA_REQ_03
I^{2}C TX	00100	DMA_REQ_04
$\overline{\text{EXT\_DMA\_REQ0}}$ (MPUIO2)	00101	DMA_REQ_05
$\overline{\text{EXT\_DMA\_REQ1}}$ (MPUIO4)	00110	DMA_REQ_06
MicroWire TX	00111	DMA_REQ_07
McBSP1 TX	01000	DMA_REQ_08
McBSP1 RX	01001	DMA_REQ_09
McBSP3 TX	01010	DMA_REQ_10
McBSP3 RX	01011	DMA_REQ_011
UART1 TX	01100	DMA_REQ_012
UART1 RX	01101	DMA_REQ_013
UART2 TX	01110	DMA_REQ_014
UART2 RX	01111	DMA_REQ_015
McBSP2 TX	10000	DMA_REQ_016
McBSP2 RX	10001	DMA_REQ_017
UART3 TX	10010	DMA_REQ_018

Table 9. DMA Request Mapping (Continued)

MPU System DMA Requests	DMA_CCR SYNC	MPU System DMA
UART3 RX	10011	DMA_REQ_019
Camera RX	10100	DMA_REQ_020
MMC TX	10101	DMA_REQ_021
MMC RX	10110	DMA_REQ_022
Reserved	10111	DMA_REQ_023
Reserved	11000	DMA_REQ_024
Reserved	11001	DMA_REQ_025
USB function RX0	11010	DMA_REQ_026
USB function RX1	11011	DMA_REQ_027
USB function RX2	11100	DMA_REQ_028
USB function TX0	11101	DMA_REQ_029
USB function TX1	11110	DMA_REQ_030
USB function TX2	11111	DMA_REQ_031

6 Registers

Table 10 describes the DMA controller registers.

Note:

All DMA control registers must be accessed as 16-bit registers.

Base address for system DMA is FFFE:D800

Table 10. DMA Controller Registers

Name	Description	Type	Size (Bits)	Address	Reset Value
DMA_GCR	Global control	R/W	16	0xFFFEDC00	0x0008
DMA_CSDP_CH0	Channel 0 source destination parameters	R/W	16	0xFFFED800	0x0000
DMA_CCR_CH0	Channel 0 control	R/W	16	0xFFFED802	0x0000
DMA_CICR_CH0	Channel 0 interrupt control	R/W	16	0xFFFED804	0x0003

Table 10. DMA Controller Registers (Continued)

Name	Description	Type	Size (Bits)	Address	Reset Value
DMA_CSR_CH0	Channel 0 status	R	16	0xFFFED806	0x0000
DMA_CSSA_L_CH0	Channel 0 source starting address—lower bits	R/W	16	0xFFFED808	U
DMA_CSSA_U_CH0	Channel 0 source starting address—upper bits	R/W	16	0xFFFED80A	U
DMA_CDSA_L_CH0	Channel 0 destination starting address—lower bits	R/W	16	0xFFFED80C	U
DMA_CDSA_U_CH0	Channel 0 destination starting address—upper bits	R/W	16	0xFFFED80E	U
DMA_CEN_CH0	Channel 0 element number	R/W	16	0xFFFED810	U
DMA_CFN_CH0	Channel 0 frame number	R/W	16	0xFFFED812	U
DMA_CFI_CH0	Channel 0 frame index	R/W	16	0xFFFED814	U
DMA_CEI_CH0	Channel 0 element index	R/W	16	0xFFFED816	U
DMA_CPC_CH0	Channel 0 channel progress counter	R/W	16	0xFFFED818	U
DMA_CSDP_CH1	Channel 1 source destination parameters	R/W	16	0xFFFED840	0x0000
DMA_CCR_CH1	Channel 1 control	R/W	16	0xFFFED842	0x0000
DMA_CICR_CH1	Channel 1 interrupt control	R/W	16	0xFFFED844	0x0003
DMA_CSR_CH1	Channel 1 status	R	16	0xFFFED846	0x0000
DMA_CSSA_L_CH1	Channel 1 source starting address lower bits	R/W	16	0xFFFED848	U
DMA_CSSA_U_CH1	Channel 1 source starting address upper bits	R/W	16	0xFFFED84A	U
DMA_CDSA_L_CH1	Channel 1 destination starting address lower bits	R/W	16	0xFFFED84C	U
DMA_CDSA_U_CH1	Channel 1 destination starting address upper bits	R/W	16	0xFFFED84E	U
DMA_CEN_CH1	Channel 1 element number	R/W	16	0xFFFED850	U
DMA_CFN_CH1	Channel 1 frame number	R/W	16	0xFFFED852	U

Table 10. DMA Controller Registers (Continued)

Name	Description	Type	Size (Bits)	Address	Reset Value
DMA_CFI_CH1	Channel 1 frame index	R/W	16	0xFFFED854	U
DMA_CEI_CH1	Channel 1 element index	R/W	16	0xFFFED856	U
DMA_CPC_CH1	Channel 1 channel progress counter	R/W	16	0xFFFED858	U
DMA_CSDP_CH2	Channel 2 source destination parameters	R/W	16	0xFFFED880	0x0000
DMA_CCR_CH2	Channel 2 control	R/W	16	0xFFFED882	0x0000
DMA_CICR_CH2	Channel 2 interrupt control	R/W	16	0xFFFED884	0x0003
DMA_CSR_CH2	Channel 2 status	R	16	0xFFFED886	0x0000
DMA_CSSA_L_CH2	Channel 2 source starting address lower bits	R/W	16	0xFFFED888	U
DMA_CSSA_U_CH2	Channel 2 source starting address upper bits	R/W	16	0xFFFED88A	U
DMA_CDSA_L_CH2	Channel 2 destination starting address lower bits	R/W	16	0xFFFED88C	U
DMA_CDSA_U_CH2	Channel 2 destination starting address upper bits	R/W	16	0xFFFED88E	U
DMA_CEN_CH2	Channel 2 element number	R/W	16	0xFFFED890	U
DMA_CFN_CH2	Channel 2 frame number	R/W	16	0xFFFED892	U
DMA_CFI_CH2	Channel 2 frame index	R/W	16	0xFFFED894	U
DMA_CEI_CH2	Channel 2 element index	R/W	16	0xFFFED896	U
DMA_CPC_CH2	Channel 2 channel progress counter	R/W	16	0xFFFED898	U
DMA_CSDP_CH3	Channel 3 source destination parameters	R/W	16	0xFFFED8C0	0x0000
DMA_CCR_CH3	Channel 3 control	R/W	16	0xFFFED8C2	0x0000
DMA_CICR_CH3	Channel 3 interrupt control	R/W	16	0xFFFED8C4	0x0003
DMA_CSR_CH3	Channel 3 status	R	16	0xFFFED8C6	0x0000
DMA_CSSA_L_CH3	Channel 3 source starting address lower bits	R/W	16	0xFFFED8C8	U

Table 10. DMA Controller Registers (Continued)

Name	Description	Type	Size (Bits)	Address	Reset Value
DMA_CSSA_U_CH3	Channel 3 source starting address upper bits	R/W	16	0xFFFED8CA	U
DMA_CDSA_L_CH3	Channel 3 destination starting address lower bits	R/W	16	0xFFFED8CC	U
DMA_CDSA_U_CH3	Channel 3 destination starting address upper bits	R/W	16	0xFFFED8CE	U
DMA_CEN_CH3	Channel 3 element number	R/W	16	0xFFFED8D0	U
DMA_CFN_CH3	Channel 3 frame number	R/W	16	0xFFFED8D2	U
DMA_CFI_CH3	Channel 3 frame index	R/W	16	0xFFFED8D4	U
DMA_CEI_CH3	Channel 3 element index	R/W	16	0xFFFED8D6	U
DMA_CPC_CH3	Channel 3 channel progress counter	R/W	16	0xFFFED8D8	U
DMA_CSDP_CH4	Channel 4 source destination parameters	R/W	16	0xFFFED900	0x0000
DMA_CCR_CH4	Channel 4 control	R/W	16	0xFFFED902	0x0000
DMA_CICR_CH4	Channel 4 interrupt control	R/W	16	0xFFFED904	0x0003
DMA_CSR_CH4	Channel 4 status	R	16	0xFFFED906	0x0000
DMA_CSSA_L_CH4	Channel 4 source starting address lower bits	R/W	16	0xFFFED908	U
DMA_CSSA_U_CH4	Channel 4 source starting address upper bits	R/W	16	0xFFFED90A	U
DMA_CDSA_L_CH4	Channel 4 destination starting address lower bits	R/W	16	0xFFFED90C	U
DMA_CDSA_U_CH4	Channel 4 destination starting address upper bit	R/W	16	0xFFFED90E	U
DMA_CEN_CH4	Channel 4 element number	R/W	16	0xFFFED910	U
DMA_CFN_CH4	Channel 4 frame number	R/W	16	0xFFFED912	U
DMA_CFI_CH4	Channel 4 frame index	R/W	16	0xFFFED914	U
DMA_CEI_CH4	Channel 4 element index	R/W	16	0xFFFED916	U
DMA_CPC_CH4	Channel 4 channel progress counter	R/W	16	0xFFFED918	U

Table 10. DMA Controller Registers (Continued)

Name	Description	Type	Size (Bits)	Address	Reset Value
DMA_CSDP_CH5	Channel 5 source destination parameters	R/W	16	0xFFFED940	0x0000
DMA_CCR_CH5	Channel 5 control	R/W	16	0xFFFED942	0x0000
DMA_CICR_CH5	Channel 5 interrupt control	R/W	16	0xFFFED944	0x0003
DMA_CSR_CH5	Channel 5 status	R	16	0xFFFED946	0x0000
DMA_CSSA_L_CH5	Channel 5 source starting address lower bits	R/W	16	0xFFFED948	U
DMA_CSSA_U_CH5	Channel 5 source starting address upper bits	R/W	16	0xFFFED94A	U
DMA_CDSA_L_CH5	Channel 5 destination starting address lower bits	R/W	16	0xFFFED94C	U
DMA_CDSA_U_CH5	Channel 5 destination starting address upper bits	R/W	16	0xFFFED94E	U
DMA_CEN_CH5	Channel 5 element number	R/W	16	0xFFFED950	U
DMA_CFN_CH5	Channel 5 frame number	R/W	16	0xFFFED952	U
DMA_CFI_CH5	Channel 5 frame index	R/W	16	0xFFFED954	U
DMA_CEI_CH5	Channel 5 element index	R/W	16	0xFFFED956	U
DMA_CPC_CH5	Channel 5 channel progress counter	R/W	16	0xFFFED958	U
DMA_CSDP_CH6	Channel 6 source destination parameters	R/W	16	0xFFFED980	0x0000
DMA_CCR_CH6	Channel 6 control	R/W	16	0xFFFED982	0x0000
DMA_CICR_CH6	Channel 6 interrupt control	R/W	16	0xFFFED984	0x0003
DMA_CSR_CH6	Channel 6 status	R	16	0xFFFED986	0x0000
DMA_CSSA_L_CH6	Channel 6 source starting address lower bit	R/W	16	0xFFFED988	U
DMA_CSSA_U_CH6	Channel 6 source starting address upper bits	R/W	16	0xFFFED98A	U
DMA_CDSA_L_CH6	Channel 6 destination starting address lower bits	R/W	16	0xFFFED98C	U
DMA_CDSA_U_CH6	Channel 6 destination starting address upper bits	R/W	16	0xFFFED98E	U

Table 10. DMA Controller Registers (Continued)

Name	Description	Type	Size (Bits)	Address	Reset Value
DMA_CEN_CH6	Channel 6 element number	R/W	16	0xFFFED990	U
DMA_CFN_CH6	Channel 6 frame number	R/W	16	0xFFFED992	U
DMA_CFI_CH6	Channel 6 frame index	R/W	16	0xFFFED994	U
DMA_CEI_CH6	Channel 6 element index	R/W	16	0xFFFED996	U
DMA_CPC_CH6	Channel 6 channel progress counter	R/W	16	0xFFFED998	U
DMA_CSDP_CH7	Channel 7 source destination parameters	R/W	16	0xFFFED9C0	0x0000
DMA_CCR_CH7	Channel 7 control	R/W	16	0xFFFED9C2	0x0000
DMA_CICR_CH7	Channel 7 interrupt control	R/W	16	0xFFFED9C4	0x0003
DMA_CSR_CH7	Channel 7 status	R	16	0xFFFED9C6	0x0000
DMA_CSSA_L_CH7	Channel 7 source starting address lower bits	R/W	16	0xFFFED9C8	U
DMA_CSSA_U_CH7	Channel 7 source starting address upper bits	R/W	16	0xFFFED9CA	U
DMA_CDSA_L_CH7	Channel 7 destination starting address lower bits	R/W	16	0xFFFED9CC	U
DMA_CDSA_U_CH7	Channel 7 destination starting address lower bits	R/W	16	0xFFFED9CE	U
DMA_CEN_CH7	Channel 7 element number	R/W	16	0xFFFED9D0	U
DMA_CFN_CH7	Channel 7 frame number	R/W	16	0xFFFED9D2	U
DMA_CFI_CH7	Channel 7 frame index	R/W	16	0xFFFED9D4	U
DMA_CEI_CH7	Channel 7 element frame	R/W	16	0xFFFED9D6	U
DMA_CPC_CH7	Channel 7 channel progress counter	R/W	16	0xFFFED9D8	U
DMA_CSDP_CH8	Channel 8 source destination parameters	R/W	16	0xFFFEDA00	0x0000
DMA_CCR_CH8	Channel 8 control	R/W	16	0xFFFEDA02	0x0000
DMA_CICR_CH8	Channel 8 interrupt control	R/W	16	0xFFFEDA04	0x0003
DMA_CSR_CH8	Channel 8 status	R	16	0xFFFEDA06	0x0000
DMA_CSSA_L_CH8	Channel 8 source starting address lower bits	R/W	16	0xFFFEDA08	U

Table 10. DMA Controller Registers (Continued)

Name	Description	Type	Size (Bits)	Address	Reset Value
DMA_CSSA_U_CH8	Channel 8 source starting address upper bits	R/W	16	0xFFFEDA0A	U
DMA_CDSA_L_CH8	Channel 8 destination starting address lower bits	R/W	16	0xFFFEDA0C	U
DMA_CDSA_U_CH8	Channel 8 destination starting address upper bits	R/W	16	0xFFFEDA0E	U
DMA_CEN_CH8	Channel 8 element number	R/W	16	0xFFFEDA10	U
DMA_CFN_CH8	Channel 8 frame number	R/W	16	0xFFFEDA12	U
DMA_CFI_CH8	Channel 8 frame index	R/W	16	0xFFFEDA14	U
DMA_CEI_CH8	Channel 8 element index	R/W	16	0xFFFEDA16	U
DMA_CPC_CH8	Channel 8 channel progress counter	R/W	16	0xFFFEDA18	U
DMA_LCD_CTRL	LCD control	R/W	16	0xFFFEDB00	0x0000
DMA_LCD_TOP_ F1_L	LCD top address for frame buffer 1 lower bits	R/W	16	0xFFFEDB02	U
DMA_LCD_TOP_ F1_U	LCD top address for frame buffer 1 upper bits	R/W	16	0xFFFEDB04	U
DMA_LCD_BOT_ F1_L	LCD bottom address for frame buffer 1 lower bits	R/W	16	0xFFFEDB06	U
DMA_LCD_BOT_ F1_U	LCD bottom address for frame buffer 1 upper bits	R/W	16	0xFFFEDB08	U
DMA_LCD_TOP_ F2_L	LCD top address for frame buffer 2 lower bits	R/W	16	0xFFFEDB0A	U
DMA_LCD_TOP_ F2_U	LCD top address for frame buffer 2 upper bits	R/W	16	0xFFFEDB0C	U
DMA_LCD_BOT_ F2_L	LCD bottom address for frame buffer 2 lower bits	R/W	16	0xFFFEDB0E	U
DMA_LCD_BOT_ F2_U	LCD bottom address for frame buffer 2 upper bits	R/W	16	0xFFFEDB10	U

Table 11 shows the global control register bit descriptions.

Table 11. DMA Global Control register (DMA_GCR)

Bit	Name	Value	Description	Type	Reset Value
15−4	RESERVED				
3	AUTOGATING_ON		DMA clock autogating is as follows:	R/W	1
		0	Reserved. Do not use this setting.		
		1	Allows the DMA to dynamically cut off its clocks according to its activity. This bit should always be set to 1.		
2	FREE		DMA reaction to the suspend signal is as follows:	R/W	0
		0	The DMA suspends all the current transfers when it receives the suspend signal from the processor. Transfers resume when the processor releases the suspend signal. The DMA clock must not be cut off when the DMA is suspended.		
		1	The DMA continues running when it receives the suspend signal from the processor (when the processor is halted for debug by a breakpoint, for example).		
1−0	RESERVED				

6.1 Generic Channel Registers

There is one set of these registers for each generic DMA channel. Although the DMA has a 32-bit TIPB, all registers are in 16-bit format and must be accessed as 16-bit data by the MPU.

Table 12. Channel Source Destination Parameters Register (DMA_CSDP)

Bit	Name	Value	Description	Type	Reset Value
15−14	DST_BURST_EN		Destination burst enable	R/W	00
			Enable/disable bursting on the destination port. When bursting is enabled, the destination port performs bursts 4 x dst_width. When bursting is disabled, the destination port performs single accesses of dst_width bits.		
		00	Single access (no burst)		
		01	Single access (no burst)		

Table 12. Channel Source Destination Parameters Register (DMA_CSDP) (Continued)

Bit	Name	Value	Description	Type	Reset Value
		10	Burst 4		
		11	Reserved (do not use this setting)		
13	DST_PACK		Destination packing	R/W	0
			The DMA ports can have a data bus width different from that of the type of data moved by the DMA channel. For example, s8 data can be read on a 32-bit DMA port. The DMA channel has the capacity to pack four consecutive s8 data reads in a single 32-bit read access to increase bandwidth.		
		0	The destination port never makes packed accesses.		
		1	The destination port makes packed accesses.		
12−9	DST		Transfer destination	R/W	0000
			A unique identifier is given to each port. This field indicates which port is the destination of the transfer.		
		0000	EMIFF		
		0001	EMIFS		
		0010	IMIF		
		0011	TIPB		
		0100	Local		
		0101	TIPB_MPUI		
			Others: Illegal (do not use this setting)		
8−7	SRC_BURST_EN		Source burst enable	R/W	00
			Enable/disable bursting on the source port. When bursting is enabled, the source port performs bursts 4 x src_width. When bursting is disabled, the source port performs single accesses of src_width bits.		
		00	Single access (no burst)		
		01	Single access (no burst)		

Table 12. Channel Source Destination Parameters Register (DMA_CSDP) (Continued)

Bit	Name	Value	Description	Type	Reset Value
		10	Burst 4		
		11	Reserved (do not use this setting)		
6	SRC_PACK		Source packing	R/W	0
			The DMA ports can have a data bus width different from that of the type of data moved by the DMA channel. For example, s8 data can be read on a 32-bit DMA port. The DMA channel has the capacity to pack four consecutive s8 data reads in a single 32-bit read access to increase bandwidth.		
		0	The source port never makes packed accesses.		
		1	The source port makes packed accesses.		
5−2	SRC		Transfer source	R/W	0000
			A unique identifier is given to each port. This field indicates which port is the originator of the transfer.		
		0000	EMIFF		
		0001	EMIFS		
		0010	IMIF		
		0011	TIPB		
		0100	Local		
		0101	TIPB_MPUI		
			Others: Illegal (do not use this setting)		

Table 12. Channel Source Destination Parameters Register (DMA_CSDP) (Continued)

Bit	Name	Value	Description	Type	Reset Value
1−0	DATA_TYPE		Defines the type of the data moved in the channel	R/W	00
		00	s8, 8 bits scalar		
		01	s16, 16 bits scalar		
		10	s32, 32 bits scalar		
		11	Illegal value		
			The starting address must be aligned on the boundary of the type of data moved. For example, if the type is s32, the source and destination starting addresses must be aligned on a 32-bit word. If the type is s8, the source and destination starting addresses can have any value. The DMA forces by hardware the starting address value on the type of data transferred.		

Table 13. DMA Channel Control Register (DMA_CCR)

Bit	Name	Value	Description	Type	Reset Value
15−14	DST_AMODE		Destination addressing mode This field is used to choose the addressing mode on the destination port of a channel.	R/W	00
		00	Constant address		
		01	Post-incremented address		
		10	Single index (element index)		
		11	Double index (element index and frame index)		

Table 13. DMA Channel Control Register (DMA_CCR) (Continued)

Bit	Name	Value	Description	Type	Reset Value
13–12	SRC_AMODE		Source addressing mode This field is used to choose the addressing mode on the source port of a channel.	R/W	00
		00	Constant address		
		01	Post-incremented address		
		10	Single index (element index)		
		11	Double index (element index and frame index)		
11	END_PROG		End of programming	R/W-RST	0
		0	Delays the channel autoinitialization if AUTO_INIT = 1 and REPEAT = 0.		
		1	Allows the channel to reinitialize itself if AUTO_INIT = 1		
10	RESERVED			R/W-RST	0
9	REPEAT		Repetitive operations	R/W	0
		0	When the current transfer is complete, the channel automatically reinitializes itself and starts a new transfer only if END_PROG = 1.		
		1	When the current transfer is complete, the channel automatically reinitializes itself and starts a new transfer disregarding END_PROG.		
8	AUTO_INIT		Autoinitialization at the end of the transfer	R/W	0
		0	The channel stops at the end of the current transfer.		
		1	When the current transfer is complete, the channel automatically reinitializes itself and starts a new transfer.		

Table 13. DMA Channel Control Register (DMA_CCR) (Continued)

Bit	Name	Value	Description	Type	Reset Value
			There are two ways to stop a channel while it is in autoinitialization mode: ❑ Write a 0 to the DMA_CCR EN bit; the channel immediately stops. ❑ Write a 0 to the DMA_CCR AUTO_INIT bit; the channel completes the current transfer and stops.		
7	EN		Enable This bit is used to enable/disable the transfer in the DMA channel.	R/W-RST	0
		0	The transfer stops, and it is reset.		
		1	The transfer starts.		
			This bit is automatically cleared by the DMA once the transfer is accomplished. Clearing of this bit by the DMA has priority over the write by the processor. If both simultaneously occur, the processor write is discarded.		
6	PRIO		Channel priority	R/W	0
		0	The channel has low priority level.		
		1	The channel has high priority level.		
5	FS		Frame synchronization This bit is used to program the way a DMA request is serviced in a synchronized transfer.	R/W	0
		0	An element is transferred each time a DMA request is made. This element can be interleaved on the DMA port with other channel requests.		
		1	An entire frame is transferred each time a DMA request is made. This frame can be interleaved on the DMA ports with other channel requests.		
4–0	SYNC		Synchronization control This field is used to specify the external DMA request, which can trigger the transfer in this channel. One DMA request among 15 possible can be chosen. The values for this field are defined in Table 9.	R/W	00000

Table 13. DMA Channel Control Register (DMA_CCR) (Continued)

Bit	Name	Value	Description	Type	Reset Value
		00000	Transfer not synchronized		
		i	Transfer synchronized on DMA request [i], i ≠ 0 regarding the table described		

Table 14. DMA Channel Interrupt Control Register (DMA_CICR)

Bit	Name	Value	Description	Type	Reset Value
15−7	RESERVED				
6	RESERVED			R	0
5	BLOCK_IE		End block interrupt enable	R/W	0
		0	The channel does not interrupt the processor when the transfer of the block completes.		
		1	The channel sends an interrupt to the processor when the transfer of the block completes.		
4	LAST_IE		Last frame interrupt enable	R/W	0
		0	The channel does not interrupt the processor when the transfer of the last frame starts.		
		1	The channel sends an interrupt to the processor when the transfer of the last frame starts.		
3	FRAME_IE		Frame interrupt enable	R/W	0
		0	The channel does not interrupt the processor when the transfer of the current frame completes.		
		1	The channel sends an interrupt to the processor when the transfer of the current frame completes.		
2	HALF_IE		Half frame interrupt enable	R/W	0
		0	The channel does not interrupt the processor when the transfer of the first half of the current frame completes.		
		1	The channel sends an interrupt to the processor when the transfer of the first half of the current frame completes.		
1	DROP_IE		Synchronization event drop interrupt enable	R/W	1
		0	The channel does not interrupt the processor when a synchronization event drop occurs.		
		1	The channel sends an interrupt to the processor if the channel transfer is synchronized on DMA requests and on successive DMA request drops. This occurs when a new DMA request is made while the service of the previous one is not finished yet.		

Table 14. DMA Channel Interrupt Control Register (DMA_CICR) (Continued)

Bit	Name	Value	Description	Type	Reset Value
0	TOUT_IE		Time-out interrupt enable	R/W	1
		0	The DMA does not send an interrupt to the processor if a time-out error occurs.		
		1	The DMA sends an interrupt to the processor if a time-out error occurs either in the source or in the destination port of the channel.		

The interrupt enable bits are used to choose the events that cause the DMA channel to send an interrupt to the processor. There are two classes of events:

- Error events: errors during the transfer (time out, event drop)
- Status events: new frame starts, end of data block to transfer is reached.

Each time an event occurs, if the corresponding interrupt enable bit is set, the channel sends an interrupt to the processor. At the same time, the corresponding status bit is set in DMA_CSR (DMA channel status register) or in DMA_TSR (DMA time-out error status register). A status bit is not set if the corresponding interrupt enable bit in DMA_CICR equals 0.

Table 15. DMA Channel Status Register (DMA_CSR)

Bit	Name	Value	Description	Type	Reset Value
15−14	RESERVED				
13−7	ALT_STATUS		Alternate status bits for channels with shared interrupts. For DMA channels with shared interrupts, these seven bits have the same function as bits 6−0 of this register, except they correspond to the other channel that shares the interrupt. For example, in register DMA_CSR_CH0, these bits correspond to channel 6 status and mirror the values present in DMA_CSR_CH6[6−0]. DMA_CSR registers for both channel 0 and 6 are cleared when either register is read. For channels without shared interrupts, these bits are reserved.		0000000

Table 15. DMA Channel Status Register (DMA_CSR) (Continued)

Bit	Name	Value	Description	Type	Reset Value
6	SYNC		Synchronization status This bit is not set to one when an interrupt is generated, but when a DMA request is made in a synchronized channel. When the DMA request is serviced, the bit returns to zero.	R	0
		0	No DMA request is in service.		
		1	A DMA request was made for this channel when it was in service.		
5	BLOCK		End block	R	0
		0	Current transfer is not finished yet.		
		1	The current transfer in the channel is finished (another one may have start if DMA_CCR2 AUTOINIT = 1).		
4	LAST		Last frame	R	0
		0	Last frame did not start yet.		
		1	The transfer of the last frame has started.		
3	FRAME		Frame	R	0
		0	Transfer of the current frame still in progress		
		1	A complete frame was transferred.		
2	HALF		Half frame	R	0
		0	First half of the current frame not transferred yet		
		1	First half of the current frame was transferred.		
1	DROP		Event drop	R	0
		0	No event drop occurred during the transfer.		
		1	An event drop occurred during the transfer.		
0	TOUT		Time-out in the channel	R	0
		0	No time-out error occurred in channel.		
		1	Time-out occurred in channel.		

This register is written by the DMA controller to reflect the channel status. The MPU can read this register (by polling or after an interrupt) to monitor the DMA status. After the MPU reads this register, all of the DMA_CSR bits will be automatically cleared. The DMA_CSR bit is not cleared with an emulation read via the debugger. The bit will be set only if the corresponding DMA_CICR bit is enabled.

The MPU must read the DMA_CSR register before the next DMA transfer generates an interrupt.

Table 16. DMA Channel Source Starting Address-Lower Bits Register (DMA_CSSA_L)

Bit	Name	Description	Type	Reset Value
15–0	Source starting address, lower bits	Lower bits of the source starting address, expressed in bytes. The source starting address output by the DMA is an up-to-32-bit byte address made of the concatenation of DMA_CSSA_U and DMA_CSSA_L.	R/W	Undefined

Table 17. DMA Channel Source Starting Address-Upper Bits Register (DMA_CSSA_U)

Bit	Name	Description	Type	Reset Value
15–0	Source starting address, upper bits	Upper bits of the source starting address, expressed in bytes. The source starting address output by the DMA is a 32-bit byte address made of the concatenation of DMA_CSSA_U and DMA_CSSA_L.	R/W	Undefined

Table 18. DMA Channel Destination Starting Address-Lower Bits Register (DMA_CDSA_L)

Bit	Name	Description	Type	Reset Value
15–0	Destination starting address, lower bits	Lower bits for the destination starting address, expressed in bytes. The destination starting address is up to an up-to-32-bit byte address made of the concatenation of DMA_CDSA_U and DMA_CDSA_L.	R/W	Undefined

Table 19. DMA Channel Destination Starting Address-Upper Bits Register (DMA_CDSA_U)

Bit	Name	Description	Type	Reset Value
15−0	Destination starting address, upper bits	Upper bits for the source starting address, expressed in bytes. The destination starting address is made of the concatenation of DMA_CDSA_U and DMA_CDSA_L.	R/W	Undefined

Table 20. DMA Channel Element Number Register (DMA_CEN)

Bit	Name	Description	Type	Reset Value
15−0	Channel element number	Number of elements within a frame. The maximum element number is 65535.	R/W	Undefined

Table 21. DMA Channel Frame Number Register (DMA_CFN)

Bit	Name	Description	Type	Reset Value
15−0	Channel frame number	Number of frames within the block to transfer. The maximum frame number is 65535. The size in bytes of the data block to transfer is DMA_CFN x DMA_CEN x DMA_CES. This size is programmed in bytes to: ❑ Allow transfer of an odd byte number ❑ Accommodate the requirement of different access sizes on source and destination ports	R/W	Undefined

Table 22. DMA Channel Frame Index Register (DMA_CFI)

Bit	Name	Description	Type	Reset Value
15−0	Frame index	Contains the frame index, expressed in bytes, used to compute the addresses when double-index addressing mode is used.	R/W	Undefined

Table 23. DMA Channel Element Index Register (DMA_CEI)

Bit	Name	Description	Type	Reset Value
15−0	Element index	Contains the element index, expressed in bytes, used to compute the addresses when single-index addressing mode is used.	R/W	Undefined

Table 24. DMA Channel Progress Counter Register (DMA_CPC)

Bit	Name	Description	Type	Reset Value
15−0	Last element/ frame address 16 LSB	This register can be used to monitor the progress of a DMA transfer: ❑ If the channel transfer is synchronized on elements (DMA_CCR SYNC ≠ 0 and DMA_CCR FS = 0), the register is updated with the address 16 LSB each time the destination port issues the last request for an element. ❑ If the channel transfer is synchronized on frames (DMA_CCR SYNC ≠ 0 and DMA_CCR FS = 1) or not synchronized (DMA_CCR SYNC = 0), the register is updated with 16 LSB of the address each time the destination port issues the last request for a frame.	R	Undefined

The DMA LCD control register contains seven bits that control the LCD channel operation. There are two cases of interruption: end frame buffer or abort on the bus (bus error). Bit IE (interrupt enable) enables the generation of the interruption.

If the COND bit and the corresponding IE bit are set, an interrupt signal is sent from the DMA channel to the MPU. The MPU reads this register to find the cause of the interruption.

- ❑ COND = 0: Condition not detected
- ❑ COND = 1: Condition detected

The COND bit is set when a transfer starts, and cleared automatically when the MPU reads the DMA LCD Control Register (DMA_LCD_CTRL).

Table 25. DMA LCD Control Register (DMA_LCD_CTRL)

Bit	Name	Value	Description	Type	Reset Value
15−7	RESERVED				
6	LCD_SOURCE		Memory source for the LCD channel This bit indicates the memory source for the next LCD transfer.	R/W	0
		0	Memory source is EMIFF.		
		1	Memory source is IMIF.		
5	BUS_ERROR_IT_COND		Status LCD channel register (must be reset after read)	R−R	0
		0	No bus error interrupt detected		
		1	Bus error interrupt detected		
4	FRAME_2_IT_COND		Status LCD channel register (must be reset after read)	R−R	0
		0	No end of frame 2 interrupt detected		
		1	End of frame 2 interrupt detected		
3	FRAME_1_IT_COND		Status LCD channel register (must be reset after read)	R−R	0
		0	No end of frame 1 interrupt detected		
		1	End of frame 1 interrupt detected		
2	BUS_ERROR_IT_IE		Bus error interrupt enable	R/W	0
		0	Interrupt disabled		
		1	Interrupt enabled		
1	FRAME_IT_IE		End frame interrupt enable	R/W	0
		0	Interrupt disabled		
		1	Interrupt enabled		

Table 25. DMA LCD Control Register (DMA_LCD_CTRL) (Continued)

Bit	Name	Value	Description	Type	Reset Value
0	FRAME_MODE		Kind of frame mode used for LCD transfer	R/W	0
		0	One frame buffer; only registers for frame 1 are used.		
		1	Two frame buffers; LCD channel reads alternatively top_frame_1 and top_frame_2		

6.1.1 LCD Top Address for Frame Buffer 1 Registers (DMA_LCD_TOP_F1_L and DMA_LCD_TOP_F1_U)

The LCD top address registers are two 16-bit registers that contain the starting address for the video RAM buffer 1. The 32-bit address is obtained by the concatenation of the two 16-bit words as described here:
LCD_TOP_F1 = DMA_LCD_TOP_F1_U & DMA_LCD_TOP_F1_L.

Note:

The top address for the frame buffer must be aligned on a 16-byte boundary.

Table 26. LCD Top Address for Frame Buffer 1—Lower Bits Register (DMA_LCD_TOP_F1_L)

Bit	Name	Description	Type	Reset Value
15−1	LCD_TOP_F1_ L[15−1]	LCD top address for frame buffer 1 lower bits [15−1]	R/W	Unde-fined
0	LCD_TOP_F1_ L[0]	Address bit 0. Fixed at 0 since address must be even.	R	0

Table 27. LCD Top Address for Frame Buffer 1—Upper Bits Register (DMA_LCD_TOP_F1_U)

Bit	Name	Description	Type	Reset Value
15−0	LCD_TOP_F1_ L[31−16]	LCD top address for frame buffer 1 upper bits [31−16]	R/W	Undefin ed

6.1.2 LCD Bottom Address for Frame Buffer 1 Registers (DMA_LCD_BOT_F1_L and DMA_LCD_BOT_F1_

The LCD bottom address registers are two 16-bit registers that contain the bottom address for the video RAM buffer 1. The 32-bit address is obtained by the concatenation of the two 16-bit words as described here:
LCD_BOTTOM_F1 = DMA_LCD_BOT_F1_U and DMA_LCD_BOT_F1_L

Note:

The bottom address for the frame buffer must be aligned on a 16−byte boundary.

Table 28. LCD Bottom Address for Frame Buffer 1 Register—Lower Bits Register (DMA_LCD_BOT_F1_L)

Bit	Name	Description	Type	Reset Value
15−1	LCD_BOT_F1_ L[15−1]	LCD bottom address for frame buffer 1 lower bits [15−1]	R/W	Undefined
0	LCD_BOT_F1_ L[0]	Address bit 0. Fixed at 0 since address must be even.	R	0

Table 29. LCD Bottom Address for Frame Buffer 1 Register—Upper Bits Register (DMA_LCD_BOT_F1_U)

Bit	Name	Description	Type	Reset Value
15−0	LCD_BOT_F1_ L[31−16]	LCD bottom address for frame buffer 1 upper bits [31−16]	R/W	Undefined

6.1.3 LCD Top Address for Frame Buffer 2 Registers (DMA_LCD_TOP_F2_L and DMA_LCD_TOP_F2_U)

The LCD top address registers are two 16-bit registers that contain the starting address for the video RAM buffer 2. The 32-bit address is obtained by the concatenation of the two 16-bit words as described here:
LCD_TOP_F2 = DMA_LCD_TOP_F2_U & DMA_LCD_TOP_F2_L

Note:

The top address for the frame buffer must be aligned on a 16-byte boundary.

Table 30. LCD Top Address for Frame Buffer 2—Lower Bits Register (DMA_LCD_TOP_F2_L)

Bit	Name	Description	Type	Reset Value
15–1	LCD_TOP_F2_ L[15–1]	LCD top address for frame buffer 2 lower bits [15–1]	R/W	Undefined
0	LCD_TOP_F2_ L[0]	Address bit 0. Fixed at 0 since address must be even.	R	0

Table 31. LCD Top Address for Frame Buffer 2—Upper Bits Register (DMA_LCD_TOP_F2_U)

Bit	Name	Description	Type	Reset Value
15–0	LCD_TOP_F2_ L[31–16]	LCD top address for frame buffer 2 upper bits [31–16]	R/W	Undefined

6.1.4 LCD Bottom Address for Frame Buffer 2 Registers (DMA_LCD_BOT_F2_L and DMA_LCD_BOT_F2_U)

The LCD bottom address registers are two 16-bit registers that contain the bottom address for the video RAM buffer 2. The 32-bit address is obtained by the concatenation of the two 16-bit words as described here:
LCD_BOTTOM_F2 = DMA_LCD_BOT_F2_U and DMA_LCD_BOT_F2_L

Note:

The bottom address for the frame buffer must be aligned on a 16-byte boundary.

Table 32. LCD Bottom Address for Frame Buffer 2—Lower Bits Register (DMA_LCD_BOT_F2_L)

Bit	Name	Description	Type	Reset Value
15−1	LCD_BOT_F2_ L[15−1]	LCD bottom address for frame buffer 2 lower bits [15−1]	R/W	Undefined
0	LCD_BOT_F2_ L[0]	Address bit 0. Fixed at 0 since address must be even.	R	0

Table 33. LCD Bottom Address for Frame Buffer 2—Upper Bits Register (DMA_LCD_BOT_F2_U)

Bit	Name	Description	Type	Reset Value
15−0	LCD_BOT_F2_ L[31−16]	LCD bottom address for frame buffer 2 upper bits [31−16]	R/W	Undefined

Index