ARM1176JZF Development Chip

Revision: r0p0

Technical Reference Manual

ARM DDI 0375A

ARM1176JZF Development Chip
Technical Reference Manual

Release Information

Change history

Date	Issue	Confidentiality	Change
31 August 2007	A	Non-Confidential	First release of r0p0.

Proprietary Notice

Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by ARM and the party that ARM delivered this document to.

Product Status

The information in this document is final, that is for a developed product.

Web Address

`http://www.arm.com`

Contents
ARM1176JZF Development Chip Technical Reference Manual

List of Tables
ARM1176JZF Development Chip Technical Reference Manual

List of Figures
ARM1176JZF Development Chip Technical Reference Manual

Preface

This preface introduces the *ARM 1176JZF Development Chip Revision r0p0 Technical Reference Manual*. It contains the following sections:

- *About this manual* on page xiv
- *Feedback* on page xix.

About this manual

This is the *Technical Reference Manual* (TRM) for the *ARM 1176JZF Development Chip* subsystem. All references to the chip in this manual refer to the ARM 1176JZF Development Chip.

Product revision status

The r*n*p*n* identifier indicates the revision status of the product described in this manual, where:

r*n* Identifies the major revision of the product.

p*n* Identifies the minor revision or modification status of the product.

Intended audience

This manual is written for system designers, system integrators, and verification engineers who are implementing a *System-on-Chip* (SoC) device based on the ARM1176JZF development chip subsystem. The manual describes the VC.

Using this manual

This manual is organized into the following chapters:

Note

Many of the components used in the ARM1176JZF development chip subsystem have a separate TRM. See the component chapter in this manual before using any of the relevant TRMs. The component information in this manual is specifically provided for use with the subsystem.

Chapter 1 *Introduction*

Read this chapter for an introduction to the chip. It contains a brief description of each of the main components that the chip contains.

Chapter 2 *Functional Overview*

Read this chapter for a description of the top-level functionality of the chip. It contains a block diagram of the development chip and describes the function of each block in detail.

Chapter 3 *Programmer's Model*

Read this chapter for a description of how to program the chip registers.

Appendix A ***Electrical and Physical Characteristics***

Read this chapter for a description of the electrical and physical characteristics of the development chip.

Appendix B ***Component Revision Status***

Read this chapter for a list of the revision status of each of the components included in the chip.

Glossary Read the Glossary for definitions of terms used in this manual.

Conventions

Conventions that this manual can use are described in:

- *Typographical*
- *Timing diagrams* on page xvi
- *Signals* on page xvi
- *Numbering* on page xvii.

Typographical

The typographical conventions are:

italic Highlights important notes, introduces special terminology, denotes internal cross-references, and citations.

bold Highlights interface elements, such as menu names. Denotes signal names. Also used for terms in descriptive lists, where appropriate.

`monospace` Denotes text that you can enter at the keyboard, such as commands, file and program names, and source code.

`mono`space Denotes a permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.

`monospace italic` Denotes arguments to monospace text where the argument is to be replaced by a specific value.

`monospace bold` Denotes language keywords when used outside example code.

< and > Angle brackets enclose replaceable terms for assembler syntax where they appear in code or code fragments. They appear in normal font in running text. For example:

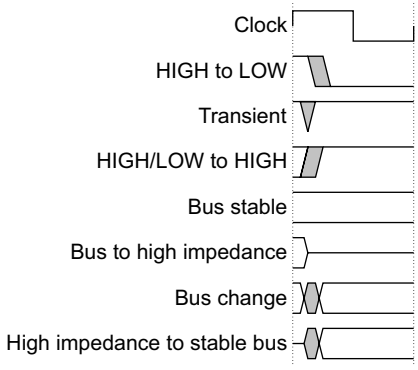
- `MRC p15, 0 <Rd>, <CRn>, <CRm>, <Opcode_2>`

- The Opcode_2 value selects the register that is accessed.

Timing diagrams

The figure named *Key to timing diagram conventions* explains the components used in timing diagrams. Variations, when they occur, have clear labels. You must not assume any timing information that is not explicit in the diagrams.

Shaded bus and signal areas are undefined, so the bus or signal can assume any value within the shaded area at that time. The actual level is unimportant and does not affect normal operation.



Key to timing diagram conventions

Signals

The signal conventions are:

Signal level The level of an asserted signal depends on whether the signal is active-HIGH or active-LOW. Asserted means HIGH for active-HIGH signals and LOW for active-LOW signals.

Prefix A Denotes *Advanced eXtensible Interface* (AXI) global and address channel signals.

Prefix B Denotes AXI write response channel signals.

Prefix C Denotes AXI low-power interface signals.

Prefix H Denotes *Advanced High-performance Bus* (AHB) signals.

Prefix n Denotes active-LOW signals except in the case of AXI, AHB, or *Advanced Peripheral Bus* (APB) reset signals.

Prefix P	Denotes APB signals.
Prefix R	Denotes AXI read channel signals.
Prefix W	Denotes AXI write channel signals.
Suffix n	Denotes AXI, AHB, and APB reset signals.

Numbering

The numbering convention is:

<size in bits>'<base><number>

This is a Verilog method of abbreviating constant numbers. For example:

- `'h7B4` is an unsized hexadecimal value.
- `'o7654` is an unsized octal value.
- `8'd9` is an eight-bit wide decimal value of 9.
- `8'h3F` is an eight-bit wide hexadecimal value of `0x3F`. This is equivalent to `b00111111`.
- `8'b1111` is an eight-bit wide binary value of `b00001111`.

Further reading

This section lists publications by ARM Limited, and by third parties.

ARM Limited periodically provides updates and corrections to its documentation. See `http://www.arm.com` for current errata sheets, addenda, and the Frequently Asked Questions list.

ARM publications

This manual contains information that is specific to the ARM1176JZF development chip. See the *Intelligent Energy Controller Technical Reference Manual* (ARM DDI 0304) for other relevant information.

Note

See the component chapter in this manual before using any of the relevant TRMs. The component information in this manual is specifically provided for use with the ARM1176JZF development chip.

- *ARM1176JZF Development Chip Electrical Specification* (PR168-PRDC-004363)

- *Flow Guidelines for Generating an IEM Enabled ARM1176 Macrocell* (PR114-PRDC-005381)
- *Generic Interrupt Controller Architecture* (PRD03-GENC-004392).

Other publications

This section lists relevant documents published by third parties:

- *Adaptive Power Controller – APC1 – (SY751),* National Semiconductor (SY751-DA-03001).

Feedback

ARM Limited welcomes feedback on the ARM1176JZF development chip subsystem and its documentation.

Feedback on this product

If you have any comments or suggestions about this product, contact your supplier giving:

- the product name
- a concise explanation of your comments.

Feedback on this manual

If you have any comments on this manual, send email to `errata@arm.com` giving:

- the title
- the number
- the relevant page number(s) to which your comments apply
- a concise explanation of your comments.

ARM Limited also welcomes general suggestions for additions and improvements.

Chapter 1
Introduction

This chapter describes the general properties and functionality of the chip and contains the following sections:

1.1 About the ARM1176JZF Development Chip

The chip is a TSMC, 130nm generic implementation of the following:

- ARM1176JZF core
- *Level 2 Cache Controller* (L2CC)
- CoreSight ETM11
- peripherals supporting TrustZone, CoreSight, and *Intelligent Energy Management* (IEM).

The ARM1176JZF, L2CC and CoreSight ETM11 operate with dynamic voltage and frequency scaling under control of the ARM IEM system.

The IEM control system comprises the following:

- ARM Intelligent Energy Controller (IEC)
- National Semiconductor's *Advanced Power Controller* (APC1)
- a custom, SoC-specific, *Dynamic Clock Generator* (DCG).

Additionally, dynamic memory controller, PL340, is implemented with a 32-bit 166MHz Mobile DDR interface.

This document defines the functional specification and programmers view of the Development Chip. The ARM1176JZF Development Chip Requirements Capture [1] summarizes the features.

Full specifications are provided for modules created specifically for the design, such as the DCG and System controller. Modifications of incoming IP are also documented.

Implementation guidelines describing the IEM enabling of the application processor sub-system is presented in a separate document [4].

1.2 Component summary

The chip consists of the following components:

- ARM1176JZF Development Chip with 32KB, I and D, caches and 8KB, I and D, TCMs
  - core, RAM and SoC voltage domains.
- level 2 cache and controller with 128KB unified cache
  - clocked 1:1 with the core, I & R/D buses
  - core, RAM and SoC voltage domains.

1.2.1 CoreSight components

The chip consists of the following CoreSight components:

- CoreSight ETM11
  - clocked 1:1 with the core
  - core and SoC voltage domains.
- CoreSight ETB11, 8KB memory
  - TPIU for off-chip trace port, 32 bits implemented
  - replicators to TPIU, ETB11 and off-chip ATB port.
- cross-trigger interface and ATB bus brought off-chip for multiple core debug support
- *Debug Access Port* (DAP)
  - access ports: APB, CoreSight, JTAG, ARM1176JZF Development Chip, AHB, system access
  - debug ports: JTAG.

1.2.2 System components

The chip consists of the following system components:

- Color LCD controller
- *Dynamic Memory Controller* (DMC), 32-bit mobile DDR at 166MHz. Memory interface operates synchronously to AXI clock for lowest latency
- *Static Memory Controller* (SMC), 32-bit
- UARTs, SSP, and GPIO
- RTC, timers & watchdog
- System controller
- *Generic Interrupt Controller* (GIC) and *TrustZone Interrupt Controller* (TZIC)

- *Configurable AXI Interconnect* (CAI).

1.2.3 TrustZone components

The chip consists of the following TrustZone components:

- TrustZone System Protection Controller, APB3 peripheral
- 16KB non-volatile RAM for boot-ROM emulation
- 512KB L3 RAM, configurable secure region watermark
- CAI modified to incorporate:
  - DMC and SMC have per chip-select, or 64MB region, security blocking
  - AXI master port has configurable security and four TrustZone decodes exported
  - AXI slave port has configurable security blocking of **AxPROT[1]**
  - CLCDC has configurable security enabling of **AxPROT[1]**.

1.2.4 Intelligent Energy Manager (IEM) components

The chip consists of the following IEM components:

- *Intelligent Energy Controller* (IEC), APB3 peripheral
- dynamic clock generation and reset controller, APB3 peripheral
- Dynamic Voltage Controller (DVC), APB3 peripheral, National Semiconductor *Advanced Power Controller1* (APC1)
- *Hardware Performance Monitor* (HPM), National Semiconductor.

1.3 Technology summary

The technology that the chip uses has the following characteristics:

- TSMC 130nm generic process, nine layer metal, the 9th layer is reserved for flipchip bumps:
  - 10⌋14mm silicon die
  - IEM subsystem target: 332MHz
  - DDR, AXI, and ATB Target: 166MHz
  - APB Target: 83MHz
  - 1.2V core, 3.3V input and output
  - 1.8V LVCMOS for mobile DDR input and output.
- packaging:
  - 1400 ball fully populated, flipchip
  - package ball pitch: 1mm
  - package bump pitch: 200µm.
- IEM subsystem libraries:
  - ARM Q3 cell library
  - ARM r2 RAMs, for the ARM1176JZF core
  - Artisan HS HC RAMs, for L2CC, characterized by Artisan for IEM operation.
- SoC libraries:
  - Artisan Sage-HS cell library
  - Artisan HS HC RAMs
  - Artisan supplied input and output libraries: 3.3V and 1.8V.
- third-party IP:
  - Artisan 533MHz de-skew PLL, three instances, one de-skew
  - Artisan 800Mbs DLL, one master & four slaves
  - National *Advanced Power Controller* (APC1) and *Hardware Performance Monitor* (HPM) soft IP.

Chapter 2
Functional Overview

This chapter describes the functionality of the chip and contains the following sections:

2.1 About the ARM 1176JZF Development Chip

This section describes the chip. Figure 2-1 shows a top-level block diagram of the chip.

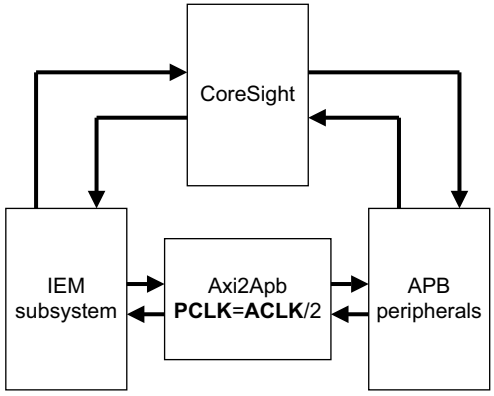


Figure 2-1 ARM1176JZF Development Chip block diagram

Figure 2-2 on page 2-3 shows the IEM subsystem in detail.

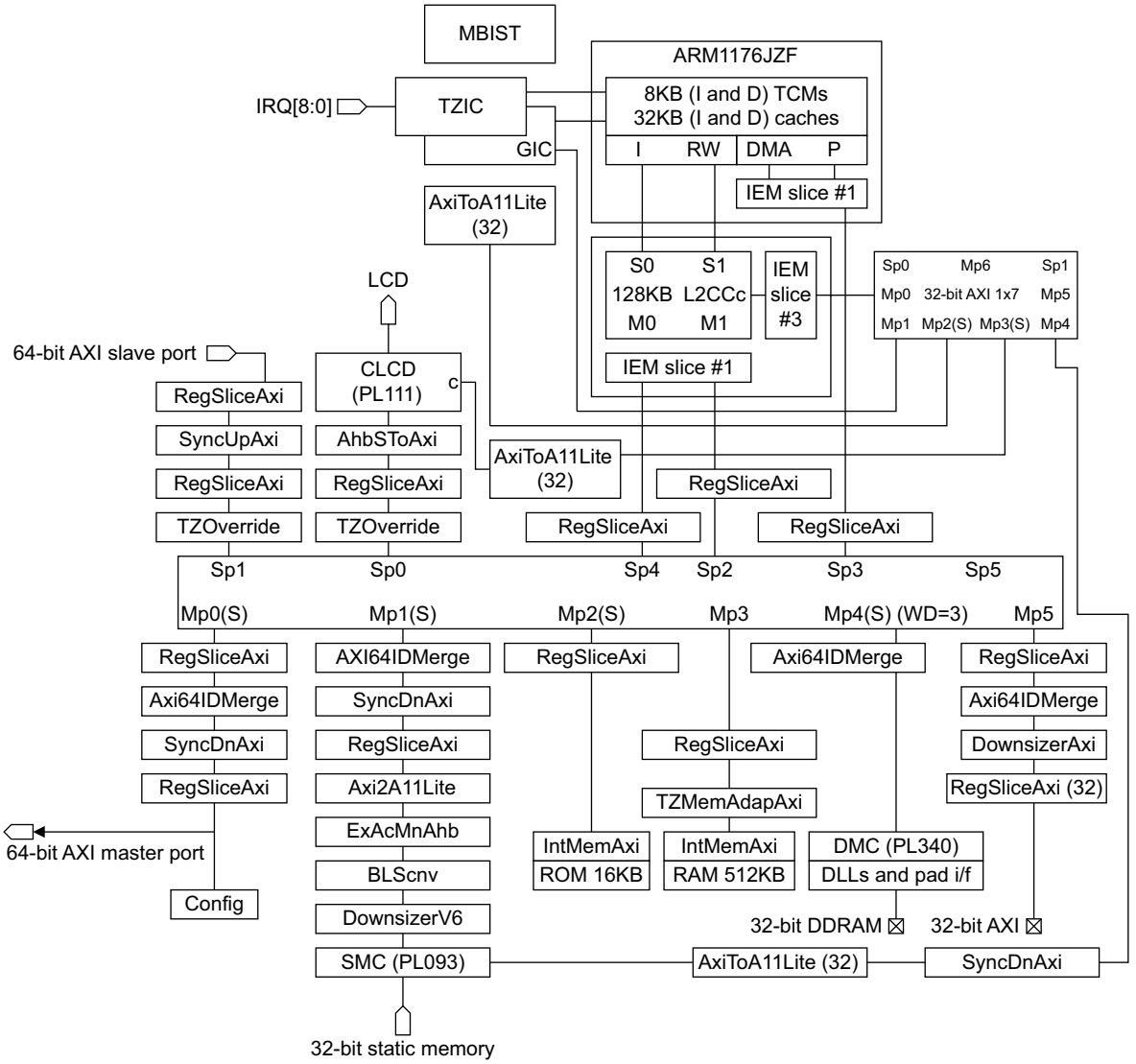


Figure 2-2 IEM subsystem

In Figure 2-2, you can configure the following components to be secure:

Sp0

- CLCD (PL111)
- `AhbSToAxi`
- `RegSliceAxi`
- `TZOverdrive`.

Sp1
- RegSliceAxi
- SyncUpAxi
- TZOverdrive.

Sp2 RegSliceAxi.

Sp3 RegSliceAxi.

Sp4 RegSliceAxi.

Mp0(S) RegSliceAxi.

Figure 2-3 shows the CoreSight subsystem in detail.

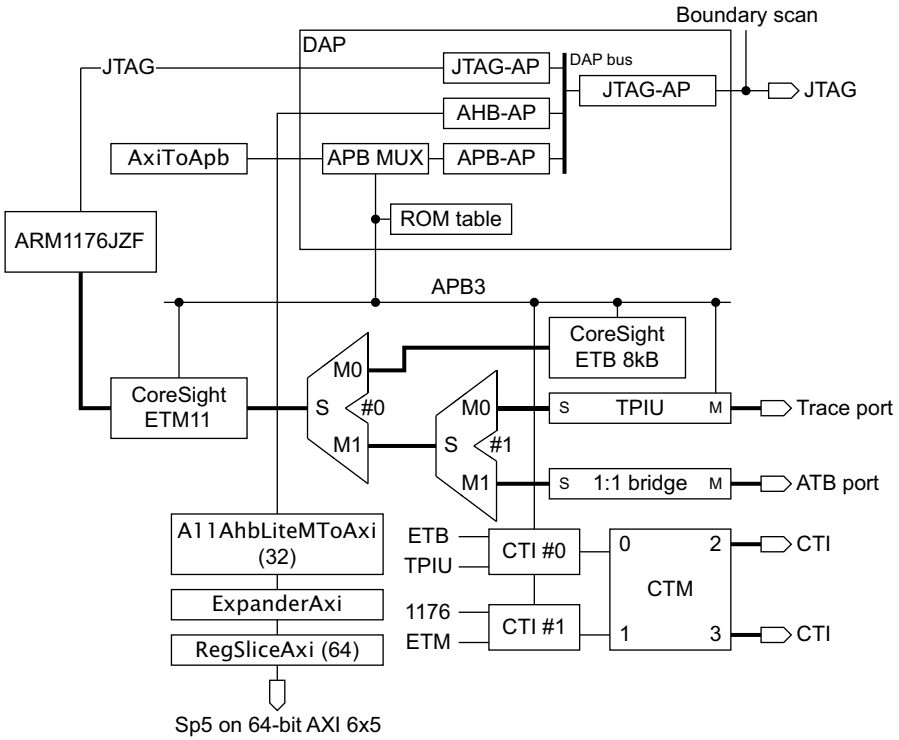


Figure 2-3 CoreSight subsystem

Figure 2-4 on page 2-5 shows the APB peripherals in detail.

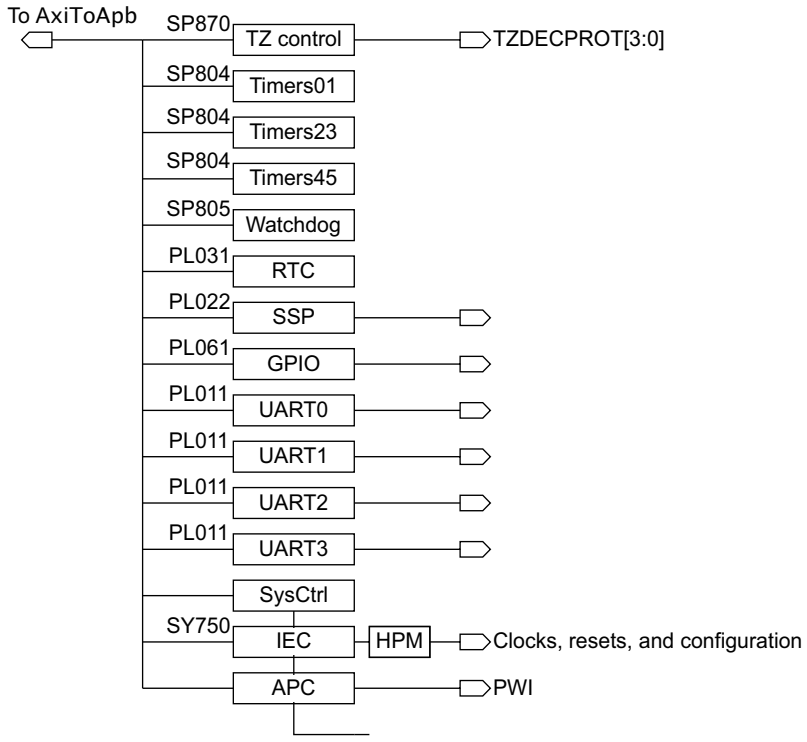


Figure 2-4 APB peripherals

This section contains the following subsections:

- *Processor subsystem* on page 2-6
- *Level 2 Cache Controller* on page 2-7
- *Intelligent Energy Management (IEM)* on page 2-8
- *Voltage domains* on page 2-8
- *AXI infrastructure* on page 2-9
- *Clocking* on page 2-15
- *Interrupt logic* on page 2-16
- *Interrupt sources* on page 2-18
- *On-chip RAM* on page 2-19
- *Exclusive access* on page 2-20
- *External static memory control* on page 2-20
- *External dynamic memory control* on page 2-21.

2.1.1 Processor subsystem

The processor sub-system consists of the following:

- ARM1176JZF core
- L2CC
- CoreSight ETM11.

The processor sub-system is subject to dynamic performance control of the core voltage domain. Level1 RAM, within the ARM1176JZF core, is a separate voltage domain that scales with the core voltage. It is isolated from it to enable data retention in dormant mode. Because the L2CC does not support data retention of Level2 RAM, the Level2 RAMs are supplied by the core voltage.

The IEM sub-system also contains:

- memory-BIST engines for the ARM1176JZF core and L2CC
- a *Hardware Performance Monitor* (HPM) to monitor the dynamic voltage
- ring oscillators to indicate voltage levels for diagnostic purposes.

The IEM sub-system is developed as a hard macro, with an approximate size of 6mm wide by 4mm high. Its primary purpose is to encapsulate all of the IEM enabling complexity, IP, and techniques within a macro.

All interfaces to and from the constant-voltage SoC domain are designed to operate asynchronously, and synchronously as appropriate. They incorporate level shifters and clamps. The whole IEM, application processor, sub-system supports:

- Run and standby modes, that is the IEM dynamic performance control of voltage and frequency.
- Dormant mode. The L1 RAMs are held at the retention voltage, the rest of the IEM sub-system is held at 0V, and the inputs and outputs are held clamped at the reset state.
- Shutdown mode. The whole IEM sub-system is held at 0V, and the inputs and outputs are held clamped at the reset state.

The ARM1176JZF DMA and peripheral port AXI buses incorporate specific synchronizing structures called IEM slices. These include *First In First Out* (FIFOs) to isolate the high-latency of the asynchronous AXI interface from the core. Similarly, the master ports and configuration port of the L2CC are isolated from AXI using the same FIFO-equipped slices.

The L2CC slave port 0, Sp0, is connected to the ARM1176JZF instruction port, and the L2CC slave port 1, Sp1, is connected to the ARM1176JZF data read and write port. These interfaces operate synchronously at a 1:1 clocking ratio.

The ARM1176JZF memories are 32KB I and D caches, and 8KB I and D *Tightly Coupled Memories* (TCMs).

Level2 cache is 128KB unified cache and is 8-way associative, so each way is 16KB. Tag parity and data parity RAMs are not implemented.

2.1.2 Level 2 Cache Controller

The L2CC is configured to use two master ports that interface to Level3 memory. This configuration enables you to position instruction fetches, that is high bandwidth demand on the DMC, at a lower arbitration priority than data and other system bus masters. Table 2-1 lists the L2C master port transactions to AXI.

Table 2-1 L2C master port transactions

Master port 0	Master port 1
Non-cached reads and non-buffered stores from Sp0	Non-cached reads and non-buffered stores from Sp1
Linefills with LF buffer 0	Linefills with LF buffer 1
Buffered stores with write buffer	Buffered stores with write buffer or eviction buffer
Write allocations with LF buffer 0	Write allocations with LF buffer 1

At reset, the L2CC is disabled, bypassed. To enable the L2C, you must write to the L2C control register, Register 1, using a read-modify-write sequence.

Note

- Only a secure access can enable or disable the L2CC.
- Security control is performed on a register-by-register basis within the L2C AXI configuration port.

When the L2CC is not enabled, transactions on Sp0 and Sp1 pass straight through to the L3 memory system via the Mp0 and Mp1 master ports, respectively. There is a penalty introduced by having the L2C disabled and in the path. This penalty is twice the depth of the internal L2C pipeline. it is made up of once for slave to master, and the other for master to slave. The minimum L2C latency is one **CLK** cycle on the slave port, one **ACLK** cycle on the master port, and potentially one **ACLK** cycle for bus.

2.1.3 Intelligent Energy Management (IEM)

Intelligent Energy Management (IEM) control systems allow you to vary the frequency and core voltage in the IEM, application processor, sub-system domain. The system is based around the ARM *Intelligent Energy Controller* (IEC). The *Advanced Power Controller* (APC1) from National Semiconductor provides voltage control, and a SoC-specific *Dynamic Clock Generator* (DCG) provides frequency control.

DCG functionality includes *Design For Test* (DFT), clock, reset, power and clamp controls, including power and, L1 memory clamp sequencing to support the following IEM and core modes.

IEC operation modes

The following are the IEC operational modes:

- *Dynamic Voltage Scaling* (DVS)
- DVS emulation.

Application processor sub-system power modes

The following are the application processor sub-system power modes:

- run mode
- standby mode
- shutdown mode
- dormant mode.

Note

The IEM implementation defaults to use one frequency per voltage performance level. Even though the IEC configuration is normally hard-wired, the configuration is soft in the development chip. Its default values at reset are stored in registers in the system controller. This enables additional exploration of multiple frequencies per voltage. Three PLLs perform dynamic clock generation and system clocks due to the flexibility the development chip requires.

2.1.4 Voltage domains

The chip contains the following voltage domains:

- *IEM subsystem voltage domains* on page 2-9
- *SoC voltage domains* on page 2-9.

IEM subsystem voltage domains

The IEM subsystem contains the following voltage domains:

V_{core} Core voltage for the ARM1176JZF core, CSETM11 and L2CC, plus MBIST logic. This voltage domain includes the L2C RAMs and is varied dynamically by the IEM control system between 0V and 1.2V.

V_{ram} RAM voltage for the RAMs within the ARM1176JZF core. This voltage is held at V_{core} except when operating in dormant mode when the RAMs are held in retention voltage, that is approximately 0.6V.

SoC voltage domains

The SoC contains the following voltage domains:

V_{soc} 1.2V constant SoC voltage.

V_{pllfa} Fixed frequency PLL digital, 1.2V, and analog, 3.3V, rails. The analog rail is brought out of the SoC and isolated from all other rails.

V_{pll1} Dynamic frequency PLL digital, 1.2V, and analog, 3.3V, rails. The analog rail is brought out of the SoC and isolated from all other rails.

V_{pll2} Dynamic frequency PLL digital, 1.2V, and analog, 3.3V rails. The analog rail is brought out of the SoC and isolated from all other rails.

V_{dll} DLL digital, 1.2V, and analog, 3.3V rails. The analog rail is brought out of the SoC and isolated from all other rails.

V_{io} 3.3V input and output supply rail for all inputs and outputs other than MDDR.

V_{ddr} 1.8V input and output supply rail for MDDR.

$V_{ddr\text{-}pre}$ 2.5V supply rail for the MDDR pre-driver.

2.1.5 AXI infrastructure

The infrastructure is created using the *Configurable AXI Interconnect* (CAI) PL300 tool. Separate 64-bit and 32-bit interconnect structures exist to partition the main L3 memory and peripherals. A link exists between the 64-bit and 32-bit structures to primarily enable an external master, or the ARM1176JZF RW channel, to access peripheral space.

Conversion to and from AHB is performed where necessary to support legacy AHB components. All bus matrices feature full connectivity. All masters can access all slaves, subject to memory map, remap and run-time security configuration.

The 64-bit infrastructure has a 6_6, fully wired structure, accommodating all system masters. All six slave ports have an interleaved write depth of one. The six master ports have interleaved write depths of one, except for Mp4. Mp4 is connected to the DMC and has an interleaved write depth of 3.

The 32-bit infrastructure has a 2×7 structure. Its purpose is to connect all the system peripherals and configuration ports to the ARM1176JZF 32-bit peripheral port and to the 64-bit AXI interconnect. This facilitates external master access to peripheral space. All slave and master ports have an interleaved write depth of one.

Accesses to undefined or reserved AXI memory regions receive a DECERR response that results in a prefetch or data abort, defined by the access type.

This design does not use the AXI low power interface, c-channel, except as a work around for implementing Standby Wait For Interrupt on the L2CC in the IEM sub-system. See *Intelligent Energy Management (IEM)* on page 2-38.

64-bit AXI arbitration

Each AXI master port has an arbiter that mediates between the slave ports in the interconnect. The arbitration is identical for all master ports. The arbitration priorities listed in Table 2-2 lists are the development chip defaults.

Table 2-2 64-bit AXI infrastructure arbitration priority

Slave port priority	Master	Alternative master
0 = highest	CLCDC	-
1	AXI off-chip master	-
2	ARM1176JZF data	Level 2 cache master port 1
3	ARM1176JZF DMA	-
4	ARM1176JZF instruction	Level 2 cache master port 0
5 = lowest	CoreSight debug	-

Note

The arbitration this section describes is the system default and is effective at reset. Registers exist within the system controller to enable you to configure the arbitration priority. See *SoC configuration* on page 3-7.

Although the configurable arbitration priority affects only the address channel, the master ports prioritize data based on the order the addresses are received.

Note

The IDs are not altered when the arbitration is modified. IDs are based on the default arbitration priority this section describes.

AXI IDs

Master ID in the system is formed by the configurable interconnect as follows:

```
ID = {max id_bit_width of any master, binary number of master}
```

The CAI appends the master ID number, a function of the default arbitration priority, to the ID from a master. Table 2-3 lists the master IDs visible in the 64-bit domain.

Table 2-3 64-bit AXI make-up

Master	Slave port priority	ID width	AXI ID, slave side
CLCDC	0 = highest	0	00000-000
AXI off-chip master	1	5	00000-001….11111-001
ARM1176JZF data	2	1	00000-010 (S0), 00001-010 (S1)
ARM1176JZF DMA	3	0	00000-011
ARM1176JZF instruction	4	0	00000-100
CoreSight debug	5 = lowest	0	00000-101

There are no ID bits between the ARM1176JZF, **I** and **RW** ports, and the L2CC, **S0** and **S1** ports. Bit [0] of the L2CC M1 port ID bus is wired to the CAI. There are five bits of ID from the AXI slave port to enable the MPCore or Cortex-A8 test chips to be attached to this SoC. The ID width becomes nine bits if the 32-bit AXI matrix is cascaded with the 64-bit.

The L2CC configuration port only accepts 8-bit ID fields. Reducing the ID width and bit mapping, provides two benefits:

- enhanced use of the PL340 *Quality of Service* (QoS), continuous 4-bit ID window
- possible to reduce the pin count of the off-chip AXI master port.

Table 2-4 lists the ID merge functional values.

Table 2-4 ID merge function

Master	AXI ID Value	Merged ID Value
CLCDC	00000-000	000000
ARM1176JZF Data S0	00000-010	010111
ARM1176JZF Data S1	00001-010	010010
ARM1176JZF DMA	00000-011	100011
ARM1176JZF instruction	00000-100	000100
CoreSight debug	00000-101	110101
AXI off-chip master, ExtSlv port	00000-001….11111-001 (= ABCDE-001)	AB1CDE[a]

a. ABCDE = **xID_ExtSlv[5:0]**, giving merged ID values of 8-15, 24-31, 40-47, and 56-63.

The DMC (PL340) has a QoS feature that can window on four contiguous ID bits and enables QoS settings for 16 masters. Table 2-5 lists the DMC QoS options, values used by moving the DMC QoS window over the merged ID field.

Table 2-5 DMC QoS options

Master	QoS using [3:0]	Value	QoS using [4:1]	Value	QoS using [5:2]	Value
CLCDC	0000	0	0000	0	0000	0
ARM1176JZF data S0	0111	7	1011	11	0101	5
ARM1176JZF data S1	0010	2	1001	9	0100	4
ARM1176JZF DMA	0011	3	0001	1	1000	8

Table 2-5 DMC QoS options (continued)

Master	QoS using [3:0]	Value	QoS using [4:1]	Value	QoS using [5:2]	Value
ARM1176JZF instruction	0100	4	0010	2	0001	1
CoreSight debug	0101	5	1010	10	1101	13
AXI off-chip master, ExtSlv Port	1CDE	8-15[a]	B1CD	4-7, 12-15[b]	AB1C	2-3, 6-7, 10-11, 14-15[c]

a. Aliasing over bits xID_ExtSlv[5:4].
b. Aliasing over bits xID_ExtSlv[5] and [0].
c. Aliasing over bits xID_ExtSlv[1:0].

Table 2-6 lists the 7-bit master IDs visible in the 32-bit domain.

Table 2-6 32-bit AXI ID make-up

Master	Slave port priority	AXI ID, slave side
ARM1176JZF peripheral	0 = highest	000000-0
AXI 64-bit infrastructure	1 = lowest	000000-1, CLCD 010111-1, ARM data, S0 010010-1, ARM data, S1 100011-1, ARM DMA 000100-1, ARM instruction 110101-1, CoreSight debug 001000-1….111111-1, AXI master

Security configuration

Each CAI master port is assigned either a secure or non-secure attribute through a set of inputs controlled by the *TrustZone Protection Controller* (TZPC). Figure 2-1 on page 2-2 shows shaded areas in the system block diagram. These indicate ports and peripherals you can configure to be secure.

Note

At reset, all CAI master ports and peripherals that you can configure to be secure, under control of the TZPC, are set as secure.

The AXI infrastructure rejects non-secure, NS, transactions to secure, S, ports or peripheral, and returns a DECERR response. Writes fail, but the write burst must complete before the AXI infrastructure returns a DECERR response. Reads also fail in a similar way, with a zero value returned and a DECERR response. The AXI infrastructure responds after the first read, when the master aborts the rest of the burst.

You can configure the off-chip AXI master port to be either fully secure or non-secure. Four security configuration bits, **DECPROT**, from the TZPC, are exported off-chip as sideband signals to the AXI Master port **TZPROTMAXI[3:0].** These are useful to AXI peripheral developers,. These enable you to construct off-chip, either a single secure peripheral or an additional CAI matrix, with up to four separate secure regions.

Security for the AXI Slave port is configurable under control of the TZPC as a gating function of secure transactions. Secure transactions can be gated from entering the AXI infrastructure.

Although the CLCDC has no security awareness, as part of the AHB to AXI conversion, you can configure transactions as secure under control of the TZPC. You can also configure the CLCDC configuration port to be secure to complete the TrustZone protection of the peripheral.

You can configure the DMC and SMC memory space to be either secure or non-secure in 64MB regions under control of the TZPC. Each of the eight SMC chip select regions is 64MB. They are potentially either a half or a full DMC chip select region. You can also configure the DMC and SMC configuration ports to be secure to complete the TrustZone protection of the peripherals.

AXI off-chip slave interface

A 64-bit AXI slave port is implemented that enables off-chip masters to access the on-chip peripherals, memory controllers and the AXI master port. It has an interleaved write depth of one.

The interface is synchronous. It supports all integer clocking ratios from 1:1 to 1:8 between the external clock and SoC AXI clock.

The AXI off-chip slave port operates synchronously with **PLLREFCLK**, the main PLL input clock. This synchronicity is achieved by de-skewing the AXI SoC clock to **PLLREFCLK**, the board-level clock.

Feedback for the de-skew PLL is taken from this off-chip interface.

Maximum flexibility for TrustZone support is provided because the AXI slave port **AxPROT[1]** signals are input to the SoC. You can override these signals to gate out secure transactions under control of the TZPC.

The *ARM1176JZF Development Chip Electrical Specification* describes the interface port signal list.

AXI off-chip master interface

A 64-bit AXI master port is implemented, that enables access to off-chip slaves from on-chip masters and the AXI slave port. It has an interleaved write depth of one. The interface is synchronous and supports all integer clocking ratios of 1:1 to 1:8 between the external clock and the SoC AXI clock.

Note

These ratios are independent of those set for the AXI slave port. The AXI off-chip Master port operates synchronously with **PLLREFCLK**, the main PLL input clock. This is achieved by de-skewing the AXI SoC clock to **PLLREFCLK**, the board-level clock.

A clock is exported at the frequency of operation of this port. You can configure the master port to be secure, under control of the TZPC, enabling development of secure peripherals. Alternatively, you can configure the port to be non-secure and configure the master ports of an external extended sub-system independently to be secure or non-secure. To facilitate this, four **DECPROT** signals are output from the TZPC as side-band signals to the AXI master port.

2.1.6 Clocking

The ARM1176JZF core, L2CC, and ETM11 operate at a variable frequency, under control of the IEM control system, up to a target frequency of 332MHz. The processor subsystem operates synchronously and asynchronously to the AXI infrastructure.

Synchronous operation, relating to an IEM performance target of 100%, is supported with core:AXI clock ratios of 2:1 and 4:1 through a static configuration at reset. This uses balanced clocks and a common enable.

The target clock frequencies for the system configuration are:

- IEM subsystem at 332MHz
- AXI clock at 166MHz
- **PCLK** at 83MHz.

Th clocking strategy includes components that operate at different clock frequencies:

Operating at 166MHz

- AXI infrastructure and bus masters, excluding the processor subsystem

- dynamic and static memory controllers, on-chip RAMs
- *AMBA Trace Bus* (ATB) for CoreSight.

Operating at 83MHz

- APB peripherals, including the DMC configuration port.

You can configure the SMC external memory clock to operate at **ACLK**, **ACLK**÷2 or **ACLK**÷3, as part of the SoC static configuration at reset. SyncFlash requirements are up to 54MHz. The DMC external clock operates at 166MHz, synchronously to **ACLK**.

2.1.7 Interrupt logic

In the development chip, secure interrupt sources are routed to the **nFIQ** input on the core. Non-secure sources are routed to the **nIRQ** input on the core. A TZIC is implemented to enable and control routing to the TZIC or non-secure-IC logic as appropriate for the application.

All interrupt sources, on-chip and off-chip, are routed to the TZIC. This enables a dynamic interrupt configuration to be defined as part of the secure boot process. It is the responsibility of the end user to ensure that a secure interrupt is generated only by a securely decoded peripheral.

Secure interrupt controller

The task of the TZIC is to create the secure interrupt **nFIQ** from the raw interrupt sources. It must also stop the sources chosen for the **nFIQ** from propagating to the non-secure interrupt controller. This creates a secure interrupt that cannot be controlled or influenced from the non-secure side of a TrustZone system. To the non-secure side, this is a non-maskable interrupt.

Although you should configure the TZIC as secure in a TrustZone aware system, you can configure it as non-secure.

Secure interrupt flow

The following flow shows how the TZIC uses interrupts to force the operating system to enter secure state:

1. A TZIC selected interrupt occurs and **nFIQ** is asserted.
2. The ARM core branches to the **FIQ** interrupt vector.
3. The ARM core switches into monitor mode and stacks the current program status.
4. The **FIQ** handler switches the operating system into secure state.
5. The interrupt service routine executes in secure state.
6. The requesting interrupt in the peripheral is cleared.

7. The system returns to non-secure state and re-enables interrupts.

Non-secure interrupt controller

The non-secure interrupt controller creates the non-secure interrupt **nIRQ** from the interrupt sources that the TZIC enables. The non-secure interrupt controller implemented in the development chip is the GIC. This resides in non-secure memory space and provides user mode interrupt control. Because of the core architecture, the **nIRQ** does not provide a path into the secure side of a TrustZone system.

You can completely bypass the TZIC. this enables the GIC to operate as a non-secure interrupt controller with control of **nIRQ** and **nFIQ**. For this configuration, **nFIQ** from the GIC is fed to the TZIC, enabling a bypass to the core.

Non-secure interrupt flow

The following is the flow for a non-secure interrupt:

1. A GIC selected interrupt occurs and **nIRQ** is asserted.
2. The ARM core branches to the FRQ interrupt vector.
3. The Interrupt Service Routine executes in non-secure state.
4. The requesting interrupt in the peripheral is cleared.
5. The system re-enables interrupts.

Figure 2-5 shows the structure for secure and non-secure interrupts.

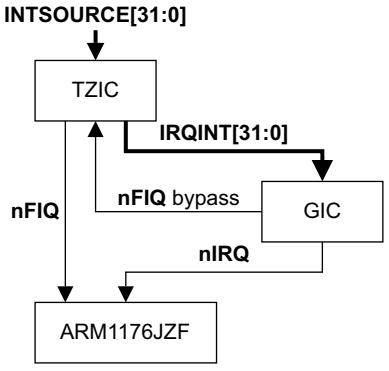


Figure 2-5 Secure and non-secure interrupt structure

2.1.8 Interrupt sources

Table 2-7 lists the interrupt sources implemented in the chip. **EXTINTSOURCE[8:0]** enable end-user developmen. They are primary inputs to the SoC. These inputs are synchronized to **ACLK** using a two-stage synchronizer. Similarly, the L2CC combined interrupt and the four interrupts from the ARM1176JZF Development Chip are also synchronized to **ACLK**.

The GIC sees these inputs as interrupt sources 32-63, but the TZIC sees them as 0-31 as Table 2-7 shows.

Table 2-7 Interrupt source allocation

Interrupt source	GIC source	TZIC source
EXTINTSOURCE[8]	63	31
EXTINTSOURCE[7]	62	30
EXTINTSOURCE[6]	61	29
EXTINTSOURCE[5]	60	28
EXTINTSOURCE[4]	59	27
EXTINTSOURCE[3]	58	26
EXTINTSOURCE[2]	57	25
EXTINTSOURCE[1]	56	24
EXTINTSOURCE[0]	55	23
CS_IRQ	54	22
UART_3	53	21
UART_2	52	20
UART_1	51	19
UART_0	50	18
SSP	49	17
GPIO	48	16
CLCDC	47	15
RTC	46	14
L2CC	45	13

Table 2-7 Interrupt source allocation (continued)

Interrupt source	GIC source	TZIC source
IEC	44	12
APC	43	11
Timers_2	42	10
Timers_1	41	9
Timers_0	40	8
Core PMU IRQ	39	7
Core Ext. Error DMA IRQ	38	6
Core secure DMA IRQ	37	5
Core DMA IRQ	36	4
Comms Tx	35	3
Comms Rx	34	2
Software interrupt	33	1
Watchdog	32	0

2.1.9 On-chip RAM

Two 64-bit wide on-chip RAMs are implemented. Both feature a write depth of one, no outstanding addresses, and a single wait state for reads.

One 16KB RAM is implemented for secure boot ROM emulation. You can configure it to be fully secure or non-secure under control of the TZPC. The RAM control inputs are qualified with SoC power-on-reset to prevent corruption during reset sequences.

At reset, the RAM defaults to secure. A second 512KB RAM is implemented for general-purpose application use. This has a secure wrapper for TrustZone support enabling the RAM to be partitioned between secure and non-secure regions under control of the TZPC. You can size the secure partition in 4KB increments, from the RAM base address, up to the maximum of 512KB. A non-secure access to the secure region returns a DECERR. At reset, the RAM defaults to being a single, fully secure region.

2.1.10 Exclusive access

Exclusive access monitoring is provided within the DMC and is added in front of PL093 using an AHB Exclusive Access Monitor.

The Dynamic Memory Controller, PL340, keeps track of up to four outstanding exclusive accesses. These are not programmed, but stored on a rolling basis.

The Static Memory Controller, PL093, has two exclusive accesses tracked. You can configure the master IDs at run time from registers within the system controller.

Note

There is no exclusive access monitoring provision for either of the two on-chip RAMs.

2.1.11 External static memory control

This memory is implemented to support eight chip selects of 64MB each with 32-bit data.

The SMC is clocked at 1:1 or 1:2 with **ACLK**, **ACLKSMC**, using a `SyncDnAxi` bridge to provide the optimal external clock frequency for synchronous devices of up to 54MHz.

The external synchronous memory clock, **SMMEMCLK** can run at 1:1, 1:2 or 1:3 with **ACLKSMC**. Both **ACLKSMC** and **SMMEMCLK** clock ratios are configured by static configuration. Figure 2-6 on page 2-21 shows the AXI to AHB logic, and the clocking conversion logic.

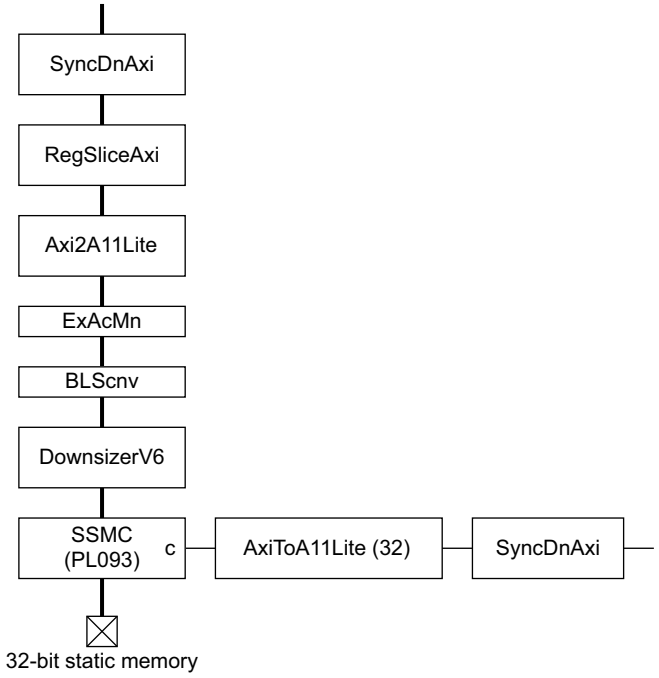


Figure 2-6 SMC AXI interface logic

2.1.12 External dynamic memory control

The *Dynamic Memory Controller* (DMC), PrimeCell PL340, is implemented with a 32-bit Mobile DDR interface. It operates at 1.8V LVCMOS thresholds and a target clock rate of 166MHz. The standard for 1.8V Mobile DDR is 1.8V LVCMOS, rather than SSTL_18.

The clock output provided on PL340 is replicated to create an inverted clock output to support Mobile DDR,. These two stages form a differential clock output that is closely skew controlled. The pad interface is clocked synchronously with **ACLK** to minimize latency and reduce the clocking complexity within the SoC.

The PL340 enables you to configure the base and range addresses of each chip select. Although the ARM1176JZF SoC implements four chip selects, fewer can be used to satisfy the maximum capacitive loading of any DDR input or output of 10pF.

Options

The following device configurations are supported:

Option 1

- four 512Mbit devices making 256MByte in total
- connected as two chip selects, each with two chips of 32M⌋16.

Option 2

- four 256Mbit devices making 128MByte total
- connected as two chip selects, each with two chips of 16M⌋16.

Option 3

- two 512Mbit devices making 128MByte total
- connected as one chip select, with two chips of 32M⌋16.

DQS pull-downs

The DMC uses four bi-directional data qualifier strobes, one per byte lane. These require pull-downs for functionality.

Note

You must provide these pull-downs on the PCB, because they are not implemented in the pads.

DLL control

To capture read data correctly, a master-slave DLL combination adjusts DQS strobe timing for each byte lane. There is one slave per byte lane strobe. The DLL applies a nominal cycle delay to each DQS input to the SoC.

Note

DQS going out of the SoC for writes is not manipulated.

The master DLL outputs a raw 7-bit delay code, updated every 32 **ACLK** cycles. It tracks voltage and temperature variations. Although this code is applied to the four slaves, each byte lane code is manipulated individually using saturating adders. These adders are user-configuarble.

Note

The saturating adder functionality ensures a value higher than 7'7F saturates at 7'7F, and a value less than 7'00 saturates at 7'00.

To create this offset from the master DLL code, the user_config Register in PL340 is extended from eight bits to 32 bits. Table 2-8 shows the user_config Register functionality. No offset is applied by default. Figure 2-7 shows the bit assignment for this register.

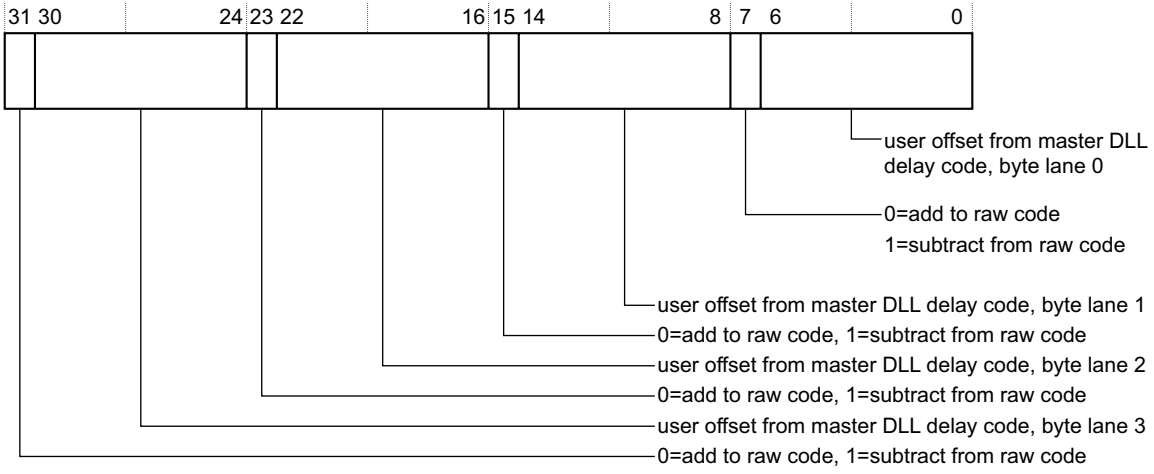


Figure 2-7 user_config Register bit assignment

Table 2-8 lists the user_config Register functionality in the PL340.

Table 2-8 user_config Register functionality in PL340

Bits	Reset	Function
[31]	0	0 = add to raw code 1 = subtract from raw code.
[30:24]	000_0000	User offset from master DLL delay code, byte lane 3
[23]	0	0 = add to raw code 1 = subtract from raw code.
[22:16]	000_0000	User offset from master DLL delay code, byte lane 2
[15]	0	0 = add to raw code 1 = subtract from raw code.

Table 2-8 user_config Register functionality in PL340 (continued)

Bits	Reset	Function
[14:8]	000_0000	User offset from master DLL delay code, byte lane 1
[7]	0	0 = add to raw code 1 = subtract from raw code.
[6:0]	000_0000	user offset from master DLL delay code, byte lane 0

You can read back the modified codes using the user_status Register in PL340. It has been extended from eight bits to 32 bits. Table 2-9 lists the reset and functions of the bits. Figure 2-8 shows the bit assignment for this register.

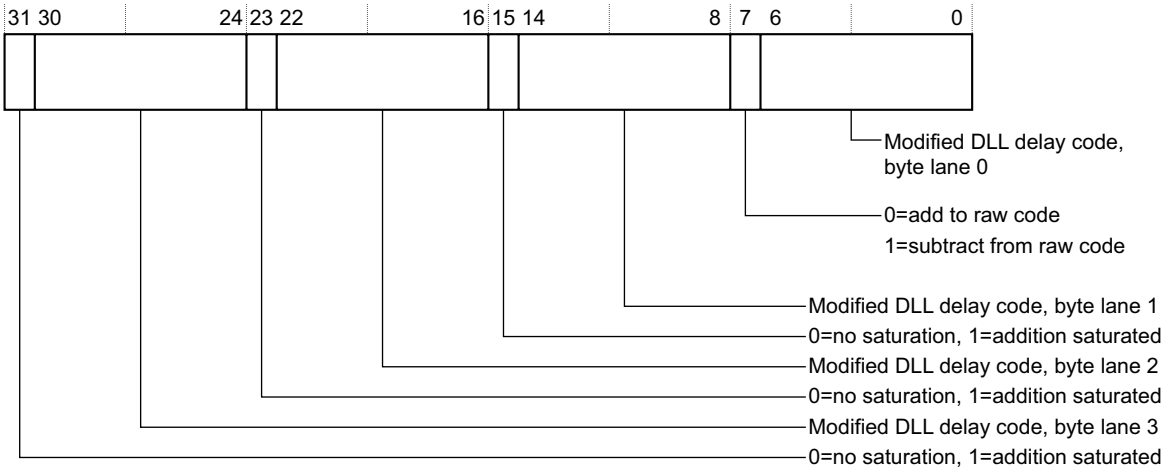


Figure 2-8 user_status Register bit assignment

Table 2-9 lists the bit assignment for this register.

Table 2-9 user_status Register functionality in PL340

Bits	Reset	Function
[31]	PVT defined	0 = no saturation 1 = addition saturated.
[30:24]		Modified DLL delay code, byte lane 3
[23]	PVT defined	0 = no saturation 1 = addition saturated.

Table 2-9 user_status Register functionality in PL340 (continued)

Bits	Reset	Function
[22:16]		Modified DLL delay code, byte lane 2
[15]	PVT defined	0 = no saturation 1 = addition saturated.
[14:8]		Modified DLL delay code, byte lane 1
[7]	PVT defined	0 = no saturation 1 = addition saturated.
[6:0]		Modified DLL delay code, byte lane 0

The slave DLLs are updated with the most recent delay code. This is ensured by the control logic implementation:

- A self-loading counter, loaded with an 8-bit value from the system controller periodically requests a DLL update. The counter runs on **mclk**, with a clock enable of **mclk**÷32, that the Master DLL provides.

  The count increment is 192ns at 166MHz.

- The default reset value for this counter is 0⌋34, 54d. This provides a delay of approximately 10µs between DLL updates.

  The range of update period is between 192ns and 48µs with a 166MHz clock.

- DLL updates cannot be disabled.

- During normal operation of the memory controller, a dummy EBI arbiter is implemented. This grants the DMC ownership of the bus by default. A DLL update request forces **ebibackoff** HIGH. This in turn forces the DMC to relinquish control of the bus at the end of the current transaction.

  Control logic responds to this bus release, and updates the slave DLLs with a low going pulse on **ASETB**. This is safe because there is no DQS activity either at this time or outstanding.

  The update process removes bus ownership from the DMC for worst case, three **mclk** cycles.

- The DLL master and slave cells are powered and clocked continuously in the chip to enhance implementation. Although you can gate the clock and remove power to these devices during prolonged periods of inactivity, this increases the DLL train complexity and delays update sequencing.

2.2 System controller

The system controller is an APB memory-mapped peripheral that incorporates system control and status registers. It enables:

- software to control the IEC configuration busses
- visibility of SoC-level configuration capture
- PLL configuration settings
- IEM mode control.

The system controller consists of:

- the APB subsystem
- the IEM performance level controller
- the PLL controller
- the remap controller
- the system reset controller.

Figure 2-9 on page 2-27 shows a top-level block overview of the system controller .

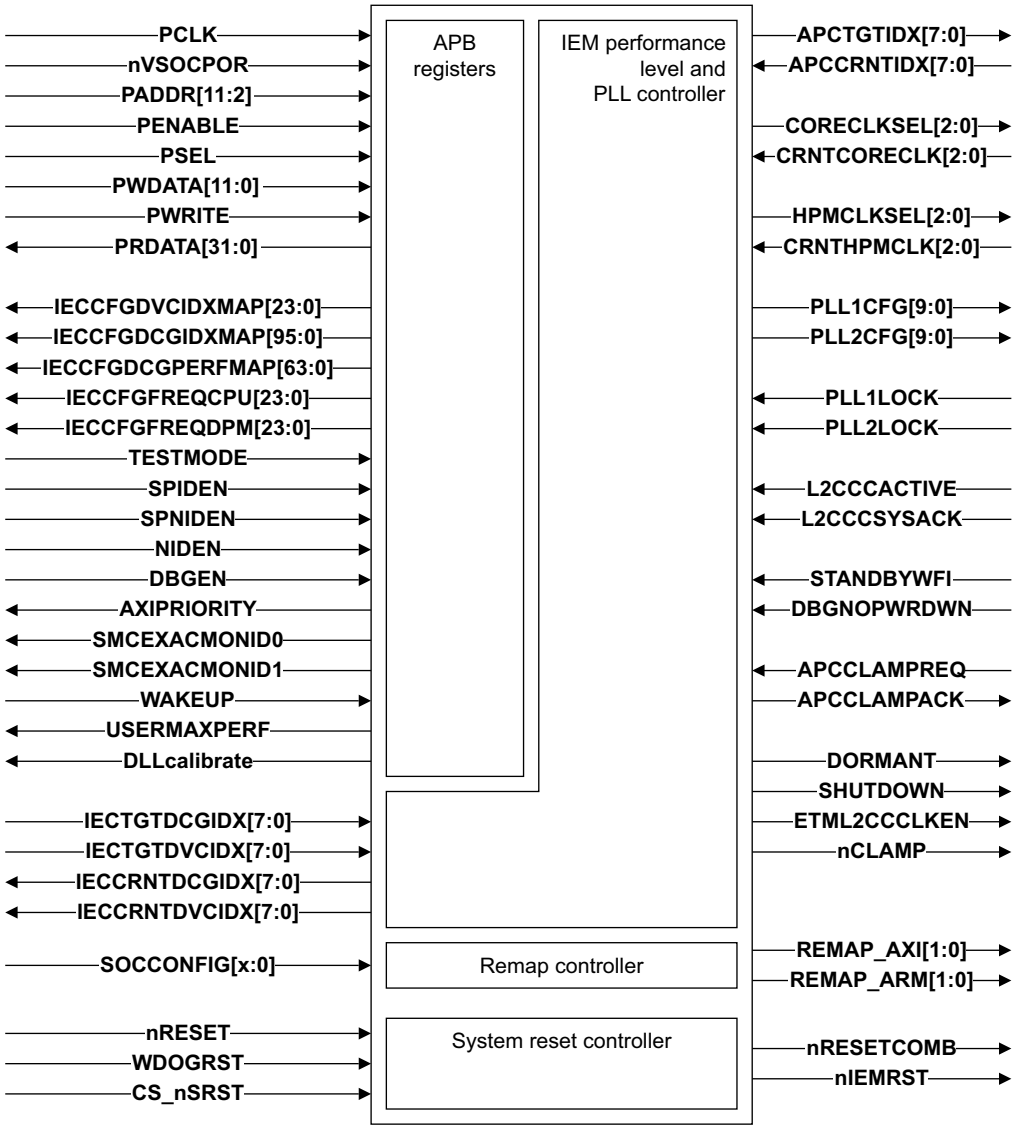


Figure 2-9 System controller top-level

This section contains the following subsections:

- *Intelligent Energy Management (IEM)* on page 2-8
- *Power modes* on page 2-29
- *Remap controller* on page 2-36
- *System reset controller* on page 2-37.

2.2.1 Intelligent Energy Management (IEM)

This section describes the IEM and contains the following subsection:

- *Performance level and PLL controller*.

Performance level and PLL controller

The IEM performance level and PLL controller is responsible for:

- receiving performance requests from the IEC
- sending and monitoring performance requests to the APC1 and DCG
- configuring the PLLs
- instructing the clock selection block concerning which PLL to select and when.

This block also controls the safe entry and exit from low power modes. This includes StandbyWFI, shutdown and dormant modes.

The IEM PLL controller consists of the performance level controller state machine and PLL configuration select blocks. Figure 2-10 shows an overview of the IEM performance level and PLL controller.

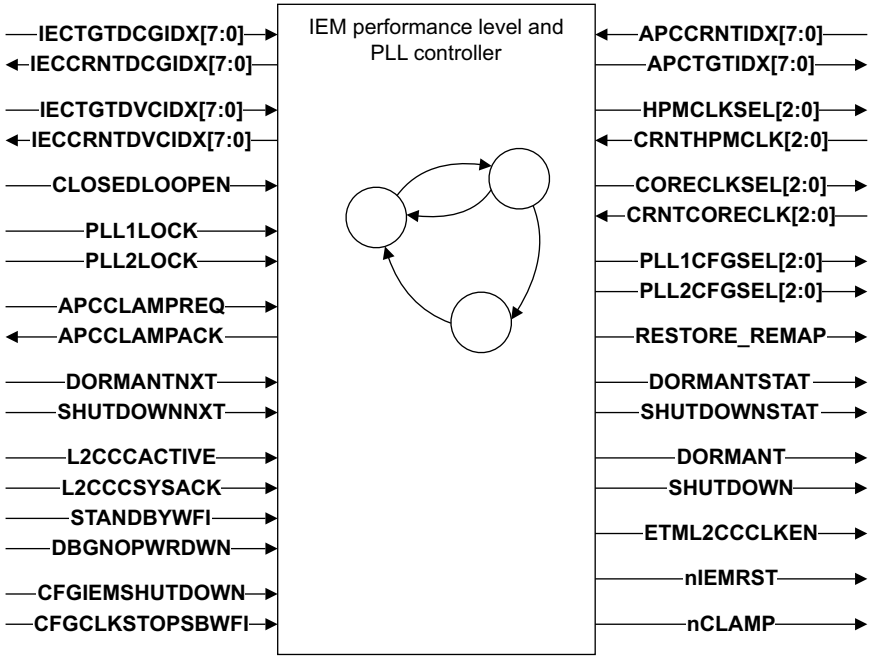


Figure 2-10 IEM performance level and PLL controller

2.2.2 Power modes

This section describes the power modes and contains the following subsections:

- *Run mode*
- *StandbyWFI mode* on page 2-34
- *Dormant and shutdown modes* on page 2-35
- *IEM subsystem permanent shutdown mode* on page 2-35
- *Wake-up sources* on page 2-35.

Run mode

This is the normal mode after reset. IEM software only operates in this mode.

Note

- After reset, the AXI slices operate asynchronously and the IEC does not generate an **IECSYNCMODEREQ** until configured.
- Additionally, after reset, the IEC defaults to requesting a 100% performance target. In Run mode, a performance level of 0% is never selected by IEM software because1% is the minimum demand the IEM software can set.

The Run Mode Performance Level Controller State Machine is shown in Figure 2-11 on page 2-30 through Figure 2-15 on page 2-34. It receives performance level requests from the IEC and ensures safe movement between performance levels. The state machine supports both closed and open loop modes.

The state machine ensures that when moving between performance levels, the required PLL is locked, and the IEM subsystem voltage level is adequate.

Note

In closed loop mode, the state machine also controls switching of the HPM clock.

New performance levels issued by the IEC can override current performance targets only when the state machine is in one of the 'PLL Wait For Lock' states.

An exit to 'Max Perf' state is implicitly applied for all of the following states:

- PLL wait for lock
- running on PLL
- dormant
- shutdown.

This state machine controls IEM through all active performance levels. If 0% performance is selected, the system either enters StandbyWFI, Shutdown or Dormant mode depending on system settings. For more information on these modes, see *Controlling the power modes* on page 3-20.

Note

For ease of illustration, the state machine is divided into six segments. States enclosed within rectangles represent those that cross between diagrams.

Figure 2-11 shows the process of switching to a lower performance on PLL1 from running on PLL2.

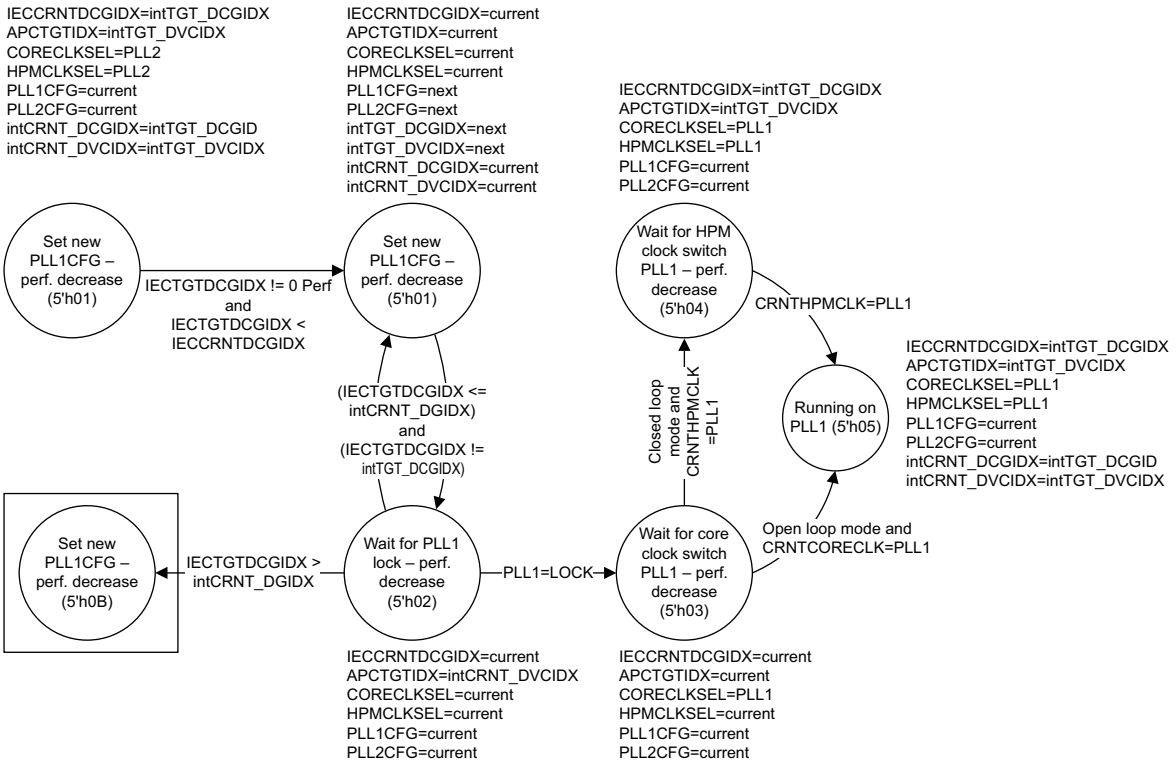


Figure 2-11 Performance decrease to running on PLL1 from PLL2

Note

When in open loop mode, **HPMCLKSEL** always drives 3'b000 to disable the HPM clock.

Figure 2-12 shows the process of switching to a lower performance on PLL2 from running on PLL1.

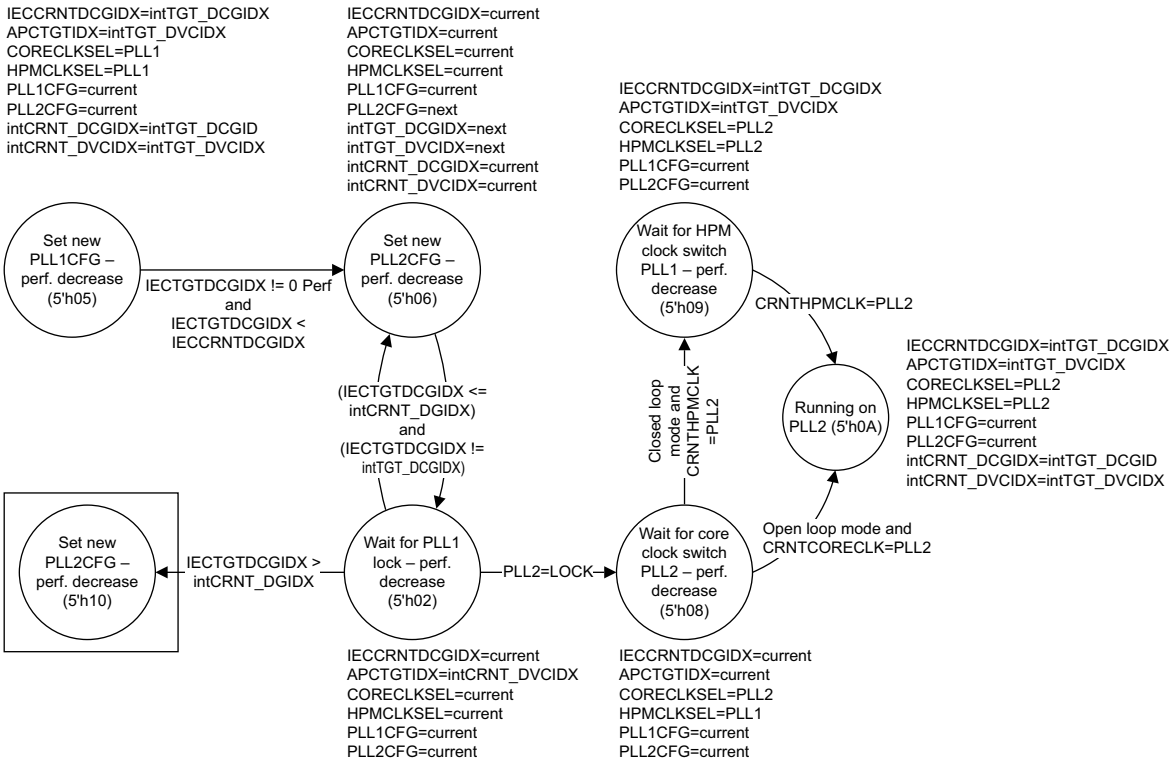


Figure 2-12 Performance decrease to running on PLL2 from PLL1

Figure 2-13 on page 2-32 shows the process of switching to a higher performance on PLL2 from running on PLL1.

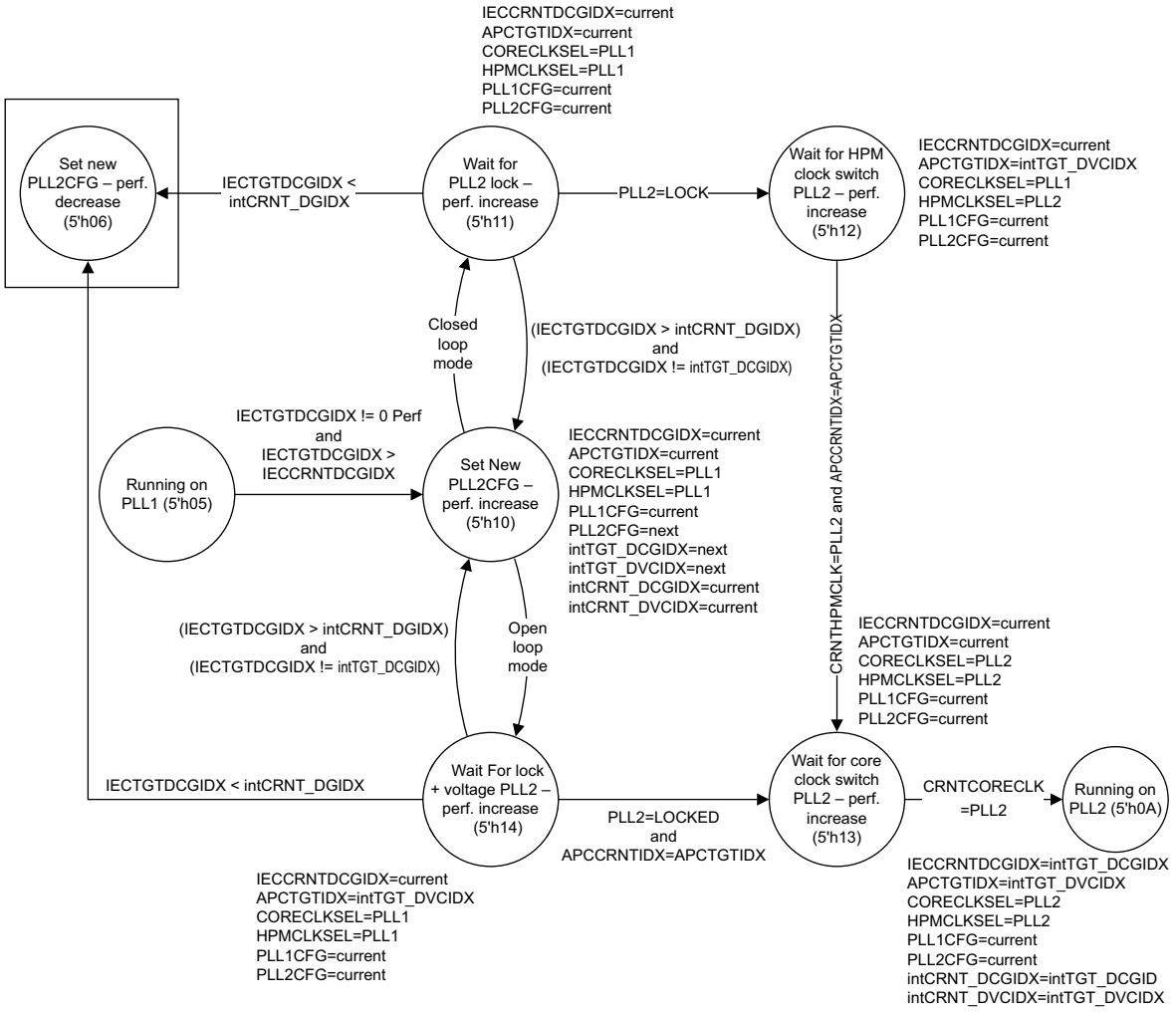


Figure 2-13 Performance increase to running on PLL2 from PLL1

Figure 2-14 on page 2-33 shows the process of switching to a higher performance on PLL1 from running on PLL2.

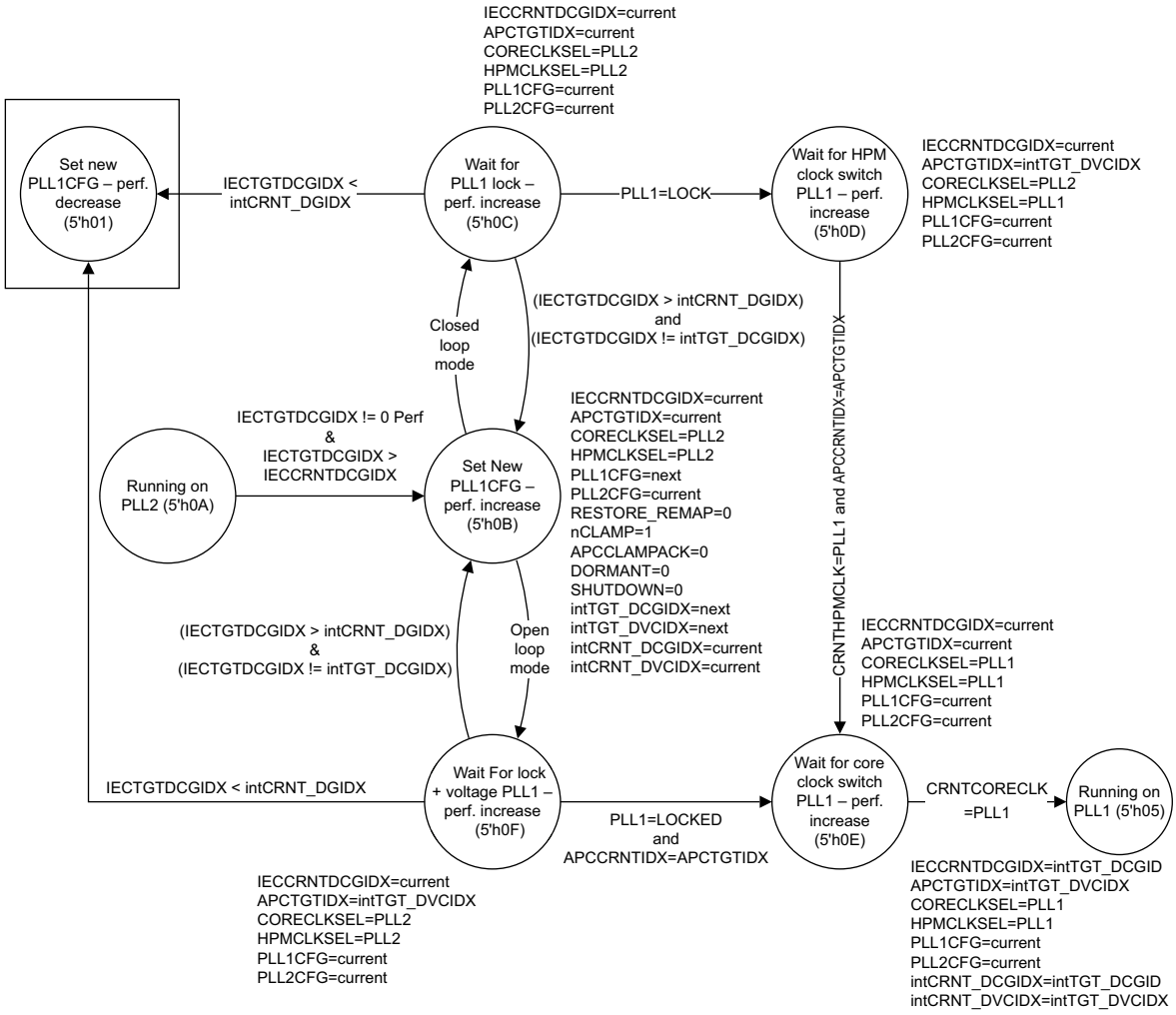


Figure 2-14 Performance increase to running on PLL1 from PLL2

Figure 2-14 shows a transition to maximum performance. When the IEC requests maximum performance, new performance targets are not considered until maximum performance is reached.

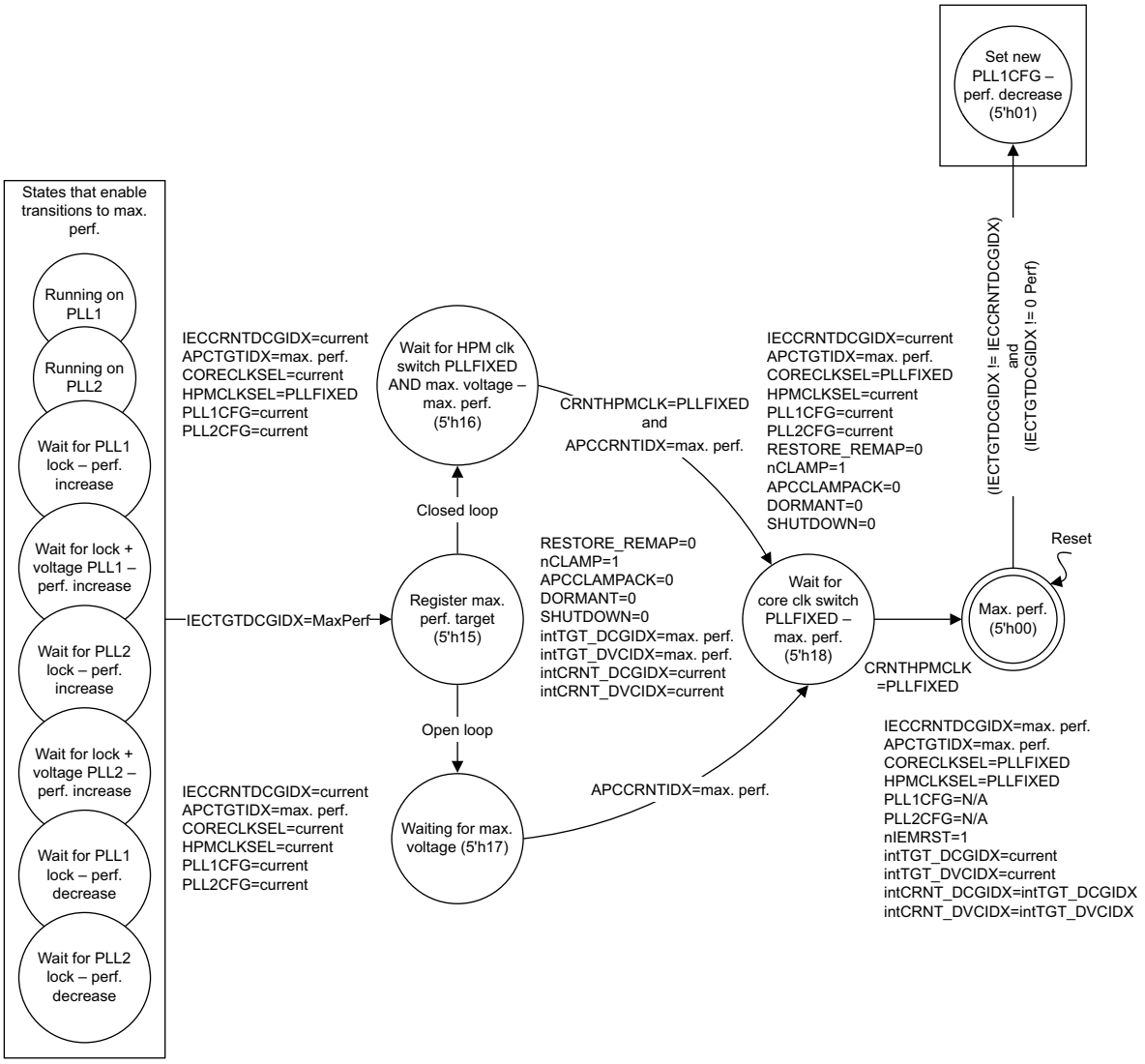


Figure 2-15 Increase to maximum performance

StandbyWFI mode

StandbyWFI mode is entered whenever the IEC selects 0% performance, and neither **SHUTDOWNNXT** nor **DORMANTNXT** flags are set.

When entering StandbyWFI mode, no new voltage requests are made to the APC. Voltage is held at the current level prior to requesting 0% performance.

If you enable closed loop, the clock to the HPM is not interrupted, and closed loop voltage control continues to operate as normal. For more information, see *StandbyWFI mode* on page 3-20.

Dormant and shutdown modes

In shutdown mode, the whole IEM subsystem is powered down. This includes the ARM1176JZF core, L2CC, and ETM11. You differentiate shutdown mode from dormant mode by setting the **SHUTDOWNNXT** or **DORMANTNXT** flag in the system controller. For more information, see *Dormant and shutdown mode* on page 3-21.

IEM subsystem permanent shutdown mode

You can completely disable the IEM subsystem to facilitate using an external master having full control of the chip. This option is configurable. This includes the ARM1176JZF core, CoreSight ETM11, and L2CC.

You can enable permanent IEM subsystem shutdown by setting configuration bit 36, **CFGIEMSHUTDOWN**. If you select permanent shutdown, the IEM subsystem is held in reset, and input and output clamps are enabled. Because of this, you can safely power-down the IEM subsystem.

Wake-up sources

Table 2-10 on page 2-36 shows wake-up sources that are implemented on the chip.

Note

All sources are *sticky*. You must clear them in hardware to enable you to write a new performance level to the IEC, prior to signal de-assertion.

Table 2-10 lists the IEC Maxperf and wake-up sources.

Table 2-10 IEC Maxperf and wake-up sources

Signal	MAXPERF functionality
nIRQ	Asserted when **nIRQ** active LOW. **Note** You must clear the interrupt source in either the GIC or TZIC.
nFIQ	Asserted when **nFIQ** active LOW. **Note** You must clear the interrupt source in either the GIC or TZIC.
USERMAXPERF	Asserted by the rising edge of SoC primary input **WAKUP**. Cleared by writing zero to the System Control Register bit 4.
DAPMAXPERF	Asserted by DAP **CSYSPWRUPREQ**. Optionally disabled by writing zero to the System Control Register bit 5. Visible after enable on bit 6.
CS_MAXPERF	Asserted by CTI #1 **TRIGGEROUT[1]**. Enables this trigger to act as a parallel trigger to **EDBGRQ**. **Note** The debugger must perform the IEC performance write and clear this trigger.

2.2.3 Remap controller

After power on reset, the state of the two remaps, **SMCREMAP** and **AXIREMAP**, depend on the values the configuration block captures. Although, you can clear each of the remaps post reset by software, you can never set them.

You must restore remap states captured in the configuration block when you return from:

- a soft reset
- a watch dog reset
- shutdown mode
- dormant mode.

Figure 2-16 on page 2-37 shows the remap control logic.

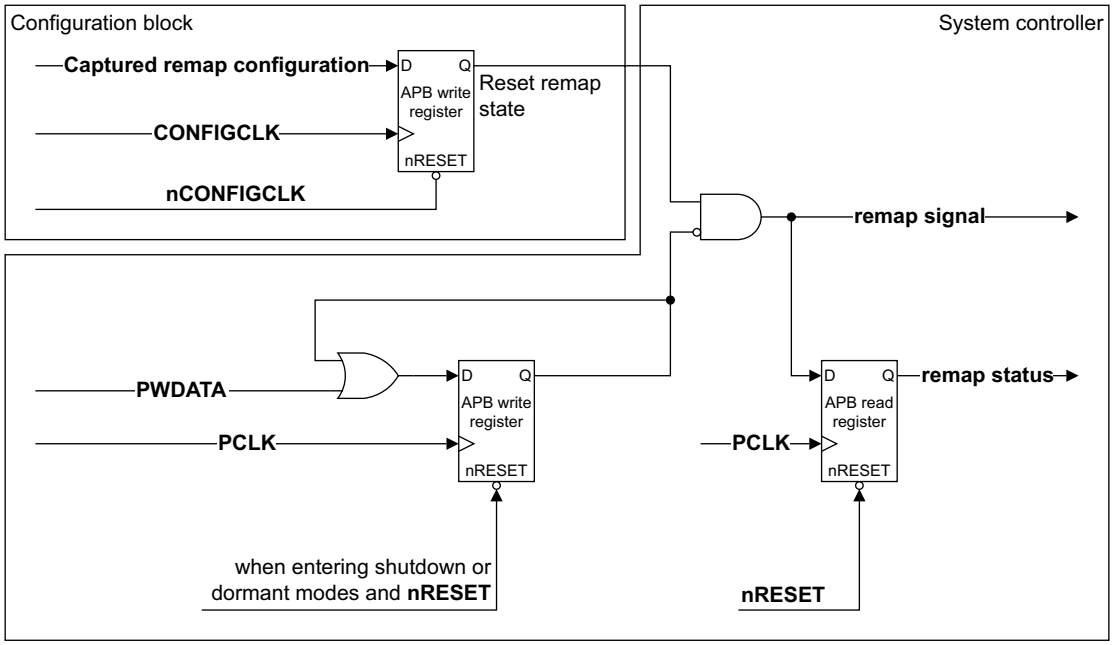


Figure 2-16 Remap logic

2.2.4 System reset controller

The following four sources can generate a system reset:

- **nRESET**
- watchdog reset
- software reset
- CoreSight reset.

The system reset controller combines these signals to generate a single **nRESETCOMB** output driven by the *Dynamic Clock Generator* (DCG).

To satisfy the ARM1176JZF core and AXI reset requirements, **nRESETCOMB** is held for eight **PCLK** cycles.

With minimum divide ratios configured, the ARM1176JZF core is held in reset for a minimum of 32 clock cycles. The AXI infrastructure is held for at least 16 cycles.

2.3 Intelligent Energy Management (IEM)

The IEM subsystem consists of the:

- *Intelligent Energy Controller* (IEC)
- *Dynamic Clock Generator* (DCG)
- System controller
- *Dynamic Voltage Controller* (DVC).

An *Advanced Power Controller* (APC1) from National Semiconductor is used as the DVC for the development chip. Both the DCG and system controller are new designs. The IEC is a standard ARM component (SY750).

The IEM software module, running on the ARM1176JZF core, requests the required performance level via the IEC. The IEC communicates with the DCG and DVC to provide the required combination of voltage and clock frequency to satisfy the performance requirements.

The IEC, system controller, and DVC, are APB memory-mapped peripherals. Figure 2-17 on page 2-39 shows a top-level overview of the IEM control system.

Note

Figure 2-17 on page 2-39 shows full connectivity only for the IEC, APC1 and HPM. For more information on the system controller and DCG see *System controller* on page 2-26 and *Dynamic Clock Generator (DCG)* on page 2-40.

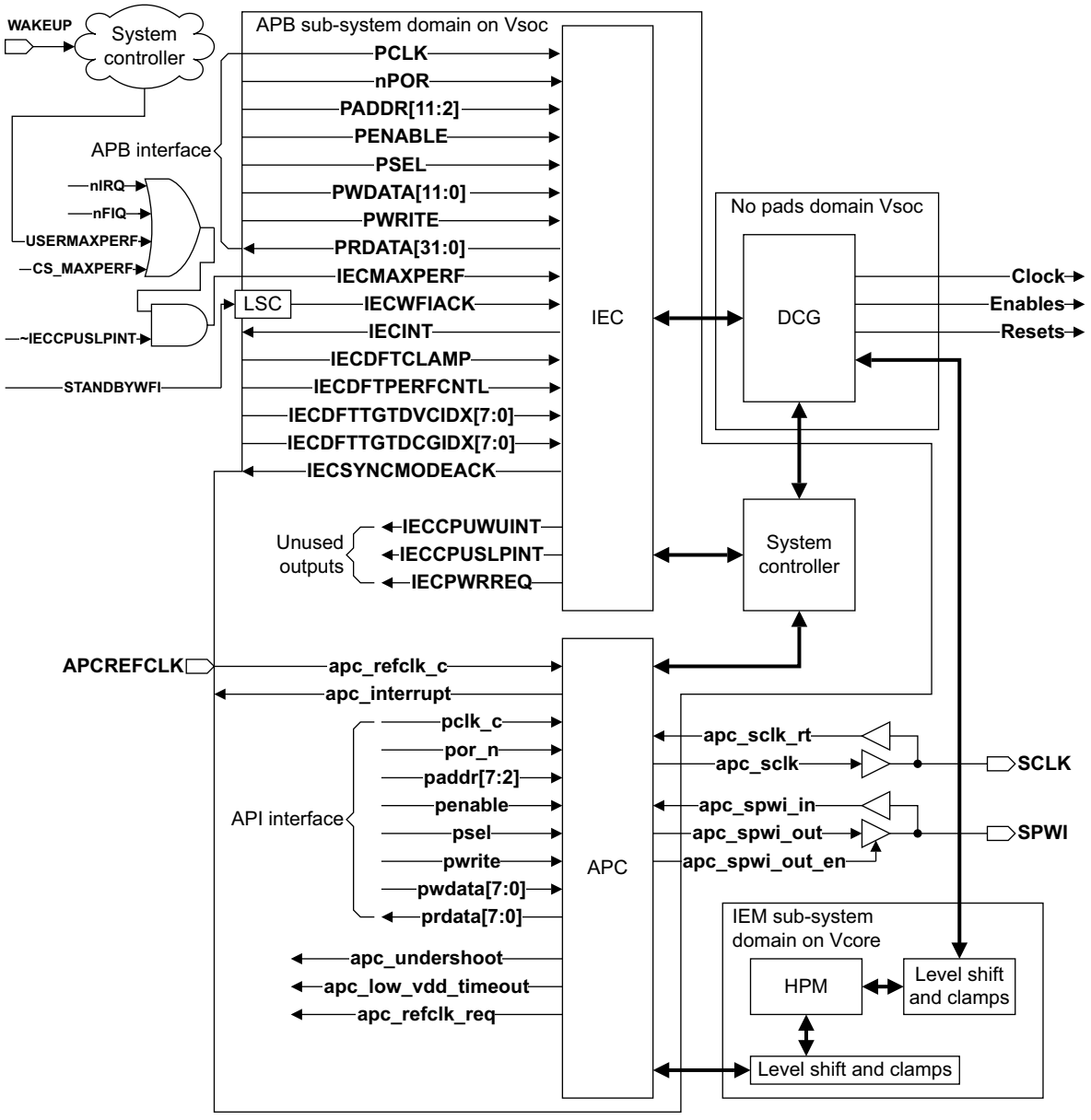


Figure 2-17 IEM block diagram

This section contains the following subsections:

- *Dynamic Clock Generator (DCG)* on page 2-40

- *Operation* on page 2-54
- *Dynamic Voltage Scaling (DVS)* on page 2-61.

2.3.1 Dynamic Clock Generator (DCG)

The DCG generates all clocks, resets, power and clamp signals, and sequencing for the chip.

The DCG consists of three PLLs. One PLL has a fixed frequency at reset, the other two are variable.

The fixed PLL clocks the processor at the 100% performance level and is used for the AXI, APB, and other fixed system clocks.

The two variable PLLs are used for all sub 100% performance levels. They are required for efficient switching between two sub-100% performance levels.

For example, if the performance level is currently set at 95%, the system runs on one of the variable PLLs. If a request is issued to drop to 90%, and only one variable PLL is in use, you must re-configure the PLL. Although the PLL is currently driving the system at 95%, it must now change to provide the 90% clock. The system must switch to the fixed PLL and wait for the variable PLL to lock at the new 90% performance frequency. If the current voltage is insufficient to support the fixed PLL 100% performance frequency, a divided down version of the fixed PLL clock is required.

A divided down clock is not efficient. The fastest divided down fixed PLL clock only delivers 50% performance. It produces a drop in performance. Using two variable PLLs enables the system to run at the current performance on one PLL, while the other PLL locks to the new performance level.

Figure 2-18 on page 2-41 shows the first part of the DCG block diagram.

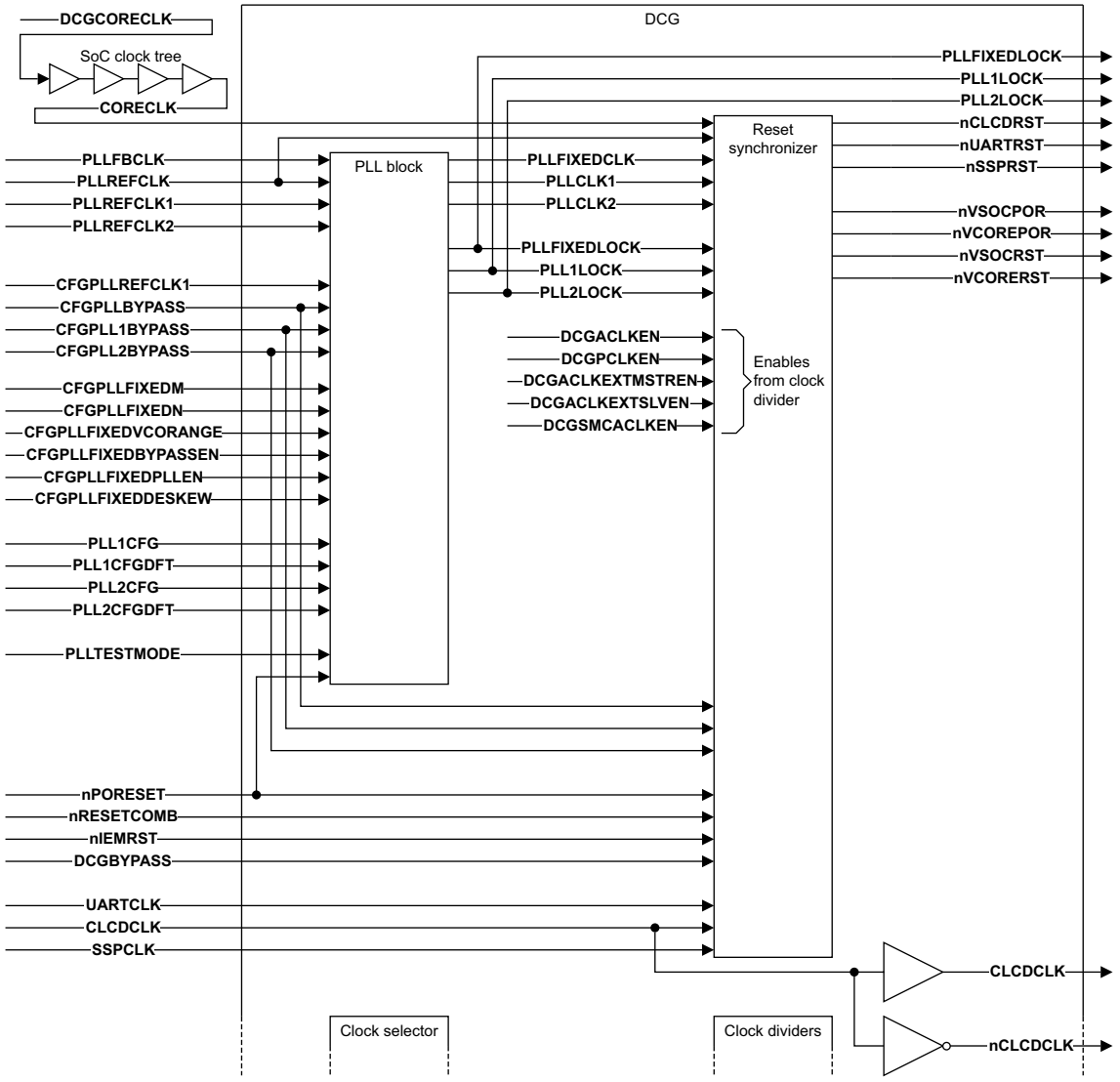


Figure 2-18 DCG block diagram part 1

Figure 2-18 shows the second part of the DCG block diagram.

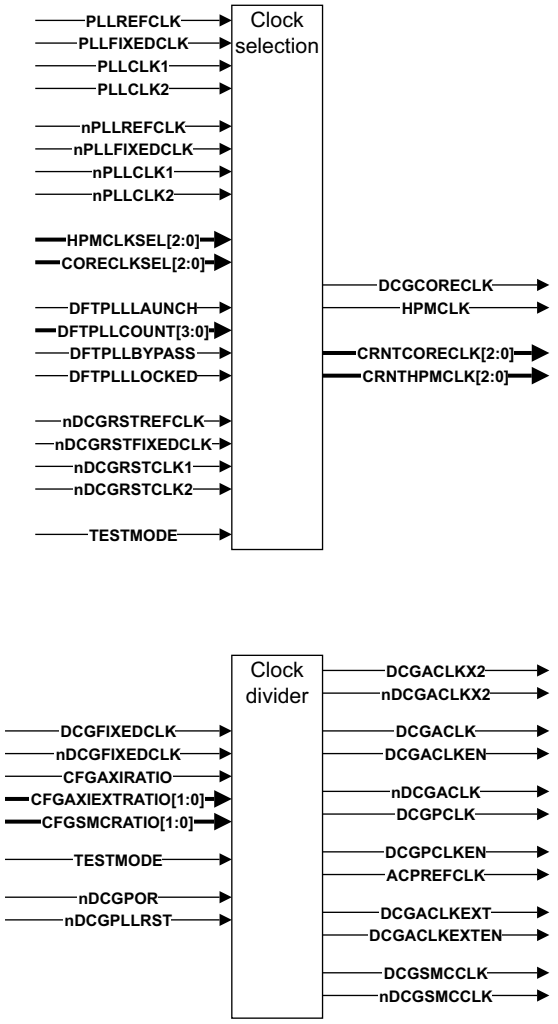


Figure 2-19 DCG block diagram part 2

PLL block

The PLL block contains three PLLs, PLL Fixed, PLL1 and PLL2.

PLL Fixed is configured at reset via the Configuration Block, and generates all fixed clocks in the design. it also clocks the IEM subsystem when running in 100% performance mode, or when IEM is disabled.

PLL Fixed has a fixed reference input, **PLLREFCLK**.

Note

The feedback clock to PLL Fixed is synchronous to the off-chip AXI slave port.

PLL1 and PLL2, generate clocks for each of the required sub-100% performance levels when IEM is enabled.

PLL1 and PLL2 have configurable reference input clocks:

- If the configuration input **CFGPLLREFCLK1** is set LOW, the reference clock for PLL1 and PLL2 is sourced from **PLLREFCLK**.
- If **CFGPLLREFCLK1** is set HIGH, the reference clock for both PLL1 and PLL2 is sourced from **PLLREFCLK1**.

For debug or test purposes, you can bypass all PLLs by setting the configuration input, **CFGPLLBYPASS**, HIGH. In this configuration, the **PLLFIXEDCLK** output is directly driven by **PLLREFCLK**. The **PLLCLK1** output is directly driven by **PLLREFCLK1**, and **PLLCLK2** is directly driven by **PLLREFCLK2**.

nPORESET controls the **PLL_EN** inputs on PLL Fixed. The respective PLLxCFG bus controls the **PLL_EN** for PLL1 and PLL2.

During PLL test mode, all PLL configuration inputs, and PLL clock and lock outputs, are made available from primary chip input and output.

Outputs from this block include the three PLL clocks, and the current PLL lock status for each PLL.

When **TESTMODE** is HIGH, manufacturing PLL test mode is selected. This enables **PLL1CFGDFT** and **PLL2CFGDFT** to control PLL1 and PLL2 respectively. Figure 2-20 on page 2-44 shows the connectivity of the PLL block.

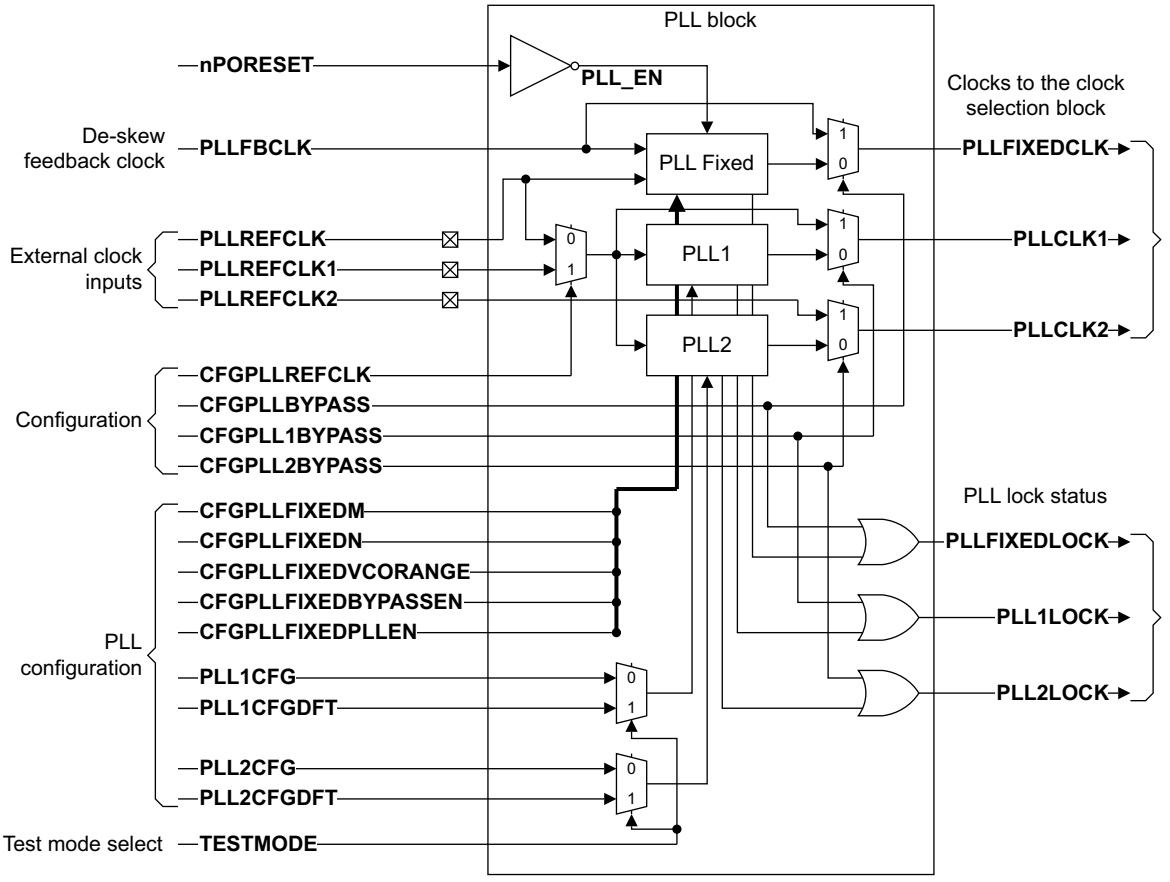


Figure 2-20 PLL block

Clock selection

In functional mode, the clock selection block enables you to dynamically switch the clock sources for **HPMCLK** and **DCGCORECLK** between the three clock sources, glitch free. You can select clock sources for both **HPMCLK** and **DCGCORECLK** independently. You select clocks by using two instantiations of the 3-way glitchless clock switch.

Clock select signals, **HPMCLKSEL** and **CORECLKSEL** are treated as asynchronous to this block, and are internally synchronized to each clock domain before use.

The selected clock outputs, **CRNTCORECLK** and **CRNTHPMCLK**, indicate the currently selected clock. The output of these signals is synchronous to the falling edge of the newly selected clock. You must therefore treat them as asynchronous before you read them in the PLL controller.

Clock divider

Figure 2-21 on page 2-46 shows a block diagram of the clock divider block.

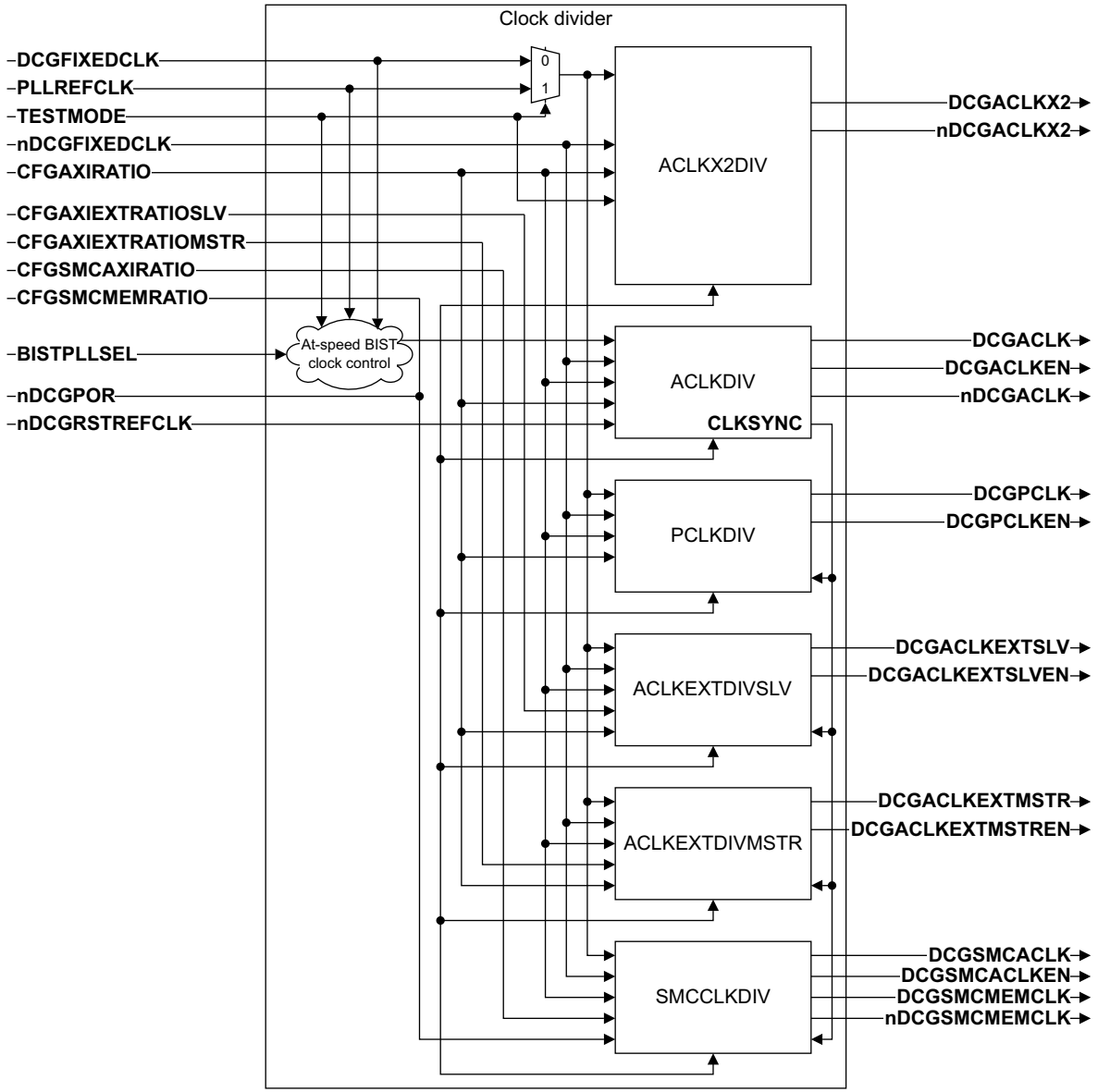


Figure 2-21 Clock divider block diagram

The clock generator block derives all the necessary system clocks and enables from **DCGFIXEDCLK**.

ACLKDIV This block generates the necessary clocks and enables for the AXI subsystem.

You define the required divide ratio between the AXI clock and the processor clock using the static **CFGAXIRATIO** configuration input. The divide ratio options are either 2:1 when **CFGAXIRATIO** = 0, or 4:1 when **CFGAXIRATIO** = 1.

ACLKX2DIV

This block generates the two AXI clocks required for the PL340 memory controller.

This block uses the **CFGAXIRATIO** input to determine the required divide ratio. This is either be 1:1 when **CFGAXIRATIO** = 0, or 2:1 when **CFGAXIRATIO** = 1.

PCLKDIV This generates the APB subsystem clock.

Note

The APB divide ratio is NOT user-programmable, and is fixed at 2:1 against the AXI clock.

ACLKEXTDIVSLV

This generates the clock and enable for the off-chip AXI slave port and generates the PLL feedback clock.

The feedback clock de-skews the system clock and enables the AXI slave ports to operate synchronously between off-chip devices.

ACLKEXTDIVSMSTR

This generates the clock and enable for the off-chip AXI master port. For this port, the clock is exported off-chip.

SMCCLKDIV

This block generates the clocks required for the SMC PL093. **CFGSMCAXIRATIO** defines the speed that the AMBA interface is clocked with respect to the AXI clock.

This can either be 1:1 when **CFGSMCAXIRATIO** = 0, if 2:1 when **CFGSMCAXIRATIO** = 1.

The memory clock for the SMC can either run at 1:1, 2:1 or 3:1 with respect to the SMC AMBA interface. This block also generates an inverted version of the memory clock that you can use as the

SMMEMCLKDELAY clock input on the SMC PL093.
SMMEMCLKDELAY re-times data and control signals to off-chip memories.

Note

All clock divided ratios are fixed at reset. You cannot alter them dynamically without a system reset.

Figure 2-22 shows the generic circuit that generates the divided-down clocks. You generate all divided clocks by gating pulses and then extending the falling edge to maintain a 50-50 mark-space ratio.

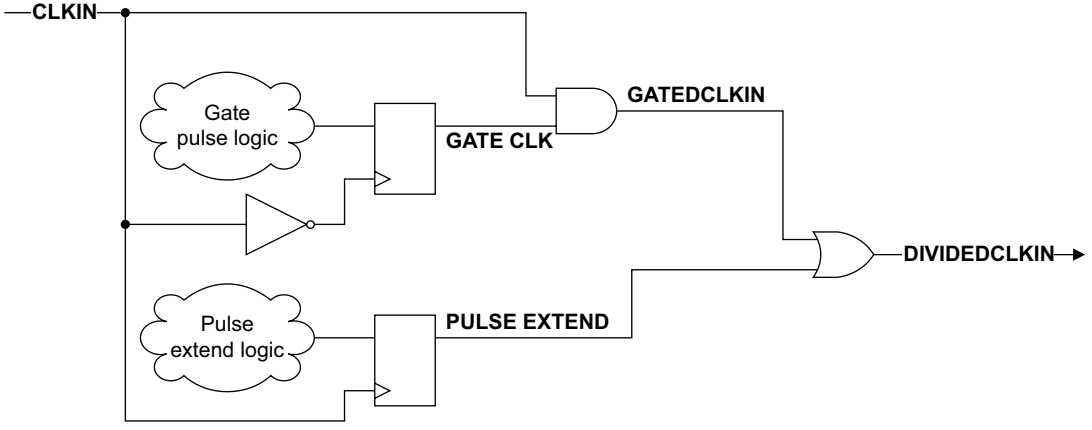


Figure 2-22 Clock divide circuit

Figure 2-23 shows an example of how to generate a divide by 4.

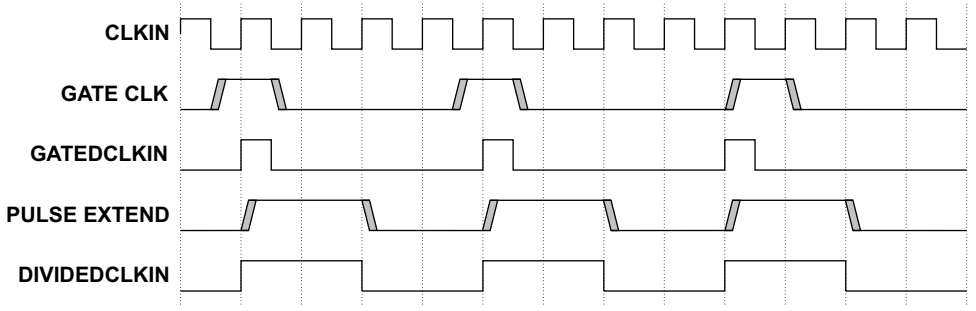


Figure 2-23 Example divide by 4

Table 2-11 lists all the system clock divide ratio options.

Table 2-11 System clock divide ratio options

AXI divide ratio	APB divide ratio	APB divide ratio overall from CPU	SMC AXI divide ratio	SMC AXI overall from CPU	SMC mem divide ratio	SMC mem divide ratio overall from CPU	AXI master divide ratio	AXI master divide ratio overall from CPU	AXI slave divide ratio	AXI slave divide ratio overall from CPU
2:1	2:1	4:1	1:1	2:1	1:1	2:1	1:1	2:1	1:1	2:1
2:1	-	-	1:1	2:1	2:1	4:1	2:1	4:1	2:1	4:1
2:1	-	-	1:1	2:1	3:1	6:1	3:1	6:1	3:1	6:1
2:1	-	-	2:1	4:1	1:1	4:1	4:1	8:1	4:1	8:1
2:1	-	-	2:1	4:1	2:1	8:1	5:1	10:1	5:1	10:1
2:1	-	-	2:1	4:1	3:1	12:1	6:1	12:1	6:1	12:1
2:1	-	-	-	-	-	-	7:1	14:1	7:1	14:1
2:1	-	-	-	-	-	-	8:1	16:1	8:1	16:1
4:1	2:1	8:1	1:1	4:1	1:1	4:1	1:1	4:1	1:1	4:1
4:1	-	-	1:1	4:1	2:1	8:1	2:1	8:1	2:1	8:1
4:1	-	-	1:1	4:1	3:1	12:1	3:1	12:1	3:1	12:1
4:1	-	-	2:1	8:1	1:1	8:1	4:1	16:1	4:1	16:1
4:1	-	-	2:1	8:1	2:1	16:1	5:1	20:1	5:1	20:1
4:1	-	-	2:1	8:1	3:1	24:1	6:1	24:1	6:1	24:1
4:1	-	-	-	-	-	-	7:1	28:1	7:1	28:1
4:1	-	-	-	-	-	-	8:1	32:1	8:1	32:1

Table 2-12 lists the possible system operating frequencies for a 320MHz CPU clock speed.

Table 2-12 Example system frequencies with 320MHz CPU speed

AXI divide ratio	AXI clock freq. MHz	APB divide ratio	APB clock freq. MHz	SMC AXI divide ratio	SMC AXI clock freq. MHz	SMC mem divide ratio	SMC mem clock freq. MHz	AXI EXT divide ratio	AXI EXT clock freq. MHz	AXI EXT divide ratio	AXI EXT clock freq. MHz
2:1	160	2:1	80	1:1	160	1:1	160	1:1	160	1:1	160
2:1	160	-	-	1:1	160	2:1	80	2:1	80	2:1	80
2:1	160	-	-	1:1	160	3:1	53.3	3:1	53.3	3:1	53.3
2:1	160	-	-	2:1	80	1:1	80	4:1	40	4:1	40
2:1	160	-	-	2:1	80	2:1	40	5:1	32	5:1	32
2:1	160	-	-	2:1	80	3:1	26.6	6:1	26.7	6:1	26.7
2:1	160	-	-	-	-	-	-	7:1	22.9	7:1	22.9
2:1	160	-	-	-	-	--	-	8:1	20	8:1	20
4:1	80	2:1	40	1:1	80	1:1	80	1:1	80	1:1	80
4:1	80	-	-	1:1	80	2:1	40	2:1	40	2:1	40
4:1	80	-	-	1:1	80	3:1	26.6	3:1	26.7	3:1	26.7
4:1	80	-	-	2:1	40	1:1	40	4:1	20	4:1	20
4:1	80	-	-	2:1	40	2:1	20	5:1	16	5:1	16
4:1	80	-	-	2:1	40	3:1	13.3	6:1	13.3	6:1	13.3
4:1	80	-	-	-	-	-	-	7:1	11.4	7:1	11.4
4:1	80	-	-	-	-	-	-	8:1	10	8:1	10

Hardware Performance Monitor (HPM)

The HPM clock is always the system clock value you require, divided by four.

ETM and L220 cache controller StandbyWFI clock gate

You can maximize power saving when the ARM1176JZF core enters StandbyWFI mode. Disable the clock to the CoreSight ETM11 and L2CC.

The system controller generates the ETM11 and L2CC clock gate enable. It is dependent on the following:

- L2CC activity
- IEC requesting 0% performance
- the ARM1176JZF core entering StandbyWFI mode.

Note

Although clock gating of the ETM and L2CC clock is enabled by default, you can disable it by setting the CFGCLKSTOPSBWFI configuration bit LOW.

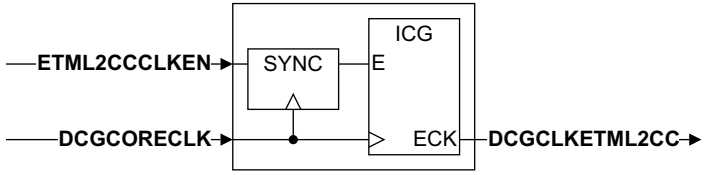


Figure 2-24 ETM and L2CC StandbyWFI clock gate

Reset synchronization

The reset synchronization block provides all required SoC resets. Resets are split into the following three domains:

DCG Resets are synchronized to each PLL clock and **PLLREFCLK**. These resets are used within the DCG only.

Asynchronous

This provides the resets for the three asynchronous clocks used for the CLCD, UARTs, and SSP.

SoC This provides the main SoC level resets. These include the power-on, and system resets.

This section contains the following subsections:

- *Reset sources* on page 2-52
- *DCG resets* on page 2-52
- *Asynchronous resets* on page 2-53
- *Reset sources* on page 2-52.

Reset sources

The chip has one main power-on-reset, **nPORESET**, and four system level resets. The system level resets are:

nRESET Board level reset. This is a primary SoC input.

Watchdog Generated by the APB Watchdog peripheral.

Software Generated by unlocking and writing to the System Control Status Control register. See *System Control Register* on page 3-35 and *System Status Register* on page 3-37.

CS_nSRST Resets the system while debugging under CoreSight control.

All these resets are combined within the system controller to generate a single system level reset, **nRESETCOMB**, at the DCG. The system controller has responsibility to record the cause of the last reset for software purposes.

The system controller holds **nRESETCOMB** for a minimum of three core clock cycles.

DCG resets

All DCG resets are from generated from **nPORESET**. Table 2-13 lists the DCG resets.

Table 2-13 DCG resets

Reset signal name	Synchronizing clock	Special function
nDCGREFRST	**PLLREFCLK**	None
nDCGPOR	**PLLFIXEDCLK**	None
nDCGRSTFIXED	**PLLFIXEDCLK**	Held in reset state when **PLLFIXEDLOCK** = 1'b0 unless **CFGPLLBYPASS** = 1'b1
nDCGRSTCLK1	**PLLCLK1**	Held in reset state when **PLL1LOCK** = 1'b0 unless **CFGPLL1BYPASS** = 1'b1
nDCGRSTCLK2	**PLLCLK2**	Held in reset state when **PLL2LOCK** = 1'b0 unless **CFGPLL2BYPASS** = 1'b1

Asynchronous resets

Either **nPORESET** or **nRESETCOMB** generate synchronous resets. Table 2-14 lists the asynchronous resets.

Table 2-14 Asynchronous resets

Reset signal name	Synchronizing clock	Special function
nUARTRST	**UARTCLK**	None
nCLCDRST	**LCDCLK**	None
nSSPRST	**SSPCLK**	None

SoC resets

There are four SoC resets as follows:

nVSOCPOR

Power-on reset for all logic on the VSOC power domain.

nVCOREPOR

Power-on reset for all logic on the VCORE power domain.

nVSOCRST

System reset for all logic on the VSOC power domain.

nVCORERST

System reset for all logic on the VCORE power domain.

You can assert both **nVCOREx** resets by setting **nIEMRST** = 1'b0.

Note

The system controller controls the **nIEMRST** signal, and enables entry-to and exit-from shutdown and dormant modes. Table 2-15 on page 2-54 lists the SoC reset signals.

Table 2-15 SoC reset signals

Reset signal name	Reset	Synchronizing clock	Special function
nVSOCPOR	**nPORESET**	**CORECLK**	Held in reset state when **PLLFIXEDLOCK** = 1'b0 unless **CFGPLLBYPASS** = 1'b1. Not released until 100 **PLLREFCLK** cycles after **PLLFIXEDLOCK** = 1'b1 unless **CFGPLLBYPASS** = 1'b0 in which case it is released 100 **PLLREFCLK** cycles after **nPORESET** is released. Is not released until all clock enables are HIGH.
nVCOREPOR	**nPORESET** **nIEMRST**	**CORECLK**	Same as **nVSOCPOR** with the addition of reset being asserted when **nIEMRST** = 1'b0
nVSOCRST	**nPORESET** **nRESETCOMB**	**CORECLK**	Behaves the same as **nVSOCPOR** when reset by **nPORESET**. If reset by **nRESETCOMB**, **nVSOCRST** is released when it has been synchronized to **CORECLK**, and all clock enables are HIGH.
nVCORERST	**nPORESET** **nRESETCOMB** **nIEMRST**	**CORECLK**	Same as **nVSOCRST** with the addition of reset being asserted when **nIEMRST** = 1'b0

2.3.2 Operation

This section describes the detailed operation of the IEM system in both closed, and open loop modes and contains the following subsections:

- *Open loop mode*
- *Closed loop mode* on page 2-58
- *Clock enable control* on page 2-60.

Open loop mode

In open loop mode, the required supply voltage is fixed. It is based on a pre-defined voltage table programmed in the APC1 'Open-loop VDD Core Registers 1-8'. Some energy is always wasted because of the extra voltage margin required to ensure the device operates across temperature variations.

The DCG can operate with or without *Performance Requirement Optimization*. If you enable Performance Requirement Optimization, the current voltage from the DVC, IECCRNTDVCIDX, and IEC configuration mapping IECCFGDVCIDXMAP, determine if the target frequency on IECTGTDCGIDX can be achieved at the current voltage. If the performance level is achievable, the PLL controller can immediately prepare the PLLs and switch when ready.

If the required voltage is not available for the target frequency, the PLL controller switches to the highest available frequency for the current voltage. However, if the current frequency is at the maximum for the current voltage, the PLL controller prepares to switch to the maximum frequency for the next voltage step. This enables maximum performance as the voltage level rises. Figure 2-25 on page 2-56 shows an example LOW to HIGH frequency switch. When the frequency level is achieved, the unused PLLs, are powered down. This is only performed if the PLL lock time from power-down is acceptable.

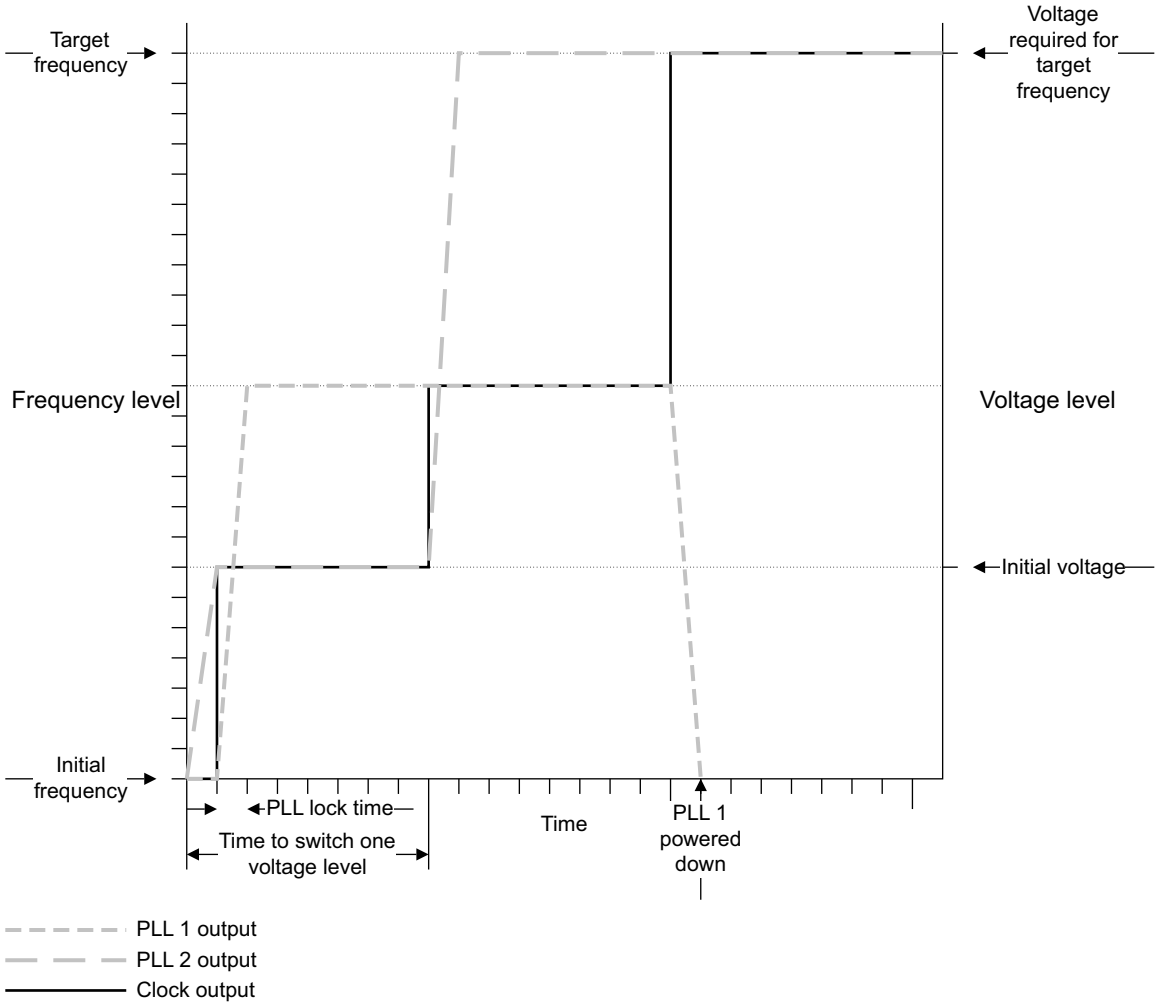


Figure 2-25 Switching with performance requirement optimization

Performance requirement optimization normally requires the DVC to monitor and report all intermediate voltage steps towards the target frequency. The National Semiconductor APC1 device does not require this. Instead, when the APC1 receives a new target performance level, it starts a timer. This timer is set to the worst case time required to switch between the minimum and maximum voltage levels. When this timer expires, the current index level is updated with the requested target level. The timer is programmed using the VDD_DELAY register in the APC1.

With performance requirement optimization turned off, the PLL controller switches directly to the new performance when the required target voltage is achieved. Figure 2-26 shows switching to a higher frequency without using performance requirement optimization.

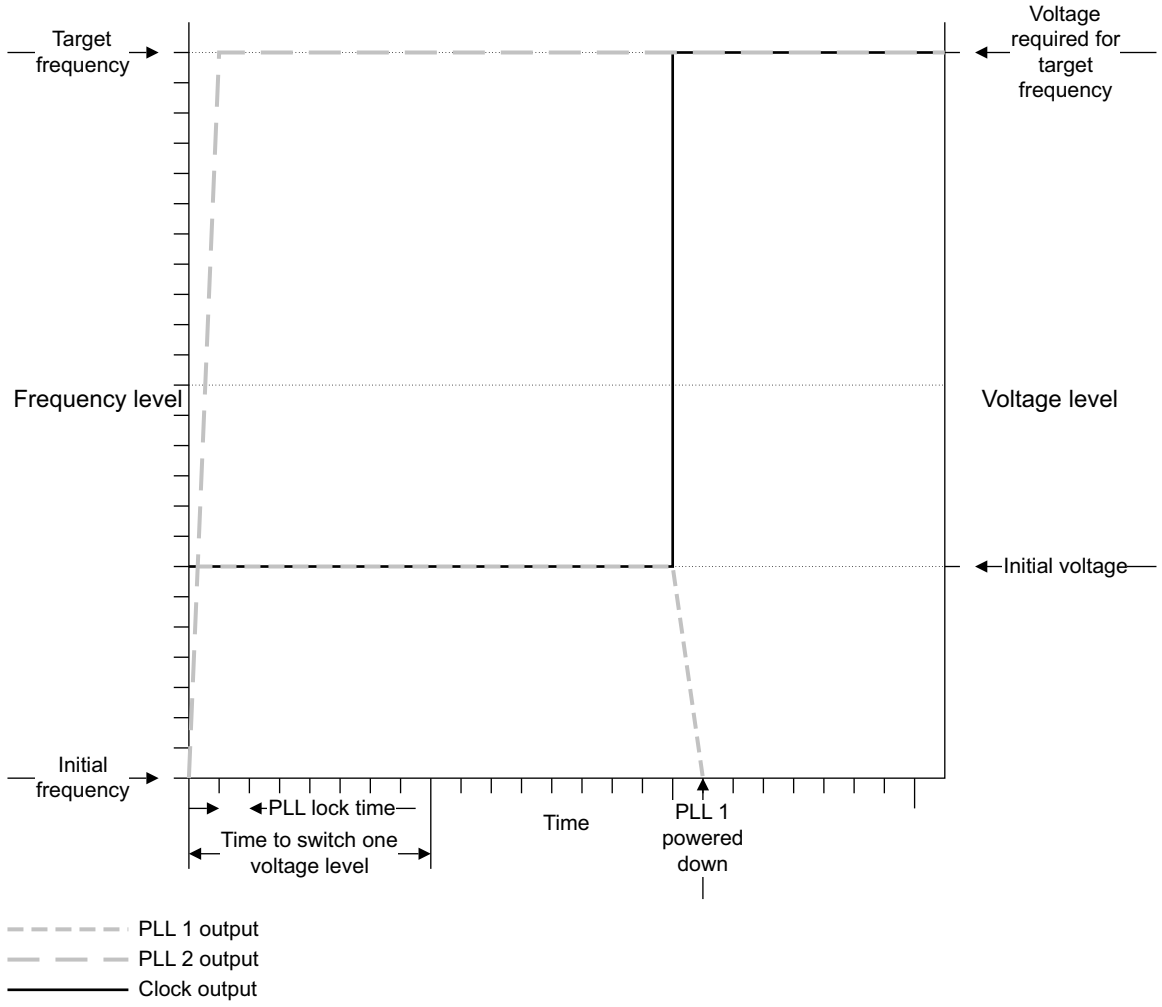


Figure 2-26 Switching without performance requirement optimization

Performance requirement optimization is disabled by default. You can enable it by setting the PERFREQOP bit in the DCG Control Register.

Note

Never enable performance requirement optimization for the chip.

When switching to a lower performance level, the DCG can switch to the lower performance clock as soon as the PLL is ready.

If a new performance target is issued by the IEC before the current target has successfully completed, the new target always overrides the current target.

Closed loop mode

A *Hardware Performance Monitor* (HPM) is used for closed loop mode. For this chip, National Semiconductor provides the HPM. The HPM is a delay line based analog to digital converter. It provides gate delay information with respect to process, temperature, and voltage. The advantage of using the HPM in a closed loop system is that you can continually adjust the supplied voltage. This ensures the minimum voltage required to support the current performance is always applied. This also reduces voltage slack in the system, and wastes less energy.

For more information on the APC1 and HPM, see *Adaptive Power Controller – APC1 – (SY751),* National Semiconductor (SY751-DA-03001).

The HPM is provided with a sampling clock that enables it to evaluate if the current voltage is sufficient to support the required frequency. This clock is directly related to the target system clock. For the development chip, the HPM clock is the target clock divided by four.

ARM recommends that you update the Target_Performance_Index simultaneously to both the DCG and APC1 when switching to a higher frequency. If the sampling clock to the HPM is not updated quickly enough, errors can result. The APC1 can incorrectly assume the voltage is sufficient for the target frequency when in fact it is still evaluating the voltage for the current frequency.

The VDD_DELAY timer in the APC1 helps to prevent this incorrect evaluation of voltage. It delays the time between the Target_Performance_Index changing and HPM evaluation. This is not ideal because the delay timer is fixed. It must be set to the maximum possible time for the HPM clock to be updated. This setting might be unacceptable because the PLL lock time is unpredictable. Although under normal circumstances it is less than 50µs, worst case locking time can be a few milliseconds.

This setting results in additional performance overhead when operating in closed loop mode. Figure 2-27 on page 2-59 shows the timing diagram for this standard mode of operation for closed loop performance switching.

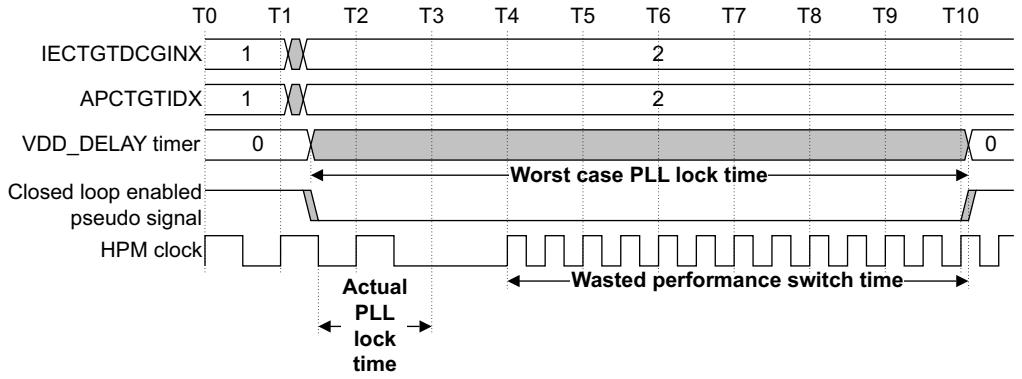


Figure 2-27 Standard APCI closed loop behavior

To address this issue, the IEC does not directly drive the Performance_Target_Index level to the APC1. Instead, the DCG drives this signal in a proxy manner. The following sequence is used when switching to a higher performance level:

1. IEM software requests a new performance level by writing to the IEC.
2. The IEC updates the **IECTGTDCGIDX** output to the DCG.
3. The DCG delays the **APCTGTIDX** request to the APC1 and then uses the PLL lock signals to more accurately predict when the HPM sampling clock is ready.
4. The DCG issues the new **APCTGTIDX** to the APC1.
5. The DCG switches the HPM sampling clock to the new performance level.

The advantage of this system is that the VDD_DELAY timer in the APC1 only requires that you set the maximum time required to switch in the new HPM clock. This time is much smaller then the worst case PLL lock time. It reduces unnecessary time in the performance level switching. Figure 2-28 on page 2-60 shows this improved operation.

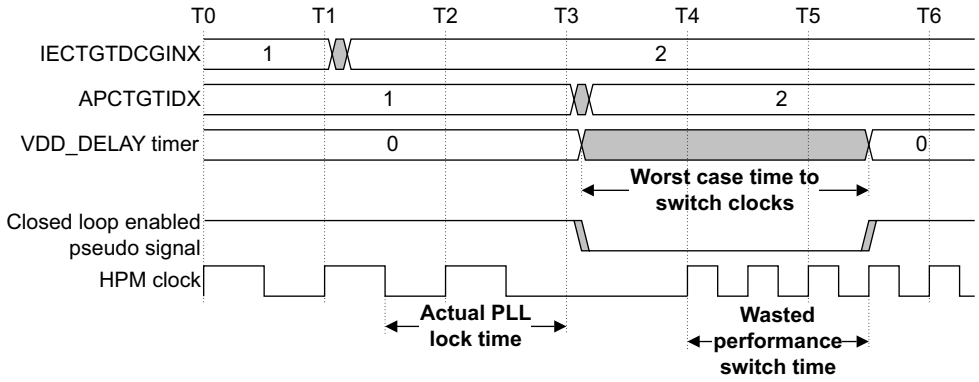


Figure 2-28 Improved closed loop with target voltage under DCG control

When switching to a lower performance level, it is assumed that lower frequencies never require a higher voltage than is currently available. A fall in voltage to the required level while the circuit is active does not affect the circuit's operation. The APC1 immediately updates **APCCRNTIDX** to reflect **APCTGTIDX**. This enables the DCG to immediately switch both the system and HPM sampling clock to the lower frequency. The APC1 then uses feedback from the HPM to lower the voltage.

Similar to open-loop mode, if the IEC issues a new performance target before the current target is achieved, it overrides the current target. Closed-loop mode is different from open-loop mode in that it can be interrupted at any stage up until the point where the command to switch clocks is issued. In closed loop mode, when **APCTGTIDX** is updated to the APC1 and the new HPM clock is selected, the system is committed to complete the initial performance transfer.

Clock enable control

You must set the clock enable HIGH for efficient operation of the AXI register slices during asynchronous mode.

The state of the **IECSYNCMODEREQ** and **IECSYNCMODEACK** signals from the IEC determine the control of the enable. Because of the operating nature of the register slices, it is not possible to determine exactly when they enter synchronous mode after the assertion of **IECSYNCMODEREQ**, or when they enter asynchronous mode after the de-assertion of **IECSYNCMODEREQ**.

Although **IECSYNCMODEACK** indicates when the transition is safely complete, internally, the point when the slices actually switch modes is indeterminant. You must assume the slices are running in synchronous mode during this undefined operating state. You must permit the enable to toggle normally to avoid data transfer corruption.

There is only time when you can guarantee the slices are in asynchronous mode and that the enable can be driven HIGH. This time is when both **IECSYNCMODEREQ** and **IECSYNCMODEACK** are LOW. Figure 2-29 shows the clock enable control between synchronous and asynchronous modes.

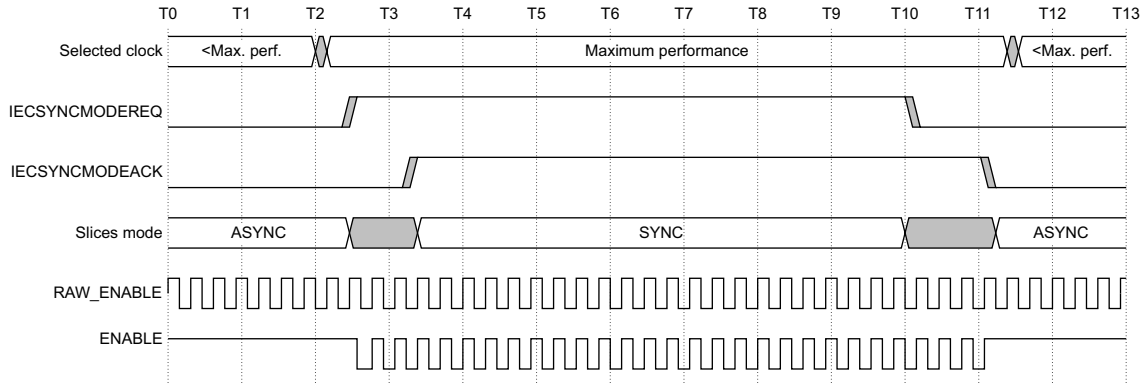


Figure 2-29 Clock enable control between synchronous and asynchronous modes

2.3.3 Dynamic Voltage Scaling (DVS)

DVS emulation enables you to carry out performance scaling on systems that do not support voltage scaling. With DVS emulation you permanently hold the voltage at the maximum level and then switch the system clock between the maximum frequency and off.

The percentage of time spent at the maximum frequency within a defined time frame is then directly related to the performance level requested by the IEM software.

For example, if the IEM software determines it requires only 50% performance, the processor is run at maximum clock speed for 50% of the frame time and stopped for the remaining 50%, until the new frame starts. You can configure the frame time completely using the IEC.

Each frame is divided into eight time slots that represent the eight performance indexes. Because you cannot configure the time slots within a frame, performance level intervals for DVS emulation are fixed at 12.5%.

If software requests a new performance mid frame, that frame is cancelled and a new one is started with the updated performance level request.

Figure 2-30 on page 2-62 shows an example of DVS emulation where the system performance request changes from 50% in the first frame, to 25% in the second frame, and then to 75% in the last frame.

Note

Figure 2-30 represents only the relative occurrence of control signals. It is not a cycle-accurate representation.

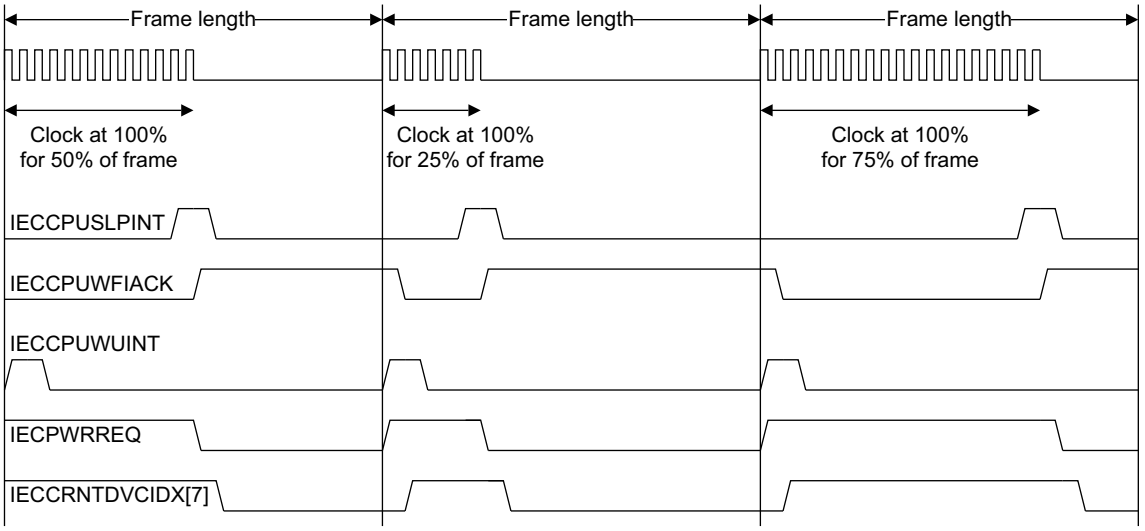


Figure 2-30 DVS emulation example

The IEC controls the DVS emulation using the following signals:

IECCPUSLPINT

The IEC asserts this interrupt to indicate the processor is no longer required for the current frame. Servicing this interrupt usually involves putting the ARM1176JZF core into StandbyWFI, and must therefore involve all relevant software steps, such as draining level 1 memories.

IECCPUWFIACK

This IEC input is connected to the **STANDBYWFI** output of the ARM1176JZF core. When this signal is asserted, the IEC requests performance level 0.

IECCPUWUINT

This interrupt is generated at the start of the new frame to wake-up the processor.

You can extend DVS emulation mode. Reduce the voltage to a retention level while the system is inactive using the following control signals:

IECPWRREQ

This signal is asserted at the start of a new frame and indicates to the power supply that maximum voltage is required.

IECCRNTDVCIDX[7]

Because of the use of thermometer encoding for **IECCRNTDVCIDX**, you must only monitor bit 7 to make sure that the maximum achieved voltage has been achieved, and that the clock can be restarted.

There is also the special case when **IECMAXPERF** is asserted while in DVS emulation mode. If this event occurs, the following sequence is performed:

1. An interrupt causes **IECMAXPERF** to be asserted.

   This can occur in either the active or sleep stage of the DVS emulation cycle.

2. Assertion of **IECMAXPERF** in the active stage causes DVS emulation to be abandoned, and the current frame timing terminates.

   IECPWRREQ remains HIGH, and **IECCPUSLPINT** is not asserted.

3. Assertion of IECMAXPERF in the sleep stage causes DVS emulation to be abandoned and the current frame timing terminates.

   IECPWRREQ is asserted together with **IECCPUWUINT**.

4. The processor clears **IECCPUWUINT** and services the maximum performance condition.

5. De-assertion of **IECMAXPERF** results in the IEC returning to DVS emulation mode and starting a new frame at the current performance level.

2.4 Generic Interrupt Controller (GIC)

This section describes the GIC. The *Generic Interrupt Controller* (GIC) is logically split into two parts:

- Interrupt prioritization and distribution to each CPU interface. This is known as the *Distributor*.
- Priority masking and preemption handling for each CPU. This is known as the *CPU Interface*.

Although the RTL for the GIC is derived from the initial MPCore development, because it is essentially parameterized, it is still relevant to the single-core ARM1176JZF Development Chip. This section describes the implemented distributor and CPU interface functions.

This implementation complies with the *Generic Interrupt Controller Architecture* (PRD03-GENC-004392) document.

The number of interrupts a GIC can handle is implementation-specific, with a maximum of 1 020. The GIC implemented in this delevlopment chip can handle 64 interrupts. Of these 64 interrupts, the lower 32 bits, [0:31], are reserved for private interrupt sources, and in a multiprocessing environment, banked for each CPU.

To support platform OS compatibility, the interrupt sources of the lowest 8 bits, [0:7], must be reserved for software use.

Each interrupt source has their own configurable priority and list of targeted CPUs. Within the chip, 16 levels of priority are supported, and one core handles all interrupts.

See *Generic Interrupt Controller (GIC)* on page 3-44 for a description of the GIC registers.

2.5 CoreSight

This section describes the CoreSight architecture implemented in the development chip. It contains the following subsections:

- *AXI access*
- *JTAG test access port* on page 2-66
- *APB modification* on page 2-67
- *ATB off-chip* on page 2-67
- *Trigger connectivity* on page 2-68
- *DapRomDefs file* on page 2-70.

Note

All CoreSight components operate at V_{soc} voltage.

2.5.1 AXI access

You can access CoreSight components from the 32-bit AXI infrastructure. Only privileged accesses, **AxPROT[0]**, are permitted to pass to the CoreSight subsystem. Non-privileged accesses are blocked and return a DECERR, resulting in a data abort.

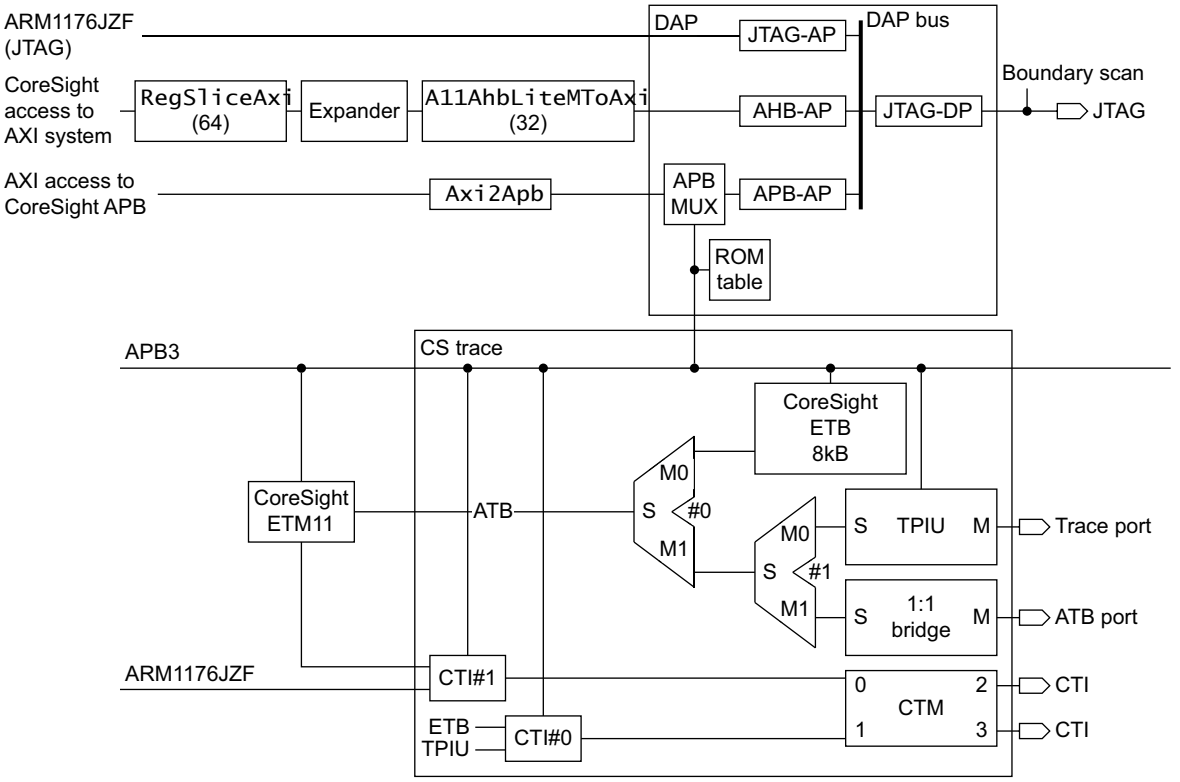


Figure 2-31 CoreSight architecture

2.5.2 JTAG test access port

The JTAG *Test Access Port* (TAP) provides access to the boundary scan TAP controller and to debug logic within the functional design. Two SoC pins, **JTAGBSTEN** and **JTAGDAPEN** define the functional path of the JTAG TAP.

When **JTAGBSTEN** is HIGH, the boundary scan TAP controller is selected. The functional debug logic is held in a safe state with TMS held HIGH. The boundary scan TAP controller is a Synopsys DesignWare component.

When **JTAGBSTEN** is LOW, the debug logic path is selected as follows:

- When **JTAGDAPEN** is HIGH, the CoreSight DAP is selected.

  The JTAG-DP, DAP JTAG debug port, is connected to the JTAG TAP and the ARM1176JZF core is connected to the JTAG-AP, DAP JTAG access port.

Note

There is no JTAG connection to the CoreSight ETM11.

- When **JTAGDAPEN** is LOW, the CoreSight DAP is bypassed.

  The JTAG TAP connects directly to the JTAG port on the ARM1176JZF core.

  Note

  RTCK is brought out to the JTAG TAP for dynamic clocking of the ARM1176JZF chip. This is essential for IEM.

Table 2-16 lists the JTAG TAP selection, depending on the values of **JTAGBSTEN** and **JTAGDAPEN** signals.

Table 2-16 JTAG TAP selection

JTAGBSTEN	JTAGDAPEN	JTAG port functional path
0	0	JTAG port to ARM1176JZF core, CoreSight DAP bypassed
0	1	JTAG port to CoreSight DAP
1	X	JTAG port to boundary scan TAP Controller

2.5.3 APB modification

When the ETM11 core voltage domain is powered down, **PREADY** remains LOW. **PSLVERR** is tied permanently LOW in the ETM11.

To enable the CoreSight APB master access to other APB peripherals, **PREADY** is pulled HIGH whenever the **SHUTDOWN** or **DORMANT** signals from the System Controller are HIGH. **PSLVERR** is also pulled HIGH with **PREADY** to indicate an error.

2.5.4 ATB off-chip

A 1:1 sync bridge is implemented to take the ATB off-chip. Because this interface only works at a lower frequency, the ATB **VALID** and **READY** signals on the bridge master and slave interfaces are tied to prevent input and output signal toggling. This also ensures the bridge cannot hold off the replicator, and therefore stop ATB transactions between the CoreSight ETM11 and ETB or TPIU.

The flush VALID and READY bridge interfaces are similarly disabled. This is accomplished using the **EXTCTLOUT[0]** output from the TPIU. The default after reset is to disable transfers.

2.5.5 Trigger connectivity

CoreSight triggering is accomplished using two *Cross Trigger Interfaces* (CTIs) and a *Cross Trigger Matrix* (CTM). The two CTIs use two of the four trigger channels on the CTM. The other two trigger channels are brought out of the SoC to enable multi-core debug. Table 2-17 summarizes the CTI #0 connectivity.

Table 2-17 CTI #0 connectivity

Bit index	Trigger in	Trigger in ack	Trigger out	Trigger out ack
[0]	Tied LOW	N/C	ETB flush	ETB flush ack
[1]	Tied LOW	N/C	ETB trig	ETB trig ack
[2]	Full, ETB	N/C	TPIU flush	TPIU flush ack
[3]	**AcqComp**, ETB	N/C	TPIU trig	TPIU trig ack
[4]	Tied LOW	N/C	N/C	Tied LOW
[5]	Tied LOW	N/C	N/C	Tied LOW
[6]	Tied LOW	N/C	N/C	Tied LOW
[7]	Tied LOW	N/C	N/C	Tied LOW

Table 2-18 summarizes the CTI #1 connectivity.

Table 2-18 CTI #1 connectivity

Bit index	Trigger in	Trigger in ack	Trigger out	Trigger out ack
[0]	**DBGACK**, 1176	N/C	EDBGRQ	DBGACK, 1176
[1]	!**nPMUIRQ**, 1176	N/C	CS_MAXPERF	Tied LOW
[2]	Tied LOW	N/C	CS_IRQ	Tied LOW
[3]	Tied LOW	N/C	N/C	Tied LOW
[4]	**ExtOut[0]**, ETM	**ExtOutAck[0]**, ETM	**ExtIn[0]**, ETM	**ExtInAck[0]**, ETM

Table 2-18 CTI #1 connectivity (continued)

Bit index	Trigger in	Trigger in ack	Trigger out	Trigger out ack
[5]	**ExtOut[1]**, ETM	**ExtOutAck[1]**, ETM	**ExtIn[1]**, ETM	**ExtInAck[1]**, ETM
[6]	**TrigOut**, ETM	**TrigOutAck**, ETM	N/C	Tied LOW
[7]	Tied LOW	N/C	N/C	Tied LOW

The **CS_MAXPERF** input is an additional input to the IEC MAXPERF logic. This enables a user to replicate the **EDBGRQ** trigger output from CTI #1 onto this signal, thereby offering the option to have 100% IEM performance during debug or not. This is invasive to the system performance, but offers better debug functionality.

Note

Although this trigger is not acknowledged by **DBGACK** and is instant, the software must acknowledge it. This enables debug software to determine when to remove **MAXPERF** and enables a return to the last requested performance level.

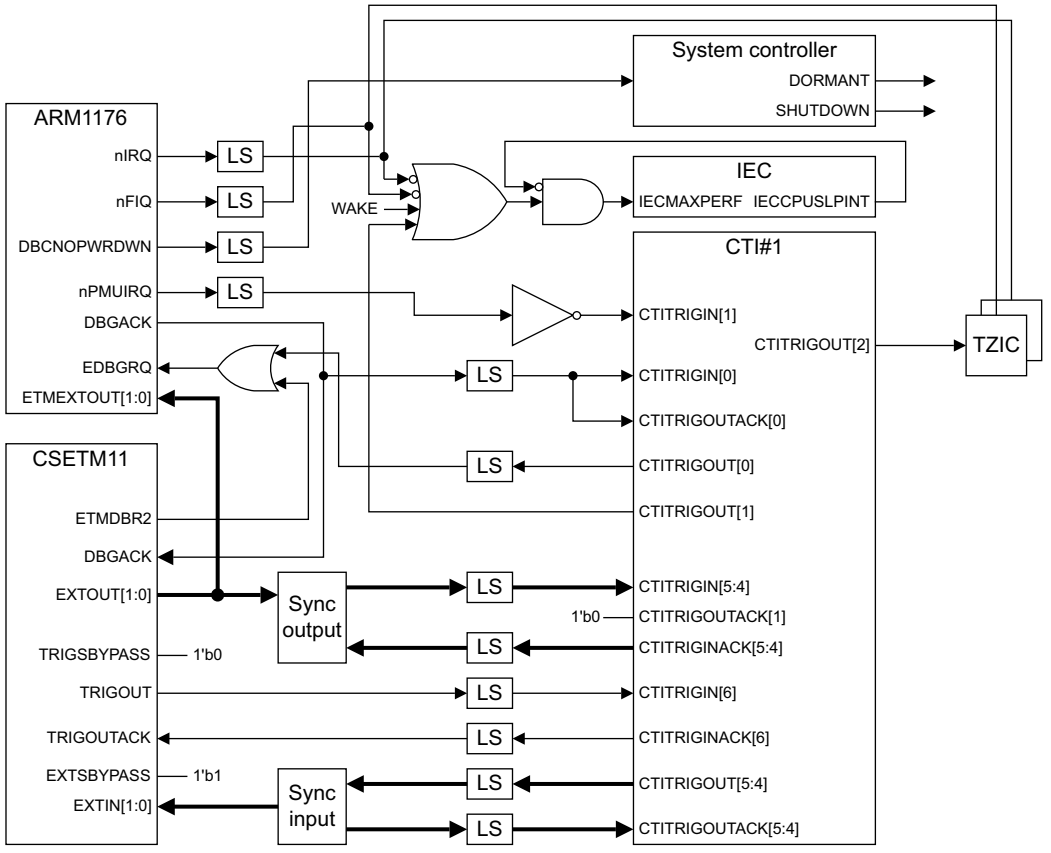


Figure 2-32 CoreSight trigger connectivity

2.5.6 DapRomDefs file

The CoreSight DAPROM has the following definitions:

```
// ----------------------------------------------------------------------------
// CoreSight Management Register Values
// ----------------------------------------------------------------------------

// PERIPHID0[7:0] = Part number [7:0] (0x11)
`define ROM_PERIPHID0_VAL       8'h11    // Peripheral ID0 reg

// PERIPHID1[7:4] = JEP106 ID Code [3:0] (0xB)
// PERIPHID1[3:0] = Part number [11:8] (0x1)
`define ROM_PERIPHID1_VAL       8'hB1    // Peripheral ID1 reg

// PERIPHID2[7:4] = Revision (0x0)
```



```
// PERIPHID2[3]   = flags validity of JEP106 code (1'b1)
// PERIPHID2[2:0] = JEP106 ID Code [6:4] (0x3)
`define ROM_PERIPHID2_VAL       8'h0B    // Peripheral ID2 reg

// PERIPHID3[7:4] = RevAnd bits
// PERIPHID3[3:0] = Customer modification
`define ROM_PERIPHID3_VAL       8'h00    // Peripheral ID3 reg

// PERIPHID4[7:4] = 4KB usage count
// PERIPHID4[3:0] = JEP106 Continuation code (0x4)
`define ROM_PERIPHID4_VAL       8'h04    // Peripheral ID4 reg

// PERIPHID5 = reserved
`define ROM_PERIPHID5_VAL       8'h00    // Peripheral ID5 reg

// PERIPHID6 = reserved
`define ROM_PERIPHID6_VAL       8'h00    // Peripheral ID6 reg

// PERIPHID7 = reserved
`define ROM_PERIPHID7_VAL       8'h00    // Peripheral ID7 reg
`define ROM_COMPONID0_VAL       8'h0D    // Component ID0 reg`define
ROM_COMPONID1_VAL       8'h10    // Component ID1 reg (ROM Class)`define
ROM_COMPONID2_VAL       8'h05    // Component ID2 reg
`define ROM_COMPONID3_VAL       8'hB1    // Component ID3 reg
```

2.6 Clock sources and domains

This section describes the clock sources and domains that the development chip requires. It contains the following subsections:

- *External clock sources*
- *Internally generated clock sources* on page 2-73
- *Exported clocks* on page 2-74.

2.6.1 External clock sources

Table 2-19 lists the external clock sources for the development chip.

Table 2-19 External clock sources

Clock	Maximum frequency	Reset	Description
PLLREFCLK	166MHz		Board level clock. AXI master and slave ports operate synchronously to this clock.
PLLREFCLK1	100MHz		IEM.
PLLREFCLK2	100MHz		IEM.
TIMCLK	5MHz	**PRESETn**	Clock for the SP804 timers, typically 1MHz. Internally synchronized to **PCLK**.
CLK1Hz	1Hz	**PRESETn**	Clock for the RTC. Internally synchronized to **PCLK**.
UARTCLK	30MHz	**nUARTRST**	UART clock, typically 24MHz.
CLCDCLK	60MHz	**nCLCLKRESET**	CLCDC LCD clock.
SSPCLK	20MHz	**nSSPRST**	SSP clock, typically 3.7MHz. The SSP bit rate is half this clock rate.
APC_REFCLK_C	50MHz		APC reference clock, typically 6-30MHz.
TCK	10MHz		JTAG-DP, DAP, and boundary scan TAP clock.
CONFIGINIT	50MHz	**CONFIGCLRn**	SoC static configuration capture clock.

2.6.2 Internally generated clock sources

Table 2-20 lists the internally generated clock sources.

Table 2-20 Internally generated clock sources

Clock	Maximum frequency	Reset	Description
CORECLK	332MHz	**nRESETIN**	ARM1176JZF core clock.
ACLK	166MHz	**nVSOCRST**	AXI subsystem & ATB clock.
ACLKn	166MHz		Used within DDR pad interface.
ACLKx2	332MHz		Used within DDR pad interface.
ACLKx2n	332MHz		Used within DDR pad interface.
ACLKEN		-	IEM(100%), AXI clock enable.
ACLKEXTSLV	83MHz	**nVSOCRST**	AXI master and slave port clock. **Note** 1:1 mode does not support 166MHz.
ACLKEXTEN		-	IEM(100%), AXI clock enable.
ACLKSMC	100MHz	**nVSOCRST**	SMC AXI clock. Clock rate limited by SMC design. Operates at 1:1 or 1:2 with **ACLK**.
ACLKSMCEN		-	
PCLK	83MHz	**nVSOCRST**	APB peripheral and CoreSight clock.
PCLKEN		-	
SMMEMCLK	60MHz	**nVSOCRST**	SSMC pad interface clock, 1:1, 1:2 or 1:3 with **ACLKSMC**.
nSMMEMCLK	60MHz	**nVSOCRST**	SSMC inverted pad interface clock.
SMMEMCLKDELAY	60MHz	-	Delayed version of **SMMEMCLK** used to drive out control and data signals for synchronous memories.
TRACECLKIN	166MHz	**nVSOCRST**	TPIU clock, double exported **TRACECLK**.
CLCDCLK	60MHz	**nVSOCRST**	CLCDC clock.

Table 2-20 Internally generated clock sources (continued)

Clock	Maximum frequency	Reset	Description
nCLCDCLK	60MHz	**nVSOCRST**	CLCDC inverted clock.
IECDPMCLKEN	30MHz	**nVSOCRST**	IEC dynamic performance monitor.
Clock enable at **PCLK**÷10			
IECDVSEMCLKEN	1MHz	**nVSOCRST**	IEC DVS emulation PWM frame advance.
Clock enable at **PCLK**÷10			
HPM Clock	83MHz	**nVCORERST**	Hardware performance monitor clock, runs in closed loop mode.

See figure 6-12 for supported clock ratios.

Note

PCLK clocks **WDOGCLK**, the APB watchdog peripheral.

2.6.3 Exported clocks

Table 2-21 lists the exported clocks.

Table 2-21 Exported clocks

Clock	Maximum frequency	Description
TRACECLK	**TRACECLKIN**√2	Trace clock, **ACLK** is used for **TRACECLKIN**
clk_out[0-3], **clk_out_n[0-3]**	-	MDDR feedback on SDRAM
SSMCFBCLKIN[3:0]	-	SMC memory fed-back clock inputs
SMCLK[3:0]	-	SMC memory clock outputs, exported
ACLK_EXTMST	-	Master port clock
ATCLKEXT	-	This is **ACLK**
SSPCLKOUT	20MHz	SSP serial clock output
APC_SCLK	-	PowerWise serial clock

Chapter 3
Programmer's Model

This chapter describes how to program the chip and contains the following sections:

3.1 About the programmer's model

This chapter describes how to program the registers in the ARM1176JZF Development Chip. It contains registers for the memory maps, SoC, power modes, IEC, system controller, and *Generic Interrupt Controller* (GIC).

3.2 Memory maps

This section describes the ARM1176JZF memory maps for each bus master you implement in the SoC. Peripherals are grouped into 4KB and 64KB blocks, or boundaries to assist software usage of small and large pages. Peripherals are grouped together functionally and by security, and reside at boundaries that are easy to decode.

Note

The ARM1176JZF core dormant and shutdown power modes for TrustZone require you to reinstate the memory remaps for the IEM subsystem when it exits these modes.

This section contains the following subsections:

- *Boot and remap options*
- *ARM1176JZF peripheral port memory map* on page 3-5
- *ARM1176JZF DMA port, CLCDC, and CoreSight memory map* on page 3-5.

3.2.1 Boot and remap options

Table 3-1 on page 3-4 lists shows the physical remap options you can configure at reset. You can capture from device pins and RDATA pins on the AXI master off-chip port into the SocConfig Register of the System Controller. See *SoC configuration* on page 3-7.

Note

Although you can use software to view and clear the remap state after reset, you cannot reset the remap state.

Table 3-1 Remap options

Remap setting	Description
Remap[1:0] = 00	No remaps are active. This is the normal run-time memory map.
Remap[1:0] = 01, TzRemap	TrustZone AXI ROM, 16KB, is aliased to address `0x0`. For simplicity of address decoding, it is aliased over the bottom 64MB of DMC bank 0. **Note** The AXI ROM is always visible at its higher non-aliased address location. This is Devchip specific, and is not the case in a true TZ SoC.
Remap[1:0] = 10, SmcRemap	SMC bank 7 is aliased to address `0x0`, over the bottom 64MB of DMC bank 0. **Note** SMC bank 7 is always visible at its higher, non-aliased, address location.
Remap[1:0] = 11, AxiRemap	When active, accesses to the bottom 64MB of memory space are directed to the AXI master port. There is no alias.

VINITHI is configured at reset to define the location of the ARM1176JZF exception vectors. **INITRAM** When HIGH, the ITCM, 8KB in this implementation, is mapped to address `0x0` for instructions and data. DTCM is not affected by this configuration option.

Note

When **VINITHI** is set HIGH, you cannot make the ITCM reside at `0xFFFF_0000`.

The ARM1176JZF peripheral port is mapped into read and write space using register CP15, c15.

Note

- The peripherals are visible to all AXI masters at the same address location. This eanbales use of the development chip as a peripherals chip by complex external masters.
- The **TzRemap**, **SMCRemap**, and **AXIRemap** signals are replicated for use by the ARM1176JZF core and the other AXI system masters. This enables you to reinstate remaps. This is especially important for TrustZone. The ARM1176JZF core must exit from shutdown or dormant mode without affecting other system masters.

3.2.2 ARM1176JZF peripheral port memory map

Figure 3-1 shows the ARM 1176JZF peripheral port memory map.

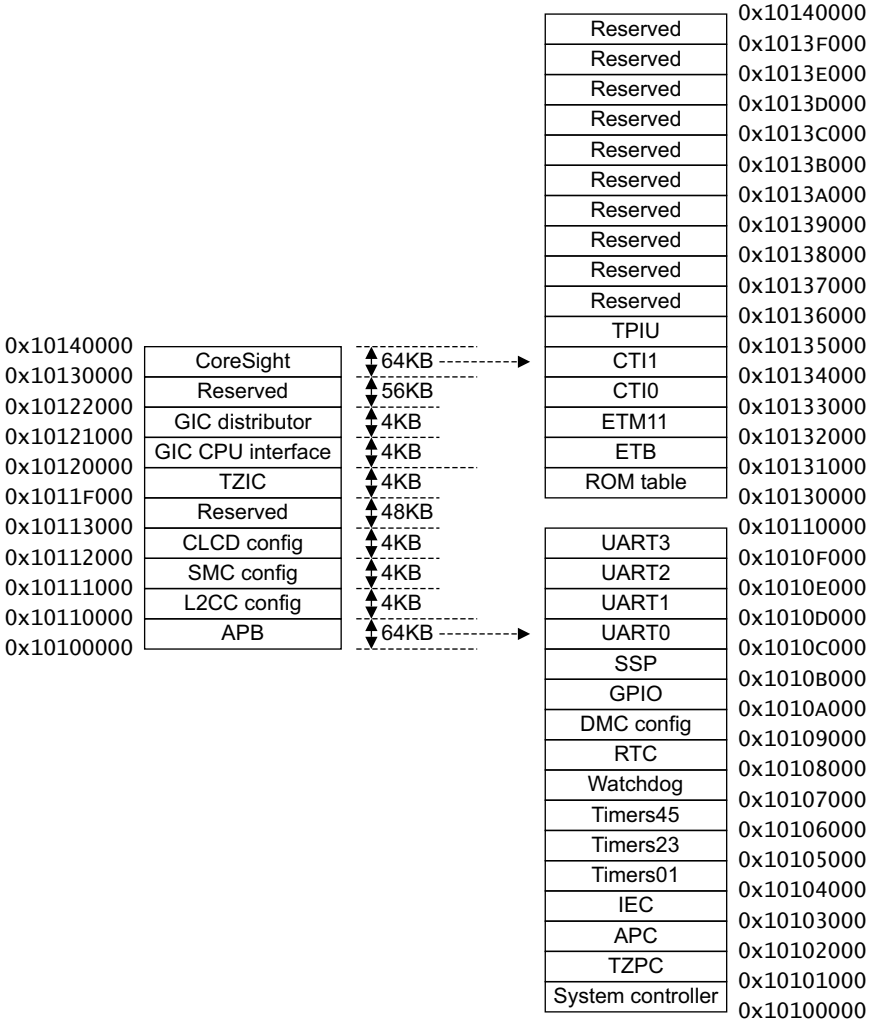


Figure 3-1 ARM 1176JZF peripheral port memory map

3.2.3 ARM1176JZF DMA port, CLCDC, and CoreSight memory map

Figure 3-2 on page 3-6 shows the ARM 1176JZF DMA port, CLCDC, and CoreSight memory map.

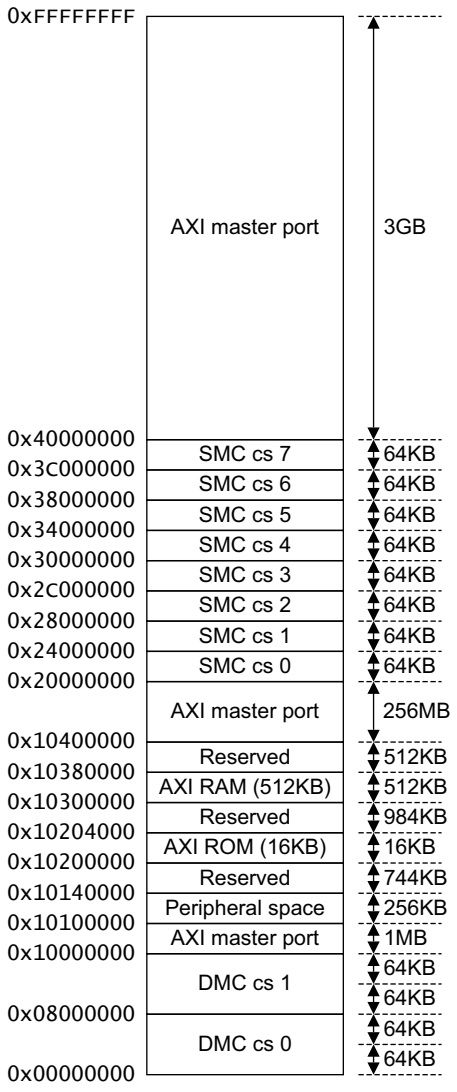


Figure 3-2 ARM 1176JZF DMA port, CLCDC, and CoreSight memory map

3.3 SoC configuration

This section describes the SoC configuration and contains the following subsections:

- *Configuring the chip*
- *Entering test mode* on page 3-8
- *Controlling debug* on page 3-11
- *Configuring security* on page 3-12.

3.3.1 Configuring the chip

The configuration block captures and holds static SoC configuration data. SoC configuration is captured from the AXI master **RDATA** bus input pads while the SoC is held at reset, sequenced by an off-chip device.

The configuration block has two main control inputs, **CONFIGINIT** and **nCONFIGCLR**. These inputs are completely independent of other clocks and resets in the design. They enable you to control the configuration block directly.

SoC configuration inputs are sampled on the rising edge of **CONFIGINIT**. You can use **nCONFIGCLR** to reset the configuration block to a default configuration. Figure 3-3 shows the reset and configuration capture sequence.

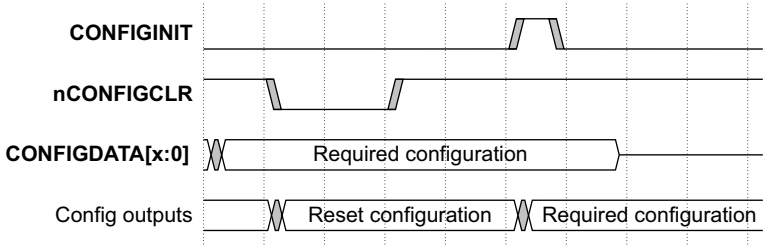


Figure 3-3 Configuration capture sequence

Registers in the system controller hold other system configuration parameters, such as IEC configuration. See *Configuration block signals* on page 3-18.

3.3.2 Entering test mode

Select test mode by setting the **TESTMODE** input HIGH. The configuration block is fully scan inserted to enable full control of the configuration inputs. Some configurations must however, be under user control or held static during test. For this purpose, the configuration block has test inputs that enable you to drive configurations directly. Table 3-2 lists the test mode configuration inputs.

Table 3-2 Test mode configuration inputs

Configuration signal	Description	RDATA[n] input sampled	Reset state	TESTMODE
CFGCOREID[3:0]	LS nibble of core **ID[7:0]**	[3:0]	4'h1	N/A
CFGREMAP[1:0]	Defines boot memory map: 00 = normal memory 01 = AXI ROM 10 = SMC 11 = AXI master port.	[5:4]	2'b01	N/A
CFGINITRAM	ITCM at `0x0` for data and instructions at reset	[6]	1'b0	N/A
CFGVINITHI	High exception vector location	[7]	1'b0	N/A
CFGBIGENDINIT	ARMv5 big endian behavior	[8]	1'b0	N/A
CFGUBITINIT	ARMv6 unaligned behavior	[9]	1'b1	N/A
CFGCP15SDISABLE	TZ write access disable	[10]	1'b0	N/A
CFGAXIRATIO	AXI: Core clock ratio: 0 = 2:1 1 = 4:1.	[11]	1'b0	1'b0

Table 3-2 Test mode configuration inputs (continued)

Configuration signal	Description	RDATA[n] input sampled	Reset state	TESTMODE
CFGAXIEXTRATIOSLV[2:0]	AXI external slave clock ratio: 000 = 1:1 001 = 2:1 010 = 3:1 011 = 4:1 100 = 5:1 101 = 6:1 110 = 7:1 111 = 8:1.	[14:12]	3'b111	TSTAXIEXTRATIOSLV
CFGAXIEXTRATIOMSTR[2:0]	AXI external master clock ratio: 000 = 1:1 001 = 2:1 010 = 3:1 011 = 4:1 100 = 5:1 101 = 6:1 110 = 7:1 111 = 8:1.	[17:15]	3'b111	TSTAXIEXTRATIOMSTR
CFGSMCAXIRATIO	SMC AXI sync down clock ratio: 0 = 1:1 1 = 2:1.	[18]	1'b0	1'b0
CFGSMCMEMRATIO[1:0]	SMC AXI to external memory clock ratio 00 = 1:1 01 = 2:1 10 = 3:1 11 = undefined.	[20:19]	2'b10	2'b00

Table 3-2 Test mode configuration inputs (continued)

Configuration signal	Description	RDATA[n] input sampled	Reset state	TESTMODE
CFGSMMWCS7[1:0]	SMC bank 7 data width 00 = 8-bit 01 = 16-bit 10 = 32-bit 11 = undefined.	[22:21]	2'b10	2'b00
CFGPLLREFCLK1		[23]	1'b0	TSTPLLREFCLK1
CFGPLLBYPASS		[24]	1'b0	TSTPLLBYPASS
CFGPLL1BYPASS		[25]	1'b0	TSTPLL1BYPASS
CFGPLL2BYPASS		[26]	1'b0	TSTPLL2BYPASS
CFGPLLFIXEDVCORANGE[a]	Selects VCO range: • HIGH > 133MHz • LOW < 133MHz.	[27]	1'b1	TSTPLLFIXEDVCORANGE
CFGPLLFIXEDM[a]	PLL feedback divider	[31:28]	4'h1	TSTPLLFIXEDM
CFGPLLFIXEDN[a]	PLL input divider	[34:32]	3'b001	TSTPLLFIXEDN
CFGPLLFIXEDBYPASSEN[a]	Bypasses VCO	[35]	1'b0	TSTPLLFIXEDBYPASSEN
CFGPLLFIXEDPLLEN[a]	PLL Enable	[36]	1'b1	TSTPLLFIXEDPLLEN
CFGPLLFIXEDDESKEW[a]	Selects external clock feedback path for on chip clock deskew	[37]	1'b1	TSTPLLFIXEDDESKEW

Table 3-2 Test mode configuration inputs (continued)

Configuration signal	Description	RDATA[n] input sampled	Reset state	TESTMODE
CFGIRQSOURCE	Selects the **nIRQ**, **nFIQ** source: 0 = external 1 = internal.	[38]	1'b1	-
CFGIEMSHUTDOWN	Selects the IEM subsystem to be powered down from reset: 0 = functional 1 = shutdown.	[39]	1'b0	-
CFGCLKSTOPSBWFI	Selects whether the Clock is stopped to the L2CC and ETM during CPU STANDBYWFI State: 0 = clock is left free running 1 = clock is stopped.	[40]	1'b1	-

a. You must disable the internal feedback and input dividers to use the PLL to de-skew the off-chip AXI interface to the on-chip SoC clock. You must also enable the PLL, not bypass the VCO, enable de-skew, and set the VCO range correctly to use PLL de-skew mode.

3.3.3 Controlling debug

The following signals are brought directly out to pins to enable debug of the development chip:

- **SPIDEN**
- **SPNIDEN**
- **DBGEN**, considered as **IDEN**
- **NIDEN**.

They are synchronized to:

- the core clock domain for use within the IEM subsystem
- the AXI clock for use within the CoreSight subsystem.

Note

These signals are driven from an external FPGA that enables debug control. An authentication module might be developed as part of the CoreSight architecture.

3.3.4 Configuring security

You configure security for the SoC under control of the TZPC registers:

- *TZPCDECPROT0 Register*
- *TZPCDECPROT1 Register* on page 3-13
- *TZPCDECPROT2 Register* on page 3-15
- **R0SIZE**, see *Internal RAM security* on page 3-17.

The SoC is fully secure immediately following a hard reset. Peripherals that you can configure to be secure are secure at reset. Table 3-3 on page 3-13, Table 3-4 on page 3-14 and Table 3-5 on page 3-16 show bit assignments.

TZPCDECPROT0 Register

Figure 3-4 shows the bit assignments for this register.

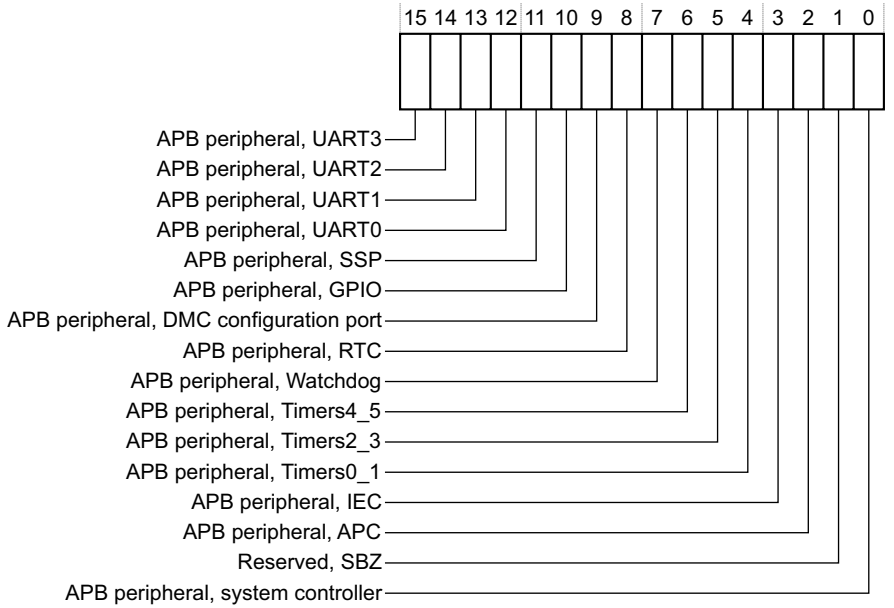


Figure 3-4 TZPCDECPROT0 Register bit assignments

Table 3-3 lists the bit assignments for this register.

Table 3-3 TZPCDECPROT0 Register bit assignments

Bits	Reset	Function
[15]	0 = secure	APB peripheral, UART3
[14]	0 = secure	APB peripheral, UART2
[13]	0 = secure	APB peripheral, UART1
[12]	0 = secure	APB peripheral, UART0
[11]	0 = secure	APB peripheral, SSP
[10]	0 = secure	APB peripheral, GPIO
[9]	0 = secure	APB peripheral, DMC configuration port
[8]	0 = secure	APB peripheral, RTC
[7]	0 = secure	APB peripheral, Watchdog
[6]	0 = secure	APB peripheral, Timers4_5
[5]	0 = secure	APB peripheral, Timers2_3
[4]	0 = secure	APB peripheral, Timers0_1
[3]	0 = secure	APB peripheral, IEC
[2]	0 = secure	APB Peripheral, APC
[1]	0	Not used, SBZ
[0]	0 = secure	APB peripheral, system controller

TZPCDECPROT1 Register

Figure 3-5 on page 3-14 shows the bit assignments for this register.

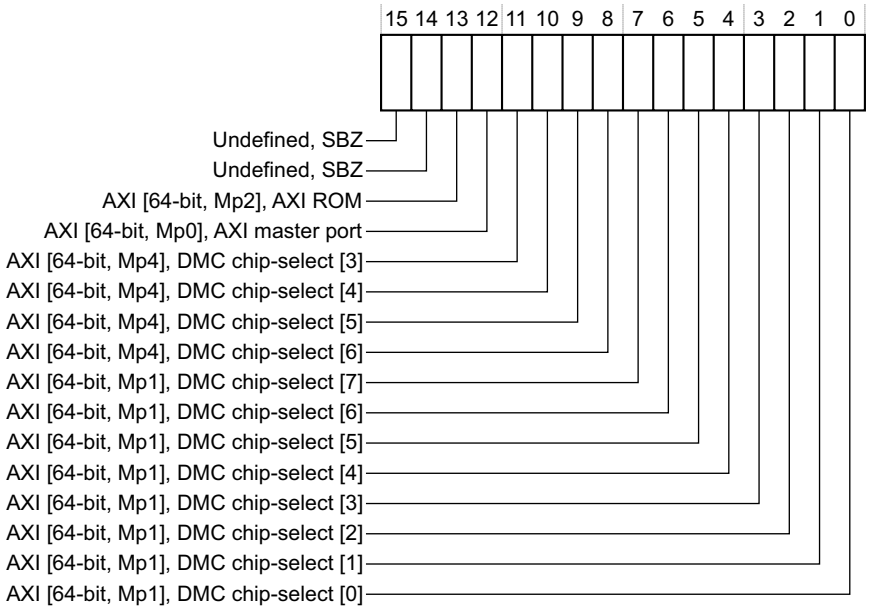


Figure 3-5 TZPCDECPROT1 Register bit assignments

Table 3-4 lists the bit assignments for this register.

Table 3-4 TZPCDECPROT1 Register bit assignments

Bits	Reset	Function
[15]	0	Undefined, SBZ
[14]	0	Undefined, SBZ
[13]	0 = secure	AXI [64-bit, Mp2], AXI ROM
[12]	0 = secure	AXI [64-bit, Mp0], AXI master port
[11]	0 = secure	AXI [64-bit, Mp4], DMC chip-select [3]
[10]	0 = secure	AXI [64-bit, Mp4], DMC chip-select [4]
[9]	0 = secure	AXI [64-bit, Mp4], DMC chip-select [5]
[8]	0 = secure	AXI [64-bit, Mp4], DMC chip-select [6]
[7]	0 = secure	AXI [64-bit, Mp1], SMC chip-select [7][a]
[6]	0 = non-secure	AXI [64-bit, Mp1], SMC chip-select [6]

Table 3-4 TZPCDECPROT1 Register bit assignments (continued)

Bits	Reset	Function
[5]	0 = non-secure	AXI [64-bit, Mp1], SMC chip-select [5]
[4]	0 = non-secure	AXI [64-bit, Mp1], SMC chip-select [4]
[3]	0 = non-secure	AXI [64-bit, Mp1], SMC chip-select [3]
[2]	0 = non-secure	AXI [64-bit, Mp1], SMC chip-select [2]
[1]	0 = non-secure	AXI [64-bit, Mp1], SMC chip-select [1]
[0]	0 = non-secure	AXI [64-bit, Mp1], SMC chip-select [0]

a. SMC chip-select 7 is secure by default. This enables the SoC to boot from external Flash as the secure boot region. See *Boot and remap options* on page 3-3. If you do not require this option, for example when booting securely from internal ROM, and a non-secure image from SMC, you must set the bit to enable a non-secure image load from this Flash region.

TZPCDECPROT2 Register

Figure 3-6 shows the bit assignments for this register.

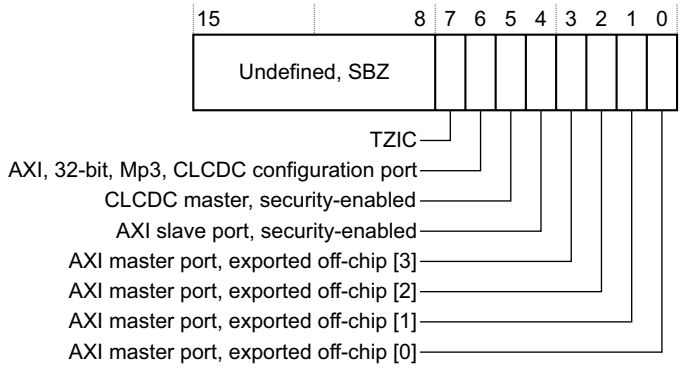


Figure 3-6 TZPCDECPROT2 Register bit assignments

Table 3-5 lists the bit assignments for this register

Table 3-5 TZPCDECPROT2 Register bit assignments

Bits	Reset	Function
[15:8]	0	Undefined, SBZ
[7]	0 = secure	TZIC
[6]	0 = secure	AXI, 32-bit, Mp3, CLCDC configuration port
[5]	0 = secure	CLCDC master, security-enabled
[4]	0 = secure	AXI slave port, security-enabled
[3]	0 = secure	AXI master port, exported off-chip [3]
[2]	0 = secure	AXI master port, exported off-chip [2]
[1]	0 = secure	AXI master port, exported off-chip [1]
[0]	0 = secure	AXI master port, exported off-chip [0]

CLCDC master security override

The CLCDC master **HPROT[3:0]** signals are hard-wired to 4'b0001. This implements simple non-privileged data access. The controller has no secure access ability. To enable CLCDC access to secure memory, configure **ARPROT[1]** and **AWPROT[1]** under the control of the TZPC signal **TZPCDECPROT2[5]**.

You can also configure the CLCDC configuration port independently as secure or non-secure.

AXI slave port security override

Although the AXI slave port **ARPROT[1]** and **AWPROT[1]** signals are bonded out, you can override them to disable secure accesses from external masters using the **TZPCDECPROT2[4]** TZPC signal.

You perform the override between the SyncUpAxi component master port and the CAI 64-bit block. Table 3-6 lists a truth table of the interaction between **AxPROT[1]** and **TZPCDECPROT2[4]**.

Table 3-6 AXI slave port security override

AxPROT[1] input signal	TZPCDECPROT2[4]	AxPROT[1] internal signal
0: secure access	1: secure access blocked[a]	1
1: non-secure access	1	1
0: secure access	0: secure access permitted	0
1: non-secure access	0	1

a. The logic to perform this function is an OR of the **AxPROT[1]** signal with the **TZPCDECPROT2[4]** signal.

Internal RAM security

The **R0SIZE** output from the TZPC is connected directly to the TrustZone Memory Adapter for the on-chip 512KB RAM. This defines the secure RAM region in 4KB increments from the RAM base address. At reset, the RAM is configured as fully secure.

DMC security

You can map DDR devices connected to the Dynamic Memory Controller in memory between 0x0000_0000 and 0x0FFF_FFFF, ignoring remaps. For security purposes, this 256MB range is split into four 64MB regions. You can configure each region as secure or non-secure using the **TZPCDECPROT1[11:8]** TZPC signals. This is a simple TrustZone enabling feature for development purposes.

SMC security

You can map static memory devices connected to the *Static Memory Controller* (SMC) in memory between 0x2000_0000 and 0x3FFF_FFFF. For security purposes this 512MB range is split into eight 64MB regions. You can configure each region, that maps directly onto an SMC chip select, as secure or non-secure using the **TZPCDECPROT1[7:0]** TZPC signals. This is a TrustZone enabling feature for development purposes.

3.3.5 Configuration block signals

Table 3-7 lists the configuration block signals.

Table 3-7 Configuration block signals

Signal name	Type
CFGAXIEXTRATIOMSTR[2:0]	Output
CFGAXIEXTRATIOSLV[2:0]	Output
CFGAXIRATIO	Output
CFGBIGENDINIT	Output
CFGBYPASS	Input
CFGCP15SDISABLE	Output
CFGINITRAM	Output
CFGIRQSOURCE	Output
CFGPLL1BYPASS	Output
CFGPLL2BYPASS	Output
CFGPLLBYPASS	Output
CFGPLLFIXEDBYPASSEN	Output
CFGPLLFIXEDDESKEW	Output
CFGPLLFIXEDM[3:0]	Output
CFGPLLFIXEDN[2:0]	Output
CFGPLLFIXEDPLLEN	Output
CFGPLLFIXEDVCORANGE	Output
CFGPLLREFCLK1	Output
CFGSMCAXIRATIO	Output
CFGSMCMEMRATIO[1:0]	Output
CFGSMMWCS7[1:0]	Output
CFGUBITINIT	Output
CFGVINITHI	Output

Table 3-7 Configuration block signals (continued)

Signal name	Type
CONFIGCLRn	Input
CONFIGDATA[41:0]	Input
CONFIGINIT	Input
SOCCONFIG[41:0]	Output
TSTPLL1BYPASS	Input
TSTPLL2BYPASS	Input
TSTPLLBYPASS	Input
TSTPLLFIXEDBYPASSEN	Input
TSTPLLFIXEDDESKEW	Input
TSTPLLFIXEDM[3:0]	Input
TSTPLLFIXEDN[2:0]	Input
TSTPLLFIXEDPLLEN	Input
TSTPLLFIXEDVCORANGE	Input
TSTPLLREFCLK1	Input

3.4 Controlling the power modes

This section describes how to control power modes and contains the following subsections:

- *StandbyWFI mode*
- *Dormant and shutdown mode* on page 3-21.

3.4.1 StandbyWFI mode

Perform the following steps to safely enter StandbyWFI mode,:

1. Ensure the SHUTDOWNNXT, DORMANTNXT, and USERMAXPERF flags are cleared in the System Controller Control register.

   Note

   It is not possible to return the system from StandbyWFI by asserting the user maximum performance **WAKEUP** input.

2. Ensure the IEC Max Perf Enable bit[2] in the IEC DPC Control Register is set HIGH.

   This enables system wake up from StandbyWFI when the system controller asserts IECMAXPERF.

3. Write 0% performance to the IEC.

   This ensures that the dynamic performance monitors are frozen during StandbyWFI.

   The IEC then issues its sleep interrupt by asserting IECCPUSLPINT. It also requests the minimum IEM operating performance level to save energy while waiting for the ARM1176-JZF to enter StandbyWFI.

4. Issue a Cache Sync request to the L2CC.

5. Issue a CP15 WFI Instruction.

   When the IEC detects that the ARM1176JZF core has safely entered STANDBYWFI, a 0% performance request is issued.

   Although the IEC also requests 0 voltage, this request is blocked by the System Controller to prevent regulator shutdown and voltage is held at the current minimum level.

   If the CFGCLKSTOPSBWFI configuration bit is set, the System Controller clears ETML2CCCLKEN when CACTIVE from the L2CC goes LOW.

   If the CFGCLKSTOPSBWFI configuration bit is not set, ETML2CCCLKEN remains HIGH, therefore the clock to both the ETM11 and L2CC remain active.

The following procedure occurs when exiting StandbyWFI mode:

- An interrupt or debug request causes the ARM1176JZF core to clear STANDBYWFI
- ETML2CCCLKEN is asserted
- IEC requests 100% performance
- The system returns to run mode.

Note

During StandbyWFI, both **FREECLKIN** and **CLKIN** to the ARM1176JZF core remain active.

3.4.2 Dormant and shutdown mode

Perform the following steps to enter shutdown or dormant modes:

1. Set the SHUTDOWNNXT or DORMANTNXT flag in the System Controller Control Register.
2. Ensure the USERMAXPERF bit is cleared in the System Controller Control Register.

   Note

   If you want the system to return from shutdown or dormant mode because of a rising edge on the user maximum performance input, **WAKEUP**, you must enable it. Set the USERMAXPERFEN bit in the System Controller Control register.

   The system software, not IEM driver, requests a 0% performance level by writing to the IEC.

   The IEC asserts an IECCPUSLPINT interrupt.

   When the ARM1176JZF core receives the IEC IECCPUSLPINT interrupt, the ARM1176JZF core must:
   - disable all interrupts
   - flush write buffers
   - enable DMA channels to drain.
   - handshake with the ETM11 to drain the trace buffers.
3. Save the level 1 and level 2 states as appropriate.

The APC_PWRDN_EN and ACP_VDD_UD bits are set to 1 in the APC_CONTROL register.

Note

It does not matter whether the APC1 is in open or closed loop when entering or exiting shutdown or dormant modes.

4. Enable system wake up from shutdown or dormant mode by assertion of IECMAXPERF by the System controller. Ensure the IEC Max Perf Enable bit[2] in the IEC DPC Control Register is set to 1. This results in:
   - The **IECCPUSLPINT** interrupt is cleared.
   - The CP15 *WaitForInterrupt* (WFI) instruction is issued.
   - When the ARM1176JZF core successfully enters standby wait for interrupt by asserting STANDBYWFI, and the L2CC has ceased activity by clearing CACTIVE, the System Controller applies reset to the IEM Subsystem. It also sets the APC target performance index APCTGTIDX to zero.
   - When the APC1 detects a target performance request of zero, it asserts the **apc_clamp_req** signal, and if in closed loop mode, the loop filter is disabled.
   - When the System Controller detects **apc_clamp_req**, it enables the input and output clamps around the IEM subsystem.
   - When the IEM subsystem is safely clamped, the System Controller asserts **apc_clamp_ack** to the APC1.
   - Finally, the APC1 issues the sleep command to the regulators via the Power Wise Interface and the IEM subsystem voltage is driven to zero. The DORMANT and SHUTDOWN SoC outputs control the off-chip power supply settings.
   - The rest of the SoC continues to be powered and clocked as normal.

Returning from shutdown or dormant mode

A maximum performance condition must occur to exit shutdown mode. Interrupts or a rising edge on the external user maximum performance request input **WAKEUPM** cause maximum performance conditions.

The maximum performance condition causes IECMAXPERF into to the IEC to be asserted. The IEC then requests maximum voltage and frequency to the System Controller:

1. When the System Controller detects that IECTGTDCGIDX and IECTGTDVCIDX are set to maximum, it sets the proxy voltage to the APC1 APCTGTIDX to the maximum level.
2. The APC1 issues the WAKEUP command to the regulators through the Power Wise Interface.
3. APC1 then polls COREVDDOK status of the regulator through the Power Wise Interface after the internal wake-up timer has timed-out.
4. When APC1 confirms COREVDDOK status is OK, **apc_clamp_req** is de-asserted.
5. APC1 then updates **apc_current_index** to the requested value on APCTGTIDX.
6. When the System Controller detects **apc_clamp_req** de-asserted, itreleases the input and output clamps around the IEM subsystem.
7. DCG switches in the maximum performance clock.

   Note

   Prior to this switch of IEM subsystem clock, if the APC1 is in closed loop mode, the HPM clock must also be started.

8. Remap is restored to power-on-reset captured configuration settings.
9. Reset is then released by the DCG. This includes the ARM1176JZF core, Level 2 Cache, and ETM11.

   The system boot code must read the reset status flags in the system controller to determine the cause of the last reset.

   Warning

   If the maximum performance condition is removed from the IEC, it reverts to the currently programmed performance level, in this case, 0%.

   If you do not require this action, you must program a new performance level. ARM recomends that the system boot code programs maximum performance before handing over control to the IEM driver following a power down.

10. Clear the maximum performance condition. Either an interrupt or a rising edge on the **WAKEUP** input causes the maximum performance condition. If a rising edge on **WAKEUP** caused the maximum performance request, you can clear it by writing a zero to the USERMAXPERF bit [4] in the System Controller Control Register.

    USERMAXPERF is also cleared if you disable this feature by clearing **USERMAXPERFEN** bit [3] in the System Controller Control Register.

11. The system is restored to pre-powerdown state.

3.5 Configuring the IEC

This section describes how to configure the IEC and contains the following subsections:

- *Dynamic Clock Generator (DCG)*
- *DCG fractional performance level mapping configuration* on page 3-28
- *Dynamic Voltage Controller (DVC) index mapping* on page 3-28
- *Processor frequency configuration* on page 3-28
- *Dynamic Performance Monitor (DPM) frequency configuration* on page 3-29
- *Target index and current index* on page 3-29.

3.5.1 Dynamic Clock Generator (DCG)

The DCG generates all:

- clocks
- resets
- power and clamp signals
- sequencing for the chip.

The DCG consists of three PLLs. One PLL has a fixed frequency at reset, and the other two are variable. The fixed PLL clocks the processor at the 100% performance level, and the variable PLLs carry out all sub 100% performance levels. The fixed PLL is also used for the AXI, APB and other fixed system clocks.

Two variable PLLs are required to enable the most efficient method of switching between two sub 100% performance levels.

For example, if the performance level is currently set at 95%, the system is running on one of the variable PLLs. A request is then issued to drop to 90%. If only one variable PLL is being used, you must re-configure the PLL currently driving the system at 95% to provide the 90% clock.

The system must therefore switch to the fixed PLL while waiting for the variable PLL to lock at the new 90% performance frequency. However, the current voltage might not be sufficient to support the fixed PLL 100% performance frequency. You must therefore use a divided down version of the fixed PLL clock.

This is not desirable, because the fastest divided down fixed PLL clock only delivers 50% performance, and produces a considerable drop-off in performance. Therefore, by using two variable PLLs, the system can always run at the current performance on one PLL, while the other PLL locks to the new performance level.

Figure 3-7 on page 3-26 shows the first part of the DCG block diagram.

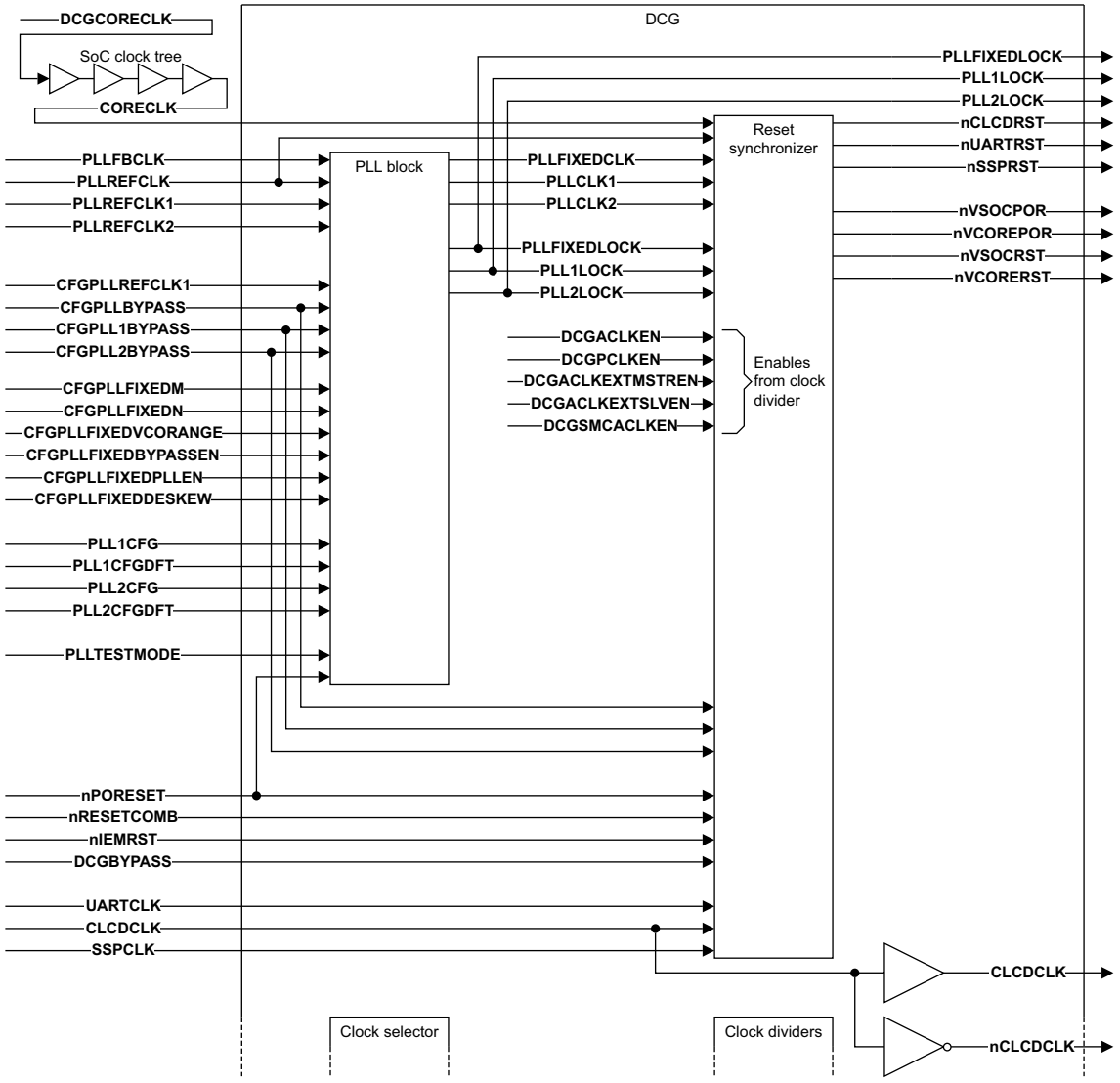


Figure 3-7 DCG block diagram part 1

Figure 3-8 on page 3-27 shows the second part of the DCG block diagram.

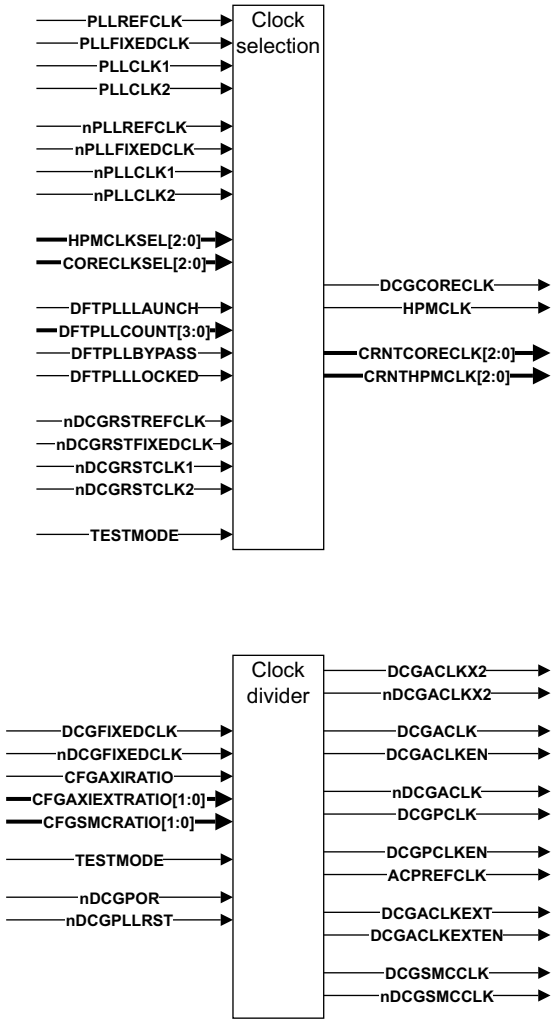


Figure 3-8 DCG block diagram part 2

3.5.2 DCG fractional performance level mapping configuration

Table 3-8 lists the performance levels for the chip.

Table 3-8 IEC performance levels

Performance level%	Performance level	Octal value	Index level
100%	28-32	7	8
83%	23-27	6	7
67%	17-22	5	6
50%	1-16	4	5
0%	0	0	1 - 4

The IEC configuration input **IECCFGDCGIDXMAP** is driven to the following octal value by the system controller at reset.

IECCFGDCGIDXMAP = `96'o77777_66666_555555_4444444444444444`

3.5.3 Dynamic Voltage Controller (DVC) index mapping

The 24-bit wide **IECCFGDVCIDXMAP** configuration bus provides the mapping between DCG index levels and the minimum voltage level required.

For the chip, each DCG index level is mapped to a unique voltage level. This is a requirement of the APC1 that performs its own mapping between performance level and voltage.

The **IECCFGDVCIDXMAP** configuration bus is therefore permanently tied to the following value:

IECCFGDVCIDXMAP = `24'o76543210`

3.5.4 Processor frequency configuration

This configuration specifies the maximum operating frequency of the processor in kHz. For the chip, this is 300000KHz. **IECCFGFREQCPU** is therefore driven to the following value by the system controller at reset:

IECCFGFREQCPU = `24'd300_000` = `24'h04_93E0`

3.5.5 Dynamic Performance Monitor (DPM) frequency configuration

This configuration specifies the rate that the DPMs are clocked. **IECDPMCLKEN** always operates at **PCLK** divided by 10.

Assuming that **PCLK** = Maximum core clock√4 = 75MHz.

Therefore, **IECCFGFREQDPM** is 75MHz√10 = 7.5MHz, and driven to the following value by the system controller at reset:

IECCFGFREQDPM = `24'd007_500 = 24'h00_1D4C`

3.5.6 Target index and current index

See the *Intelligent Energy Controller Technical Reference Manual* for information on target index and current index.

3.6 System controller

This section describes the system controller and contains the following subsections:

- *Register summary*
- *Register descriptions* on page 3-31.

3.6.1 Register summary

Table 3-9 lists the system controller registers.

Table 3-9 System controller register summary

Register name	Base offset	Type	Reset	Description
CoreID	0x000	R	Variable	Core identification register
SoCConfig1	0x004	RW	Variable	SoC configuration register
SoCConfig2	0x008	R	Variable	SoC configuration register
SysControl	0x00C	RW	Variable	System controller control register
SysStatus	0x010	R	Variable	System status register
DCGIDXMAP0	0x014	RW	0x00924924	IEC configuration index mapping configuration level 1-8
DCGIDXMAP1	0x018	RW	0x00924924	IEC configuration index mapping configuration level 9-16
DCGIDXMAP2	0x01C	RW	0x00DADB6D	IEC configuration index mapping configuration level 17-24
DCGIDXMAP3	0x020	RW	0x00FFFFB6	IEC configuration index mapping configuration level 25-32
DCGPERFMAP0	0x024	RW	0x00000000	Performance level mapping index 1-4
DCGPERFMAP1	0x028	RW	0x806A5640	Performance level mapping index 5-8
DVCIDXMAP	0x02C	RW	0xFAC688	IEC configuration DVC index map levels 1-8
CFGFREQCPU	0x030	RW	0x0004E200	Maximum CPU clock speed, 320MHz
CFGFREQDPM	0x034	RW	0x00001D4C	DPM frequency
PLLCFGPERF1	0x038	RW	0x00000000	PLL configuration setting for performance level 1
PLLCFGPERF2	0x03C	RW	0x00000000	PLL configuration setting for performance level 2

Table 3-9 System controller register summary (continued)

Register name	Base offset	Type	Reset	Description
PLLCFGPERF3	0x040	RW	0x00000000	PLL configuration setting for performance level 3.
PLLCFGPERF4	0x044	RW	0x00000000	PLL configuration setting for performance level 4.
PLLCFGPERF5	0x048	RW	0x00000294	PLL configuration setting for performance level 5. The default is 50% of maximum performance.
PLLCFGPERF6	0x04C	RW	0x000002B0	PLL Configuration setting for performance level 6. The default is 67% of maximum performance.
PLLCFGPERF7	0x050	RW	0x000002AD	PLL Configuration setting for performance level 7. The default is 81% of maximum performance.
DCGSTATUS	0x054	R	32'h11223344	Current DCG system status.
AXIPRIORITY	0x060	RW	0x00543210	64-bit AXI arbitration priority.
SMCEXTACMONID	0x064	RW	0x812 ID0 = 010010 ID1 = 001000	SMC exclusive access monitor ID value.
DLLCalibrate	0x068	RW	0x34	DLL update period, in **mclk**÷32 increments.
SCPeriphID0	0xFE0	R	0x10	Peripheral ID part number LOW 8 bits.
SCPeriphID1	0xFE4	R	0x18	Peripheral ID part number HIGH 8 bits.
SCPeriphID2	0xFE8	R	0x04	Peripheral ID designer.
SCPeriphID3	0xFEC	R	0x00	Peripheral ID configuration.
SCPCellID0	0xFF0	R	0x0D	Standard PrimeCell ID coding.
SCPCellID1	0xFF4	R	0xF0	Standard PrimeCell ID coding.
SCPCellID2	0xFF8	R	0x05	Standard PrimeCell ID coding.
SCPCellID3	0xFFC	R	0xB1	Standard PrimeCell ID coding.

3.6.2 Register descriptions

This section describes all of the ARM1176 development chip registers. It contains the following subsections:

- *Core Identification Register* on page 3-32
- *SoC Configuration Registers* on page 3-32
- *System Control Register* on page 3-35

- *System Status Register* on page 3-37
- *IEC Configuration Index Mapping Configuration Registers* on page 3-38.

Core Identification Register

The CoreID register holds the Core Identification number for the ARM1176JZF core. The development chip has only one core, but this ID value enables it to either be cascaded or to form part of a multi-core system.

CoreID[7:0] is a read-only value. **CoreID[7:4]** is hard-wired to `0x0` and **CoreID[3:0]** is configurable at reset as part of the static SoC configuration. Figure 3-9 shows the bit assignments for this register.

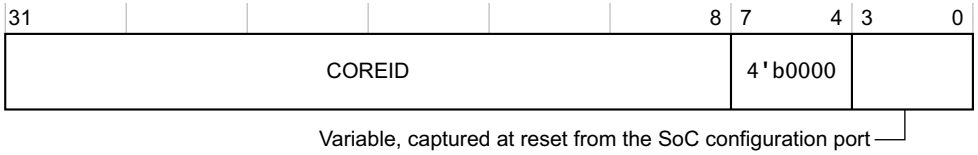


Figure 3-9 CoreID Register bit assignments

Table 3-10 lists the bit assignments for this register.

Table 3-10 CoreID Register bit assignments

Bits	Offset	Name	Type	Function
[31:8]	`0x000`	COREID	R	Undefined, SBZ
[7:4]				`4'b0000`
[3:0]				Variable, captured at reset from the SoC configuration port

SoC Configuration Registers

The SoCConfig registers 1 and 2 hold the SoC configuration, captured during reset from the SoC pins.

SoCConfig1 Register

Figure 3-10 on page 3-33 shows the bit assignments for this register.

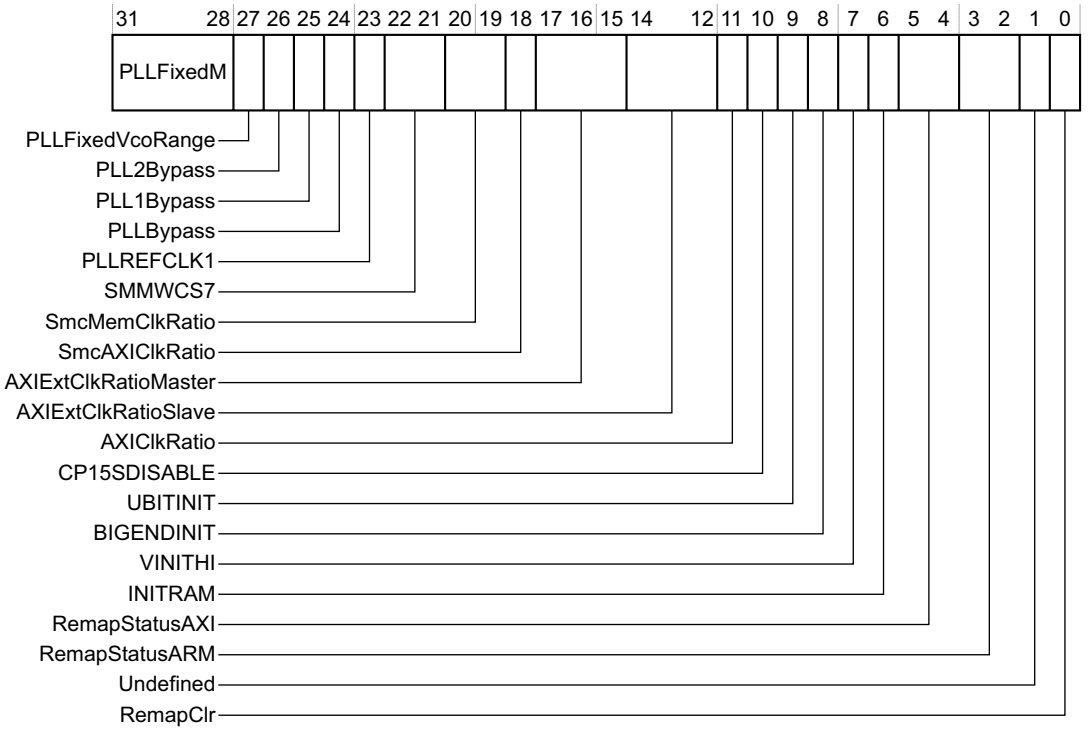


Figure 3-10 SoCConfig1 Register bit assignments

Table 3-11 lists the bit assignments for this register.

Table 3-11 SoCConfig1 Register bit assignments

Bits	Offset	Name	Type	Function
	0x004	SoCConfig1	-	
[31:28]		PLLFixedM	R	PLL fixed internal feedback divider
[27]		PLLFixedVcoRange	R	PLL fixed VCO range setting
[26]		PLL2Bypass	R	PLL 2 bypass
[25]		PLL1Bypass	R	PLL 1 bypass
[24]		PLLBypass	R	PLL fixed bypass
[23]		PLLREFCLK1	R	PLL reference clock select
[22:21]		SMMWCS7	R	SMC bank7 data width

Table 3-11 SoCConfig1 Register bit assignments (continued)

Bits	Offset	Name	Type	Function
[20:19]		SmcMemClkRatio	R	SMC memory clock ratio
[18]		SmcAXIClkRatio	R	SMC AXI sync down clock ratio
[17:15]		AXIExtClkRatioMaster	R	AXI external master synchronous clock ratio
[14:12]		AXIExtClkRatioSlave	R	AXI external slave synchronous clock ratio
[11]		AXIClkRatio	R	AXI core to AXI synchronous clock ratio
[10]		CP15SDISABLE	R	TZ write access disable
[9]		UBITINIT	R	ARMv6 unaligned behavior
[8]		BIGENDINIT	R	ARMv5 big endian behavior
[7]		VINITHI	R	High exception vector location
[6]		INITRAM	R	ITCM at 0x0
[5:4]		RemapStatusAXI	R	AXI remap status
[3:2]		RemapStatusARM	R	ARM I & RW remap status
[1]		-	-	Undefined, SBZ
[0]		RemapClr	W	Remap clear

SoCConfig2 Register

Figure 3-11 shows the bit assignments for this register.

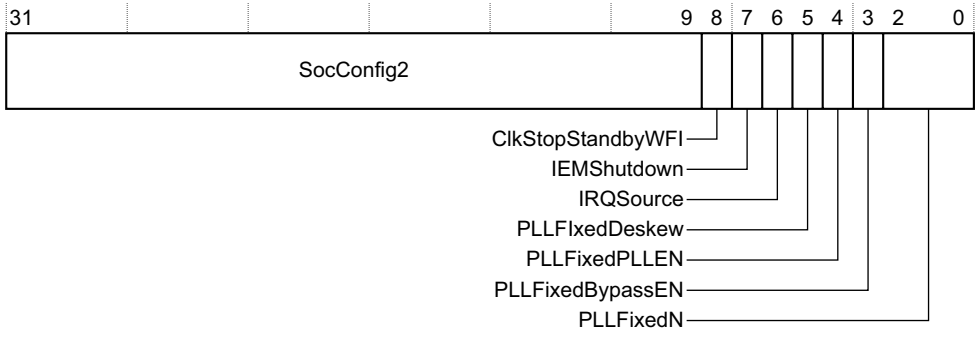


Figure 3-11 SocConfig2 Register bit assignments

Table 3-12 lists the bit assignments for this register.

Table 3-12 SoCConfig2 Register bit assignments

Bits	Offset	Name	Type	Function
[31:9]	0x008	SocConfig2	-	Undefined, SBZ
[8]		ClkStopStandbyWFI	R	When set indicates that the clock to the ETM11 and L2CC is stopped during CPU standbyWFI mode.
[7]		IEMShutdown	R	IEM subsystem in permanent shutdown mode.
[6]		IRQSource	R	Selects nIRQ, nFIQ source: 0 = external 1 = internal.
[5]		PLLFIxedDeskew	R	PLL fixed de-skew enable.
[4]		PLLFixedPLLEN	R	PLL fixed enable.
[3]		PLLFixedBypassEN	R	PLL fixed VCO bypass enable.
[2:0]		PLLFixedN	R	PLL fixed internal input divider.

System Control Register

Figure 3-12 shows the bit assignments for this register.

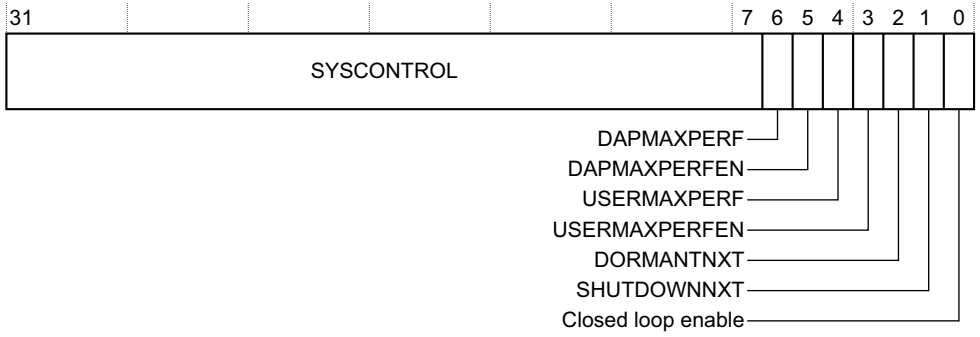


Figure 3-12 System Control Register bit assignments

Table 3-13 lists the bit assignments for this register.

Table 3-13 System Control Register bit assignments

Bits	Offset	Name	Type	Function
[31:7]	0x00C	SYSCONTROL	RW	Undefined, SBZ.
[6]		DAPMAXPERF	RW	DAP maximum performance request.
[5]		DAPMAXPERFEN	RW	DAP maximum performance request enable.
[4]		USERMAXPERF	RW	Indicates that an external maximum performance request occurred. A rising edge on the SoC **WAKEUP** input sets this bit, and writing a zero clears it. You cannot set it.
[3]		USERMAXPERFEN	RW	Bit set enables user maximum performance from **WAKEUP** input, default. Bit clear disables user maximum performance from **WAKEUP** input.
[2]		DORMANTNXT	RW	Bit set indicates that dormant mode is entered on next power down.
[1]		SHUTDOWNNXT	RW	Bit set indicates that shutdown mode is entered on next power down.
[0]		Closed loop enable	RW	Bit set indicates IEM closed loop mode. Bit clear indicates IEM open loop mode, default.

USERMAXPERF holds the IEM subsystem at maximum performance after a rising edge on the **WAKEUP** input. After a user maximum performance request, software can program a new IEC performance level before it clears USERMAXPERF. If a new IEC performance level is not programmed before USERMAXPERF is cleared, the IEC selects the previous performance level prior to **WAKEUP**.

It is important that when returning from shutdown, dormant or standbyWFI power down modes, to write a >0% performance level to the IEC before clearing USERMAXPERF. This is to prevent the IEC trying to re-enter 0% by asserting its sleep interrupt when USERMACPERF is cleared.

Note

To enable USERMAXPERF register bit [3], you must set **USERMAXPERFEN** HIGH. If re-enabled, USERMAXPERF is not asserted until a rising edge occurs on **WAKEUP**.

At reset, SHUTDOWNNXT = 0, and DORMANTNXT = 0. This indicates that STANDBYWFI is entered if 0% performance is requested.

Note

It is not possible to set both the SHUTDOWNNXT and DORMANTNXT bits. If you attempt to do this, both SHUTDOWNNXT and DORMANTNXT default to their reset values of 0.

System Status Register

Figure 3-13 shows the bit assignments for this register.

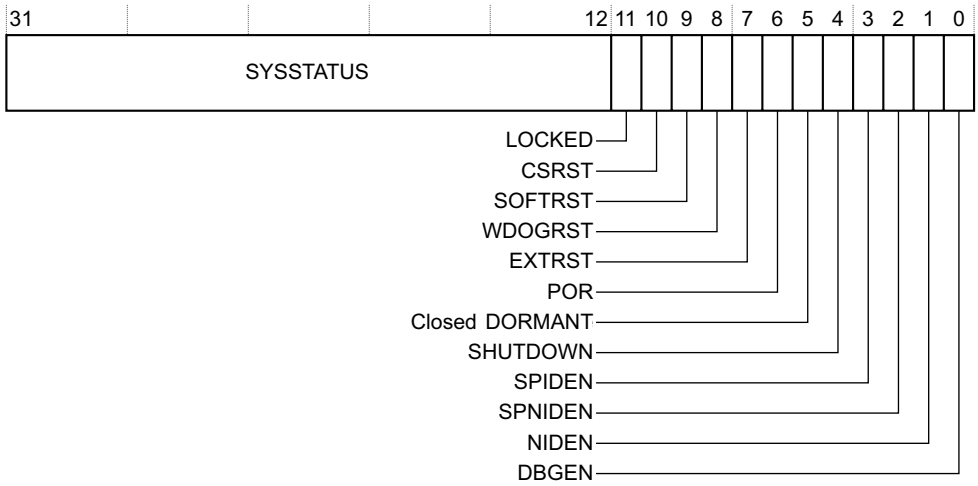


Figure 3-13 System Status Register bit assignments

Table 3-14 lists the bit assignments for this register.

Table 3-14 System Status Register bit assignments

Bits	Offset	Name	Type	Function
[31:12]	0x00	SYSSTATUS	R	Undefined, SBZ
[11]		LOCKED	R	Indicates that software reset request is locked when HIGH, write 0xA05FA05F to unlock
[10]		CSRST	R	CoreSight reset flag, warm reset
[9]		SOFTRST	R	Software reset flag, warm reset
[8]		WDOGRST	R	Watchdog reset flag, warm reset
[7]		EXTRST	R	External reset

Table 3-14 System Status Register bit assignments (continued)

Bits	Offset	Name	Type	Function
[6]		POR	R	Power-on reset flag
[5]		DORMANT	R	Dormant mode flag, indicates previous power down mode was dormant
[4]		SHUTDOWN	R	Shutdown mode flag, indicates previous power down mode was shutdown
[3]		SPIDEN	R	Secure privilege invasive debug enable
[2]		SPNIDEN	R	Secure privilege non-invasive debug enable
[1]		NIDEN	R	Non-invasive debug enable to ETM11
[0]		DBGEN	R	Debug enable, consider this as IDEN

The System Status Register also generates a software reset. To issue a software reset, you must first unlock the register by writing `0xA05FA05F`. You can then issue a software reset request by writing `0x76543210`.

The register is automatically locked when a software reset has been performed, or the incorrect software reset request value is written.

Note

Issuing a software reset request causes **nRESETCOMB** to be held LOW for eight **PCLK** cycles. The status of the system controller, APC1 and IEC are not affected.

Note

SPIDEN, **SPNIDEN**, **NIDEN** and **DBGEN** are SoC input pins, and they are not captured like the static SoC configuration on the AXI master RDATA bus.

IEC Configuration Index Mapping Configuration Registers

This section describes the Configuration Index Mapping Configuration Registers and contains the following subsections:

- *DCG Index Mapping Register* on page 3-39
- *DCG Fractional Performance Level Mapping Register* on page 3-39
- *DVC Index Level Mapping Register* on page 3-40
- *Processor Frequency Configuration Register* on page 3-40
- *PLL Configuration Settings Registers* on page 3-40

- *AXI Priority Register* on page 3-41
- *SMC Exclusive Access Monitor ID Register* on page 3-42
- *DLL Calibrate Register* on page 3-43.

DCG Index Mapping Register

Four 32-bit registers specify the IEC DCG index mappings. Figure 3-14 shows the mapping between the System Controller registers and the **IECCFGDCGIDXMAP** configuration input.

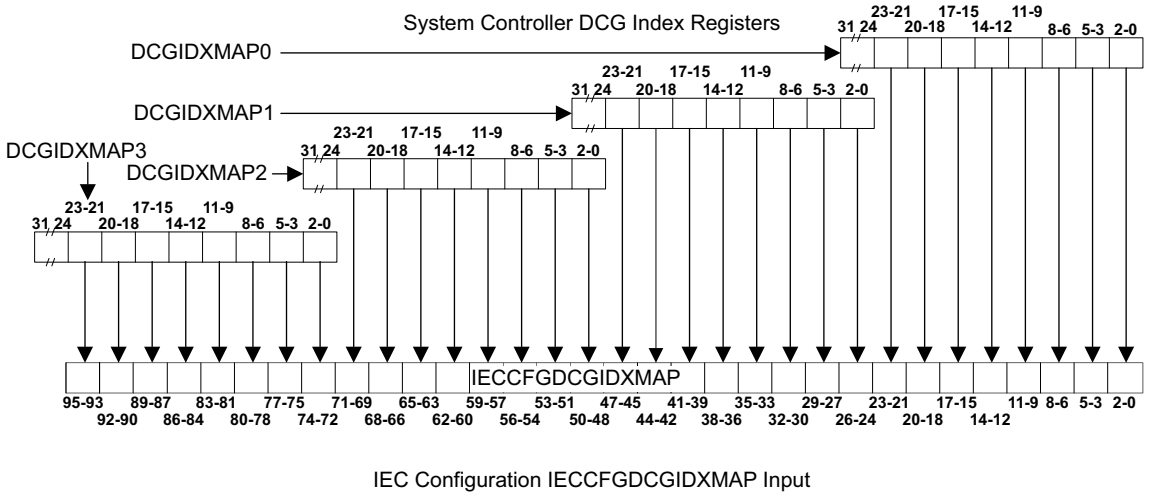


Figure 3-14 DCGIDXMAP registers to IECCFGDCGIDXMAP configuration inputs

DCG Fractional Performance Level Mapping Register

Two 32-bit registers specify the IEC fractional performance level mapping. Figure 3-15 shows the mapping between the System Controller registers and the **IECCFGDCGIDXMAP** configuration input.

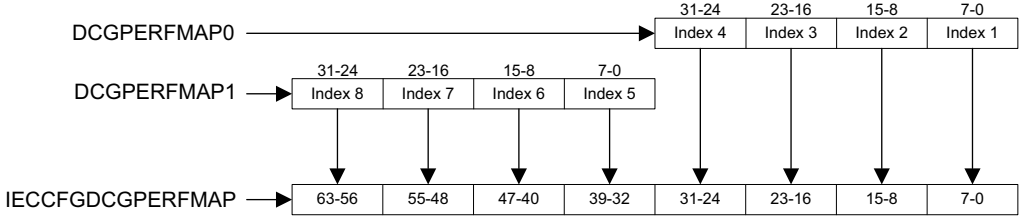


Figure 3-15 DCGPERFMAP registers to IECCFGDCGPERFMAP configuration inputs

DVC Index Level Mapping Register

The 24-bit DVCIDXMAP register is directly mapped to the **IECCFGDVCIDXMAP** configuration input on the IEC.

Processor Frequency Configuration Register

The 24-bit CFGFREQCPU register is directly mapped to the **IECCFGFRQCPU** configuration input on the IEC.

PLL Configuration Settings Registers

The system controller contains seven PLL configuration registers to define the PLL configuration settings for performance levels 1-7. Figure 3-16 shows the bit assignments for this register.

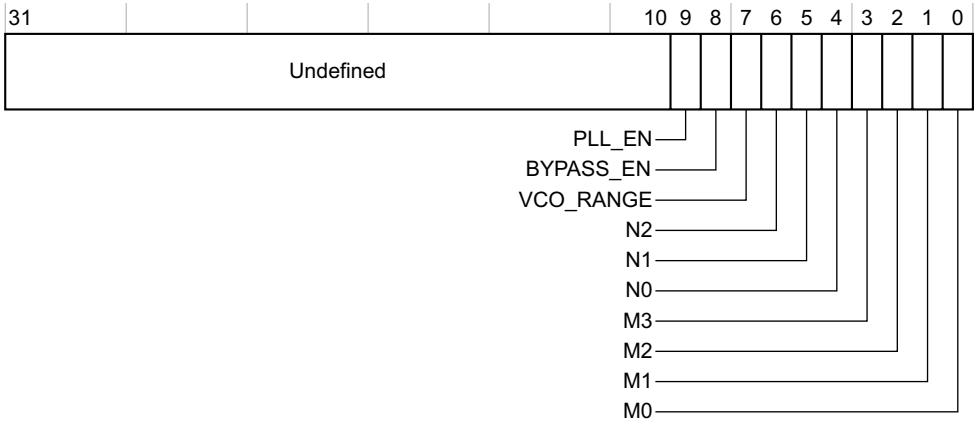


Figure 3-16 PLL Configuration Settings Register bit assignments

Table 3-15 lists the bit assignments for this register.

Table 3-15 PLL Configuration Settings Registers bit assignments

Bits	Offset	Name	Type	Function
[31:10]	0x038 -		R	Undefined, SBZ.
[9]	0x050	PLL_EN	RW	Bit set enables PLL for normal operation. Bit clear selects PLL power down mode.
[8]		BYPASS_EN	RW	Bit set bypasses the PLL VCO. Bit clear for normal PLL operation.

Table 3-15 PLL Configuration Settings Registers bit assignments (continued)

Bits	Offset	Name	Type	Function
[7]		VCO_RANGE	RW	Bit set for clock output > 133MHz.Bit clear when clock output < 133MHz.
[6]		N2	RW	PLL Input divider setting bit 2.
[5]		N1	RW	PLL Input divider setting bit 1.
[4]		N0	RW	PLL Input divider setting bit 0.
[3]		M3	RW	Internal feedback clock divider setting bit 3.
[2]		M2	RW	Internal feedback clock divider setting bit 2.
[1]		M1	RW	Internal feedback clock divider setting bit 1.
[0]		M0	RW	Internal feedback clock divider setting bit 0.

Note

The PLL input **DESKEW** is tied LOW to disable PLL deskew mode.

AXI Priority Register

Table 3-16 shows the default slave interface priorities on the 64-bit CAI. This is the system default after reset. The AXI Priority Register enables you to modify all six master arbitration priorities.

Note

Changing AXI priority does not change CAI ID values.

The reset value for this register is `0x002_C688`, octal is `18o543210`.

Table 3-16 lists the 64-bit AXI Priority Register bit assignments.

Table 3-16 64-bit AXI Priority Register bit assignments

Bits	Offset	Name	Type	Function
[31:23]	`0x060`	AXIPRIORITY	R	Undefined, SBZ
[22:20]	-	-	RW	CAI lowest priority slave **interfaceReset** value 3o5 = slave interface #6 = lowest
[19]	-	-	-	Undefined, SBZ

Table 3-16 64-bit AXI Priority Register bit assignments (continued)

Bits	Offset	Name	Type	Function
[18:16]	-	-	RW	CAI next highest priority slave **interfaceReset** value 3o4
[15]	-	-	-	Undefined, SBZ
[14:12]	-	-	RW	CAI next highest priority slave **interfaceReset** value 3o3
[11]	-	-	-	Undefined, SBZ
[10:8]	-	-	RW	CAI next highest priority slave **interfaceReset** value 3o2
[7]	-	-	-	Undefined, SBZ
[6:4]	-	-	RW	CAI next highest priority slave **interfaceReset** value 3o1
[3]	-	-	-	Undefined, SBZ
[2:0]	-	-	RW	CAI highest priority slave **interfaceReset** value 3o0 = slave interface #0 = highest

SMC Exclusive Access Monitor ID Register

Figure 3-17 shows the bit assignments for this register.

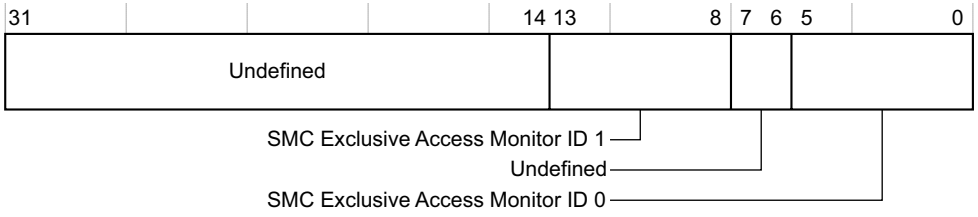


Figure 3-17 SMC Exclusive Access Monitor ID Register bit assignments

Table 3-17 lists the bit assignments for this register.

Table 3-17 SMC Exclusive Access Monitor ID Register bit assignments

Bits	Offset	Name	Type	Function
[31:14]	0x064	SMCEXACMONID	R	Undefined, SBZ
[13:8]	-	-	RW	SMC Exclusive Access Monitor ID 1
[7:6]	-	-	-	Undefined, SBZ
[5:0]	-	-	RW	SMC Exclusive Access Monitor ID 0

DLL Calibrate Register

Figure 3-18 shows the bit assignments for this register.

Figure 3-18 DLL Calibrate Register

Table 3-18 lists the bit assignments for this register.

Table 3-18 DLL Calibrate Outputs Register bit assignments

Bits	Offset	Name	Type	Function
[31:8]	0x068	DLLcalibrate	-	Undefined, SBZ
[7:0]	-	DLLcalibrate	R	DLL calibrate settings

3.7 Generic Interrupt Controller (GIC)

This section describes the GIC registers and contains the following subsections:

- *Register summary*
- *Distributor Register descriptions* on page 3-46.

3.7.1 Register summary

The offset of any particular register from the base address is fixed. Some register locations are reserved, and you must not used them during normal operation. Registers at the following locations are not used in this implementation of the GIC distributor:

- 0x008
- 0x0FF
- 0x108- 0x17C
- 0x188 - 0x1FF
- 0x208 - 27C
- 0x288 - 0x2FF
- 0x308 - 0x3FF
- 0x440 - 0x7FF
- 0x840 - 0xBFF
- 0xC10 - 0xDFF.

Table 3-19 Distributor Register summary

Register name	Base offset	Type	Width	Reset	Description
Control	0x000	RW	1	00000000	Control register
IrqControllerType	0x004	RW	8	00000001	Interrupt controller type register
IrqEnSet0	0x100	RW	32	0000FFFF	Irq 0 to 31 set-enable register
IrqEnSet1	0x104	RW	32	00000000	Irq 32 to 63 set-enable register
IrqEnClr0	0x180	RW	32	0000FFFF	Irq 0 to 31 clear-enable register
IrqEnClr1	0x184	RW	32	00000000	Irq 32 to 63 clear-enable register
IrqPenSet0	0x200	RW	16	00000000	Irq 0 to 31 set-pending register
IrqPenSet1	0x204	RW	32	00000000	Irq 32 to 63 set-pending register
IrqPenClr0	0x280	RW	16	00000000	Irq 0 to 31 clear-pending register
IrqPenClr1	0x284	RW	32	00000000	Irq 32 to 63 clear-pending register

Table 3-19 Distributor Register summary (continued)

Register name	Base offset	Type	Width	Reset	Description
IrqActiveBit0	0x300	RO	32	00000000	Irq 0 to 31 active bit register
IrqActiveBit1	0x304	RO	32	00000000	Irq 32 to 63 active bit register
IrqPriority0	0x400	RW	16	00000000	Irq 0 to 3 priority register
IrqPriority1	0x404	RW	16	00000000	Irq 4 to 7 priority register
IrqPriority2	0x408	RW	16	00000000	Irq 8 to 11 priority register
IrqPriority3	0x40C	RW	16	00000000	Irq 12 to 15 priority register
IrqPriority4	0x410	RO	16	00000000	Irq 16 to 19 priority register
IrqPriority5	0x414	RO	16	00000000	Irq 20 to 23 priority register
IrqPriority6	0x418	RO	16	00000000	Irq 24 to 27 priority register
IrqPriority7	0x41C	RO	16	00000000	Irq 28 to 31 priority register
IrqPriority8	0x420	RW	16	00000000	Irq 32 to 35 priority register
IrqPriority9	0x424	RW	16	00000000	Irq 36 to 39 priority register
IrqPriority10	0x428	RW	16	00000000	Irq 40 to 43 priority register
IrqPriority11	0x42C	RW	16	00000000	Irq 44 to 47 priority register
IrqPriority12	0x430	RW	16	00000000	Irq 48 to 51 priority register
IrqPriority13	0x434	RW	16	00000000	Irq 52 to 55 priority register
IrqPriority14	0x438	RW	16	00000000	Irq 56 to 59 priority register
IrqPriority15	0x43C	RW	16	00000000	Irq 60 to 63 priority register
IrqCPUTargets0	0x800	RO	32	00000000	Irq 0 to 3 CPU targets register
IrqCPUTargets1	0x804	RO	32	00000000	Irq 4 to 7 CPU targets register
IrqCPUTargets2	0x808	RO	32	00000000	Irq 8 to 11 CPU targets register
IrqCPUTargets3	0x80C	RO	32	00000000	Irq 12 to 15 CPU targets register
IrqCPUTargets4	0x810	RO	32	00000000	Irq 16 to 19 CPU targets register
IrqCPUTargets5	0x814	RO	32	00000000	Irq 20 to 23 CPU targets register
IrqCPUTargets6	0x818	RO	32	00000000	Irq 24 to 27 CPU targets register

Table 3-19 Distributor Register summary (continued)

Register name	Base offset	Type	Width	Reset	Description
IrqCPUTargets7	0x81C	RO	32	01010100	Irq 28 to 31 CPU targets register
IrqCPUTargets8	0x820	RW	4	00000000	Irq 32 to 35 CPU targets register
IrqCPUTargets9	0x824	RW	4	00000000	Irq 36 to 39 CPU targets register
IrqCPUTargets10	0x828	RW	4	00000000	Irq 40 to 43 CPU targets register
IrqCPUTargets11	0x82C	RW	4	00000000	Irq 44 to 47 CPU targets register
IrqCPUTargets12	0x830	RW	4	00000000	Irq 48 to 51 CPU targets register
IrqCPUTargets13	0x834	RW	4	00000000	Irq 52 to 55 CPU targets register
IrqCPUTargets14	0x838	RW	4	00000000	Irq 56 to 59 CPU targets register
IrqCPUTargets15	0x83C	RW	4	00000000	Irq 60 to 63 CPU targets register
IrqConfig0	0xC00	RW	32	00000000	Irq 0 to 15 configuration register
IrqConfig1	0xC04	RO	32	00000000	Irq 16 to 31 configuration register
IrqConfig2	0xC08	RW	32	00000000	Irq 32 to 47 configuration register
IrqConfig3	0xC0C	RW	32	00000000	Irq 48 to 63 configuration register
SoftwareIrq	0xF00	WO	20	-	Software interrupt register
PeriphID0	0xFE0	RO	8	0x000000	Peripheral identification register 0
PeriphID1	0xFE4	RO	8	0x0000001	Peripheral identification register 1
PeriphID2	0xFE8	RO	8	0x00000004	Peripheral identification register 2
PeriphID3	0xFEC	RO	8	0x00000000	Peripheral identification register 3
PrimeCellID0	0xFF0	RO	8	0x0000000D	PrimeCell identification register 0
PrimeCellID1	0xFF4	RO	8	0x000000F0	PrimeCell identification register 1
PrimeCellID2	0xFF8	RO	8	0x00000005	PrimeCell identification register 2
PrimeCellID3	0xFFC	RO	8	0x000000B1	PrimeCell identification register 3

3.7.2 Distributor Register descriptions

This section describes the registers and contains the following subsections:

- *Distributor Register Control Register* on page 3-47

- *Interrupt Controller Type Register*
- *Interrupt Set-Enable Registers* on page 3-48
- *Interrupt Clear-Enable Registers* on page 3-50
- *Interrupt Set-Pending Registers* on page 3-51
- *Interrupt Clear-Pending Registers* on page 3-53
- *Active Bit Registers* on page 3-55
- *Interrupt Priority Registers* on page 3-55
- *Interrupt CPU Target Registers* on page 3-58
- *Interrupt Configuration Registers* on page 3-61
- *Software Interrupt Register* on page 3-66.

Distributor Register Control Register

Figure 3-19 shows the bit assignments for this register.

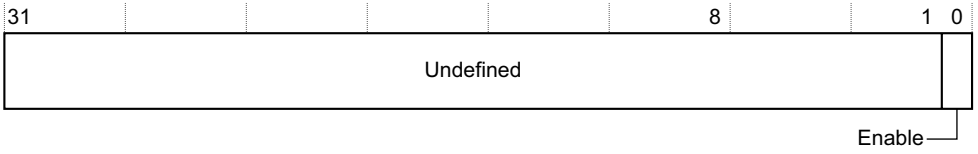


Figure 3-19 Distributor Register Control Register bit assignments

Table 3-20 lists the bit assignments for this register.

Table 3-20 Distributor Register Control Register bit assignments

Bits	Offset	Name	Type	Function
[31:1]	0x000		RW	Undefined, SBZ.
[0]	-	Enable	RW	Distributor enable. If this bit is 0, no interrupts are sent out of the distributor. Disabling the distributor while there are pending interrupts might give rise to spurious interrupts as a result of the inherent timing race between the disabling of the interrupt and the CPU response to pending interrupts.

Interrupt Controller Type Register

Figure 3-20 on page 3-48 shows the bit assignments for this register.

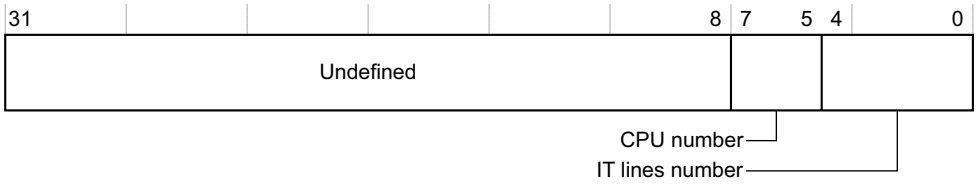


Figure 3-20 Interrupt Controller Type Register bit assignments

Table 3-21 lists the bit assignments for this register.

Table 3-21 Interrupt Controller Type Register bit assignments

Bits	Offset	Name	Type	Function
[31:8]	0x004		RW	Undefined, SBZ
[7:5]	-	CPU number	RW	CPU number encoding for the ARM1176 development chip is:000: System contains 1 CPU.
[4:0]	-	IT lines number	RW	IT lines number encoding for the GIC in the ARM1176 development chip is as follows:00001: 64 interrupts support, 32 interrupt lines support.

You can use the number of IT lines to determine which sets of registers in the Distributor register map are populated. In the ARM1176JZF core, IT lines is hard-coded to `5'b00001`, indicating 64 interrupt sources.

The CPU number is encoded as:

000 System contains 1 CPU.

001 System contains 2 CPUs.

010 System contains 3 CPUs.

011 System contains 4 CPUs.

0xx Reserved for future extension.

Note

The CPU number indicates the number of CPUs in the system. The CPU number for the ARM 1176JZF development chip is always 1.

Interrupt Set-Enable Registers

These registers control an enable bit for each interrupt in the interrupt distributor.

The GIC in the chip supports 64 interrupts:

- 32 software interrupts

- 32 hardware interrupts.

Interrupt Set-Enable Register 0

Figure 3-21 shows the bit assignments for this register.

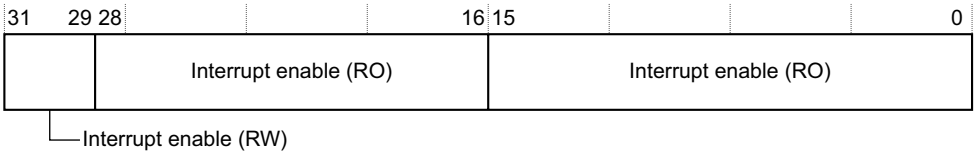


Figure 3-21 Interrupt Set-Enable Register 0 bit assignments

Table 3-22 lists the bit assignments for this register.

Table 3-22 Interrupt Set-Enable Register 0 bit assignments

Bits	Offset	Name	Type	Function
[31:29]	0x100	Interrupt enable	RW	Bit 31 is for the legacy **nIRQ** pin. Bit 30 is for a private watchdog in an MPCore system and is not used in this SoC. Bit 29 is for a private timer in an MPCore system and is not used in this SoC.
[28:16]		Interrupt enable	RO	Undefined interrupts, read as 0.
[15:0]		Interrupt enable	RO	Interrupts 0 to 15 are inter-processor, or reserved for software use. Read as 1.

Interrupt Set-Enable Register 1

Figure 3-22 shows the bit assignments for this register.

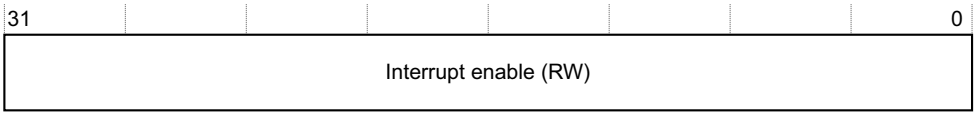


Figure 3-22 Interrupt Set-Enable Register 1 bit assignments

Table 3-23 lists the bit assignments for this register.

Table 3-23 Interrupt Set-Enable Register 1 bit assignments

Bits	Offset	Name	Type	Function
[31:0]	0x104	Interrupt enable	RW	Writing a 1 sets the interrupt enable bit and so, the interrupt is transmitted to the targeted CPU.

In these registers, writing a 1 sets the corresponding interrupt enable bit, and a read reads back a 1. When this is set, you can only clear it by using the Enable-Clear Register. In other words, writing a 0 to the Set-Enable Register does not write a 0 to the corresponding interrupt enable bit.

The values read in the Set-Enable Registers, for a particular range of interrupts, represent currently enabled interrupts. Non-present interrupts, depending on the Interrupt Number field of the Interrupt Controller Type Register related fields are read as zero, and writing to these fields has no effect.

Interrupt Clear-Enable Registers

These registers control an enable bit for each interrupt in the interrupt distributor. There can be up to eight Clear-Enable registers. Because the GIC in the ARM1176JZF Development Chip only supports 64 interrupts, it has two such registers and the rest are unused.

Interrupt Clear-Enable Register 0

Figure 3-23 shows the bit assignments for this register.

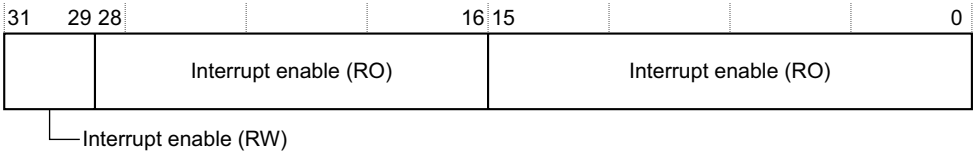


Figure 3-23 Interrupt Clear-Enable Register 0 bit assignments

Table 3-24 lists the bit assignments for this register.

Table 3-24 Interrupt Clear-Enable Register 0 bit assignments

Bits	Offset	Name	Type	Function
[31:29]	0x180	Interrupt enable	RW	Writing a 1 clears the interrupt enable bit and it reads a 0. Bit 31 is for the legacy **nIRQ** pin. Bit 30 is for a private watchdog in an MPCore system and is not used in this SoC. Bit 29 is for a private timer in an MPCore system and is not used in this SoC.
[28:16]	-	Interrupt enable	RO	Undefined interrupts, read as 0.
[15:0]	-	Interrupt enable	RO	Interrupts 0 to 15 are inter-processor or reserved for software use. Read as 0.

Interrupt Clear-Enable Register 1

Figure 3-24 shows the bit assignments for this register.

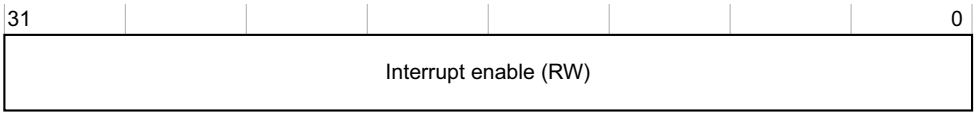


Figure 3-24 Interrupt Clear-Enable Register 1 bit assignments

Table 3-25 lists the bit assignments for this register.

Table 3-25 Interrupt Clear-Enable Register 1 bit assignments

Bits	Offset	Name	Type	Function
[31:0]	0x184	Interrupt enable	RW	Writing a 1 clears the interrupt enable bit and it reads a 0.

Writing a 1 clears the corresponding interrupt enable bit and a read reads back a 0. When cleared, you can only set this bit by using the Set-Enable Register. In other words, writing a 1 to the Clear-Enable Register does not write a 1 to the corresponding interrupt enable bit.

The values read in the Clear-Enable Registers, for a particular range of interrupts, represent currently enabled interrupts. Non-present interrupts, depending on the Interrupt number field of the Interrupt Controller Type Register, related fields are read as zero and writing to these fields has no effect.

Interrupt Set-Pending Registers

These registers indicate the interrupts that are currently in pending state, bit read as 1, or to force some interrupts to enter pending state by overriding event detection on interrupt lines. Each set register is used for 32 interrupts therefore, the ARM1176 development chip uses two such registers.

Interrupt Set-Pending Register 0

Figure 3-37 on page 3-63 shows the bit assignments for this register.

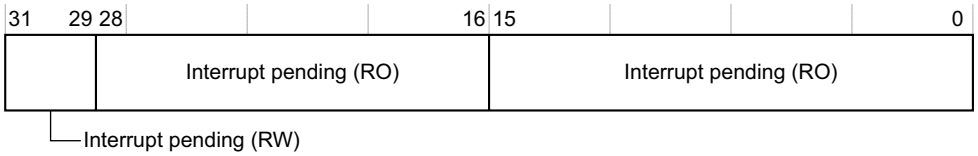


Figure 3-25 Interrupt Set-Pending Register 0 bit assignments

Table 3-26 lists the bit assignments for this register.

Table 3-26 Interrupt Set-Pending Register 0 bit assignments

Bits	Offset	Name	Type	Function
[31:29]	0x200	Interrupt pending	RW	Writing 1 sets the interrupt pending bit and the corresponding interrupt enters pending state. Bit 31 is for the legacy **nIRQ** pin. Bit 30 is for private watchdog in an MPCore system and is not used in this SoC. Bit 29 is for private timer in an MPCore system and is not used in this SoC.
[28:16]	-	Interrupt pending	RO	Undefined interrupts, read as 0.
[15:0]	-	Interrupt pending	RO	Writing to IT0 to IT15 has not effect. Only the Software Interrupt Register can trigger these interrupts.

Interrupt Set-Pending Register 1

Figure 3-26 shows the bit assignments for this register.

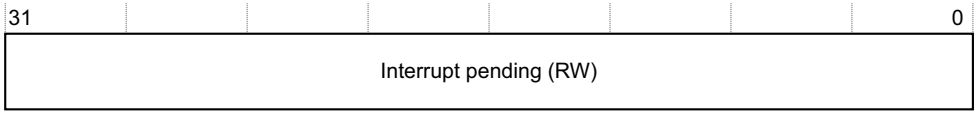


Figure 3-26 Interrupt Set-Pending Register 1 bit assignments

Table 3-27 lists the bit assignments for this register.

Table 3-27 Interrupt Set-Pending Register 1 bit assignments

Bits	Offset	Name	Type	Function
[31:0]	0x204	Interrupt pending	RW	Writing 1 sets the interrupt pending bit and the corresponding interrupt enters pending state

Note

- You can only perform reads and writes by enabling the distributor through the distributor control register. described in *Distributor Register Control Register* on page 3-47.
- For read status, you must set interrupt targets in the CPU targets registers, corresponding to hardware IRQs 32-63.

Writing a 1 to a bit through the Set-Pending Register means that the corresponding interrupt enters pending state. Writing a 1 sets the corresponding interrupt pending bit and a read reads back a 1. When set, you can only clear this bit using the Clear-Pending Register. Writing a 0 to the Set-Pending Register does not write a 0 to the corresponding interrupt pending bit.

Note

All RESERVED interrupts, spurious interrupts, and non-present interrupts, depending on the Interrupt Number field of the Interrupt Controller Type Register, and related fields are read as zero and writing to these fields has no effect.

The values read in the Set-Pending and Clear-Pending registers for the same interrupts range are the same. They both represent current pending interrupts. If a bit is read as 1, it implies that the interrupt is pending for at least one CPU.

Interrupt Clear-Pending Registers

These registers either determine that the registers that are currently in pending state and the bit is read as 1, or they force pending interrupts to return to the inactive state. Each set register is used for 32 interrupts. The ARM1176 Development Chip uses two such registers. The remaining ones are unused.

Interrupt Clear-Pending Register o

Figure 3-27 shows the bit assignments for this register.

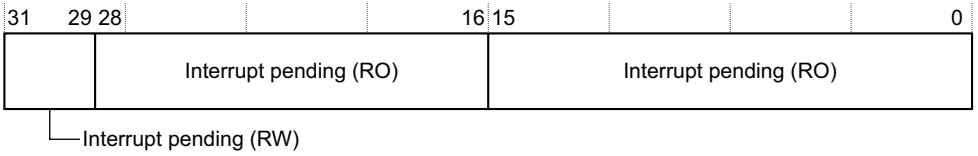


Figure 3-27 Interrupt Clear-Pending Register 0 bit assignments

Table 3-28 lists the bit assignments for this register.

Table 3-28 Interrupt Clear-Pending Register 0 bit assignments

Bits	Offset	Name	Type	Function
[31:29]	0x280	Interrupt pending	RW	Writing a 1 clears the interrupt enable bit and the corresponding interrupt enters pending state. A read returns a 0. Bit 31 is for the legacy **nIRQ** pin. Bit 30 is for private watchdog in an MPCore system and is not used in this SoC. Bit 29 is for private timer in an MPCore system and is not used in this SoC.
[28:16]	-	Interrupt pending	RO	Undefined interrupts, read as 0.
[15:0]	-	Interrupt pending	RO	Writing to bits corresponding to IT0 to IT15 has no effect. Only the Software Interrupt Register can trigger these interrupts.

Interrupt Clear-Pending Register 1

Figure 3-28 shows the bit assignments for this register.

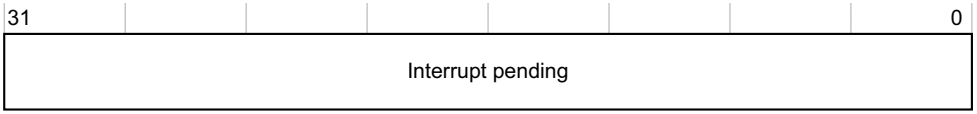


Figure 3-28 Interrupt Clear-Pending Register 1 bit assignments

Table 3-29 lists the bit assignments for this register.

Table 3-29 Interrupt Clear-Pending Register 1 bit assignments

Bits	Offset	Name	Type	Function
[31:0]	0x284	Interrupt pending	RW	Writing a 1 clears the interrupt enable bit and the corresponding interrupt enters pending state. A read returns a 0.

Writing a 1 clears the corresponding interrupt pending bit and a read reads back a 0. When cleared, you can only set this bit by using the Enable-Set Register. In other words, writing a 1 to the Enable-Clear Register does not write a 1 to the corresponding interrupt enable bit.

Note

All RESERVED interrupts, spurious interrupts, and non-present interrupts, depending on the Interrupt number field of the Interrupt Controller Type Register,and related fields are read as zero and writing to these fields has no effect.

The values read in the Pending-Set and Pending-Clear registers for the same interrupts range are the same, and represent current pending interrupts. If a bit is read as 1, it implies that the interrupt is pending.

Active Bit Registers

The Active Bit Register enables the software to find out the interrupts that are currently active, bit read as 1. They are read-only registers and writes to these registers are ignored. Because each register only handles 32 interrupts, there are two such registers in the chip. Figure 3-29 shows the bit assignments for these registers.

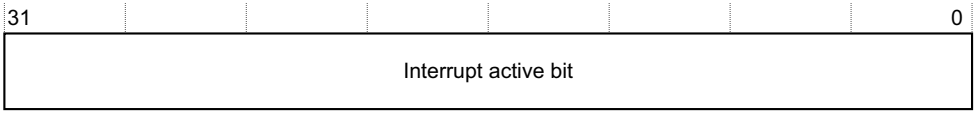


Figure 3-29 Active Bit Register 0 and 1 bit assignments

Table 3-30 lists the bit assignments for these registers.

Table 3-30 Active Bit Register 0 and 1 bit assignments

Bits	Offset	Name	Type	Function
[31:0]	0x300 0x304	Interrupt active bit	RO	Bit reads as 1: Corresponding interrupt is active Bit reads as 0: Corresponding interrupt is inactive

Interrupt Priority Registers

These registers store the individual interrupt priority and16 levels of priority are supported.

Because each register sets the priority of four interrupts, there are 16 such registers in the ARM1176 Development Chip. You can set the priority for each interrupt by writing a value between 0x0-0xF to the corresponding bits for the interrupts in the registers described in this section. An interrupt with priority 0x0 has the highest priority, and an interrupt with priority 0xF is the lowest priority.

Note

You must never set the priority for an interrupt equal to the Priority Mask Register under normal operation. This is because the CPU interface does a strict comparison between the pending interrupt priority and the priority mask set in the Priority Mask Register. See *Register summary* on page 3-44. Because of this, if the priority mask is set to `0xF`, interrupts of priority `0xF` are not executed.

For interrupts with equal priority, the interrupt with the lowest ID is handled first, the interrupt with the second lowest ID is handled second, and the interrupt with the third lowest ID is handled third. If the two interrupts have the same ID, this can be the case for software interrupts, the lowest CPU source ID is executed first.

Interrupt Priority Register 0-3

Figure 3-30 shows the bit assignments for these registers.

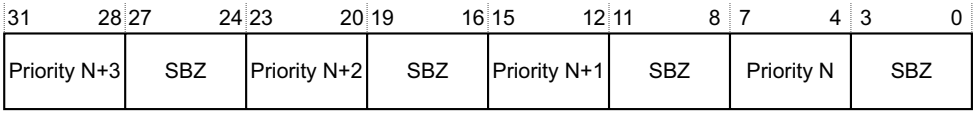


Figure 3-30 Interrupt Priority Register 0-3 bit assignments

Table 3-31 lists the bit assignments for these registers.

Table 3-31 Interrupt Priority Register 0-3 bit assignments

Bits	Offset	Name	Type	Function
[31:28]	`0x400`	Priority N+3	RW	Interrupt priority
[27:24]	`0x404`	SBZ		
[23:20]	`0x408`	Priority N+2		
[19:16]	`0x40C`	SBZ		
[15:12]	-	Priority N+1		
[11:8]	-	SBZ		
[7:4]	-	Priority N		
[3:0]	-	SBZ		

Interrupt Priority Register 4-6

Figure 3-31 shows the bit assignments for these registers.

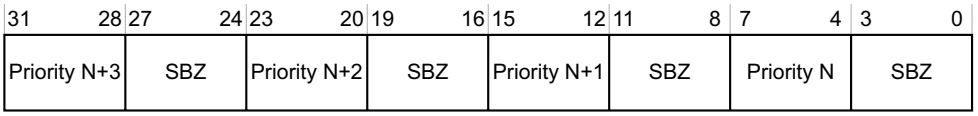


Figure 3-31 Interrupt Priority Register 4-6 bit assignments

Table 3-32 lists the bit assignments for these registers.

Table 3-32 Interrupt Priority Register 4-6 bit assignments

Bits	Offset	Name	Type	Function
[31:28]	0x410	Priority N+3	RO	Interrupt priority
[27:24]	0x414	SBZ		
[23:20]	0x418	Priority N+2		
[19:16]	-	SBZ		
[15:12]	-	Priority N+1		
[11:8]	-	SBZ	RO	Interrupt priority
[7:4]	-	Priority N		
[3:0]	-	SBZ		

Interrupt Priority Register 7-15

Figure 3-32 shows the bit assignments for these registers.

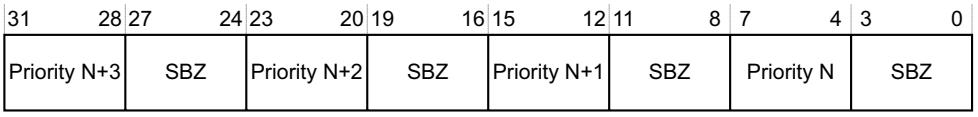


Figure 3-32 Interrupt Priority Register 7-15 bit assignments

Table 3-33 lists the bit assignments for these registers.

Table 3-33 Interrupt Priority 7-15 Register bit assignments

Bits	Offset	Name	Type	Function
[31:28]	0x41C	Priority N+3	RW	Interrupt priority
[27:24]	0x420	SBZ		
[23:20]	0x424	Priority N+2		
[19:16]	0x428	SBZ		
[15:12]	0x42C	Priority N+1		
[11:8]	0x430	SBZ		
[7:4]	0x434	Priority N		
[3:0]	0x438 0x43C	SBZ		

Interrupt CPU Target Registers

These registers store the list of CPUs that each interrupt is sent to if the event defined by the Interrupt Controller Type Register occurs. Each bit in the Interrupt CPU Targets Register refers to one CPU. For the ARM1176JZF Development Chip, only one core is relevant. These registers are ignored in the case of software triggered interrupts.

This section describes the format of interrupt Interrupt CPU Targets Registers. Because each register is used for four interrupts, there are 16 registers in the development chip.

Interrupt CPU Target Registers 0-6

Figure 3-33 shows the bit assignments for these registers.

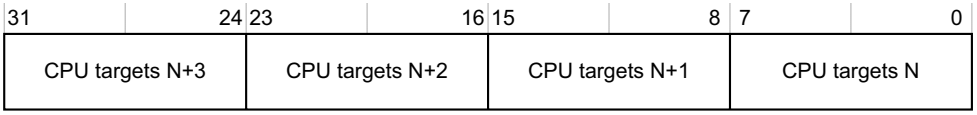


Figure 3-33 Interrupt CPU Target Registers 0-6 bit assignments

Table 3-34 lists the bit assignments for these registers.

Table 3-34 Interrupt CPU Target Registers 0-6 bit assignments

Bits	Offset	Name	Type	Function
[31:24]	0x800	CPU targets N+3[a]	RO	Store list of CPUs that are sent interrupts
[23:16]	0x804	CPU targets N+2		
[15:8]	0x808	CPU targets N+1		
[7:0]	0x80C 0x810 0x814 0x818	CPU targets N		

a. Only one cpu is active and it is always CPU0.

Interrupt CPU Target Register 7

Figure 3-34 shows the bit assignments for this register.

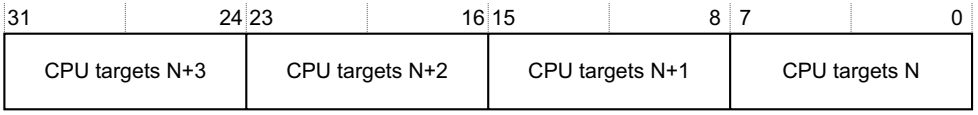


Figure 3-34 Interrupt CPU Target Register 7 bit assignments

Table 3-35 lists the Interrupt CPU Target Register 7 bit assignments.

Table 3-35 Interrupt CPU Target Register 7 bit assignments

Bits	Offset	Name	Type	Function
[31:24]	0x81C	CPU targets N+3[a]	RO	Store list of CPUs that are sent interrupts
[23:16]	-	CPU targets N+2		
[15:8]	-	CPU targets N+1		
[7:0]	-	CPU targets N		

a. Only one cpu is active and it is always CPU0.

This register is for interrupts IT28, IT29, IT30, and IT31.

Note

For IT29, IT30 and IT31, values read in corresponding fields depend on the accessing CPU because these interrupt sources are private:

- For CPU 0: CPU targets 29, 30 and 31 are read as `0x1`. Writes are ignored.

In the particular implementation of the GIC in the ARM1176JZF Development Chip, there is only one CPU, CPU 0. The reset value of the CPU Target Register 8 is `0x01010100`.

Interrupt CPU Target Register 8-15

Figure 3-35 shows the bit assignments for these registers.

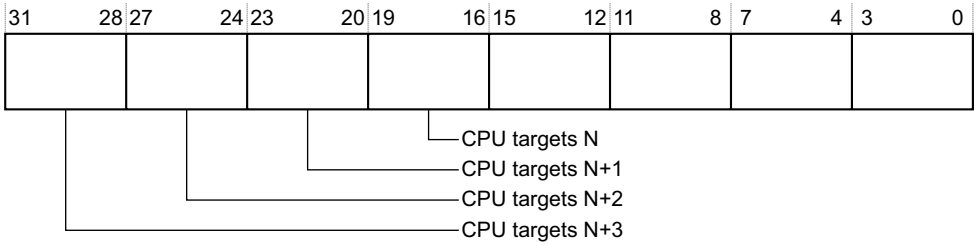


Figure 3-35 Interrupt CPU Target Register 8-15

Table 3-36 lists the bit assignments for these registers. These bits correspond to a hardware interrupt. The reset value of these bits is 0.

Table 3-36 Interrupt CPU Target Register 8-15 bit assignments

Bits	Offset	Name	Type	Function
[31:28]	`0x820`	CPU targets N+3	RO[a]	Store the list of CPUs that interrupts are sent to
[27:24]	`0x824`	CPU targets N+2		
[23:20]	`0x828`	CPU targets N+1		
[19:16]	`0x82C`	CPU targets N		
[15:12]	`0x830`	-		
[11:8]	`0x834`	-		
[7:4]	`0x838`	-		
[3:0]	`0x83C`	-		

a. For bits assigned to registers 0x820 to 0x83c, only bits [0], [8], [16] and [24] are RW. The others are R0.

Interrupt Configuration Registers

Interrupt Configuration Registers define the interrupt line event that it is considered active and the software model. There can be up to 16 interrupt configuration registers.

Because each register handles only 16 interrupts, the ARM1176JZF Development Chip uses four such registers.

Note

IT 0 to 31 and 1023 are special interrupts for each individual CPU:

- IT0 to IT15 are defined as inter-processor interrupts
- IT16 to IT28 are reserved
- IT29 is defined as private timer interrupt
- IT30 is defined as private watchdog interrupt, if in Timer mode
- IT31 is defined as legacy **nIRQ** pin
- IT1023 is defined as the spurious interrupt.

Corresponding fields are all read as `00` and write accesses to these fields are ignored for all interrupts except for the inter-processor interrupts, IT0 to IT15. You can configure IPI software models and apply them to the interrupts sent from the writing CPU.

These interrupts are aliased. An additional field in the Interrupt Acknowledge Register specifies the requesting processor.

Interrupt Configuration Register 0

Figure 3-36 shows the bit assignments for this register.

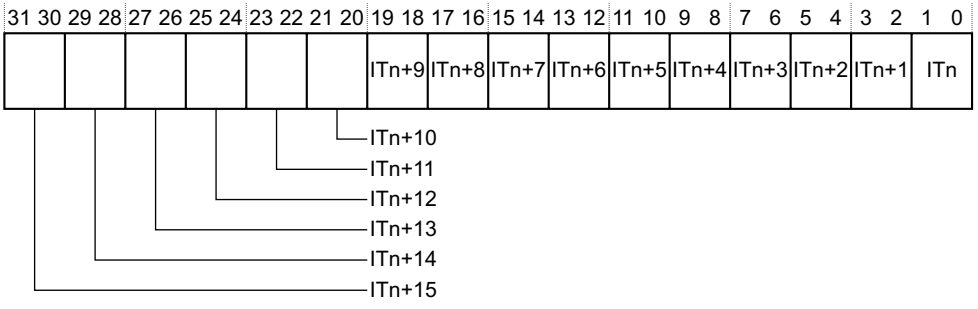


Figure 3-36 Interrupt Configuration Register 0 bit assignments

Table 3-37 lists the bit definition for this register.

Table 3-37 Interrupt Configuration Register 0 bit definition

Bits	Offset	Name	Type	Function
[31:30]	0xC00	ITn+15	RW	Configure software model for all 16 IPIs
[29:28]	-	ITn+14		
[27:26]	-	ITn+13		
[25:24]	-	ITn+12		
[23:22]	-	ITn+11		
[21:20]	-	ITn+10	RW	Configure software model for all 16 IPIs
[19:18]	-	ITn+9		
[17:16]	-	ITn+8		
[15:14]	-	ITn+7		
[13:12]	-	ITn+6		
[11:10]	-	ITn+5		
[9:8]	-	ITn+4		
[7:6]	-	ITn+3		
[5:4]	-	ITn+2		
[3:2]	-	ITn+1		
[1:0]	-	ITn		

Table 3-38 lists the Individual ITn encoding.

Table 3-38 Interrupt definition encoding

IT bit 0	Meaning
0	Interrupt line uses the N-NN software model
1	Interrupt line uses the 1-N software model

Table 3-38 Interrupt definition encoding (continued)

IT bit 0	Meaning
IT bit 1	Meaning
0	Interrupt line is considered as level HIGH active
1	Interrupt line is considered as rising edge sensitive

You can configure the software model of the interrupts applied to in this register, but not the edge and level configuration. The testmask of this register, for example, for PMAP testing, is `0x55555555`.

Interrupt Configuration Register 1

Figure 3-37 shows the bit assignments for this register.

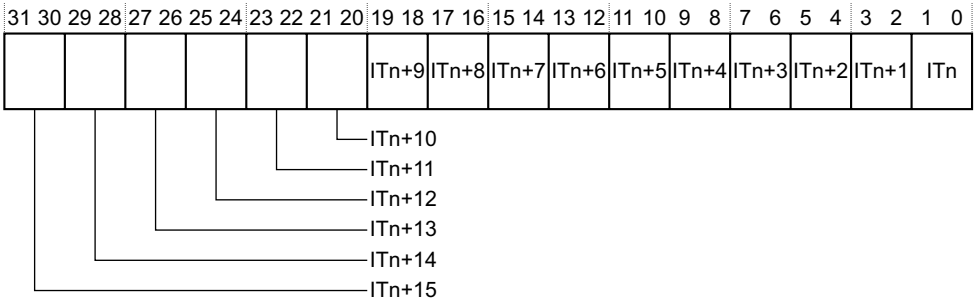


Figure 3-37 Interrupt Configuration Register 1 bit assignments

Table 3-39 lists the bit assignments for this register.

Table 3-39 Interrupt Configuration Register 1 bit assignments

Bits	Offset	Name	Type	Function
[31:30]	0xC04	ITn+15	RO	Configuration for IDs 16-28 and internal IRQs, 29, 30, and 31
[29:28]	-	ITn+14		
[27:26]	-	ITn+13		
[25:24]	-	ITn+12		
[23:22]	-	ITn+11		
[21:20]	-	ITn+10		
[19:18]	-	ITn+9		
[17:16]	-	ITn+8		
[15:14]	-	ITn+7		
[13:12]	-	ITn+6		
[11:10]	-	ITn+5		
[9:8]	-	ITn+4		
[7:6]	-	ITn+3		
[5:4]	-	ITn+2		
[3:2]	-	ITn+1		
[1:0]	-	ITn		

This register is reserved for IDs 16-28 and internal IRQs, 29, 30, and 31. This register is read-only and you cannot configure it.

Interrupt Configuration Register 2-3

Figure 3-38 on page 3-65 shows the bit assignments for these registers.

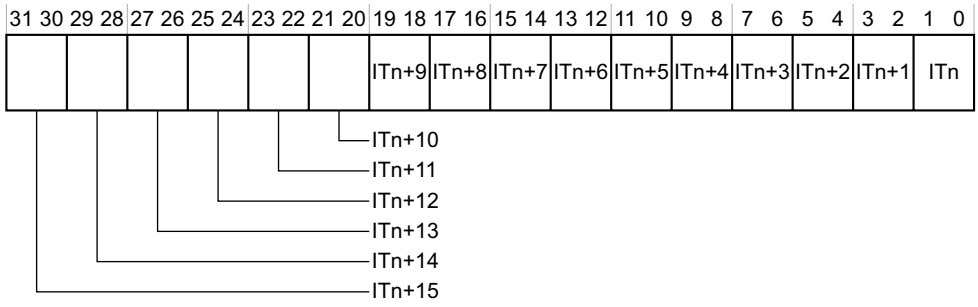


Figure 3-38 Interrupt Configuration Register 2-3 bit assignments

Table 3-40 lists the bit definition for these registers.

Table 3-40 Interrupt Configuration Register 2-3 bit definition

Bits	Offset	Name	Type	Function
[31:30]	0xC08	ITn+15	RW	Configure software model and edge and level for IDs 32-47, and 48-63
[29:28]	0xC0C	ITn+14		
[27:26]	-	ITn+13		
[25:24]	-	ITn+12		
[23:22]	-	ITn+11		
[21:20]	-	ITn+10		
[19:18]	-	ITn+9		
[17:16]	-	ITn+8		
[15:14]	-	ITn+7		
[13:12]	-	ITn+6		
[11:10]	-	ITn+5		
[9:8]	-	ITn+4		
[7:6]	-	ITn+3	RW	Configure software model and edge and level for IDs 32-47, and 48-63
[5:4]	-	ITn+2		
[3:2]	-	ITn+1		
[1:0]	-	ITn		

These two registers support hardware IRQs of ID 32-47, and 48-63. You can fully configure them.

Software Interrupt Register

The Software Interrupt Register is a write-only register that triggers an interrupt, identified by its own ID, to a list of CPUs. Figure 3-39 shows the bit assignments for this register.

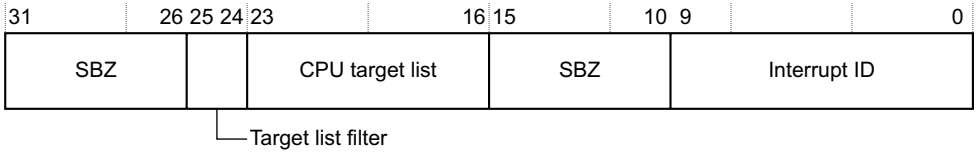


Figure 3-39 Software Interrupt Register bit assignments

Table 3-41 lists the bit definition for this register.

Table 3-41 Software Interrupt Register bit definition

Bits	Offset	Name	Type	Function
[31:26]	0xF00	SBZ	WO	Trigger specific interrupt---
[25:24]	-	Target list filter[a]		
[23:16]	-	CPU target list[a]		
[15:10]	-	SBZ		
[9:0]	-	Interrupt ID		Interrupt identifier

a. Allowed values that trigger an interrupt on CPU0 are:
target list filter = 0x00 and bit [0] of CPU target set
target list filter = 0x10.

The CPU target list can be different from the one defined in the CPU Target List Register for the specified interrupt ID.

The Target List filter definition is:

- 00: Interrupt sent to CPUs listed in CPU Target List
- 01: CPU target list is ignored, interrupt is sent to all but the requesting CPU
- 10: CPU target list is ignored, interrupt is sent to the requesting CPU only
- 11: Reserved.

Note

If you attempt to trigger an interrupt with an ID larger than the number of supported interrupts, or that references a CPU that is not present, there can be unpredictable effects in the interrupt distributor.

3.7.3 CPU interface summary

The CPU interface registers are described in this section. Table 3-42 shows the CPU interface registers and the following sections provide details about each individual register.

Table 3-42 CPU interface registers

Name	Offset	Type	Width	Reset	Description
Control	0x000	RW	1	0x00000000	Control register
PriorityMask	0x004	RW	4	0x00000000	Priority Mask register
BinaryPoint	0x008	RW	3	0x00000003	Binary Point register
IrqAck	0x00C	RO	13	0x000003FF	Interrupt Ack register
EndIrq	0x010	WO	13	-	End of Interrupt (EIO) register
RunningPriority	0x014	RO	4	0x000000FF	Running Rriority register
HighestPendingIrq	0x018	RO	13	0x000003FF	Highest Pending Interrupt register

Control register

Table 3-43 shows bit assignments for the CPU Interface Control register.

Table 3-43 CPU Interface Control register bit assignments

Name	Bits	Type	Function
-	31:1	RW	Reserved. SBZ
Enable	0	RW	Enable

Priority Mask register

Use the Priority Mask to prevent interrupts from being sent to the CPU. The CPU interface asserts an interrupt request to the CPU if, and only if, the priority of the highest pending interrupt sent by the interrupt distributor is strictly higher than the mask value set in the Priority Mask Register.

One consequence of the strict comparison is that lowest priority interrupt will never cause the assertion of interrupt request to the CPU. This allows an extra level of interrupt enabling. Although an interrupt can be pending, it will never be seen by the CPU. Table 3-44 shows bit assignments for the Priority Mask register.

Table 3-44 Priority Mask Register bit assignments

Name	Bits	Type	Function
Priority Mask	7:4	RW	Used to determine if the CPU interface asserts an interrupt request to CPU, if and only if the priority of the highest pending interrupt sent by the interrupt distributor is strictly higher than the mask set in this register.

Binary Point register

Binary point register is used to specify a certain number of bits to ignore in the priority comparison made in the CPU interface for pre-emption stack. Table 3-45 shows bit assignments for the Binary Pointer register.

Table 3-45 Binary Pointer Register bit assignments

Name	Bits	Type	Function
SBZ/UNP	31:3	RW	-
Binary Point	2:0		If this is set, ignore bits in priority comparison made in CPU interface for pre-emption stack.

Table 3-46 shows the meanings for Binary Point values:

Table 3-46 Binary Point value meanings

Value	Meaning
000	Bits [7:1] of the priority are used to determine pre-emption
001	Bits [7:2] of the priority are compared for pre-emption.
010	Bits [7:3] of the priority are compared for pre-emption.
011	Bits [7:4] of the priority are compared for pre-emption.
100	Bits [7:5] of the priority are compared for pre-emption.
101	Bits [7:6] of the priority are compared for pre-emption.
110	Bits [7:7] of the priority are compared for pre-emption.
111	No pre-emption is performed. All bits of the priority are used for prioritization.

You can restrict the lowest value to any value from `0` to `4`. If you attempt to specify a value less than the set minimum, the minimum value is used. This value can be read back to enable software discovery. At reset, the register takes its minimum supported value.

Interrupt Acknowledge Register

The Interrupt Acknowledge Register is a read-only register used by the CPU to determine the ID of the interrupt asserted by the CPU interface. Table 3-47 shows bit assignments for the Interrupt Acknowledge Register.

Table 3-47 Interrupt Acknowledge Register bit assignments

Name	Bits	Type	Function
SBZ/RAZ	31:13	RO	-
CPU source ID	12:10		CPU source ID value. Depends on Interrupt ID field: • If Interrupt ID field is 0 to 15, contains ID of the CPU that requested the IPI. • In all other cases, CPU source ID field is read as zero. This can be ignored.
Interrupt ID	9:0		Interrupt identifier

End of Interrupt (EOI) Register

This write-only register is used when the software finishes handling an interrupt. Table 3-48 shows bit assignments for the EOI Register.

Table 3-48 End of Interrupt Register bit assignments

Name	Bits	Type	Function
SBZ/RAZ	31:13	WO	-
CPU source ID	12:10		CPU source ID value. Depends on Interrupt ID field: • If Interrupt ID field is 0 to 15, contains ID of the CPU that requested the IPI. • In all other cases, CPU source ID field is read as zero. This can be ignored.
Interrupt ID	9:0		Interrupt identifier

Running Interrupt Register

This read-only register contains the priority level of the currently running interrupts on the CPU. Table 3-49 shows bit assignments for the Running Interrupt Register.

Table 3-49 Running interrupt Register bit assignments

Name	Bits	Type	Function
SBZ/UNP	31:8	RO	-
Priority	7:4		Indicates priority level of the current running interrupt.
SBZ	3:0		-

When no interrupt is running, defined as acknowledged by reading acknowledge register but not ended by writing to EOI register, the priority value read is `0xFF`.

Highest Pending Interrupt Register

The Highest Pending Interrupt Register contains the Interrupt ID and CPU ID of the Highest Pending Interrupt for this CPU. If no interrupt is pending, the Interrupt ID returned is `0x3FF`. This indicates a spurious interrupt.

The format of the register is the same as the Interrupt Acknowledge Register. Table 3-50 shows bit assignments for the Highest Pending Interrupt Register.

Table 3-50 Highest Pending interrupt Register bit assignments

Name	Bits	Type	Function
SBZ/RAZ	31:13	WO	-
CPU Source ID	12:10		CPU source ID value. Depends on Interrupt ID field: • If Interrupt ID field is 0 to 15, contains ID of the CPU that requested the IPI. • In all other cases, CPU source ID field is read as zero. This can be ignored.
Interrupt ID	9:0		Interrupt identifier

Appendix A
Electrical and Physical Characteristics

This chapter describes the electrical and physical characteristics of the ARM1176JZF Development Chip and contains the following sections:

A.1 About electrical and physical characteristics

Electrical specifications are defined in the separate document, *ARM1176JZF Development Chip Electrical Specification* (PR168-PRDC-004363), to reduce the size of this TRM.

The physical characteristics of this chip include:

TSMC 130nm Generic Process

- 9 layer metal with 9th layer reserved for flipchip bumps)
- 10x14mm Silicon Die
- IEM Subsystem Target: 332MHz; DDR, AXI and ATB Target: 166MHz; APB Target: 83MHz
- 1.2V core, 3.3V I/O, 1.8V LVCMOS for Mobile DDR I/O.

Packaging

- 1400 ball fully populated, Flipchip
- Package ball pitch: 1mm
- Package bump pitch 200um.

A.2 Electrical requirements

The electrical requirements are defined in the *ARM1176JZF Development Chip Electrical Specification*(PR168-PRDC-004363).

A.3 Bonding and pinout

This section defines the input and output cells and die bonding for the ARM1176JZF Development Chip. It contains the following subsections:

- *Pad types*
- *Die bonding* on page A-5.

A.3.1 Pad types

Table A-1 lists a key to the pad types that the die bonding information uses.

Table A-1 Pad type

Pad type	Cell name	Description
MDO	pnl_hstl_classi_m	Mobile DDR output
MDOB	pnl_hstl_classi_m	Mobile DDR output bi-directional
IPD	pnl_it2pd8	Digital input with pull-down
IPU	pnl_it2pu8	Digital input with pull-up
O4	pnl_tf04it0nn2	Digital output 4mA
I	pnl_it2nn8	Digital input
O8	pnl_tf08it0nn2	Digital output 8mA
O12	pnl_tf12it0nn2	Digital output 12mA
T8	pnl_tf08it0nn2	Digital tristate output
B	pnl_tf08it0nn2	Digital bi-directional
A	pnl_aio	Analog through cell
DP	pnl_vc	Digital power
DG	pnl_gcs	Digital ground
AP	pnl_av	Analog power
AG	pnl_ag	Analog ground
IOP	pnl_vop	I/O power
IOG	pnl_go	I/O ground
MDPD	pnl_hstl_vp_m	Mobile DDR pre driver 2v5

Table A-1 Pad type (continued)

Pad type	Cell name	Description
MDOP	pnl_hstl_vq	Mobile DDR output driver 1v8
MDOG	pnl_hstl_go	Mobile DDR output driver V_{SS}
MDCP	pnl_hstl_vc	Mobile DDR core power 1v2
MDCG	pnl_hstl_gcs	Mobile DDR core power V_{SS}
MDVRF	pnl_hstl_vref	Mobile DDR VRF

A.3.2 Die bonding

Table A-2 lists the die bonding. All inputs and outputs operate at 3.3V unless otherwise indicated, then they operate at 1.8V.

Table A-2 Die bonding

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
-	**VSS**	Common ground	IOG	1	C39
-	**VD33**	I/O power, 3.3V	IOP	2	D39
-	VSS	Common ground	DG	3	D38
-	**VD33**	I/O power, 3.3V	IOP	4	E39
AXI master port	**AWADDR_ExtMst[0]**	Write address bus	O8	5	E38
AXI master port	**AWADDR_ExtMst[1]**	Write address bus	O8	6	E37
AXI master port	**AWADDR_ExtMst[2]**	Write address bus	O8	7	F39
AXI master port	**AWADDR_ExtMst[3]**	Write address bus	O8	8	F38
AXI master port	**AWADDR_ExtMst[4]**	Write address bus	O8	9	F37
AXI master port	**AWADDR_ExtMst[5]**	Write address bus	O8	10	F36
AXI master port	**AWADDR_ExtMst[6]**	Write address bus	O8	11	G39
AXI master port	**AWADDR_ExtMst[7]**	Write address bus	O8	12	G38
AXI master port	**AWADDR_ExtMst[8]**	Write address bus	O8	13	G37

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI master port	**AWADDR_ExtMst[9]**	Write address bus	O8	14	G36
AXI master port	**AWADDR_ExtMst[10]**	Write address bus	O8	15	G35
AXI master port	**AWADDR_ExtMst[11]**	Write address bus	O8	16	H39
AXI master port	**AWADDR_ExtMst[12]**	Write address bus	O8	17	H38
AXI master port	**AWADDR_ExtMst[13]**	Write address bus	O8	18	H37
AXI master port	**AWADDR_ExtMst[14]**	Write address bus	O8	19	H36
AXI master port	**AWADDR_ExtMst[15]**	Write address bus	O8	20	H35
AXI master port	**AWADDR_ExtMst[16]**	Write address bus	O8	21	H34
AXI master port	**AWADDR_ExtMst[17]**	Write address bus	O8	22	J39
AXI master port	**AWADDR_ExtMst[18]**	Write address bus	O8	23	J38
AXI master port	**AWADDR_ExtMst[19]**	Write address bus	O8	24	J37
AXI master port	**AWADDR_ExtMst[20]**	Write address bus	O8	25	J36
AXI master port	**AWADDR_ExtMst[21]**	Write address bus	O8	26	J35
AXI master port	**AWADDR_ExtMst[22]**	Write address bus	O8	27	J34
AXI master port	**AWADDR_ExtMst[23]**	Write address bus	O8	28	J33
AXI master port	**AWADDR_ExtMst[24]**	Write address bus	O8	29	K39
AXI master port	**AWADDR_ExtMst[25]**	Write address bus	O8	30	K38
AXI master port	**AWVALID_ExtMst**	Write address valid	O8	31	K37
AXI master port	**AWREADY_ExtMst**	Write address ready	I	32	K36
AXI master port	**AWADDR_ExtMst[26]**	Write address bus	O8	33	K35
AXI master port	**AWADDR_ExtMst[27]**	Write address bus	O8	34	K34
AXI master port	**AWADDR_ExtMst[28]**	Write address bus	O8	35	K33
AXI master port	**AWADDR_ExtMst[29]**	Write address bus	O8	36	K32

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI master port	**AWADDR_ExtMst[30]**	Write address bus	O8	37	L39
AXI master port	**AWADDR_ExtMst[31]**	Write address bus	O8	38	L38
AXI master port	**AWLEN_ExtMst[0]**	Write address burst size	O8	39	L37
AXI master port	**AWLEN_ExtMst[1]**	Write address burst size	O8	40	L36
AXI master port	**AWLEN_ExtMst[2]**	Write address burst size	O8	41	L35
AXI master port	**AWLEN_ExtMst[3]**	Write address burst size	O8	42	L34
AXI master port	**AWSIZE_ExtMst[0]**	Write address transfer size	O8	43	L33
AXI master port	**AWSIZE_ExtMst[1]**	Write address transfer size	O8	44	L32
AXI master port	**AWBURST_ExtMst[0]**	Write address burst type	O8	45	M39
AXI master port	**AWBURST_ExtMst[1]**	Write address burst type	O8	46	M38
AXI master port	**AWLOCK_ExtMst[0]**	Write address atomic type	O8	47	M37
AXI master port	**AWLOCK_ExtMst[1]**	Write address atomic type	O8	48	M36
AXI master port	**AWCACHE_ExtMst[0]**	Write address cache attribute	O8	49	M35
AXI master port	**AWCACHE_ExtMst[1]**	Write address cache attribute	O8	50	M34
AXI master port	**AWCACHE_ExtMst[2]**	Write address cache attribute	O8	51	M33
AXI master port	**AWCACHE_ExtMst[3]**	Write address cache attribute	O8	52	M32

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI master port	**TZPROTMAXI[0]**	TZ sideband signals on master port	O8	53	N39
AXI master port	**TZPROTMAXI[1]**	TZ sideband signals on master port	O8	54	N38
AXI master port	**TZPROTMAXI[2]**	TZ sideband signals on master port	O8	55	N37
AXI master port	**TZPROTMAXI[3]**	TZ sideband signals on master port	O8	56	N36
AXI master port	**AWPROT_ExtMst[0]**	Write address protection level	O8	57	N35
AXI master port	**AWPROT_ExtMst[1]**	Write address protection level	O8	58	N34
AXI master port	**AWPROT_ExtMst[2]**	Write address protection level	O8	59	N33
AXI master port	**AWID_ExtMst[0]**	Write address ID	O8	60	N32
AXI master port	**AWID_ExtMst[1]**	Write address ID	O8	61	P39
AXI master port	**AWID_ExtMst[2]**	Write address ID	O8	62	P38
AXI master port	**AWID_ExtMst[3]**	Write address ID	O8	63	P37
AXI master port	**AWID_ExtMst[4]**	Write address ID	O8	64	P36
AXI master port	**AWID_ExtMst[5]**	Write address ID	O8	65	P35
AXI master port	**ACLK_ExtMst**	Exported ACLK	O12	66	P34
AXI master port	**ARPROT_ExtMst[0]**	Read address protection level	O8	67	P33
AXI master port	**ARPROT_ExtMst[1]**	Read address protection level	O8	68	P32

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI master port	**ARPROT_ExtMst[2]**	Read address protection level	O8	69	R39
AXI master port	**ARADDR_ExtMst[0]**	Read address bus	O8	70	R38
AXI master port	**ARADDR_ExtMst[1]**	Read address bus	O8	71	R37
AXI master port	**ARADDR_ExtMst[2]**	Read address bus	O8	72	R36
AXI master port	**ARADDR_ExtMst[3]**	Read address bus	O8	73	R35
AXI master port	**ARADDR_ExtMst[4]**	Read address bus	O8	74	R34
AXI master port	**ARADDR_ExtMst[5]**	Read address bus	O8	75	R33
AXI master port	**ARADDR_ExtMst[6]**	Read address bus	O8	76	R32
AXI master port	**ARADDR_ExtMst[7]**	Read address bus	O8	77	T39
AXI master port	**ARADDR_ExtMst[8]**	Read address bus	O8	78	T38
AXI master port	**ARADDR_ExtMst[9]**	Read address bus	O8	79	T37
AXI master port	**ARADDR_ExtMst[10]**	Read address bus	O8	80	T36
AXI master port	**ARADDR_ExtMst[11]**	Read address bus	O8	81	T35
AXI master port	**ARADDR_ExtMst[12]**	Read address bus	O8	82	T34
AXI master port	**ARADDR_ExtMst[13]**	Read address bus	O8	83	T33
AXI master port	**ARADDR_ExtMst[14]**	Read address bus	O8	84	T32
AXI master port	**ARADDR_ExtMst[15]**	Read address bus	O8	85	U39
AXI master port	**ARADDR_ExtMst[16]**	Read address bus	O8	86	U38
AXI master port	**ARADDR_ExtMst[17]**	Read address bus	O8	87	U37
AXI master port	**ARADDR_ExtMst[18]**	Read address bus	O8	88	U36
AXI master port	**ARADDR_ExtMst[19]**	Read address bus	O8	89	U35
AXI master port	**ARADDR_ExtMst[20]**	Read address bus	O8	90	U34
AXI master port	**ARADDR_ExtMst[21]**	Read address bus	O8	91	U33

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI master port	**ARADDR_ExtMst[22]**	Read address bus	O8	92	U32
AXI master port	**ARADDR_ExtMst[23]**	Read address bus	O8	93	V39
AXI master port	**ARREADY_ExtMst**	Read address ready	I	94	V38
AXI master port	**ARVALID_ExtMst**	Read address valid	O8	95	V37
AXI master port	**ARADDR_ExtMst[24]**	Read address bus	O8	96	V36
AXI master port	**ARADDR_ExtMst[25]**	Read address bus	O8	97	V35
AXI master port	**ARADDR_ExtMst[26]**	Read address bus	O8	98	V34
AXI master port	**ARADDR_ExtMst[27]**	Read address bus	O8	99	V33
AXI master port	**ARADDR_ExtMst[28]**	Read address bus	O8	100	V32
AXI master port	**ARADDR_ExtMst[29]**	Read address bus	O8	101	W36
AXI master port	**ARADDR_ExtMst[30]**	Read address bus	O8	102	W32
AXI master port	**ARADDR_ExtMst[31]**	Read address bus	O8	103	Y39
AXI master port	**ARLEN_ExtMst[0]**	Read address burst size	O8	104	Y38
AXI master port	**ARLEN_ExtMst[1]**	Read address burst size	O8	105	Y37
AXI master port	**ARLEN_ExtMst[2]**	Read address burst size	O8	106	Y36
AXI master port	**ARLEN_ExtMst[3]**	Read address burst size	O8	107	Y35
AXI master port	**ARSIZE_ExtMst[0]**	Read address transfer size	O8	108	Y34
AXI master port	**ARSIZE_ExtMst[1]**	Read address transfer size	O8	109	Y33
AXI master port	**ARBURST_ExtMst[0]**	Read address burst type	O8	110	Y32

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI master port	**ARBURST_ExtMst[1]**	Read address burst type	O8	111	AA39
AXI master port	**ARLOCK_ExtMst[0]**	-	O8	112	AA38
AXI master port	**ARLOCK_ExtMst[1]**	-	O8	113	AA37
AXI master port	**ARCACHE_ExtMst[0]**	-	O8	114	AA36
AXI master port	**ARCACHE_ExtMst[1]**	-	O8	115	AA35
AXI master port	**ARCACHE_ExtMst[2]**	-	O8	116	AA34
AXI master port	**ARCACHE_ExtMst[3]**	-	O8	117	AA33
AXI master port	**ARID_ExtMst[0]**	-	O8	118	AA32
AXI master port	**ARID_ExtMst[1]**	-	O8	119	AB39
AXI master port	**ARID_ExtMst[2]**	-	O8	120	AB38
AXI master port	**ARID_ExtMst[3]**	-	O8	121	AB37
AXI master port	**ARID_ExtMst[4]**	-	O8	122	AB36
AXI master port	**ARID_ExtMst[5]**	-	O8	123	AB35
AXI master port	**WLAST_ExtMst**	Write data last	O8	124	AB34
AXI master port	**WDATA_ExtMst[0]**	Write data	O8	125	AB33
AXI master port	**WDATA_ExtMst[1]**	Write data	O8	126	AB32
AXI master port	**WDATA_ExtMst[2]**	Write data	O8	127	AC39
AXI master port	**WDATA_ExtMst[3]**	Write data	O8	128	AC38
AXI master port	**WDATA_ExtMst[4]**	Write data	O8	129	AC37
AXI master port	**WDATA_ExtMst[5]**	Write data	O8	130	AC36
AXI master port	**WDATA_ExtMst[6]**	Write data	O8	131	AC35
AXI master port	**WDATA_ExtMst[7]**	Write data	O8	132	AC34
AXI master port	**WDATA_ExtMst[8]**	Write data	O8	133	AC33

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI master port	**WDATA_ExtMst[9]**	Write data	O8	134	AC32
AXI master port	**WDATA_ExtMst[10]**	Write data	O8	135	AD39
AXI master port	**WDATA_ExtMst[11]**	Write data	O8	136	AD38
-	**VSOC**	Digital power, 1.2V	DP	137	AD37
-	**VD33**	I/O power, 3.3V	IOP	138	AD36
-	**VSS**	Common ground	DG	139	AD35
-	**VSS**	Common ground	IOG	140	AD34
AXI master port	**WDATA_ExtMst[12]**	Write data	O8	141	AD33
AXI master port	**WDATA_ExtMst[13]**	Write data	O8	142	AD32
AXI master port	**WDATA_ExtMst[14]**	Write data	O8	143	AE39
AXI master port	**WDATA_ExtMst[15]**	Write data	O8	144	AE38
AXI master port	**WDATA_ExtMst[16]**	Write data	O8	145	AE37
AXI master port	**WDATA_ExtMst[17]**	Write data	O8	146	AE36
AXI master port	**WDATA_ExtMst[18]**	Write data	O8	147	AE35
AXI master port	**WDATA_ExtMst[19]**	Write data	O8	148	AE34
AXI master port	**WDATA_ExtMst[20]**	Write data	O8	149	AE33
AXI master port	**WDATA_ExtMst[21]**	Write data	O8	150	AE32
AXI master port	**WDATA_ExtMst[22]**	Write data	O8	151	AF39
AXI master port	**WDATA_ExtMst[23]**	Write data	O8	152	AF38
AXI master port	**WDATA_ExtMst[24]**	Write data	O8	153	AF37
AXI master port	**WDATA_ExtMst[25]**	Write data	O8	154	AF36
AXI master port	**WDATA_ExtMst[26]**	Write data	O8	155	AF35
AXI master port	**WDATA_ExtMst[27]**	Write data	O8	156	AF34
AXI master port	**WDATA_ExtMst[28]**	Write data	O8	157	AF33

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI master port	**WDATA_ExtMst[29]**	Write data	O8	158	AF32
AXI master port	**WDATA_ExtMst[30]**	Write data	O8	159	AG39
AXI master port	**WDATA_ExtMst[31]**	Write data	O8	160	AG38
AXI master port	**WDATA_ExtMst[32]**	Write data	O8	161	AG37
AXI master port	**WDATA_ExtMst[33]**	Write data	O8	162	AG36
AXI master port	**WDATA_ExtMst[34]**	Write data	O8	163	AG35
AXI master port	**WDATA_ExtMst[35]**	Write data	O8	164	AG34
AXI master port	**WDATA_ExtMst[36]**	Write data	O8	165	AG33
AXI master port	**WDATA_ExtMst[37]**	Write data	O8	166	AG32
AXI master port	**WDATA_ExtMst[38]**	Write data	O8	167	AH39
AXI master port	**WVALID_ExtMst**	Write data valid	O8	168	AH38
AXI master port	**WREADY_ExtMst**	Write data ready	I	169	AH37
AXI master port	**WDATA_ExtMst[39]**	Write data	O8	170	AH36
AXI master port	**WDATA_ExtMst[40]**	Write data	O8	171	AH35
AXI master port	**WDATA_ExtMst[41]**	Write data	O8	172	AH34
AXI master port	**WDATA_ExtMst[42]**	Write data	O8	173	AH33
AXI master port	**WDATA_ExtMst[43]**	Write data	O8	174	AH32
AXI master port	**WDATA_ExtMst[44]**	Write data	O8	175	AJ39
AXI master port	**WDATA_ExtMst[45]**	Write data	O8	176	AJ38
AXI master port	**WDATA_ExtMst[46]**	Write data	O8	177	AJ37
AXI master port	**WDATA_ExtMst[47]**	Write data	O8	178	AJ36
AXI master port	**WDATA_ExtMst[48]**	Write data	O8	179	AJ35
AXI master port	**WDATA_ExtMst[49]**	Write data	O8	180	AJ34
AXI master port	**WDATA_ExtMst[50]**	Write data	O8	181	AJ33

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI master port	**WDATA_ExtMst[51]**	Write data	O8	182	AJ32
AXI master port	**WDATA_ExtMst[52]**	Write data	O8	183	AK39
AXI master port	**WDATA_ExtMst[53]**	Write data	O8	184	AK38
AXI master port	**WDATA_ExtMst[54]**	Write data	O8	185	AK37
AXI master port	**WDATA_ExtMst[55]**	Write data	O8	186	AK36
AXI master port	**WDATA_ExtMst[56]**	Write data	O8	187	AK35
AXI master port	**WDATA_ExtMst[57]**	Write data	O8	188	AK34
AXI master port	**WDATA_ExtMst[58]**	Write data	O8	189	AK33
AXI master port	**WDATA_ExtMst[59]**	Write data	O8	190	AK32
AXI master port	**WDATA_ExtMst[60]**	Write data	O8	191	AL39
AXI master port	**WDATA_ExtMst[61]**	Write data	O8	192	AL38
AXI master port	**WDATA_ExtMst[62]**	Write data	O8	193	AL37
AXI master port	**WDATA_ExtMst[63]**	Write data	O8	194	AL36
AXI master port	**WSTRB_ExtMst[0]**	-	O8	195	AL35
AXI master port	**WSTRB_ExtMst[1]**	-	O8	196	AL34
AXI master port	**WSTRB_ExtMst[2]**	-	O8	197	AL33
AXI master port	**WSTRB_ExtMst[3]**	-	O8	198	AL32
AXI master port	**WSTRB_ExtMst[4]**	-	O8	199	AM39
AXI master port	**WSTRB_ExtMst[5]**	-	O8	200	AM38
AXI master port	**WSTRB_ExtMst[6]**	-	O8	201	AM37
AXI master port	**WSTRB_ExtMst[7]**	-	O8	202	AM36
AXI master port	**WID_ExtMst[0]**	Write data ID	O8	203	AM35
AXI master port	**WID_ExtMst[1]**	Write data ID	O8	204	AM34
AXI master port	**WID_ExtMst[2]**	Write data ID	O8	205	AM33

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI master port	**WID_ExtMst[3]**	Write data ID	O8	206	AN39
AXI master port	**WID_ExtMst[4]**	Write data ID	O8	207	AN38
AXI master port	**WID_ExtMst[5]**	Write data ID	O8	208	AN37
AXI master port	**RDATA_ExtMst[0]**	Read data	I	209	AN36
AXI master port	**RDATA_ExtMst[1]**	Read data	I	210	AN35
AXI master port	**RDATA_ExtMst[2]**	Read data	I	211	AN34
AXI master port	**RDATA_ExtMst[3]**	Read data	I	212	AP39
AXI master port	**RDATA_ExtMst[4]**	Read data	I	213	AP38
AXI master port	**RDATA_ExtMst[5]**	Read data	I	214	AP37
AXI master port	**RDATA_ExtMst[6]**	Read data	I	215	AP36
AXI master port	**RDATA_ExtMst[7]**	Read data	I	216	AP35
AXI master port	**RDATA_ExtMst[8]**	Read data	I	217	AR39
-	**VD33**	I/O power, 3.3V	IOP	218	AR38
-	**VSS**	Common ground	DG	219	AR37
-	**VCORE**	IEM V_{core} voltage, 1.2V nominal	DP	220	AR36
-	**VSOC**	Digital power, 1.2V	DP	221	AT39
-	**VSS**	Common ground	IOG	222	AT38
-	**VSS**	Common ground	DG	223	AT37
AXI master port	**RDATA_ExtMst[9]**	Read data	I	224	AU39
AXI master port	**RDATA_ExtMst[10]**	Read data	I	225	AU38
AXI master port	**RDATA_ExtMst[11]**	Read data	I	226	AV39
AXI master port	**RDATA_ExtMst[12]**	Read data	I	227	AW38
AXI master port	**RDATA_ExtMst[13]**	Read data	I	228	AV37

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI master port	**RDATA_ExtMst[14]**	Read data	I	229	AW37
AXI master port	**RDATA_ExtMst[15]**	Read data	I	230	AU36
AXI master port	**RDATA_ExtMst[16]**	Read data	I	231	AV36
AXI master port	**RDATA_ExtMst[17]**	Read data	I	232	AW36
AXI master port	**RDATA_ExtMst[18]**	Read data	I	233	AT35
AXI master port	**RDATA_ExtMst[19]**	Read data	I	234	AU35
AXI master port	**RDATA_ExtMst[20]**	Read data	I	235	AV35
AXI master port	**RDATA_ExtMst[21]**	Read data	I	236	AW35
AXI master port	**RDATA_ExtMst[22]**	Read data	I	237	AR34
AXI master port	**RDATA_ExtMst[23]**	Read data	I	238	AT34
AXI master port	**RDATA_ExtMst[24]**	Read data	I	239	AU34
AXI master port	**RDATA_ExtMst[25]**	Read data	I	240	AV34
AXI master port	**RDATA_ExtMst[26]**	Read data	I	241	AW34
AXI master port	**RDATA_ExtMst[27]**	Read data	I	242	AP33
AXI master port	**RDATA_ExtMst[28]**	Read data	I	243	AR33
AXI master port	**RDATA_ExtMst[29]**	Read data	I	244	AT33
AXI master port	**RDATA_ExtMst[30]**	Read data	I	245	AU33
AXI master port	**RDATA_ExtMst[31]**	Read data	I	246	AV33
AXI master port	**RDATA_ExtMst[32]**	Read data	I	247	AW33
AXI master port	**RREADY_ExtMst**	Read data ready	O8	248	AN32
AXI master port	**RVALID_ExtMst**	Read data valid	I	249	AP32
AXI master port	**RDATA_ExtMst[33]**	Read data	I	250	AR32
AXI master port	**RDATA_ExtMst[34]**	Read data	I	251	AT32
AXI master port	**RDATA_ExtMst[35]**	Read data	I	252	AU32

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI master port	**RDATA_ExtMst[36]**	Read data	I	253	AV32
AXI master port	**RDATA_ExtMst[37]**	Read data	I	254	AW32
AXI master port	**RDATA_ExtMst[38]**	Read data	I	255	AK31
AXI master port	**RDATA_ExtMst[39]**	Read data	I	256	AL31
AXI master port	**RDATA_ExtMst[40]**	Read data	I	257	AM31
AXI master port	**RDATA_ExtMst[41]**	Read data	I	258	AN31
AXI master port	**RDATA_ExtMst[42]**	Read data	I	259	AP31
AXI master port	**RDATA_ExtMst[43]**	Read data	I	260	AR31
AXI master port	**RDATA_ExtMst[44]**	Read data	I	261	AT31
AXI master port	**RDATA_ExtMst[45]**	Read data	I	262	AU31
AXI master port	**RDATA_ExtMst[46]**	Read data	I	263	AV31
AXI master port	**RDATA_ExtMst[47]**	Read data	I	264	AW31
AXI master port	**RDATA_ExtMst[48]**	Read data	I	265	AK30
AXI master port	**RDATA_ExtMst[49]**	Read data	I	266	AL30
AXI master port	**RDATA_ExtMst[50]**	Read data	I	267	AM30
AXI master port	**RDATA_ExtMst[51]**	Read data	I	268	AN30
AXI master port	**RDATA_ExtMst[52]**	Read data	I	269	AP30
AXI master port	**RDATA_ExtMst[53]**	Read data	I	270	AR30
AXI master port	**RDATA_ExtMst[54]**	Read data	I	271	AT30
AXI master port	**RDATA_ExtMst[55]**	Read data	I	272	AU30
AXI master port	**RDATA_ExtMst[56]**	Read data	I	273	AV30
AXI master port	**RDATA_ExtMst[57]**	Read data	I	274	AW30
AXI master port	**RDATA_ExtMst[58]**	Read data	I	275	AK29
AXI master port	**RDATA_ExtMst[59]**	Read data	I	276	AL29

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI master port	**RDATA_ExtMst[60]**	Read data	I	277	AM29
AXI master port	**RDATA_ExtMst[61]**	Read data	I	278	AN29
AXI master port	**RDATA_ExtMst[62]**	Read data	I	279	AP29
AXI master port	**RDATA_ExtMst[63]**	Read data	I	280	AR29
AXI master port	**RID_ExtMst[0]**	Read data ID	I	281	AT29
AXI master port	**RID_ExtMst[1]**	Read data ID	I	282	AU29
AXI master port	**RID_ExtMst[2]**	Read data ID	I	283	AV29
AXI master port	**RID_ExtMst[3]**	Read data ID	I	284	AW29
AXI master port	**RID_ExtMst[4]**	Read data ID	I	285	AK28
AXI master port	**RID_ExtMst[5]**	Read data ID	I	286	AL28
AXI master port	**RRESP_ExtMst[0]**	-	I	287	AM28
AXI master port	**RLAST_ExtMst**	-	I	288	AN28
AXI master port	**RRESP_ExtMst[1]**	-	I	289	AP28
AXI master port	**BREADY_ExtMst**	-	O8	290	AR28
AXI master port	**BVALID_ExtMst**	-	I	291	AT28
AXI master port	**BRESP_ExtMst[0]**	-	I	292	AU28
AXI master port	**BRESP_ExtMst[1]**	-	I	293	AV28
AXI master port	**BID_ExtMst[0]**	-	I	294	AW28
AXI master port	**BID_ExtMst[1]**	-	I	295	AK27
AXI master port	**BID_ExtMst[2]**	-	I	296	AL27
AXI master port	**BID_ExtMst[3]**	-	I	297	AM27
AXI master port	**BID_ExtMst[4]**	-	I	298	AN27
AXI master port	**BID_ExtMst[5]**	-	I	299	AP27
SoC config control	**CONFIGINIT**	-	I	300	AR27

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
SoC config control	**CONFIGCLRn**	-	I	301	AT27
-	**TESTMODE**	-	I	302	AU27
-	**nPORESET**	SOC power on reset	I	303	AV27
-	**nRESET**	SOC soft reset	I	304	AW27
Diagnostic	**SHUTDOWN**	-	O8	305	AK26
Diagnostic	**DORMANT**	-	O8	306	AL26
Diagnostic	**IECPWRREQ**	-	O8	307	AM26
Diagnostic	**DVCCRNTEQTGT**	-	O8	308	AN26
Diagnostic	**DCGCRNTEQTGT**	-	O8	309	AP26
Diagnostic	**PLLFIXEDLOCK**	-	O8	310	AR26
Diagnostic	**PLL1LOCK**	-	O8	311	AT26
Diagnostic	**PLL2LOCK**	-	O8	312	AU26
Diagnostic	**nVSOCRST**	-	O8	313	AV26
RTC clock	**CLK1HZ**	-	I	314	AW26
Timer clock	**TIMCLK**	-	I	315	AK25
APC powerwise clock	**APC_SCLK**	-	B	316	AL25
APC powerwise data	**APC_SPWI**	-	B	317	AM25
APC Clock	**APC_REFCLK_C**	-	I	318	AN25
External IEC Maxperf input	**WAKEUP**	-	I	319	AP25
-	**VSS**	Common ground	DG	320	AR25
-	**VSOC**	Digital power, 1.2V	DP	321	AT25
-	**VD33**	I/O power, 3.3V	IOP	322	AU25
CLCD PL111	**CLAC**	LCD	O8	323	AV25

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
CLCD PL111	**CLD[0]**	LCD panel data	O8	324	AW25
CLCD PL111	**CLD[1]**	LCD panel data	O8	325	AK24
CLCD PL111	**CLD[2]**	LCD panel data	O8	326	AL24
CLCD PL111	**CLD[3]**	LCD panel data	O8	327	AM24
CLCD PL111	**CLD[4]**	LCD panel data	O8	328	AN24
CLCD PL111	**CLD[5]**	LCD panel data	O8	329	AP24
CLCD PL111	**CLD[6]**	LCD panel data	O8	330	AR24
CLCD PL111	**CLD[7]**	LCD panel data	O8	331	AT24
CLCD PL111	**CLD[8]**	LCD panel data	O8	332	AU24
CLCD PL111	**CLD[9]**	LCD panel data	O8	333	AV24
CLCD PL111	**CLD[10]**	LCD panel data	O8	334	AW24
CLCD PL111	**CLD[11]**	LCD panel data	O8	335	AK23
CLCD PL111	**CLD[12]**	LCD panel data	O8	336	AL23
CLCD PL111	**CLD[13]**	LCD panel data	O8	337	AM23
CLCD PL111	**CLD[14]**	LCD panel data	O8	338	AN23
CLCD PL111	**CLD[15]**	LCD panel data	O8	339	AP23
CLCD PL111	**CLD[16]**	LCD panel data	O8	340	AR23
CLCD PL111	**CLD[17]**	LCD panel data	O8	341	AT23
CLCD PL111	**CLD[18]**	LCD panel data	O8	342	AU23
CLCD PL111	**CLD[19]**	LCD panel data	O8	343	AV23
CLCD PL111	**CLD[20]**	LCD panel data	O8	344	AW23
CLCD PL111	**CLD[21]**	LCD panel data	O8	345	AJ22
CLCD PL111	**CLD[22]**	LCD panel data	O8	346	AK22
CLCD PL111	**CLD[23]**	LCD panel data	O8	347	AL22

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
CLCD PL111	**CLFP**	Frame pulse/vertical sync	O8	348	AM22
CLCD PL111	**CLLE**	Line end signal	O8	349	AN22
CLCD PL111	**CLLP**	Line sync pulse	O8	350	AP22
CLCD PL111	**CLPOWER**	LCD panel enable	O8	351	AR22
CLCD PL111	**CLCP**	-	O8	352	AT22
CLCD PL111	**LCDCLK**	-	I	353	AU22
-	**PLLREFCLK**	Clock input to main PLL	I	354	AV22
-	**VDDPLLFA**	PLL fixed analogue power, 3.3V	AP	355	AW22
-	**VSSPLLFA**	PLL fixed analogue ground	AG	356	AJ21
-	**PLLREFCLK1**	Clock input to PLL1	I	357	AK21
-	**VDDPLL1A**	PLL #1 analogue power, 3.3V	AP	358	AL21
-	**VSSPLL1A**	PLL #1 analogue ground	AG	359	AM21
-	**PLLREFCLK2**	Clock input to PLL1	I	360	AN21
-	**VDDPLL2A**	PLL #2 analogue power, 3.3V	AP	361	AP21
-	**VSSPLL2A**	PLL #2 analogue ground	AG	362	AR21
DMC PL340	**DLL_GND**	DLL common ground	AG	363	AT21
DMC PL340	**DLL_VAA**	DLL analogue power, 3.3V	AP	364	AU21

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
DMC PL340	**DLL_VDD**	DLL digital power, 1.2V	AP	365	AV21
-	**VD18**	I/O power, 1.8V	MDOP	366	AW21
-	**VSS**	Common ground	MDOG	367	AJ20
-	**VSOC**	Digital power, 1.2V	MDCP	368	AK20
-	**VSS**	Common ground	MDCG	369	AL20
-	**VRF**	I/O power, 0.9V - 1.8V differential input reference	MDVRF	370	AM20
-	**VD25**	I/O power, 2.5V - 1.8V predrive	MDPD	371	AN20
DMC PL340	**DQ[0]**	Mobile DDR data	MDOB	372	AP20
DMC PL340	**DQ[1]**	Mobile DDR data	MDOB	373	AR20
DMC PL340	**DQ[2]**	Mobile DDR data	MDOB	374	AT20
DMC PL340	**DQ[3]**	Mobile DDR data	MDOB	375	AU20
DMC PL340	**DQS[0]**	Mobile DDR data qualifier strobe	MDOB	376	AV20
DMC PL340	**DQ[4]**	Mobile DDR data	MDOB	377	AW20
DMC PL340	**DQ[5]**	Mobile DDR data	MDOB	378	AJ19
DMC PL340	**DQ[6]**	Mobile DDR data	MDOB	379	AK19
DMC PL340	**DQ[7]**	Mobile DDR data	MDOB	380	AL19
-	**VD25**	I/O power, 2.5V - 1.8V predrive	MDPD	381	AM19
DMC PL340	**DQ[8]**	Mobile DDR data	MDOB	382	AN19
DMC PL340	**DQ[9]**	Mobile DDR data	MDOB	383	AP19
DMC PL340	**DQ[10]**	Mobile DDR data	MDOB	384	AR19

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
DMC PL340	**DQ[11]**	Mobile DDR data	MDOB	385	AT19
DMC PL340	**DQS[1]**	Mobile DDR data qualifier strobe	MDOB	386	AU19
DMC PL340	**DQ[12]**	Mobile DDR data	MDOB	387	AV19
DMC PL340	**DQ[13]**	Mobile DDR data	MDOB	388	AW19
DMC PL340	**DQ[14]**	Mobile DDR data	MDOB	389	AJ18
DMC PL340	**DQ[15]**	Mobile DDR data	MDOB	390	AK18
-	**VD25**	I/O power, 2.5V - 1.8V predrive	MDPD	391	AL18
DMC PL340	**DQ[16]**	Mobile DDR data	MDOB	392	AM18
DMC PL340	**DQ[17]**	Mobile DDR data	MDOB	393	AN18
DMC PL340	**DQ[18]**	Mobile DDR data	MDOB	394	AP18
DMC PL340	**DQ[19]**	Mobile DDR data	MDOB	395	AR18
DMC PL340	**DQS[2]**	Mobile DDR data qualifier strobe	MDOB	396	AT18
DMC PL340	**DQ[20]**	Mobile DDR data	MDOB	397	AU18
DMC PL340	**DQ[21]**	Mobile DDR data	MDOB	398	AV18
DMC PL340	**DQ[22]**	Mobile DDR data	MDOB	399	AW18
DMC PL340	**DQ[23]**	Mobile DDR data	MDOB	400	AK17
-	**VD25**	I/O power, 2.5V - 1.8V predrive	MDPD	401	AL17
DMC PL340	**DQ[24]**	Mobile DDR data	MDOB	402	AM17
DMC PL340	**DQ[25]**	Mobile DDR data	MDOB	403	AN17
DMC PL340	**DQ[26]**	Mobile DDR data	MDOB	404	AP17
DMC PL340	**DQ[27]**	Mobile DDR data	MDOB	405	AR17

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
DMC PL340	**DQS[3]**	Mobile DDR data qualifier strobe	MDOB	406	AT17
DMC PL340	**DQ[28]**	Mobile DDR data	MDOB	407	AU17
DMC PL340	**DQ[29]**	Mobile DDR data	MDOB	408	AV17
DMC PL340	**DQ[30]**	Mobile DDR data	MDOB	409	AW17
DMC PL340	**DQ[31]**	Mobile DDR data	MDOB	410	AK16
-	**VD25**	I/O power, 2.5V - 1.8V predrive	MDPD	411	AL16
DMC PL340	**clk_out_n[0]**	Mobile DDR CK#	MDO	412	AM16
DMC PL340	**clk_out[0]**	Mobile DDR CK	MDO	413	AN16
DMC PL340	**clk_out_n[1]**	Mobile DDR CK#	MDO	414	AP16
DMC PL340	**clk_out[1]**	Mobile DDR CK	MDO	415	AR16
DMC PL340	**clk_out_n[2]**	Mobile DDR CK#	MDO	416	AT16
DMC PL340	**clk_out[2]**	Mobile DDR CK	MDO	417	AU16
DMC PL340	**clk_out_n[3]**	Mobile DDR CK#	MDO	418	AV16
DMC PL340	**clk_out[3]**	Mobile DDR CK	MDO	419	AW16
-	**VD25**	I/O power, 2.5V - 1.8V predrive	MDPD	420	AK15
DMC PL340	**ras_n**	Mobile DDR nRAS	MDO	421	AL15
DMC PL340	**cas_n**	Mobile DDR nCAS	MDO	422	AM15
DMC PL340	**we_n**	Mobile DDR nWE	MDO	423	AN15
DMC PL340	**cke**	Mobile DDR CKE	MDO	424	AP15
DMC PL340	**cs_n[0]**	Mobile DDR nCS	MDO	425	AR15
DMC PL340	**cs_n[1]**	Mobile DDR nCS	MDO	426	AT15

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
DMC PL340	**cs_n[2]**	Mobile DDR nCS	MDO	427	AU15
DMC PL340	**cs_n[3]**	Mobile DDR nCS	MDO	428	AV15
DMC PL340	**add[0]**	Mobile DDR address	MDO	429	AW15
DMC PL340	**add[1]**	Mobile DDR address	MDO	430	AK14
-	**VD25**	I/O power, 2.5V - 1.8V predrive	MDPD	431	AL14
DMC PL340	**add[2]**	Mobile DDR address	MDO	432	AM14
DMC PL340	**add[3]**	Mobile DDR address	MDO	433	AN14
DMC PL340	**add[4]**	Mobile DDR address	MDO	434	AP14
DMC PL340	**add[5]**	Mobile DDR address	MDO	435	AR14
DMC PL340	**add[6]**	Mobile DDR address	MDO	436	AT14
DMC PL340	**add[7]**	Mobile DDR address	MDO	437	AU14
DMC PL340	**add[8]**	Mobile DDR address	MDO	438	AV14
-	**VD25**	I/O power, 2.5V - 1.8V predrive	MDPD	439	AW14
DMC PL340	**add[9]**	Mobile DDR address	MDO	440	AK13
DMC PL340	**add[10]**	Mobile DDR address	MDO	441	AL13
DMC PL340	**add[11]**	Mobile DDR address	MDO	442	AM13

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
DMC PL340	**add[12]**	Mobile DDR address	MDO	443	AN13
DMC PL340	**add[13]**	Mobile DDR address	MDO	444	AP13
DMC PL340	**add[14]**	Mobile DDR address	MDO	445	AR13
DMC PL340	**add[15]**	Mobile DDR address	MDO	446	AT13
DMC PL340	**ba[0]**	Mobile DDR bank select	MDO	447	AU13
DMC PL340	**ba[1]**	Mobile DDR bank select	MDO	448	AV13
DMC PL340	**dqm[0]**	Mobile DDR data qualifier mask	MDO	449	AW13
DMC PL340	**dqm[1]**	Mobile DDR data qualifier mask	MDO	450	AK12
DMC PL340	**dqm[2]**	Mobile DDR data qualifier mask	MDO	451	AL12
DMC PL340	**dqm[3]**	Mobile DDR data qualifier mask	MDO	452	AM12
-	**VD25**	I/O power, 2.5V - 1.8V predrive	MDPD	453	AN12
-	**VD18**	I/O power, 1.8V	MDOP	454	AP12
-	**VSS**	Common ground	MDOG	455	AR12
-	**VSOC**	Digital power, 1.2V	MDCP	456	AT12
-	**VSS**	Common ground	MDCG	457	AU12
-	**VRF**	I/O power, 0.9V - 1.8V differential input reference	MDVRF	458	AV12

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
CTM	**CTMCHIN2[0]**	-	I	459	AW12
CTM	**CTMCHIN2[1]**	-	I	460	AK11
CTM	**CTMCHIN2[2]**	-	I	461	AL11
CTM	**CTMCHIN2[3]**	-	I	462	AM11
CTM	**CTMCHINACK2[0]**	-	O8	463	AN11
CTM	**CTMCHINACK2[1]**	-	O8	464	AP11
CTM	**CTMCHINACK2[2]**	-	O8	465	AR11
CTM	**CTMCHINACK2[3]**	-	O8	466	AT11
CTM	**CTMCHOUT2[0]**	-	O8	467	AU11
CTM	**CTMCHOUT2[1]**	-	O8	468	AV11
CTM	**CTMCHOUT2[2]**	-	O8	469	AW11
CTM	**CTMCHOUT2[3]**	-	O8	470	AK10
CTM	**CTMCHOUTACK2[0]**	-	I	471	AL10
CTM	**CTMCHOUTACK2[1]**	-	I	472	AM10
CTM	**CTMCHOUTACK2[2]**	-	I	473	AN10
CTM	**CTMCHOUTACK2[3]**	-	I	474	AP10
CTM	**CTMCHIN3[0]**	-	I	475	AR10
CTM	**CTMCHIN3[1]**	-	I	476	AT10
CTM	**CTMCHIN3[2]**	-	I	477	AU10
CTM	**CTMCHIN3[3]**	-	I	478	AV10
CTM	**CTMCHINACK3[0]**	-	O8	479	AW10
CTM	**CTMCHINACK3[1]**	-	O8	480	AK9
CTM	**CTMCHINACK3[2]**	-	O8	481	AL9
CTM	**CTMCHINACK3[3]**	-	O8	482	AM9

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
CTM	**CTMCHOUT3[0]**	-	O8	483	AN9
CTM	**CTMCHOUT3[1]**	-	O8	484	AP9
CTM	**CTMCHOUT3[2]**	-	O8	485	AR9
CTM	**CTMCHOUT3[3]**	-	O8	486	AT9
CTM	**CTMCHOUTACK3[0]**	-	I	487	AU9
CTM	**CTMCHOUTACK3[1]**	-	I	488	AV9
CTM	**CTMCHOUTACK3[2]**	-	I	489	AW9
CTM	**CTMCHOUTACK3[3]**	-	I	490	AM8
SSMC PL093	**nSMBLS[0]**	Byte lane select	O8	491	AN8
SSMC PL093	**nSMBLS[1]**	Byte lane select	O8	492	AP8
SSMC PL093	**nSMBLS[2]**	Byte lane select	O8	493	AR8
SSMC PL093	**nSMBLS[3]**	Byte lane select	O8	494	AT8
SSMC PL093	**nSMBURSTWAITIN**	Synchronous burst wait	I	495	AU8
SSMC PL093	**nSMWEN**	Write enable	O8	496	AV8
SSMC PL093	**nSMCS[0]**	Chip enable	O8	497	AW8
SSMC PL093	**nSMCS[1]**	Chip enable	O8	498	AN7
SSMC PL093	**nSMCS[2]**	Chip enable	O8	499	AP7
SSMC PL093	**nSMCS[3]**	Chip enable	O8	500	AR7
SSMC PL093	**nSMCS[4]**	Chip enable	O8	501	AT7
SSMC PL093	**nSMCS[5]**	Chip enable	O8	502	AU7
SSMC PL093	**nSMCS[6]**	Chip enable	O8	503	AV7
SSMC PL093	**nSMCS[7]**	Chip enable	O8	504	AW7
SSMC PL093	**nSMOEN**	Output enable	O8	505	AP6

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
SSMC PL093	**SMADDR[0]**	-	O8	506	AR6
SSMC PL093	**SMADDR[1]**	-	O8	507	AT6
SSMC PL093	**SMADDR[2]**	-	O8	508	AU6
SSMC PL093	**SMADDR[3]**	-	O8	509	AV6
SSMC PL093	**SMADDR[4]**	-	O8	510	AW6
SSMC PL093	**SMADDR[5]**	-	O8	511	AR5
SSMC PL093	**SMADDR[6]**	-	O8	512	AT5
SSMC PL093	**SMADDR[7]**	-	O8	513	AU5
SSMC PL093	**SMADDR[8]**	-	O8	514	AV5
SSMC PL093	**SMADDR[9]**	-	O8	515	AW5
SSMC PL093	**SMADDR[10]**	-	O8	516	AT4
SSMC PL093	**SMADDR[11]**	-	O8	517	AU4
SSMC PL093	**SMADDR[12]**	-	O8	518	AV4
SSMC PL093	**SMADDR[13]**	-	O8	519	AW4
SSMC PL093	**SMADDR[14]**	-	O8	520	AU3
SSMC PL093	**SMADDR[15]**	-	O8	521	AV3
SSMC PL093	**SMADDR[16]**	-	O8	522	AW3
-	**VD33**	I/O power, 3.3V	IOP	523	AV2
-	**VSS**	Common ground	DG	524	AW2
-	**VD33**	I/O power, 3.3V	IOP	525	AW1
-	**VSS**	Common ground	IOG	526	AU1
-	**VSOC**	Digital power, 1.2V	DP	527	AT1
-	**VSS**	Common ground	DG	528	AT2
-	**VD33**	I/O power, 3.3V	IOP	529	AR1

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
SSMC PL093	**SMADDR[17]**	-	O8	530	AR2
SSMC PL093	**SMADDR[18]**	-	O8	531	AR3
SSMC PL093	**SMADDR[19]**	-	O8	532	AP1
SSMC PL093	**SMADDR[20]**	-	O8	533	AP2
SSMC PL093	**SMADDR[21]**	-	O8	534	AP3
SSMC PL093	**SMADDR[22]**	-	O8	535	AP4
SSMC PL093	**SMADDR[23]**	-	O8	536	AN1
SSMC PL093	**SMADDR[24]**	-	O8	537	AN2
SSMC PL093	**SMADDR[25]**	-	O8	538	AN3
SSMC PL093	**SMADDRVALID**	-	O8	539	AN4
SSMC PL093	**SMBAA**	-	O8	540	AN5
SSMC PL093	**SMCANCELWAIT**	-	I	541	AM1
SSMC PL093	**SMCLK[0]**	-	O12	542	AM2
SSMC PL093	**SMCLK[1]**	-	O12	543	AM3
SSMC PL093	**SMCLK[2]**	-	O12	544	AM4
SSMC PL093	**SMCLK[3]**	-	O12	545	AM5
SSMC PL093	**SMFBCLKIN**	-	I	546	AM6
SSMC PL093	**SMDATA[0]**	-	B	547	AL1
SSMC PL093	**SMDATA[1]**	-	B	548	AL2
SSMC PL093	**SMDATA[2]**	-	B	549	AL3
SSMC PL093	**SMDATA[3]**	-	B	550	AL4
SSMC PL093	**SMDATA[4]**	-	B	551	AL5
SSMC PL093	**SMDATA[5]**	-	B	552	AL6
SSMC PL093	**SMDATA[6]**	-	B	553	AL7

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
SSMC PL093	**SMDATA[7]**	-	B	554	AK1
SSMC PL093	**SMDATA[8]**	-	B	555	AK2
SSMC PL093	**SMDATA[9]**	-	B	556	AK3
SSMC PL093	**SMDATA[10]**	-	B	557	AK4
SSMC PL093	**SMDATA[11]**	-	B	558	AK5
SSMC PL093	**SMDATA[12]**	-	B	559	AK6
SSMC PL093	**SMDATA[13]**	-	B	560	AK7
SSMC PL093	**SMDATA[14]**	-	B	561	AK8
SSMC PL093	**SMDATA[15]**	-	B	562	AJ1
SSMC PL093	**SMDATA[16]**	-	B	563	AJ2
SSMC PL093	**SMDATA[17]**	-	B	564	AJ3
SSMC PL093	**SMDATA[18]**	-	B	565	AJ4
SSMC PL093	**SMDATA[19]**	-	B	566	AJ5
SSMC PL093	**SMDATA[20]**	-	B	567	AJ6
SSMC PL093	**SMDATA[21]**	-	B	568	AJ7
SSMC PL093	**SMDATA[22]**	-	B	569	AJ8
SSMC PL093	**SMDATA[23]**	-	B	570	AH1
SSMC PL093	**SMDATA[24]**	-	B	571	AH2
SSMC PL093	**SMDATA[25]**	-	B	572	AH3
SSMC PL093	**SMDATA[26]**	-	B	573	AH4
SSMC PL093	**SMDATA[27]**	-	B	574	AH5
SSMC PL093	**SMDATA[28]**	-	B	575	AH6
SSMC PL093	**SMDATA[29]**	-	B	576	AH7
SSMC PL093	**SMDATA[30]**	-	B	577	AH8

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
SSMC PL093	**SMDATA[31]**	-	B	578	AG1
SSMC PL093	**SMWAIT**	-	I	579	AG2
UART	**UARTCLK**	-	I	580	AG3
UART 0 PL011	**nUART0CTS**	-	I	581	AG4
UART 0 PL011	**UART0RXD**	-	I	582	AG5
UART 0 PL011	**nUART0DCD**	-	I	583	AG6
UART 0 PL011	**nUART0DSR**	-	I	584	AG7
UART 0 PL011	**nUART0RI**	-	I	585	AG8
UART 0 PL011	**UART0SIRIN**	-	I	586	AF1
UART 0 PL011	**UART0TXD**	-	O8	587	AF2
UART 0 PL011	**nUART0RTS**	-	O8	588	AF3
UART 0 PL011	**nUART0SIROUT**	-	O8	589	AF4
UART 0 PL011	**nUART0OUT1**	-	O8	590	AF5
UART 0 PL011	**nUART0OUT2**	-	O8	591	AF6
UART 0 PL011	**nUART0DTR**	-	O8	592	AF7
UART 0 PL011	**UART1RXD**	-	I	593	AF8
UART 0 PL011	**nUART1CTS**	-	I	594	AE1
UART 1 PL011	**UART1TXD**	-	O8	595	AE2
UART1PL011	**nUART1RTS**	-	O8	596	AE3
UART2PL011	**UART2RXD**	-	I	597	AE4
UART2PL011	**nUART2CTS**	-	I	598	AE5
UART2PL011	**UART2TXD**	-	O8	599	AE6
UART2PL011	**nUART2RTS**	-	O8	600	AE7
UART3PL011	**UART3RXD**	-	I	601	AE8

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
UART3PL011	**nUART3CTS**	-	I	602	AD1
UART3PL011	**UART3TXD**	-	O8	603	AD2
UART3PL011	**nUART3RTS**	-	O8	604	AD3
SSP PL022	**RingOscOut1**	-	O8	605	AD4
SSP PL022	**RingOscOut2**	-	O8	606	AD5
SSP PL022	**RingOscOut3**	-	O8	607	AD6
SSP PL022	**RingOscEn**	-	I	608	AD7
SSP PL022	**RingOscTest**	-	I	609	AD8
SSP PL022	**RingOscDiv4**	-	I	610	AC1
SSP PL022	**GPIO[0]**	-	B	611	AC2
SSP PL022	**GPIO[1]**	-	B	612	AC3
SSP PL022	**GPIO[2]**	-	B	613	AC4
SSP PL022	**GPIO[3]**	-	B	614	AC5
SSP PL022	**GPIO[4]**	-	B	615	AC6
SSP PL022	**GPIO[5]**	-	B	616	AC7
SSP PL022	**GPIO[6]**	-	B	617	AC8
SSP PL022	**GPIO[7]**	-	B	618	AB1
SSP PL022	**SSPCLK**	External clock for SSp	I	619	AB2
SSP PL022	**SSPFSSIN**	-	I	620	AB3
SSP PL022	**SSPRXD**	-	I	621	AB4
SSP PL022	**SSPCLKIN**	-	I	622	AB5
SSP PL022	**SSPFSSOUT**	-	O8	623	AB6
SSP PL022	**SSPTXD**	-	T8	624	AB7

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
SSP PL022	**SSPCLKOUT**	-	T8	625	AB8
-	**VSOC**	Digital power, 1.2V	DP	626	AA1
-	**VD33**	I/O power, 3.3V	IOP	627	AA2
-	**VSS**	Common ground	IOG	628	AA3
-	**VSS**	Common ground	DG	629	AA4
DAP/BSCAN	**RTCK**	ARM1176JZF return clock	O12	630	AA5
DAP/BSCAN	**TCK**	Test clock	I	631	AA6
DAP/BSCAN	**TMS**	Test mode select	IPU	632	AA7
DAP/BSCAN	**nTRST**	Test reset	IPU	633	AA8
DAP/BSCAN	**TDI**	Test data input	IPU	634	Y1
DAP/BSCAN	**TDO**	Test data output	T8	635	Y2
DAP/SCAN	**JTAGBSTEN**	Test, Bscan enable	I	636	Y3
DAP/SCAN	**JTAGDAPEN**	Test, DAP enable	I	637	Y4
DAP/SCAN	**JTAGSYNCBYPASS**	IEM JTAG test bypass	I	638	Y5
ATB	**ATCLKEXT**	ATB exported clock	O12	639	Y6
ATB	**ATREADYEXT**	ATB ready	I	640	Y7
ATB	**AFVALIDEXT**	-	I	641	Y8
ATB	**ATVALIDEXT**	ATB valid	O8	642	W4
ATB	**AFREADYEXT**	-	O8	643	W8
ATB	**ATIDEXT[0]**	ATB ID	O8	644	V1
ATB	**ATIDEXT[1]**	ATB ID	O8	645	V2
ATB	**ATIDEXT[2]**	ATB ID	O8	646	V3

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
ATB	**ATIDEXT[3]**	ATB ID	O8	647	V4
ATB	**ATIDEXT[4]**	ATB ID	O8	648	V5
ATB	**ATIDEXT[5]**	ATB ID	O8	649	V6
ATB	**ATIDEXT[6]**	ATB ID	O8	650	V7
ATB	**ATBYTESEXT[0]**	ATB bytes	O8	651	V8
ATB	**ATBYTESEXT[1]**	ATB bytes	O8	652	U1
ATB	**ATDATAEXT[0]**	ATB data	O8	653	U2
ATB	**ATDATAEXT[1]**	ATB data	O8	654	U3
ATB	**ATDATAEXT[2]**	ATB data	O8	655	U4
ATB	**ATDATAEXT[3]**	ATB data	O8	656	U5
ATB	**ATDATAEXT[4]**	ATB data	O8	657	U6
ATB	**ATDATAEXT[5]**	ATB data	O8	658	U7
ATB	**ATDATAEXT[6]**	ATB data	O8	659	U8
ATB	**ATDATAEXT[7]**	ATB data	O8	660	T1
ATB	**ATDATAEXT[8]**	ATB data	O8	661	T2
ATB	**ATDATAEXT[9]**	ATB data	O8	662	T3
ATB	**ATDATAEXT[10]**	ATB data	O8	663	T4
ATB	**ATDATAEXT[11]**	ATB data	O8	664	T5
ATB	**ATDATAEXT[12]**	ATB data	O8	665	T6
ATB	**ATDATAEXT[13]**	ATB data	O8	666	T7
ATB	**ATDATAEXT[14]**	ATB data	O8	667	T8
ATB	**ATDATAEXT[15]**	ATB data	O8	668	R1
ATB	**ATDATAEXT[16]**	ATB data	O8	669	R2
ATB	**ATDATAEXT[17]**	ATB data	O8	670	R3

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
ATB	**ATDATAEXT[18]**	ATB data	O8	671	R4
ATB	**ATDATAEXT[19]**	ATB data	O8	672	R5
ATB	**ATDATAEXT[20]**	ATB data	O8	673	R6
ATB	**ATDATAEXT[21]**	ATB data	O8	674	R7
ATB	**ATDATAEXT[22]**	ATB data	O8	675	R8
ATB	**ATDATAEXT[23]**	ATB data	O8	676	P1
ATB	**ATDATAEXT[24]**	ATB data	O8	677	P2
ATB	**ATDATAEXT[25]**	ATB data	O8	678	P3
ATB	**ATDATAEXT[26]**	ATB data	O8	679	P4
ATB	**ATDATAEXT[27]**	ATB data	O8	680	P5
ATB	**ATDATAEXT[28]**	ATB data	O8	681	P6
ATB	**ATDATAEXT[29]**	ATB data	O8	682	P7
ATB	**ATDATAEXT[30]**	ATB data	O8	683	P8
ATB	**ATDATAEXT[31]**	ATB data	O8	684	N1
TPIU	**TRACECLK**	Trace port clock	O12	685	N2
TPIU	**TRACECTL**	Trace port control	O12	686	N3
TPIU	**TRACEDATA[0]**	Trace port data	O12	687	N4
TPIU	**TRACEDATA[1]**	Trace port data	O12	688	N5
TPIU	**TRACEDATA[2]**	Trace port data	O12	689	N6
TPIU	**TRACEDATA[3]**	Trace port data	O12	690	N7
TPIU	**TRACEDATA[4]**	Trace port data	O12	691	N8
TPIU	**TRACEDATA[5]**	Trace port data	O12	692	M1
TPIU	**TRACEDATA[6]**	Trace port data	O12	693	M2
TPIU	**TRACEDATA[7]**	Trace port data	O12	694	M3

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
TPIU	**TRACEDATA[8]**	Trace port data	O12	695	M4
TPIU	**TRACEDATA[9]**	Trace port data	O12	696	M5
TPIU	**TRACEDATA[10]**	Trace port data	O12	697	M6
TPIU	**TRACEDATA[11]**	Trace port data	O12	698	M7
TPIU	**TRACEDATA[12]**	Trace port data	O12	699	M8
TPIU	**TRACEDATA[13]**	Trace port data	O12	700	L1
TPIU	**TRACEDATA[14]**	Trace port data	O12	701	L2
TPIU	**TRACEDATA[15]**	Trace port data	O12	702	L3
TPIU	**TRACEDATA[16]**	Trace port data	O12	703	L4
TPIU	**TRACEDATA[17]**	Trace port data	O12	704	L5
TPIU	**TRACEDATA[18]**	Trace port data	O12	705	L6
TPIU	**TRACEDATA[19]**	Trace port data	O12	706	L7
TPIU	**TRACEDATA[20]**	Trace port data	O12	707	L8
TPIU	**TRACEDATA[21]**	Trace port data	O12	708	K1
TPIU	**TRACEDATA[22]**	Trace port data	O12	709	K2
TPIU	**TRACEDATA[23]**	Trace port data	O12	710	K3
TPIU	**TRACEDATA[24]**	Trace port data	O12	711	K4
TPIU	**TRACEDATA[25]**	Trace port data	O12	712	K5
TPIU	**TRACEDATA[26]**	Trace port data	O12	713	K6
TPIU	**TRACEDATA[27]**	Trace port data	O12	714	K7
TPIU	**TRACEDATA[28]**	Trace port data	O12	715	K8
TPIU	**TRACEDATA[29]**	Trace port data	O12	716	J1
TPIU	**TRACEDATA[30]**	Trace port data	O12	717	J2
TPIU	**TRACEDATA[31]**	Trace port data	O12	718	J3

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
DEBUG	**SPIDEN**	Debug control	I	719	J4
DEBUG	**SPNIDEN**	Debug control	I	720	J5
DEBUG	**DBGEN**	Debug control	I	721	J6
DEBUG	**NIDEN**	Debug control	I	722	J7
	VSS	Common ground	IOG	723	J8
IRQ	**nFIQIN**	External **nFIQ** source	I	724	H1
IRQ	**nIRQIN**	External **nIRQ** source	I	725	H2
IRQ	**INTSOURCE[0]**	Exported interrupt source	O8	726	H3
IRQ	**INTSOURCE[1]**	Exported interrupt source	O8	727	H4
IRQ	**INTSOURCE[2]**	Exported interrupt source	O8	728	H5
IRQ	**INTSOURCE[3]**	Exported interrupt source	O8	729	H6
IRQ	**INTSOURCE[4]**	Exported interrupt source	O8	730	H7
IRQ	**INTSOURCE[5]**	Exported interrupt source	O8	731	G1
IRQ	**INTSOURCE[6]**	Exported interrupt source	O8	732	G2
IRQ	**INTSOURCE[7]**	Exported interrupt source	O8	733	G3
IRQ	**INTSOURCE[8]**	Exported interrupt source	O8	734	G4
IRQ	**INTSOURCE[9]**	Exported interrupt source	O8	735	G5

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
IRQ	**INTSOURCE[10]**	Exported interrupt source	O8	736	G6
IRQ	**INTSOURCE[11]**	Exported interrupt source	O8	737	F1
IRQ	**INTSOURCE[12]**	Exported interrupt source	O8	738	F2
IRQ	**INTSOURCE[13]**	Exported interrupt source	O8	739	F3
IRQ	**INTSOURCE[14]**	Exported interrupt source	O8	740	F4
IRQ	**INTSOURCE[15]**	Exported interrupt source	O8	741	F5
IRQ	**INTSOURCE[16]**	Exported interrupt source	O8	742	E1
-	**VSOC**	Digital power, 1.2V	DP	743	E2
-	**VSS**	Common ground	DG	744	E3
-	VRAM	IEM V_{ram} voltage, 1.2V nominal	DP	745	E4
-	**VSS**	Common ground	IOG	746	D1
-	**VSOC**	Digital power, 1.2V	DP	747	D2
-	**VSS**	Common ground	DG	748	D3
IRQ	**INTSOURCE[17]**	Exported interrupt source	O8	749	C1
IRQ	**INTSOURCE[18]**	Exported interrupt source	O8	750	C2
IRQ	**INTSOURCE[19]**	Exported interrupt source	O8	751	B1
IRQ	**INTSOURCE[20]**	Exported interrupt source	O8	752	A2

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
IRQ	**INTSOURCE[21]**	Exported interrupt source	O8	753	A3
IRQ	**nFIQOUT**	Exported nFIQ source	O8	754	B3
IRQ	**nIRQOUT**	Exported nIRQ source	O8	755	A4
IRQ	**EXTINTSOURCE[0]**	External interrupt source	I	756	B4
IRQ	**EXTINTSOURCE[1]**	External interrupt source	I	757	C4
IRQ	**EXTINTSOURCE[2]**	External interrupt source	I	758	A5
IRQ	**EXTINTSOURCE[3]**	External interrupt source	I	759	B5
IRQ	**EXTINTSOURCE[4]**	External interrupt source	I	760	C5
IRQ	**EXTINTSOURCE[5]**	External interrupt source	I	761	D5
IRQ	**EXTINTSOURCE[6]**	External interrupt source	I	762	A6
IRQ	**EXTINTSOURCE[7]**	External interrupt source	I	763	B6
IRQ	**EXTINTSOURCE[8]**	External interrupt source	I	764	C6
AXI slave port	**AWADDR_ExtSlv[0]**	-	I	765	D6
AXI slave port	**AWADDR_ExtSlv[1]**	-	I	766	E6
AXI slave port	**AWADDR_ExtSlv[2]**	-	I	767	A7
AXI slave port	**AWADDR_ExtSlv[3]**	-	I	768	B7
AXI slave port	**AWADDR_ExtSlv[4]**	-	I	769	C7

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI slave port	**AWADDR_ExtSlv[5]**	-	I	770	D7
AXI slave port	**AWADDR_ExtSlv[6]**	-	I	771	E7
AXI slave port	**AWADDR_ExtSlv[7]**	-	I	772	F7
AXI slave port	**AWADDR_ExtSlv[8]**	-	I	773	A8
AXI slave port	**AWADDR_ExtSlv[9]**	-	I	774	B8
AXI slave port	**AWADDR_ExtSlv[10]**	-	I	775	C8
AXI slave port	**AWADDR_ExtSlv[11]**	-	I	776	D8
AXI slave port	**AWADDR_ExtSlv[12]**	-	I	777	E8
AXI slave port	**AWADDR_ExtSlv[13]**	-	I	778	F8
AXI slave port	**AWADDR_ExtSlv[14]**	-	I	779	G8
AXI slave port	**AWADDR_ExtSlv[15]**	-	I	780	A9
AXI slave port	**AWADDR_ExtSlv[16]**	-	I	781	B9
AXI slave port	**AWADDR_ExtSlv[17]**	-	I	782	C9
AXI slave port	**AWADDR_ExtSlv[18]**	-	I	783	D9
AXI slave port	**AWADDR_ExtSlv[19]**	-	I	784	E9
AXI slave port	**AWADDR_ExtSlv[20]**	-	I	785	F9
AXI slave port	**AWADDR_ExtSlv[21]**	-	I	786	G9
AXI slave port	**AWADDR_ExtSlv[22]**	-	I	787	H9
AXI slave port	**AWADDR_ExtSlv[23]**	-	I	788	J9
AXI slave port	**AWREADY_ExtSlv**	-	O8	789	K9
AXI slave port	**AWVALID_ExtSlv**	-	I	790	A10
AXI slave port	**AWADDR_ExtSlv[24]**	-	I	791	B10
AXI slave port	**AWADDR_ExtSlv[25]**	-	I	792	C10
AXI slave port	**AWADDR_ExtSlv[26]**	-	I	793	D10

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI slave port	**AWADDR_ExtSlv[27]**	-	I	794	E10
AXI slave port	**AWADDR_ExtSlv[28]**	-	I	795	F10
AXI slave port	**AWADDR_ExtSlv[29]**	-	I	796	G10
AXI slave port	**AWADDR_ExtSlv[30]**	-	I	797	H10
AXI slave port	**AWADDR_ExtSlv[31]**	-	I	798	J10
AXI slave port	**AWLEN_ExtSlv[0]**	Write address burst length	I	799	K10
AXI slave port	**AWLEN_ExtSlv[1]**	Read address burst length	I	800	A11
AXI slave port	**AWLEN_ExtSlv[2]**	Write address burst length	I	801	B11
AXI slave port	**AWLEN_ExtSlv[3]**	Read address burst length	I	802	C11
AXI slave port	**AWSIZE_ExtSlv[0]**	Write address transfer size	I	803	D11
AXI slave port	**AWSIZE_ExtSlv[1]**	Read address transfer size	I	804	E11
AXI slave port	**AWBURST_ExtSlv[0]**	Write address burst type	I	805	F11
AXI slave port	**AWBURST_ExtSlv[1]**	Read address burst type	I	806	G11
AXI slave port	**AWLOCK_ExtSlv[0]**	-	I	807	H11
AXI slave port	**AWLOCK_ExtSlv[1]**	-	I	808	J11
AXI slave port	**AWCACHE_ExtSlv[0]**	-	I	809	K11
AXI slave port	**AWCACHE_ExtSlv[1]**	-	I	810	A12
AXI slave port	**AWCACHE_ExtSlv[2]**	-	I	811	B12
AXI slave port	**AWCACHE_ExtSlv[3]**	-	I	812	C12

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI slave port	**AWPROT_ExtSlv[0]**	-	I	813	D12
AXI slave port	**AWPROT_ExtSlv[1]**	-	I	814	E12
AXI slave port	**AWPROT_ExtSlv[2]**	-	I	815	F12
AXI slave port	**AWID_ExtSlv[0]**	-	I	816	G12
AXI slave port	**AWID_ExtSlv[1]**	-	I	817	H12
AXI slave port	**AWID_ExtSlv[2]**	-	I	818	J12
AXI slave port	**AWID_ExtSlv[3]**	-	I	819	K12
AXI slave port	**AWID_ExtSlv[4]**	-	I	820	A13
AXI slave port	**ARADDR_ExtSlv[0]**	-	I	821	B13
AXI slave port	**ARADDR_ExtSlv[1]**	-	I	822	C13
AXI slave port	**ARADDR_ExtSlv[2]**	-	I	823	D13
AXI slave port	**ARADDR_ExtSlv[3]**	-	I	824	E13
AXI slave port	**ARADDR_ExtSlv[4]**	-	I	825	F13
AXI slave port	**ARADDR_ExtSlv[5]**	-	I	826	G13
AXI slave port	**ARADDR_ExtSlv[6]**	-	I	827	H13
AXI slave port	**ARADDR_ExtSlv[7]**	-	I	828	J13
AXI slave port	**ARADDR_ExtSlv[8]**	-	I	829	K13
AXI slave port	**ARADDR_ExtSlv[9]**	-	I	830	A14
AXI slave port	**ARADDR_ExtSlv[10]**	-	I	831	B14
AXI slave port	**ARADDR_ExtSlv[11]**	-	I	832	C14
AXI slave port	**ARADDR_ExtSlv[12]**	-	I	833	D14
AXI slave port	**ARADDR_ExtSlv[13]**	-	I	834	E14
AXI slave port	**ARADDR_ExtSlv[14]**	-	I	835	F14
AXI slave port	**ARADDR_ExtSlv[15]**	-	I	836	G14

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI slave port	**ARADDR_ExtSlv[16]**	-	I	837	H14
AXI slave port	**ARADDR_ExtSlv[17]**	-	I	838	J14
AXI slave port	**ARADDR_ExtSlv[18]**	-	I	839	K14
AXI slave port	**ARADDR_ExtSlv[19]**	-	I	840	A15
AXI slave port	**ARADDR_ExtSlv[20]**	-	I	841	B15
AXI slave port	**ARADDR_ExtSlv[21]**	-	I	842	C15
AXI slave port	**ARADDR_ExtSlv[22]**	-	I	843	D15
AXI slave port	**ARADDR_ExtSlv[23]**	-	I	844	E15
AXI slave port	**ARREADY_ExtSlv**	-	O8	845	F15
AXI slave port	**ARVALID_ExtSlv**	-	I	846	G15
AXI slave port	**ARADDR_ExtSlv[24]**	-	I	847	H15
AXI slave port	**ARADDR_ExtSlv[25]**	-	I	848	J15
AXI slave port	**ARADDR_ExtSlv[26]**	-	I	849	K15
AXI slave port	**ARADDR_ExtSlv[27]**	-	I	850	A16
AXI slave port	**ARADDR_ExtSlv[28]**	-	I	851	B16
AXI slave port	**ARADDR_ExtSlv[29]**	-	I	852	C16
AXI slave port	**ARADDR_ExtSlv[30]**	-	I	853	D16
AXI slave port	**ARADDR_ExtSlv[31]**	-	I	854	E16
AXI slave port	**ARLEN_ExtSlv[0]**	Write address burst length	I	855	F16
AXI slave port	**ARLEN_ExtSlv[1]**	Write address burst length	I	856	G16
AXI slave port	**ARLEN_ExtSlv[2]**	Read address burst length	I	857	H16

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI slave port	**ARLEN_ExtSlv[3]**	Write address burst length	I	858	J16
AXI slave port	**ARSIZE_ExtSlv[0]**	Write address transfer size	I	859	K16
AXI slave port	**ARSIZE_ExtSlv[1]**	Read address transfer size	I	860	A17
AXI slave port	**ARBURST_ExtSlv[0]**	Write address burst type	I	861	B17
AXI slave port	**ARBURST_ExtSlv[1]**	Read address burst type	I	862	C17
AXI slave port	**ARLOCK_ExtSlv[0]**	-	I	863	D17
AXI slave port	**ARLOCK_ExtSlv[1]**	-	I	864	E17
AXI slave port	**ARCACHE_ExtSlv[0]**	-	I	865	F17
AXI slave port	**ARCACHE_ExtSlv[1]**	-	I	866	G17
AXI slave port	**ARCACHE_ExtSlv[2]**	-	I	867	H17
AXI slave port	**ARCACHE_ExtSlv[3]**	-	I	868	J17
AXI slave port	**ARPROT_ExtSlv[0]**	-	I	869	K17
AXI slave port	**ARPROT_ExtSlv[1]**	-	I	870	A18
AXI slave port	**ARPROT_ExtSlv[2]**	-	I	871	B18
AXI slave port	**ARID_ExtSlv[0]**	-	I	872	C18
AXI slave port	**ARID_ExtSlv[1]**	-	I	873	D18
AXI slave port	**ARID_ExtSlv[2]**	-	I	874	E18
AXI slave port	**ARID_ExtSlv[3]**	-	I	875	F18
AXI slave port	**ARID_ExtSlv[4]**	-	I	876	G18
AXI slave port	**WLAST_ExtSlv**	-	I	877	H18
AXI slave port	**WDATA_ExtSlv[0]**	-	I	878	J18

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI slave port	**WDATA_ExtSlv[1]**	-	I	879	K18
AXI slave port	**WDATA_ExtSlv[2]**	-	I	880	L18
AXI slave port	**WDATA_ExtSlv[3]**	-	I	881	A19
AXI slave port	**WDATA_ExtSlv[4]**	-	I	882	B19
AXI slave port	**WDATA_ExtSlv[5]**	-	I	883	C19
AXI slave port	**WDATA_ExtSlv[6]**	-	I	884	D19
AXI slave port	**WDATA_ExtSlv[7]**	-	I	885	E19
AXI slave port	**WDATA_ExtSlv[8]**	-	I	886	F19
AXI slave port	**WDATA_ExtSlv[9]**	-	I	887	G19
AXI slave port	**WDATA_ExtSlv[10]**	-	I	888	H19
AXI slave port	**WDATA_ExtSlv[11]**	-	I	889	J19
AXI slave port	**WDATA_ExtSlv[12]**	-	I	890	K19
-	**VD33**	I/O power, 3.3V	IOP	891	L19
-	**VSS**	Common ground	DG	892	A20
-	**VSOC**	Digital power, 1.2V	DP	893	B20
-	**VD33**	I/O power, 3.3V	IOP	894	C20
AXI slave port	**WDATA_ExtSlv[13]**	-	I	895	D20
AXI slave port	**WDATA_ExtSlv[14]**	-	I	896	E20
AXI slave port	**WDATA_ExtSlv[15]**	-	I	897	F20
AXI slave port	**WDATA_ExtSlv[16]**	-	I	898	G20
AXI slave port	**WDATA_ExtSlv[17]**	-	I	899	H20
AXI slave port	**WDATA_ExtSlv[18]**	-	I	900	J20
AXI slave port	**WDATA_ExtSlv[19]**	-	I	901	K20
AXI slave port	**WDATA_ExtSlv[20]**	-	I	902	L20

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI slave port	**WDATA_ExtSlv[21]**	-	I	903	A21
AXI slave port	**WDATA_ExtSlv[22]**	-	I	904	B21
AXI slave port	**WDATA_ExtSlv[23]**	-	I	905	C21
AXI slave port	**WDATA_ExtSlv[24]**	-	I	906	D21
AXI slave port	**WDATA_ExtSlv[25]**	-	I	907	E21
AXI slave port	**WDATA_ExtSlv[26]**	-	I	908	F21
AXI slave port	**WDATA_ExtSlv[27]**	-	I	909	G21
AXI slave port	**WDATA_ExtSlv[28]**	-	I	910	H21
AXI slave port	**WDATA_ExtSlv[29]**	-	I	911	J21
AXI slave port	**WDATA_ExtSlv[30]**	-	I	912	K21
AXI slave port	**WDATA_ExtSlv[31]**	-	I	913	L21
AXI slave port	**WVALID_ExtSlv**	-	I	914	A22
AXI slave port	**WREADY_ExtSlv**	-	O8	915	B22
AXI slave port	**WDATA_ExtSlv[32]**	-	I	916	C22
AXI slave port	**WDATA_ExtSlv[33]**	-	I	917	D22
AXI slave port	**WDATA_ExtSlv[34]**	-	I	918	E22
AXI slave port	**WDATA_ExtSlv[35]**	-	I	919	F22
AXI slave port	**WDATA_ExtSlv[36]**	-	I	920	G22
AXI slave port	**WDATA_ExtSlv[37]**	-	I	921	H22
AXI slave port	**WDATA_ExtSlv[38]**	-	I	922	J22
AXI slave port	**WDATA_ExtSlv[39]**	-	I	923	K22
AXI slave port	**WDATA_ExtSlv[40]**	-	I	924	L22
AXI slave port	**WDATA_ExtSlv[41]**	-	I	925	A23
AXI slave port	**WDATA_ExtSlv[42]**	-	I	926	B23

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI slave port	**WDATA_ExtSlv[43]**	-	I	927	C23
AXI slave port	**WDATA_ExtSlv[44]**	-	I	928	D23
AXI slave port	**WDATA_ExtSlv[45]**	-	I	929	E23
AXI slave port	**WDATA_ExtSlv[46]**	-	I	930	F23
AXI slave port	**WDATA_ExtSlv[47]**	-	I	931	G23
AXI slave port	**WDATA_ExtSlv[48]**	-	I	932	H23
AXI slave port	**WDATA_ExtSlv[49]**	-	I	933	J23
AXI slave port	**WDATA_ExtSlv[50]**	-	I	934	K23
AXI slave port	**WDATA_ExtSlv[51]**	-	I	935	A24
AXI slave port	**WDATA_ExtSlv[52]**	-	I	936	B24
AXI slave port	**WDATA_ExtSlv[53]**	-	I	937	C24
AXI slave port	**WDATA_ExtSlv[54]**	-	I	938	D24
AXI slave port	**WDATA_ExtSlv[55]**	-	I	939	E24
AXI slave port	**WDATA_ExtSlv[56]**	-	I	940	F24
AXI slave port	**WDATA_ExtSlv[57]**	-	I	941	G24
AXI slave port	**WDATA_ExtSlv[58]**	-	I	942	H24
AXI slave port	**WDATA_ExtSlv[59]**	-	I	943	J24
AXI slave port	**WDATA_ExtSlv[60]**	-	I	944	K24
AXI slave port	**WDATA_ExtSlv[61]**	-	I	945	A25
AXI slave port	**WDATA_ExtSlv[62]**	-	I	946	B25
AXI slave port	**WDATA_ExtSlv[63]**	-	I	947	C25
AXI slave port	**WSTRB_ExtSlv[0]**	-	I	948	D25
AXI slave port	**WSTRB_ExtSlv[1]**	-	I	949	E25
AXI slave port	**WSTRB_ExtSlv[2]**	-	I	950	F25

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI slave port	**WSTRB_ExtSlv[3]**	-	I	951	G25
AXI slave port	**WSTRB_ExtSlv[4]**	-	I	952	H25
AXI slave port	**WSTRB_ExtSlv[5]**	-	I	953	J25
AXI slave port	**WSTRB_ExtSlv[6]**	-	I	954	K25
AXI slave port	**WSTRB_ExtSlv[7]**	-	I	955	A26
AXI slave port	**WID_ExtSlv[0]**	-	I	956	B26
AXI slave port	**WID_ExtSlv[1]**	-	I	957	C26
AXI slave port	**WID_ExtSlv[2]**	-	I	958	D26
AXI slave port	**WID_ExtSlv[3]**	-	I	959	E26
AXI slave port	**WID_ExtSlv[4]**	-	I	960	F26
AXI slave port	**RLAST_ExtSlv**	-	O8	961	G26
AXI slave port	**RDATA_ExtSlv[0]**	-	O8	962	H26
AXI slave port	**RDATA_ExtSlv[1]**	-	O8	963	J26
AXI slave port	**RDATA_ExtSlv[2]**	-	O8	964	K26
AXI slave port	**RDATA_ExtSlv[3]**	-	O8	965	A27
AXI slave port	**RDATA_ExtSlv[4]**	-	O8	966	B27
AXI slave port	**RDATA_ExtSlv[5]**	-	O8	967	C27
AXI slave port	**RDATA_ExtSlv[6]**	-	O8	968	D27
AXI slave port	**RDATA_ExtSlv[7]**	-	O8	969	E27
AXI slave port	**RDATA_ExtSlv[8]**	-	O8	970	F27
-	**VD33**	I/O power, 3.3V	IOP	971	G27
-	**VSS**	Common ground	DG	972	H27
-	**VSOC**	Digital power, 1.2V	DP	973	J27
AXI slave port	**RDATA_ExtSlv[9]**	-	O8	974	K27

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI slave port	**RDATA_ExtSlv[10]**	-	O8	975	A28
AXI slave port	**RDATA_ExtSlv[11]**	-	O8	976	B28
AXI slave port	**RDATA_ExtSlv[12]**	-	O8	977	C28
AXI slave port	**RDATA_ExtSlv[13]**	-	O8	978	D28
AXI slave port	**RDATA_ExtSlv[14]**	-	O8	979	E28
AXI slave port	**RDATA_ExtSlv[15]**	-	O8	980	F28
AXI slave port	**RDATA_ExtSlv[16]**	-	O8	981	G28
AXI slave port	**RDATA_ExtSlv[17]**	-	O8	982	H28
AXI slave port	**RDATA_ExtSlv[18]**	-	O8	983	J28
AXI slave port	**RDATA_ExtSlv[19]**	-	O8	984	K28
AXI slave port	**RDATA_ExtSlv[20]**	-	O8	985	A29
AXI slave port	**RDATA_ExtSlv[21]**	-	O8	986	B29
AXI slave port	**RDATA_ExtSlv[22]**	-	O8	987	C29
AXI slave port	**RDATA_ExtSlv[23]**	-	O8	988	D29
AXI slave port	**RDATA_ExtSlv[24]**	-	O8	989	E29
AXI slave port	**RDATA_ExtSlv[25]**	-	O8	990	F29
AXI slave port	**RDATA_ExtSlv[26]**	-	O8	991	G29
AXI slave port	**RDATA_ExtSlv[27]**	-	O8	992	H29
AXI slave port	**RDATA_ExtSlv[28]**	-	O8	993	J29
AXI slave port	**RDATA_ExtSlv[29]**	-	O8	994	K29
AXI slave port	**RVALID_ExtSlv**	-	O8	995	A30
AXI slave port	**RREADY_ExtSlv**	-	I	996	B30
AXI slave port	**RDATA_ExtSlv[30]**	-	O8	997	C30
AXI slave port	**RDATA_ExtSlv[31]**	-	O8	998	D30

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI slave port	**RDATA_ExtSlv[32]**	-	O8	999	E30
AXI slave port	**RDATA_ExtSlv[33]**	-	O8	1000	F30
AXI slave port	**RDATA_ExtSlv[34]**	-	O8	1001	G30
AXI slave port	**RDATA_ExtSlv[35]**	-	O8	1002	H30
AXI slave port	**RDATA_ExtSlv[36]**	-	O8	1003	J30
AXI slave port	**RDATA_ExtSlv[37]**	-	O8	1004	K30
AXI slave port	**RDATA_ExtSlv[38]**	-	O8	1005	A31
AXI slave port	**RDATA_ExtSlv[39]**	-	O8	1006	B31
AXI slave port	**RDATA_ExtSlv[40]**	-	O8	1007	C31
AXI slave port	**RDATA_ExtSlv[41]**	-	O8	1008	D31
AXI slave port	**RDATA_ExtSlv[42]**	-	O8	1009	E31
AXI slave port	**RDATA_ExtSlv[43]**	-	O8	1010	F31
AXI slave port	**RDATA_ExtSlv[44]**	-	O8	1011	G31
AXI slave port	**RDATA_ExtSlv[45]**	-	O8	1012	H31
AXI slave port	**RDATA_ExtSlv[46]**	-	O8	1013	J31
AXI slave port	**RDATA_ExtSlv[47]**	-	O8	1014	K31
AXI slave port	**RDATA_ExtSlv[48]**	-	O8	1015	A32
AXI slave port	**RDATA_ExtSlv[49]**	-	O8	1016	B32
AXI slave port	**RDATA_ExtSlv[50]**	-	O8	1017	C32
AXI slave port	**RDATA_ExtSlv[51]**	-	O8	1018	D32
AXI slave port	**RDATA_ExtSlv[52]**	-	O8	1019	E32
AXI slave port	**RDATA_ExtSlv[53]**	-	O8	1020	F32
AXI slave port	**RDATA_ExtSlv[54]**	-	O8	1021	G32
AXI slave port	**RDATA_ExtSlv[55]**	-	O8	1022	H32

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
AXI slave port	**RDATA_ExtSlv[56]**	-	O8	1023	A33
AXI slave port	**RDATA_ExtSlv[57]**	-	O8	1024	B33
AXI slave port	**RDATA_ExtSlv[58]**	-	O8	1025	C33
AXI slave port	**RDATA_ExtSlv[59]**	-	O8	1026	D33
AXI slave port	**RDATA_ExtSlv[60]**	-	O8	1027	E33
AXI slave port	**RDATA_ExtSlv[61]**	-	O8	1028	F33
AXI slave port	**RDATA_ExtSlv[62]**	-	O8	1029	G33
AXI slave port	**RDATA_ExtSlv[63]**	-	O8	1030	A34
AXI slave port	**RRESP_ExtSlv[0]**	-	O8	1031	B34
AXI slave port	**RRESP_ExtSlv[1]**	-	O8	1032	C34
AXI slave port	**RID_ExtSlv[0]**	-	O8	1033	D34
AXI slave port	**RID_ExtSlv[1]**	-	O8	1034	E34
AXI slave port	**RID_ExtSlv[2]**	-	O8	1035	F34
AXI slave port	**RID_ExtSlv[3]**	-	O8	1036	A35
AXI slave port	**RID_ExtSlv[4]**	-	O8	1037	B35
AXI slave port	**BRESP_ExtSlv[0]**	-	O8	1038	C35
AXI slave port	**BRESP_ExtSlv[1]**	-	O8	1039	D35
AXI slave port	**BVALID_ExtSlv**	-	O8	1040	E35
AXI slave port	**BREADY_ExtSlv**	-	I	1041	A36
AXI slave port	**BID_ExtSlv[0]**	-	O8	1042	B36
AXI slave port	**BID_ExtSlv[1]**	-	O8	1043	C36
AXI slave port	**BID_ExtSlv[2]**	-	O8	1044	D36
AXI slave port	**BID_ExtSlv[3]**	-	O8	1045	A37
AXI slave port	**BID_ExtSlv[4]**	-	O8	1046	B37

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
-	**VSS**	Common ground	IOG	1047	C37
-	**VD33**	I/O power, 3.3V	IOP	1048	A38
-	**VSS**	Common ground	IOG	1049	B38
-	**VSOC**	Digital power, 1.2V	DP	1050	A39
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	V25
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	Y25
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	AB25
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	U24
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	W24
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	AA24
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	V23
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	Y23
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	AB23
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	U22
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	W22
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	V21

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	Y21
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	AB21
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	U20
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	W20
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	AA20
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	V19
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	Y19
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	AB19
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	V17
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	Y17
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	AB17
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	AA16
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	V15
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	Y15
Substrate plane	**VCORE**	IEM V_{core} voltage, 1.2V nominal	-	-	AB15

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
Substrate plane	**VRAM**	IEM V_{ram} voltage, 1.2V nominal	-	-	U16
Substrate plane	**VRAM**	IEM V_{ram} voltage, 1.2V nominal	-	-	U18
Substrate plane	**VRAM**	IEM V_{ram} voltage, 1.2V nominal	-	-	W16
Substrate plane	**VRAM**	IEM V_{ram} voltage, 1.2V nominal	-	-	W18
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	B39
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	E36
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	H33
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AM32
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AV38
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	W38
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	W34
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AR35
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AV1
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AR4
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AM7
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	W2
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	W6
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	B2
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	E5
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	H8
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	N9

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	N11
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	N14
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	N17
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	N20
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	N23
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	N26
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	N31
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	M30
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	M27
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	M24
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	M21
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	M18
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	M15
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	M12
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	M10
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	L11
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	L14
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	L17
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	L23
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	L26
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	L29
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	R10
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	R12
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	R14

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	R16
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	R18
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	R20
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	R22
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	R24
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	R26
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	R28
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	R30
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	P29
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	P25
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	P22
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	P19
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	P16
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	P13
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	P11
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	T31
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	T27
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	T25
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	T23
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	T21
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	T19
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	T17
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	T15
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	T13

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	T11
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	T9
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	U12
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	U14
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	U26
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	U29
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	V10
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	V13
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	V27
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	V30
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	W9
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	W11
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	W12
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	W14
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	W26
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	W28
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	W31
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	Y29
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	Y27
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	Y13
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AA10
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AA12
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AA14
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AA18

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AA22
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AA26
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AA28
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AA30
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AB31
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AB29
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AB27
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AB13
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AB11
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AB9
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AC12
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AC14
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AC16
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AC18
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AC20
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AC22
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AC24
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AC26
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AC28
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AD30
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AD27
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AD25
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AD23
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AD21

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AD19
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AD17
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AD15
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AD13
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AD10
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AE9
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AE12
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AE14
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AE16
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AE18
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AE20
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AE22
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AE24
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AE26
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AE28
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AE31
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AF29
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AF27
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AF23
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AF20
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AF17
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AF14
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AF11
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AG10

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AG12
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AG15
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AG18
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AG21
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AG24
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AG27
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AG30
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AH9
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AH13
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AH16
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AH19
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AH22
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AH25
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AH28
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AH31
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AJ29
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AJ26
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AJ23
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AJ17
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AJ14
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	AJ11
Substrate plane	**VSOC**	Digital power, 1.2V	-	-	U28
Substrate plane	**VD18**	I/O power, 1.8V	-	-	AH24
Substrate plane	**VD18**	I/O power, 1.8V	-	-	AG23

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
Substrate plane	**VD18**	I/O power, 1.8V	-	-	AH21
Substrate plane	**VD18**	I/O power, 1.8V	-	-	AG20
Substrate plane	**VD18**	I/O power, 1.8V	-	-	AH18
Substrate plane	**VD18**	I/O power, 1.8V	-	-	AG17
Substrate plane	**VD18**	I/O power, 1.8V	-	-	AJ16
Substrate plane	**VD33**	I/O power, 3.3V	-	-	C38
Substrate plane	**VD33**	I/O power, 3.3V	-	-	F35
Substrate plane	**VD33**	I/O power, 3.3V	-	-	J32
Substrate plane	**VD33**	I/O power, 3.3V	-	-	W33
Substrate plane	**VD33**	I/O power, 3.3V	-	-	W37
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AP34
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AU37
Substrate plane	**VD33**	I/O power, 3.3V	-	-	L31
Substrate plane	**VD33**	I/O power, 3.3V	-	-	N28
Substrate plane	**VD33**	I/O power, 3.3V	-	-	L25
Substrate plane	**VD33**	I/O power, 3.3V	-	-	L15
Substrate plane	**VD33**	I/O power, 3.3V	-	-	L12
Substrate plane	**VD33**	I/O power, 3.3V	-	-	U9
Substrate plane	**VD33**	I/O power, 3.3V	-	-	L9
Substrate plane	**VD33**	I/O power, 3.3V	-	-	F6
Substrate plane	**VD33**	I/O power, 3.3V	-	-	C3
Substrate plane	**VD33**	I/O power, 3.3V	-	-	W7
Substrate plane	**VD33**	I/O power, 3.3V	-	-	W3
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AL8

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AP5
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AU2
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AJ9
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AJ25
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AJ28
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AJ31
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AH10
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AH12
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AH15
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AH27
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AH30
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AG11
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AG14
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AG26
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AG29
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AF31
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AF28
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AF25
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AF16
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AF13
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AF9
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AE10
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AE30
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AD29

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AC31
Substrate plane	**VD33**	I/O power, 3.3V	-	-	V29
Substrate plane	**VD33**	I/O power, 3.3V	-	-	Y31
Substrate plane	**VD33**	I/O power, 3.3V	-	-	U31
Substrate plane	**VD33**	I/O power, 3.3V	-	-	N30
Substrate plane	**VD33**	I/O power, 3.3V	-	-	P31
Substrate plane	**VD33**	I/O power, 3.3V	-	-	T30
Substrate plane	**VD33**	I/O power, 3.3V	-	-	W30
Substrate plane	**VD33**	I/O power, 3.3V	-	-	L28
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AB30
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AA29
Substrate plane	**VD33**	I/O power, 3.3V	-	-	R29
Substrate plane	**VD33**	I/O power, 3.3V	-	-	M29
Substrate plane	**VD33**	I/O power, 3.3V	-	-	M26
Substrate plane	**VD33**	I/O power, 3.3V	-	-	M23
Substrate plane	**VD33**	I/O power, 3.3V	-	-	M20
Substrate plane	**VD33**	I/O power, 3.3V	-	-	M17
Substrate plane	**VD33**	I/O power, 3.3V	-	-	M14
Substrate plane	**VD33**	I/O power, 3.3V	-	-	M11
Substrate plane	**VD33**	I/O power, 3.3V	-	-	P27
Substrate plane	**VD33**	I/O power, 3.3V	-	-	P24
Substrate plane	**VD33**	I/O power, 3.3V	-	-	P21
Substrate plane	**VD33**	I/O power, 3.3V	-	-	P18
Substrate plane	**VD33**	I/O power, 3.3V	-	-	P15

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
Substrate plane	**VD33**	I/O power, 3.3V	-	-	P12
Substrate plane	**VD33**	I/O power, 3.3V	-	-	P9
Substrate plane	**VD33**	I/O power, 3.3V	-	-	N22
Substrate plane	**VD33**	I/O power, 3.3V	-	-	N19
Substrate plane	**VD33**	I/O power, 3.3V	-	-	N16
Substrate plane	**VD33**	I/O power, 3.3V	-	-	N13
Substrate plane	**VD33**	I/O power, 3.3V	-	-	N10
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AB10
Substrate plane	**VD33**	I/O power, 3.3V	-	-	V11
Substrate plane	**VD33**	I/O power, 3.3V	-	-	R11
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AJ13
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AD11
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AC9
Substrate plane	**VD33**	I/O power, 3.3V	-	-	Y9
Substrate plane	**VD33**	I/O power, 3.3V	-	-	W10
Substrate plane	**VD33**	I/O power, 3.3V	-	-	T10
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AA11
Substrate plane	**VD33**	I/O power, 3.3V	-	-	N25
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AF22
Substrate plane	**VD33**	I/O power, 3.3V	-	-	AF19
Substrate plane	**VSS**	Common ground	-	-	D37
Substrate plane	**VSS**	Common ground	-	-	W39
Substrate plane	**VSS**	Common ground	-	-	AW39
Substrate plane	**VSS**	Common ground	-	-	G34

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
Substrate plane	**VSS**	Common ground	-	-	AN33
Substrate plane	**VSS**	Common ground	-	-	AT36
Substrate plane	**VSS**	Common ground	-	-	W35
Substrate plane	**VSS**	Common ground	-	-	A1
Substrate plane	**VSS**	Common ground	-	-	W1
Substrate plane	**VSS**	Common ground	-	-	AT3
Substrate plane	**VSS**	Common ground	-	-	AN6
Substrate plane	**VSS**	Common ground	-	-	D4
Substrate plane	**VSS**	Common ground	-	-	G7
Substrate plane	**VSS**	Common ground	-	-	W5
Substrate plane	**VSS**	Common ground	-	-	AJ10
Substrate plane	**VSS**	Common ground	-	-	AJ12
Substrate plane	**VSS**	Common ground	-	-	AJ15
Substrate plane	**VSS**	Common ground	-	-	AJ24
Substrate plane	**VSS**	Common ground	-	-	AJ27
Substrate plane	**VSS**	Common ground	-	-	AJ30
Substrate plane	**VSS**	Common ground	-	-	AD31
Substrate plane	**VSS**	Common ground	-	-	AD28
Substrate plane	**VSS**	Common ground	-	-	AD26
Substrate plane	**VSS**	Common ground	-	-	AD24
Substrate plane	**VSS**	Common ground	-	-	AD22
Substrate plane	**VSS**	Common ground	-	-	AD20
Substrate plane	**VSS**	Common ground	-	-	AD18
Substrate plane	**VSS**	Common ground	-	-	AD16

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
Substrate plane	**VSS**	Common ground	-	-	AD14
Substrate plane	**VSS**	Common ground	-	-	AD12
Substrate plane	**VSS**	Common ground	-	-	AD9
Substrate plane	**VSS**	Common ground	-	-	AF30
Substrate plane	**VSS**	Common ground	-	-	AF26
Substrate plane	**VSS**	Common ground	-	-	AF24
Substrate plane	**VSS**	Common ground	-	-	AF21
Substrate plane	**VSS**	Common ground	-	-	AF18
Substrate plane	**VSS**	Common ground	-	-	AF15
Substrate plane	**VSS**	Common ground	-	-	AF12
Substrate plane	**VSS**	Common ground	-	-	AF10
Substrate plane	**VSS**	Common ground	-	-	AG31
Substrate plane	**VSS**	Common ground	-	-	AG28
Substrate plane	**VSS**	Common ground	-	-	AG25
Substrate plane	**VSS**	Common ground	-	-	AG22
Substrate plane	**VSS**	Common ground	-	-	AG19
Substrate plane	**VSS**	Common ground	-	-	AG16
Substrate plane	**VSS**	Common ground	-	-	AG13
Substrate plane	**VSS**	Common ground	-	-	AG9
Substrate plane	**VSS**	Common ground	-	-	AH29
Substrate plane	**VSS**	Common ground	-	-	AH26
Substrate plane	**VSS**	Common ground	-	-	AH23
Substrate plane	**VSS**	Common ground	-	-	AH20
Substrate plane	**VSS**	Common ground	-	-	AH17

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
Substrate plane	**VSS**	Common ground	-	-	AH14
Substrate plane	**VSS**	Common ground	-	-	AH11
Substrate plane	**VSS**	Common ground	-	-	AE29
Substrate plane	**VSS**	Common ground	-	-	AE27
Substrate plane	**VSS**	Common ground	-	-	AE25
Substrate plane	**VSS**	Common ground	-	-	AE23
Substrate plane	**VSS**	Common ground	-	-	AE21
Substrate plane	**VSS**	Common ground	-	-	AE19
Substrate plane	**VSS**	Common ground	-	-	AE17
Substrate plane	**VSS**	Common ground	-	-	AE15
Substrate plane	**VSS**	Common ground	-	-	AE13
Substrate plane	**VSS**	Common ground	-	-	AE11
Substrate plane	**VSS**	Common ground	-	-	AC30
Substrate plane	**VSS**	Common ground	-	-	AC29
Substrate plane	**VSS**	Common ground	-	-	AC27
Substrate plane	**VSS**	Common ground	-	-	AC25
Substrate plane	**VSS**	Common ground	-	-	AC23
Substrate plane	**VSS**	Common ground	-	-	AC21
Substrate plane	**VSS**	Common ground	-	-	AC19
Substrate plane	**VSS**	Common ground	-	-	AC17
Substrate plane	**VSS**	Common ground	-	-	AC15
Substrate plane	**VSS**	Common ground	-	-	AC13
Substrate plane	**VSS**	Common ground	-	-	AC11
Substrate plane	**VSS**	Common ground	-	-	AC10

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
Substrate plane	**VSS**	Common ground	-	-	AB28
Substrate plane	**VSS**	Common ground	-	-	AB26
Substrate plane	**VSS**	Common ground	-	-	AB24
Substrate plane	**VSS**	Common ground	-	-	AB22
Substrate plane	**VSS**	Common ground	-	-	AB20
Substrate plane	**VSS**	Common ground	-	-	AB18
Substrate plane	**VSS**	Common ground	-	-	AB16
Substrate plane	**VSS**	Common ground	-	-	AB14
Substrate plane	**VSS**	Common ground	-	-	AB12
Substrate plane	**VSS**	Common ground	-	-	AA31
Substrate plane	**VSS**	Common ground	-	-	AA27
Substrate plane	**VSS**	Common ground	-	-	AA25
Substrate plane	**VSS**	Common ground	-	-	AA23
Substrate plane	**VSS**	Common ground	-	-	AA21
Substrate plane	**VSS**	Common ground	-	-	AA19
Substrate plane	**VSS**	Common ground	-	-	AA17
Substrate plane	**VSS**	Common ground	-	-	AA15
Substrate plane	**VSS**	Common ground	-	-	AA13
Substrate plane	**VSS**	Common ground	-	-	AA9
Substrate plane	**VSS**	Common ground	-	-	Y30
Substrate plane	**VSS**	Common ground	-	-	Y28
Substrate plane	**VSS**	Common ground	-	-	Y26
Substrate plane	**VSS**	Common ground	-	-	Y24
Substrate plane	**VSS**	Common ground	-	-	Y22

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
Substrate plane	**VSS**	Common ground	-	-	Y20
Substrate plane	**VSS**	Common ground	-	-	Y18
Substrate plane	**VSS**	Common ground	-	-	Y16
Substrate plane	**VSS**	Common ground	-	-	Y14
Substrate plane	**VSS**	Common ground	-	-	Y12
Substrate plane	**VSS**	Common ground	-	-	Y11
Substrate plane	**VSS**	Common ground	-	-	Y10
Substrate plane	**VSS**	Common ground	-	-	W29
Substrate plane	**VSS**	Common ground	-	-	W27
Substrate plane	**VSS**	Common ground	-	-	W25
Substrate plane	**VSS**	Common ground	-	-	W23
Substrate plane	**VSS**	Common ground	-	-	W21
Substrate plane	**VSS**	Common ground	-	-	W19
Substrate plane	**VSS**	Common ground	-	-	W17
Substrate plane	**VSS**	Common ground	-	-	W15
Substrate plane	**VSS**	Common ground	-	-	W13
Substrate plane	**VSS**	Common ground	-	-	V31
Substrate plane	**VSS**	Common ground	-	-	V28
Substrate plane	**VSS**	Common ground	-	-	V26
Substrate plane	**VSS**	Common ground	-	-	V24
Substrate plane	**VSS**	Common ground	-	-	V22
Substrate plane	**VSS**	Common ground	-	-	V20
Substrate plane	**VSS**	Common ground	-	-	V18
Substrate plane	**VSS**	Common ground	-	-	V16

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
Substrate plane	**VSS**	Common ground	-	-	V14
Substrate plane	**VSS**	Common ground	-	-	V12
Substrate plane	**VSS**	Common ground	-	-	V9
Substrate plane	**VSS**	Common ground	-	-	U30
Substrate plane	**VSS**	Common ground	-	-	U27
Substrate plane	**VSS**	Common ground	-	-	U25
Substrate plane	**VSS**	Common ground	-	-	U23
Substrate plane	**VSS**	Common ground	-	-	U21
Substrate plane	**VSS**	Common ground	-	-	U19
Substrate plane	**VSS**	Common ground	-	-	U17
Substrate plane	**VSS**	Common ground	-	-	U15
Substrate plane	**VSS**	Common ground	-	-	U13
Substrate plane	**VSS**	Common ground	-	-	U11
Substrate plane	**VSS**	Common ground	-	-	U10
Substrate plane	**VSS**	Common ground	-	-	T29
Substrate plane	**VSS**	Common ground	-	-	T28
Substrate plane	**VSS**	Common ground	-	-	T26
Substrate plane	**VSS**	Common ground	-	-	T24
Substrate plane	**VSS**	Common ground	-	-	T22
Substrate plane	**VSS**	Common ground	-	-	T20
Substrate plane	**VSS**	Common ground	-	-	T18
Substrate plane	**VSS**	Common ground	-	-	T16
Substrate plane	**VSS**	Common ground	-	-	T14
Substrate plane	**VSS**	Common ground	-	-	T12

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
Substrate plane	**VSS**	Common ground	-	-	R31
Substrate plane	**VSS**	Common ground	-	-	R27
Substrate plane	**VSS**	Common ground	-	-	R25
Substrate plane	**VSS**	Common ground	-	-	R23
Substrate plane	**VSS**	Common ground	-	-	R21
Substrate plane	**VSS**	Common ground	-	-	R19
Substrate plane	**VSS**	Common ground	-	-	R17
Substrate plane	**VSS**	Common ground	-	-	R15
Substrate plane	**VSS**	Common ground	-	-	R13
Substrate plane	**VSS**	Common ground	-	-	R9
Substrate plane	**VSS**	Common ground	-	-	P30
Substrate plane	**VSS**	Common ground	-	-	P28
Substrate plane	**VSS**	Common ground	-	-	P26
Substrate plane	**VSS**	Common ground	-	-	P23
Substrate plane	**VSS**	Common ground	-	-	P20
Substrate plane	**VSS**	Common ground	-	-	P17
Substrate plane	**VSS**	Common ground	-	-	P14
Substrate plane	**VSS**	Common ground	-	-	P10
Substrate plane	**VSS**	Common ground	-	-	N29
Substrate plane	**VSS**	Common ground	-	-	N27
Substrate plane	**VSS**	Common ground	-	-	N24
Substrate plane	**VSS**	Common ground	-	-	N21
Substrate plane	**VSS**	Common ground	-	-	N18
Substrate plane	**VSS**	Common ground	-	-	N15

Table A-2 Die bonding (continued)

Functional block	Signal name	Description	Pad type/ strength	Die bond pad number	BGA grid reference
Substrate plane	**VSS**	Common ground	-	-	N12
Substrate plane	**VSS**	Common ground	-	-	M31
Substrate plane	**VSS**	Common ground	-	-	M28
Substrate plane	**VSS**	Common ground	-	-	M25
Substrate plane	**VSS**	Common ground	-	-	M22
Substrate plane	**VSS**	Common ground	-	-	M19
Substrate plane	**VSS**	Common ground	-	-	M16
Substrate plane	**VSS**	Common ground	-	-	M13
Substrate plane	**VSS**	Common ground	-	-	M9
Substrate plane	**VSS**	Common ground	-	-	L30
Substrate plane	**VSS**	Common ground	-	-	L27
Substrate plane	**VSS**	Common ground	-	-	L24
Substrate plane	**VSS**	Common ground	-	-	L16
Substrate plane	**VSS**	Common ground	-	-	L13
Substrate plane	**VSS**	Common ground	-	-	L10

A.4 Physical characteristics

This section describes the physical characteristics of the ARM1176JZF Development Chip. It contains the following subsections:

- *Flipchip package*
- *Process*
- *Bumping*
- *IO pad ring design* on page A-76
- *Power and ground sources* on page A-76.

A.4.1 Flipchip package

This section describes the packaging for the ARM1176JZF Development Chip. Figure A-1 shows a cross-section of the flipchip package.

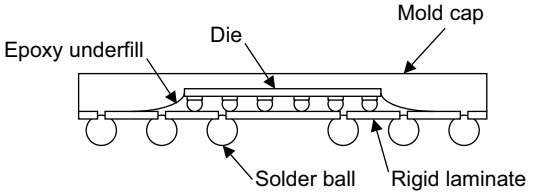


Figure A-1 Flipchip CABGA package cross-section

The 10 x 13mm die has bumps with a 200µm pitch, that bond directly to the substrate using solder.

Note

The bump array is discontinuous, not a regular rectangular array.

The substrate mounts on a PCB using 1521 solder balls on a regular, 39 x 39, 1mm pitch array.

A.4.2 Process

The process for this chip is TSMC130G.

A.4.3 Bumping

Bumping for this chip uses the following dimensions:

- Bump height (F) = 90µ / 100µ
- Bump width = 100µ

- CBD (E), bump opening in die, approximately = 58µ
- Under Bump Metal (UBM) width = 90µ
- Bump pitch (C) = 200µ, minimum.

A.4.4 Package marking

This chip has packages labeled with the following markings:

ARM chip marking logo

The ARM chip making logo is defined in the file `arm_underbar_white.eps`.

Device revision and version number

The ARM part number for the packaged silicon has the format PX111-SI-00001, and includes revision information. This information is defined by ARM when the dies are packaged.

Manufacturer

This is the location of the chip design center.

Date code This is the manufacturing date with the format of week-number and year. This information is defined by ARM when the dies are packaged.

Batch or lot code

This is optional information, defined by ARM when the dies are packaged.

Wafer number

This code number identifies the wafer and the location on the wafer where the die originated.

Figure A-2 on page A-76 shows example package markings.

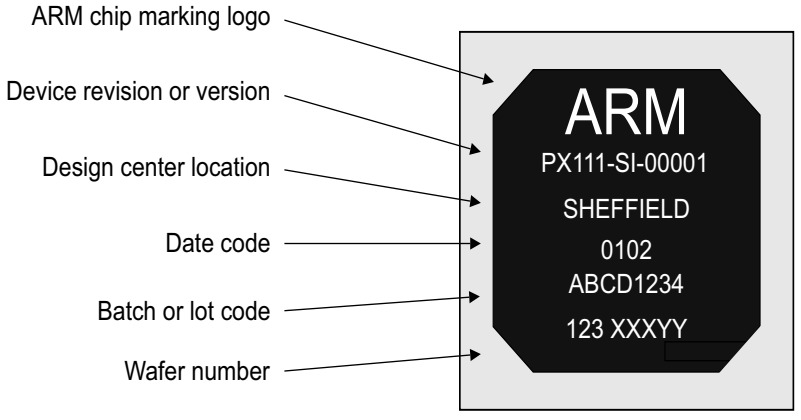


Figure A-2 Example package markings

A.4.5 IO pad ring design

The Operatic PAD ring is broken into two distinct regions. Cutters exist in the I/O ring to break the driver power rail into these two distinct regions:

3.4V region

Region for the majority of the interfaces.

MDDR region

This region uses 1.8v logic with an additional 2.5V reference supply.

This development chip has a flip chip design that supports approximately1000 signal I/Os. Metal 8 is used to route from the I/Os to each signal bump.

A.4.6 Power and ground sources

This section describes the physical characteristics of the ARM1176JZF Development Chip. It contains the following subsections:

- *MDDR region strapping* on page A-78
- *Artisan cell usage recommendations* on page A-79.

1.2V supplies

There are two 1.2V domains:

VSOC domain

1.32V maximum, 1.08V minimum.

VCORE domain

IEM operation, 1.32V maximum, 0.8V minimum.

Input and output 3.3V supply, V_{DDP} and V_{DDO}

These have 3.6V maximum and 3.0V minimum values.

PLL digital power

PLL digital power is strapped to the VSOC domain.

PLL analog power

There are 3 PLLs, each with an idividual, isolated PLL VDD domain rated at 3.3V. They have a 3.3V maximum and 3.0V minimum. There is a maximum of +/- 100mV ripple on both VDD and VSS.

V_{SS}

There is a single VSS domain throughout the device. Core bumps are used for the majority of current. In the I/O region, bumps are added in rows 5 and 6 and strapped into the I/O region MET6 using MET7. These connect to the I/O power rings on GND, VSSO and SGND.

Electrostatic Discharge (ESD) protection is provided by 4 pnls per side:

2 pnl_gcs GND

2 pnl_go VSS and SGND.

MDDR region of the pad ring

The MDDR region of the pad ring contains:

VDD, 1.2V This is VSOC power. This current comes from bumps in rows 5 and 6. Two pad cells, `pnl_hstl_vc`, provide arge protection.

GND This is tied to VSS. This current comes from over strapping from bumps in rows 5 and 6. Two pad cells, `pnl_hstl_gcs`, provide ESD protection.

VDDQ, 1.8V This is provided by over strapping. Two I/O cells, `pnl_hstl_vq`, provide ESD protection.

VDDP, 2.5V This is powered by the I/O cells, `pnl_hstl_vp_m`. Six are required.

VSSO This is tied to main VSS, through over strapping from bumps on rows 5 and 6. Two cells, `pnl_hstl_go`, provide ESD protection.

SGND This is tied to main VSS, through strapping and using the same straps used by GND.

VGG This is a path that exists throughout the I/O pad ring that requires no special connections.

There are two voltage supplies:

1.8V 1.9V maximum, 1.7V minimum

2.5V 2.75V maximum, 2.25V minimum.

Cutters

MDDR cutters break VD25, VD18, VDDQ and VDDP. Other I/O signals connect through as normal. Use cutter cell `pnl_hstl_std2hstl`.

Over strapping Metal 7 density

The overstrapping in MET7, from bumps 5 and 6 out to the edge of the die, is devoted entirely to I/O power straps.

MDDR region strapping

The MDDR region includes:

- 50% over strapping for VSS.
- 40% for VD18.
- 10% VSOC, 1.2V.

Note

2 extra bumps are added for VSOC to power the PL340. This is also powered by the MET7 strapping that passes through the PL340 macro.

Artisan cell usage recommendations

Table A-3 shows the recommendations for usage of input and output cells with in-line designs.

Table A-3 I/O cell usage for inline design

Cell	I/O cell number
pnl_hstl_vq	1 per every 8, 1.8V
pnl_hstl_go	1 per every 8, VSSO
pnl_hstl_vc	1 per every 16, 1.2V
pnl_hstl_gcs	1 per every 12, GND
pnl_hstl_vp_m	1 per every 16, 2.5V

Appendix B
Component Revision Status

This chapter describes the component revision status of the ARM1176JZF Development Chip and contains the following sections:

B.1 Components

Table B-1 lists the product revisions.

Table B-1 Product revisions

IP product	Revision
ARM1176JZF-S Processor Core	r0p0
L2CC	r1p1
CoreSight ETM11, Embedded Trace Macrocell	r0p0
MBIST for L2CC	r1p0
MBIST for ARM1176JZF	r0p0
ARM r2 RAM components	r2p0v0-02bet0
ARM q3 Cell library, with level shifters and clamps	q3cells_r0p3v0
CoreSight ETB11	r0p0
MBIST for ETB11	r0p0
CoreSight TPIU	r0p0
CoreSight Replicator	r0p0
CoreSight Cross-trigger Matrix	r0p0
CoreSight Cross-trigger Interface	r0p0
CoreSight 1:1 Synchronizing bridge	r0p0
CoreSight DAP, APs, DPs, Multiplexer (MUX), bridges and ROM	r0p0
DMC	PL340: r0p0-00rel1
SMC	PL093: r0p4-00dev1
CLCD	PL111: r0p0-00ltd1. Modified version, bug fix applied.
TZIC	SP890: r0p0
GIC	Exp
TZPC	SP870: Modified for use in SoC.

Table B-1 Product revisions (continued)

IP product	Revision
Timers	SP804: Exp
Watchdog	SP805: r2p0-00rel0
RTC	PL031: REL1v0
SSP	PL022: REL1v2
GPIO	PL061: REL1v0
UART	PL011: REL1v3
Intelligent Energy Controller (IEC)	SY750: r0p0-00rel0 Hardware workaround for Errata.
Configurable AXI interconnect	PL300- r0p0-00rel0
ADK AXI Components: `RegSliceAxi`, `SyncDnAxi`, `SyncUpAxi`, `AxiToA11Lite`, `A11AhbLiteMToAxi`, `ExpanderAxi`, `DownsizerAxi`, `IntMemAxi`, `TZMemAdapAxi`, `AhbSToAxi`, `Axi2Apb`.	r0p0-00rel0
ADK 1136 AHB Extension Components: `BLScnv`, `ExAcMnAhb`, `DownsizerV6`	BP010-AC-41002-r0p1-01rel0
MBIST for RAM and ROM	N/A - This is a custom design based on ARM1176 BIST.

B.2 Third party components

Table B-2 lists the third party IP revisions the ARM1176JZF Development Chip uses.

Note

All IP is for TSMC 130nm generic technology node.

Table B-2 Third party IP revisions

IP Product	Vendor	Version
Advanced Power Controller (APC1) and *Hardware Performance Monitor* (HPM)	National Semiconductor	SY751: r0p0-00bet0
Flipchip input and output library, 3.3V	Artisan	fc_io_2004q4v1_r0p0v0/
Flipchip input and output library LVCMOS for Mobile DDR, 1.8V	Artisan	ddr_io_2004q4v3/
800Mbps DDR2 DLL, master and two slave cells	Artisan	dll_800mbps_2004q4v1/
533MHz Deskew PLL, three instances	Artisan	pll_533mhz_2004q4v1/
Sage-HS Cell Library for SoC voltage domain	Artisan	sage-hs_2003q1v1/
High-Speed, High-Current (HS-HC) RAMs for L2C RAMs, three RAM configurations characterized for IEM voltage control	Artisan	sram-sp-hs-hc_2004q1v1/
HS-HC RAMs for CLCDC, ETB11 and internal RAMs	Artisan	sram-dp-hs_2003q2v1/

B.3 IP modifications

The IP listed in Table B-3 has been modified for use with the ARM1176JZF Development Chip.

Table B-3 IP modifications

IP Product	Modification
TZPC	APB Peripheral. Extended DECPROT0, 1 and 2 registers from 8 to 16 bits.
AXI register slice, AxiRegSlice	Enables registers replicated to reduce fan-out and improve timing.
AHB exclusive access monitor	Master IDs, two, are soft configurable from the System Controller, rather than defined in the RTL.
DMC, pad interface	Replicated `data_en` pad data enable signal for each of the 32 DQ and each of the four DQS outputs to reduce fan-out and improve timing. Inverted versions of MDDR clock **clk_out[0-3]** added as **clk_out_n[0-3]** to support differential clock crossing required by SDRAM.
DMC	Configuration built with four exclusive access monitors, rather than the default of two.
DAP	-
MBIST	SOC level MBIST is custom code recorded from the 1176 MBIST solution.
AXI, arbitration	Made address arbitration soft configurable from the system controller. Separated AXI remap for IEM and SoC masters. Modified address decoders.
GIC	Modified peripheral bus timing logic to work with AXI. Documented registers for single core use.

B.4 Custom IP

The IP listed in Table B-4 is created specifically for the ARM1176JZF Development Chip.

Table B-4 Custom IP descriptions

IP Product	Description
System controller	APB peripheral. See Chapter 3 *Programmer's Model*.
Dynamic Clock Generator (DCG)	Custom clock generator. See *Intelligent Energy Management (IEM)* on page 2-38.
AXI ID merge block, Axi64IdMerge	Block created to reduce eight ID bits to six to: • Make better use of PL340 Quality of Service (QoS) • Reduce maximum ID width in SoC, the L2CC configuration port can only accept eight ID bits. Cascaded AXI blocks require nine bits. • Reduce ID bit width on AXI master port. See Chapter 2 *Functional Overview*.
Ring oscillators, RingOscGate	Ring oscillators created from gate delay elements to measure voltage in V_{soc} and V_{core} domains.
SoC Configuration Capture Block, SoCConfig	SOC Config block added custom code.

B.5 Configured IP

The AXI CAI tool PL300 was used to create two AXI interconnect modules. XML creates these modules.

Glossary

This glossary describes some of the terms used in technical documents from ARM Limited.

Abort A mechanism that indicates to a core that the value associated with a memory access is invalid. An abort can be caused by the external or internal memory system as a result of attempting to access invalid instruction or data memory. An abort is classified as either a Prefetch or Data Abort, and an internal or External Abort.

See also Data Abort, External Abort and Prefetch Abort.

Abort model An abort model is the defined behavior of an ARM processor in response to a Data Abort exception. Different abort models behave differently with regard to load and store instructions that specify base register write-back.

Addressing modes A mechanism, shared by many different instructions, for generating values used by the instructions. For four of the ARM addressing modes, the values generated are memory addresses (which is the traditional role of an addressing mode). A fifth addressing mode generates values to be used as operands by data-processing instructions.

Advanced eXtensible Interface (AXI)
A bus protocol that supports separate address/control and data phases, unaligned data transfers using byte strobes, burst-based transactions with only start address issued, separate read and write data channels to enable low-cost DMA, ability to issue multiple

outstanding addresses, out-of-order transaction completion, and easy addition of register stages to provide timing closure. The AXI protocol also includes optional extensions to cover signaling for low-power operation.

AXI is targeted at high performance, high clock frequency system designs and includes a number of features that make it very suitable for high speed sub-micron interconnect.

Advanced High-performance Bus (AHB)
A bus protocol with a fixed pipeline between address/control and data phases. It only supports a subset of the functionality provided by the AMBA AXI protocol. The full AMBA AHB protocol specification includes a number of features that are not commonly required for master and slave IP developments and ARM Limited recommends only a subset of the protocol is usually used. This subset is defined as the AMBA AHB-Lite protocol.

See also Advanced Microcontroller Bus Architecture and AHB-Lite.

Advanced Microcontroller Bus Architecture (AMBA)
A family of protocol specifications that describe a strategy for the interconnect. AMBA is the ARM open standard for on-chip buses. It is an on-chip bus specification that details a strategy for the interconnection and management of functional blocks that make up a *System-on-Chip* (SoC). It aids in the development of embedded processors with one or more CPUs or signal processors and multiple peripherals. AMBA complements a reusable design methodology by defining a common backbone for SoC modules.

Advanced Peripheral Bus (APB)
A simpler bus protocol than AXI and AHB. It is designed for use with ancillary or general-purpose peripherals such as timers, interrupt controllers, UARTs, and I/O ports. Connection to the main system bus is through a system-to-peripheral bus bridge that helps to reduce system power consumption.

AHB *See* Advanced High-performance Bus.

AHB Access Port (AHB-AP)
An optional component of the DAP that provides an AHB interface to a SoC.

AHB-AP *See* AHB Access Port.

AHB-Lite A subset of the full AMBA AHB protocol specification. It provides all of the basic functions required by the majority of AMBA AHB slave and master designs, particularly when used with a multi-layer AMBA interconnect. In most cases, the extra facilities provided by a full AMBA AHB interface are implemented more efficiently by using an AMBA AXI protocol interface.

Aligned A data item stored at an address that is divisible by the number of bytes that defines the data size is said to be aligned. Aligned words and halfwords have addresses that are divisible by four and two respectively. The terms word-aligned and halfword-aligned therefore stipulate addresses that are divisible by four and two respectively.

AMBA *See* Advanced Microcontroller Bus Architecture.

Advanced Trace Bus (ATB)
A bus used by trace devices to share CoreSight capture resources.

APB *See* Advanced Peripheral Bus.

Application Specific Integrated Circuit (ASIC)
An integrated circuit that has been designed to perform a specific application function. It can be custom-built or mass-produced.

Application Specific Standard Part/Product (ASSP)
An integrated circuit that has been designed to perform a specific application function. Usually consists of two or more separate circuit functions combined as a building block suitable for use in a range of products for one or more specific application markets.

Architecture The organization of hardware and/or software that characterizes a processor and its attached components, and enables devices with similar characteristics to be grouped together when describing their behavior, for example, Harvard architecture, instruction set architecture, ARMv6 architecture.

Arithmetic instruction
Any VFPv2 *Coprocessor Data Processing* (CDP) instruction except FCPY, FABS, and FNEG.

See also CDP instruction.

ARM instruction A word that specifies an operation for an ARM processor to perform. ARM instructions must be word-aligned.

ARM state A processor that is executing ARM (32-bit) word-aligned instructions is operating in ARM state.

ASIC *See* Application Specific Integrated Circuit.

ASSP *See* Application Specific Standard Part/Product.

ATB *See* Advanced Trace Bus.

ATB bridge A synchronous ATB bridge provides a register slice to facilitate timing closure through the addition of a pipeline stage. It also provides a unidirectional link between two synchronous ATB domains.

An asynchronous ATB bridge provides a unidirectional link between two ATB domains with asynchronous clocks. It is intended to support connection of components with ATB ports residing in different clock domains.

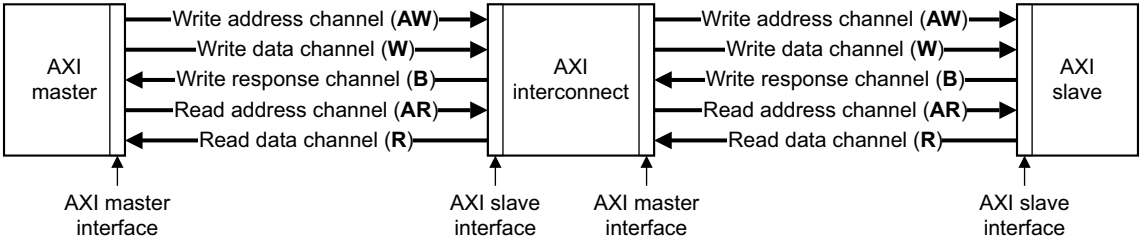
ATPG *See* Automatic Test Pattern Generation.

Automatic Test Pattern Generation (ATPG)
The process of automatically generating manufacturing test vectors for an ASIC design, using a specialized software tool.

AXI *See* Advanced eXtensible Interface.

AXI channel order and interfaces
The block diagram shows:

- the order in which AXI channel signals are described
- the master and slave interface conventions for AXI components.



AXI terminology The following AXI terms are general. They apply to both masters and slaves:

Active read transaction

A transaction for which the read address has transferred, but the last read data has not yet transferred.

Active transfer

A transfer for which the **xVALID**[1] handshake has asserted, but for which **xREADY** has not yet asserted.

1. The letter **x** in the signal name denotes an AXI channel as follows:

AW	Write address channel.
W	Write data channel.
B	Write response channel.
AR	Read address channel.
R	Read data channel.

Active write transaction

A transaction for which the write address or leading write data has transferred, but the write response has not yet transferred.

Completed transfer

A transfer for which the **xVALID/xREADY** handshake is complete.

Payload The non-handshake signals in a transfer.

Transaction An entire burst of transfers, comprising an address, one or more data transfers and a response transfer (writes only).

Transmit An initiator driving the payload and asserting the relevant **xVALID** signal.

Transfer A single exchange of information. That is, with one **xVALID/xREADY** handshake.

The following AXI terms are master interface attributes. To obtain optimum performance, they must be specified for all components with an AXI master interface:

Combined issuing capability

The maximum number of active transactions that a master interface can generate. This is specified instead of write or read issuing capability for master interfaces that use a combined storage for active write and read transactions.

Read ID capability

The maximum number of different **ARID** values that a master interface can generate for all active read transactions at any one time.

Read ID width

The number of bits in the **ARID** bus.

Read issuing capability

The maximum number of active read transactions that a master interface can generate.

Write ID capability

The maximum number of different **AWID** values that a master interface can generate for all active write transactions at any one time.

Write ID width

The number of bits in the **AWID** and **WID** buses.

Write interleave capability

The number of active write transactions for which the master interface is capable of transmitting data. This is counted from the earliest transaction.

Write issuing capability

The maximum number of active write transactions that a master interface can generate.

The following AXI terms are slave interface attributes. To obtain optimum performance, they must be specified for all components with an AXI slave interface

Combined acceptance capability

The maximum number of active transactions that a slave interface can accept. This is specified instead of write or read acceptance capability for slave interfaces that use a combined storage for active write and read transactions.

Read acceptance capability

The maximum number of active read transactions that a slave interface can accept.

Read data reordering depth

The number of active read transactions for which a slave interface can transmit data. This is counted from the earliest transaction.

Write acceptance capability

The maximum number of active write transactions that a slave interface can accept.

Write interleave depth

The number of active write transactions for which the slave interface can receive data. This is counted from the earliest transaction.

Back-annotation The process of applying timing characteristics from the implementation process onto a model.

Banked registers Those physical registers whose use is defined by the current processor mode. The banked registers are r8 to r14.

Base register A register specified by a load or store instruction that is used to hold the base value for the instruction's address calculation. Depending on the instruction and its addressing mode, an offset can be added to or subtracted from the base register value to form the virtual address that is sent to memory.

Base register write-back
: Updating the contents of the base register used in an instruction target address calculation so that the modified address is changed to the next higher or lower sequential address in memory. This means that it is not necessary to fetch the target address for successive instruction transfers and enables faster burst accesses to sequential memory.

Beat
: Alternative word for an individual transfer within a burst. For example, an INCR4 burst comprises four beats.

  See also Burst.

BE-8
: Big-endian view of memory in a byte-invariant system.

  See also BE-32, LE, Byte-invariant and Word-invariant.

BE-32
: Big-endian view of memory in a word-invariant system.

  See also BE-8, LE, Byte-invariant and Word-invariant.

Big-endian
: Byte ordering scheme in which bytes of decreasing significance in a data word are stored at increasing addresses in memory.

  See also Little-endian and Endianness.

Big-endian memory
: Memory in which:

  - a byte or halfword at a word-aligned address is the most significant byte or halfword within the word at that address
  - a byte at a halfword-aligned address is the most significant byte within the halfword at that address.

  See also Little-endian memory.

Block address
: An address that comprises a tag, an index, and a word field. The tag bits identify the way that contains the matching cache entry for a cache hit. The index bits identify the set being addressed. The word field contains the word address that can be used to identify specific words, halfwords, or bytes within the cache entry.

  See also Cache terminology diagram on the last page of this glossary.

Bounce
: The VFP coprocessor bounces an instruction when it fails to signal the acceptance of a valid VFP instruction to the ARM processor. This action initiates Undefined instruction processing by the ARM processor. The VFP support code is called to complete the instruction that was found to be exceptional or unsupported by the VFP coprocessor.

  See also Trigger instruction, Potentially exceptional instruction, and Exceptional state.

Boundary scan chain

A boundary scan chain is made up of serially-connected devices that implement boundary scan technology using a standard JTAG TAP interface. Each device contains at least one TAP controller containing shift registers that form the chain connected between **TDI** and **TDO**, through which test data is shifted. Processors can contain several shift registers to enable you to access selected parts of the device.

Branch folding

Branch folding is a technique where, on the prediction of most branches, the branch instruction is completely removed from the instruction stream presented to the execution pipeline. Branch folding can significantly improve the performance of branches, taking the CPI for branches below one.

Branch phantom

The condition codes of a predicted taken branch.

Branch prediction

The process of predicting if conditional branches are to be taken or not in pipelined processors. Successfully predicting if branches are to be taken enables the processor to prefetch the instructions following a branch before the condition is fully resolved. Branch prediction can be done in software or by using custom hardware. Branch prediction techniques are categorized as static, in which the prediction decision is decided before run time, and dynamic, in which the prediction decision can change during program execution.

Breakpoint

A breakpoint is a mechanism provided by debuggers to identify an instruction at which program execution is to be halted. Breakpoints are inserted by the programmer to enable inspection of register contents, memory locations, variable values at fixed points in the program execution to test that the program is operating correctly. Breakpoints are removed after the program is successfully tested.

See also Watchpoint.

Burst

A group of transfers to consecutive addresses. Because the addresses are consecutive, there is no requirement to supply an address for any of the transfers after the first one. This increases the speed at which the group of transfers can occur. Bursts over AMBA are controlled using signals to indicate the length of the burst and how the addresses are incremented.

See also Beat.

Byte

An 8-bit data item.

Byte-invariant

In a byte-invariant system, the address of each byte of memory remains unchanged when switching between little-endian and big-endian operation. When a data item larger than a byte is loaded from or stored to memory, the bytes making up that data item are arranged into the correct order depending on the endianness of the memory access.

The ARM architecture supports byte-invariant systems in ARMv6 and later versions. When byte-invariant support is selected, unaligned halfword and word memory accesses are also supported. Multi-word accesses are expected to be word-aligned.

See also Word-invariant.

Byte lane strobe A signal that is used for unaligned or mixed-endian data accesses to determine which byte lanes are active in a transfer. One bit of this signal corresponds to eight bits of the data bus.

Byte swizzling The reverse ordering of bytes in a word.

Cache A block of on-chip or off-chip fast access memory locations, situated between the processor and main memory, used for storing and retrieving copies of often used instructions and/or data. This is done to greatly increase the average speed of memory accesses and so improve processor performance.

See also Cache terminology diagram on the last page of this glossary.

Cache contention When the number of frequently-used memory cache lines that use a particular cache set exceeds the set-associativity of the cache. In this case, main memory activity increases and performance decreases.

Cache hit A memory access that can be processed at high speed because the instruction or data that it addresses is already held in the cache.

Cache line The basic unit of storage in a cache. It is always a power of two words in size (usually four or eight words), and is required to be aligned to a suitable memory boundary.

See also Cache terminology diagram on the last page of this glossary.

Cache line index The number associated with each cache line in a cache way. Within each cache way, the cache lines are numbered from 0 to (set associativity) -1.

See also Cache terminology diagram on the last page of this glossary.

Cache lockdown To fix a line in cache memory so that it cannot be overwritten. Cache lockdown enables critical instructions and/or data to be loaded into the cache so that the cache lines containing them are not subsequently reallocated. This ensures that all subsequent accesses to the instructions/data concerned are cache hits, and therefore complete as quickly as possible.

Cache miss A memory access that cannot be processed at high speed because the instruction/data it addresses is not in the cache and a main memory access is required.

Cache set A cache set is a group of cache lines (or blocks). A set contains all the ways that can be addressed with the same index. The number of cache sets is always a power of two.

See also Cache terminology diagram on the last page of this glossary.

Cache way
A group of cache lines (or blocks). It is 2 to the power of the number of index bits in size.

See also Cache terminology diagram on the last page of this glossary.

CAM
See Content Addressable Memory.

Cast out
See Victim.

CDP instruction
Coprocessor data processing instruction. For the VFP11 coprocessor, CDP instructions are arithmetic instructions and FCPY, FABS, and FNEG.

See also Arithmetic instruction.

Clean
A cache line that has not been modified while it is in the cache is said to be clean. To clean a cache is to write dirty cache entries into main memory. If a cache line is clean, it is not written on a cache miss because the next level of memory contains the same data as the cache.

See also Dirty.

Clock gating
Gating a clock signal for a macrocell with a control signal and using the modified clock that results to control the operating state of the macrocell.

Clocks Per Instruction (CPI)
See Cycles Per Instruction (CPI).

Coherency
See Memory coherency.

Cold reset
Also known as power-on reset. Starting the processor by turning power on. Turning power off and then back on again clears main memory and many internal settings. Some program failures can lock up the processor and require a cold reset to enable the system to be used again. In other cases, only a warm reset is required.

See also Warm reset.

Communications channel
The hardware used for communicating between the software running on the processor, and an external host, using the debug interface. When this communication is for debug purposes, it is called the Debug Comms Channel. In an ARMv6 compliant core, the communications channel includes the Data Transfer Register, some bits of the Data Status and Control Register, and the external debug interface controller, such as the DBGTAP controller in the case of the JTAG interface.

Condensed Reference Format (CRF)
An ARM proprietary file format for specifying test vectors.

Condition field
A four-bit field in an instruction that specifies a condition under which the instruction can execute.

Conditional execution
If the condition code flags indicate that the corresponding condition is true when the instruction starts executing, it executes normally. Otherwise, the instruction does nothing.

Content Addressable Memory (CAM)
Memory that is identified by its contents. Content Addressable Memory is used in CAM-RAM architecture caches to store the tags for cache entries.

CAM includes comparison logic with each bit of storage. A data value is broadcast to all words of storage and compared with the values there. Words that match are flagged in some way. Subsequent operations can then work on flagged words. It is possible to read the flagged words out one at a time or write to certain bit positions in all of them.

Context The environment that each process operates in for a multitasking operating system. In ARM processors, this is limited to mean the physical address range that it can access in memory and the associated memory access permissions.

See also Fast context switch.

Control bits The bottom eight bits of a Program Status Register. The control bits change when an exception arises and can be altered by software only when the processor is in a privileged mode.

Coprocessor A processor that supplements the main processor. It carries out additional functions that the main processor cannot perform. Usually used for floating-point math calculations, signal processing, or memory management.

Copy back *See* Write-back.

Core A core is that part of a processor that contains the ALU, the datapath, the general-purpose registers, the Program Counter, and the instruction decode and control circuitry.

Core module In the context of an ARM Integrator, a core module is an add-on development board that contains an ARM processor and local memory. Core modules can run standalone, or can be stacked onto Integrator motherboards.

Core reset *See* Warm reset.

CPI *See* Cycles per instruction.

CPSR *See* Current Program Status Register

CRF *See* Condensed Reference Format.

Cross Trigger Interface (CTI)
Part of an Embedded Cross Trigger device. The CTI provides the interface between a core/ETM and the CTM within an ECT.

Cross Trigger Matrix (CTM)
The CTM combines the trigger requests generated from CTIs and broadcasts them to all CTIs as channel triggers within an Embedded Cross Trigger device.

CTI *See* Cross Trigger Interface.

CTM *See* Cross Trigger Matrix.

Current Program Status Register (CPSR)
The register that holds the current operating processor status.

Cycles Per instruction (CPI)
Cycles per instruction (or clocks per instruction) is a measure of the number of computer instructions that can be performed in one clock cycle. This figure of merit can be used to compare the performance of different CPUs that implement the same instruction set against each other. The lower the value, the better the performance.

CoreSight The infrastructure for monitoring, tracing, and debugging a complete system on chip.

Data Abort An indication from a memory system to the core of an attempt to access an illegal data memory location. An exception must be taken if the processor attempts to use the data that caused the abort.

See also Abort, External Abort, and Prefetch Abort.

Data cache A block of on-chip fast access memory locations, situated between the processor and main memory, used for storing and retrieving copies of often used data. This is done to greatly increase the average speed of memory accesses and so improve processor performance.

DBGTAP *See* Debug Test Access Port.

Debug Access Port (DAP)
A TAP block that acts as an AMBA, AHB or AHB-Lite, master for access to a system bus. The DAP is the term used to encompass a set of modular blocks that support system wide debug. The DAP is a modular component, intended to be extendable to support optional access to multiple systems such as memory mapped AHB and CoreSight APB through a single debug interface.

Debugger A debugging system that includes a program, used to detect, locate, and correct software faults, together with custom hardware that supports software debugging.

Debug Test Access Port (DBGTAP)
The collection of four mandatory and one optional terminals that form the input/output and control interface to a JTAG boundary-scan architecture. The mandatory terminals are **DBGTDI**, **DBGTDO**, **DBGTMS**, and **TCK**. The optional terminal is **TRST**. This signal is mandatory in ARM cores because it is used to reset the debug logic.

Default NaN mode A mode in which all operations that result in a NaN return the default NaN, regardless of the cause of the NaN result. This mode is compliant with the IEEE 754 standard but implies that all information contained in any input NaNs to an operation is lost.

Delta cycle A simulation cycle in which the simulation time at the beginning of the cycle is the same as at the end of the cycle. That is, simulation time is not advanced in a delta cycle.

Delta-sweeping The process by which the VHDL simulator advances through delta cycles. A sweep covers many delta cycles.

Denormalized value *See* Subnormal value.

Design Simulation Model (DSM)
A functional simulation model of the device that is derived from the *Register Transfer Level* (RTL) but that does not reveal its internal structure. The DSM does not model any features added during synthesis such as internal scan chains. The DSM provides higher speed for functional simulation than that of the Sign-Off Model (SOM).

Device Validation Suite (DVS)
A set of tests to check the functionality of a device against the functionality defined in the Technical Reference Manual. Also stresses *Bus Interface Unit* (BIU), and low-level memory sub-system, pipeline, cache and *Tightly Coupled Memory* (TCM) behavior.

Direct-mapped cache
A one-way set-associative cache. Each cache set consists of a single cache line, so cache look-up selects and checks a single cache line.

Direct Memory Access (DMA)
An operation that accesses main memory directly, without the processor performing any accesses to the data concerned.

Dirty A cache line in a write-back cache that has been modified while it is in the cache is said to be dirty. A cache line is marked as dirty by setting the dirty bit. If a cache line is dirty, it must be written to memory on a cache miss because the next level of memory contains data that has not been updated. The process of writing dirty data to main memory is called cache cleaning.

See also Clean.

Disabled exception An exception is disabled when its exception enable bit in the FPCSR is not set. For these exceptions, the IEEE 754 standard defines the result to be returned. An operation that generates an exception condition can bounce to the support code to produce the result defined by the IEEE 754 standard. The exception is not reported to the user trap handler.

DMA *See* Direct Memory Access.

DNM *See* Do Not Modify.

Do Not Modify (DNM)
: In Do Not Modify fields, the value must not be altered by software. DNM fields read as Unpredictable values, and must only be written with the same value read from the same field on the same processor. DNM fields are sometimes followed by RAZ or RAO in parentheses to show which way the bits must read for future compatibility, but programmers must not rely on this behavior.

Double-precision value
: Consists of two 32-bit words that must appear consecutively in memory and must both be word-aligned, and that is interpreted as a basic double-precision floating-point number according to the IEEE 754-1985 standard.

Doubleword
: A 64-bit data item. The contents are taken as being an unsigned integer unless otherwise stated.

Doubleword-aligned
: A data item having a memory address that is divisible by eight.

DSM
: *See* Design Simulation Model.

DVS
: *See* Device Validation Suite.

ECT
: *See* Embedded Cross Trigger.

Embedded Cross Trigger (ECT)
: The ECT is a modular component to support the interaction and synchronization of multiple triggering events with an SoC.

EmbeddedICE logic
: An on-chip logic block that provides TAP-based debug support for ARM processor cores. It is accessed through the TAP controller on the ARM core using the JTAG interface.

EmbeddedICE-RT
: The JTAG-based hardware provided by debuggable ARM processors to aid debugging in real-time.

Embedded Trace Buffer
: The ETB provides on-chip storage of trace data using a configurable sized RAM.

Embedded Trace Macrocell (ETM)
: A hardware macrocell that, when connected to a processor core, outputs instruction and data trace information on a trace port. The ETM provides processor driven trace through a trace port compliant to the ATB protocol.

Enabled exception
: An exception is enabled when its exception enable bit in the FPCSR is set. When an enabled exception occurs, a trap to the user handler is taken. An operation that generates an exception condition might bounce to the support code to produce the result defined by the IEEE 754 standard. The exception is then reported to the user trap handler.

Endianness Byte ordering. The scheme that determines the order that successive bytes of a data word are stored in memory. An aspect of the system's memory mapping.

See also Little-endian and Big-endian

ETB *See* Embedded Trace Buffer.

ETM See *Embedded Trace Macrocell*.

Event 1 (Simple) An observable condition that can be used by an ETM to control aspects of a trace.

2 (Complex) A boolean combination of simple events that is used by an ETM to control aspects of a trace.

Exception A fault or error event that is considered serious enough to require that program execution is interrupted. Examples include attempting to perform an invalid memory access, external interrupts, and undefined instructions. When an exception occurs, normal program flow is interrupted and execution is resumed at the corresponding exception vector. This contains the first instruction of the interrupt handler to deal with the exception.

Exceptional state When a potentially exceptional instruction is issued, the VFP11 coprocessor sets the EX bit, FPEXC[31], and loads a copy of the potentially exceptional instruction in the FPINST register. If the instruction is a short vector operation, the register fields in FPINST are altered to point to the potentially exceptional iteration. When in the exceptional state, the issue of a trigger instruction to the VFP11 coprocessor causes a bounce.

See also Bounce, Potentially exceptional instruction, and Trigger instruction.

Exception service routine *See* Interrupt handler.

Exception vector *See* Interrupt vector.

Exponent The component of a floating-point number that normally signifies the integer power to which two is raised in determining the value of the represented number.

External Abort An indication from an external memory system to a core that the value associated with a memory access is invalid. An external abort is caused by the external memory system as a result of attempting to access invalid memory.

See also Abort, Data Abort and Prefetch Abort.

eXtensible Verification Component (XVC) A model that is used to provide system/device stimulus and monitor responses.

See also XVC Test Scenario Manager.

Fast context switch

In a multitasking system, the point at which the time-slice allocated to one process stops and the one for the next process starts. If processes are switched often enough, they can appear to a user to be running in parallel, in addition to being able to respond quicker to external events that might affect them.

In ARM processors, a fast context switch is caused by the selection of a non-zero PID value to switch the context to that of the next process. A fast context switch causes each Virtual Address for a memory access, generated by the ARM processor, to produce a Modified Virtual Address that is sent to the rest of the memory system to be used in place of a normal Virtual Address. For some cache control operations Virtual Addresses are passed to the memory system as data. In these cases no address modification takes place.

See also Fast Context Switch Extension.

Fast Context Switch Extension (FCSE)

An extension to the ARM architecture that enables cached processors with an MMU to present different addresses to the rest of the memory system for different software processes, even when those processes are using identical addresses.

See also Fast context switch.

FCSE *See* Fast Context Switch Extension.

Fd The destination register and the accumulate value in triadic operations. Sd for single-precision operations and Dd for double-precision.

Flash Patch and Breakpoint unit (FPB)

A set of address matching tags, that reroute accesses into flash to a special part of SRAM. This permits patching flash locations for breakpointing and quick fixes or changes.

Flat address mapping

A system of organizing memory in which each Physical Address contained within the memory space is the same as its corresponding Virtual Address.

Flush-to-zero mode In this mode, the VFP11 coprocessor treats the following values as positive zeros:

- arithmetic operation inputs that are in the subnormal range for the input precision
- arithmetic operation results, other than computed zero results, that are in the subnormal range for the input precision before rounding.

The VFP11 coprocessor does not interpret these values as subnormal values or convert them to subnormal values.

The subnormal range for the input precision is $-2^{Emin} < x < 0$ or $0 < x < 2^{Emin}$.

Fm The second source operand in dyadic or triadic operations. Sm for single-precision operations and Dm for double-precision.

Fn The first source operand in dyadic or triadic operations. Sn for single-precision operations and Dn for double-precision.

Formatter The formatter is an internal input block in the ETB and TPIU that embeds the trace source ID within the data to create a single trace stream.

Fraction The floating-point field that lies to the right of the implied binary point.

Front of queue pointer
Pointer to the next entry to be written to in the write buffer.

Fully-associative cache
A cache that has one cache set that consists of the entire cache. The number of cache entries is the same as the number of cache ways.

See also Direct-mapped cache.

Gray code Continuous binary code in which only one bit changes for a change to the next state up or down.

Half-rate clocking (ETM)
Dividing the trace clock by two so that the TPA can sample trace data signals on both the rising and falling edges of the trace clock. The primary purpose of half-rate clocking is to reduce the signal transition rate on the trace clock of an ASIC for very high-speed systems.

Halfword A 16-bit data item.

Halt mode One of two mutually exclusive debug modes. In halt mode all processor execution halts when a breakpoint or watchpoint is encountered. All processor state, coprocessor state, memory and input/output locations can be examined and altered by the JTAG interface.

See also Monitor debug-mode.

High vectors Alternative locations for exception vectors. The high vector address range is near the top of the address space, rather than at the bottom.

Hit-Under-Miss (HUM)
A buffer that enables program execution to continue, even though there has been a data miss in the cache.

Host A computer that provides data and other services to another computer. Especially, a computer providing debugging services to a target being debugged.

HUM *See* Hit-Under-Miss.

IEEE 754 standard *IEEE Standard for Binary Floating-Point Arithmetic, ANSI/IEEE Std. 754-1985*. The standard that defines data types, correct operation, exception types and handling, and error bounds for floating-point systems. Most processors are built in compliance with the standard in either hardware or a combination of hardware and software.

IEM *See* Intelligent Energy Management.

IGN *See* Ignore.

Ignore (IGN) Must ignore memory writes.

Illegal instruction An instruction that is architecturally Undefined.

IMB *See* Instruction Memory Barrier.

Implementation-defined
The behavior is not architecturally defined, but is defined and documented by individual implementations.

Implementation-specific
The behavior is not architecturally defined, and does not have to be documented by individual implementations. Used when there are a number of implementation options available and the option chosen does not affect software compatibility.

Imprecise tracing A filtering configuration where instruction or data tracing can start or finish earlier or later than expected. Most cases cause tracing to start or finish later than expected.

For example, if **TraceEnable** is configured to use a counter so that tracing begins after the fourth write to a location in memory, the instruction that caused the fourth write is not traced, although subsequent instructions are. This is because the use of a counter in the **TraceEnable** configuration always results in imprecise tracing.

Index *See* Cache index.

Index register A register specified in some load or store instructions. The value of this register is used as an offset to be added to or subtracted from the base register value to form the virtual address that is sent to memory. Some addressing modes optionally enable the index register value to be shifted prior to the addition or subtraction.

Infinity In the IEEE 754 standard format to represent infinity, the exponent is the maximum for the precision and the fraction is all zeros.

Input exception A VFP exception condition in which one or more of the operands for a given operation are not supported by the hardware. The operation bounces to support code for processing.

Instruction cache A block of on-chip fast access memory locations, situated between the processor and main memory, used for storing and retrieving copies of often used instructions. This is done to greatly increase the average speed of memory accesses and so improve processor performance.

Instruction cycle count
The number of cycles for which an instruction occupies the Execute stage of the pipeline.

Instruction Memory Barrier (IMB)
An operation to ensure that the prefetch buffer is flushed of all out-of-date instructions.

Instrumentation trace
A component for debugging real-time systems through a simple memory-mapped trace interface, providing `printf` style debugging.

Intelligent Energy Management (IEM)
A technology that enables dynamic voltage scaling and clock frequency variation to be used to reduce power consumption in a device.

Intermediate result An internal format used to store the result of a calculation before rounding. This format can have a larger exponent field and fraction field than the destination format.

Internal scan chain A series of registers connected together to form a path through a device, used during production testing to import test patterns into internal nodes of the device and export the resulting values.

Interrupt handler A program that control of the processor is passed to when an interrupt occurs.

Interrupt vector One of a number of fixed addresses in low memory, or in high memory if high vectors are configured, that contains the first instruction of the corresponding interrupt handler.

Invalidate To mark a cache line as being not valid by clearing the valid bit. This must be done whenever the line does not contain a valid cache entry. For example, after a cache flush all lines are invalid.

Jazelle architecture The ARM Jazelle architecture extends the Thumb and ARM operating states by adding a Java state to the processor. Instruction set support for entering and exiting Java applications, real-time interrupt handling, and debug support for mixed Java/ARM applications is present. When in Java state, the processor fetches and decodes Java bytecodes and maintains the Java operand stack.

Joint Test Action Group (JTAG)
The name of the organization that developed standard IEEE 1149.1. This standard defines a boundary-scan architecture used for in-circuit testing of integrated circuit devices. It is commonly known by the initials JTAG.

JTAG *See* Joint Test Action Group.

JTAG Access Port (JTAG-AP)
An optional component of the DAP that provides JTAG access to on-chip components, operating as a JTAG master port to drive JTAG chains throughout a SoC.

JTAG-AP *See* JTAG Access Port.

JTAG Debug Port (JTAG-DP)
An optional external interface for the DAP that provides a standard JTAG interface for debug access.

JTAG-DP *See* JTAG Debug Port.

LE Little endian view of memory in both byte-invariant and word-invariant systems. See also Byte-invariant, Word-invariant.

Line *See* Cache line.

Little-endian Byte ordering scheme in which bytes of increasing significance in a data word are stored at increasing addresses in memory.

See also Big-endian and Endianness.

Little-endian memory
Memory in which:

- a byte or halfword at a word-aligned address is the least significant byte or halfword within the word at that address
- a byte at a halfword-aligned address is the least significant byte within the halfword at that address.

See also Big-endian memory.

Load/store architecture
A processor architecture where data-processing operations only operate on register contents, not directly on memory contents.

Load Store Unit (LSU)
The part of a processor that handles load and store transfers.

LSU *See* Load Store Unit.

Macrocell A complex logic block with a defined interface and behavior. A typical VLSI system comprises several macrocells (such as a processor, an ETM, and a memory block) plus application-specific logic.

Memory bank One of two or more parallel divisions of interleaved memory, usually one word wide, that enable reads and writes of multiple words at a time, rather than single words. All memory banks are addressed simultaneously and a bank enable or chip select signal

determines which of the banks is accessed for each transfer. Accesses to sequential word addresses cause accesses to sequential banks. This enables the delays associated with accessing a bank to occur during the access to its adjacent bank, speeding up memory transfers.

Memory coherency A memory is coherent if the value read by a data read or instruction fetch is the value that was most recently written to that location. Memory coherency is made difficult when there are multiple possible physical locations that are involved, such as a system that has main memory, a write buffer and a cache.

Memory Management Unit (MMU)
Hardware that controls caches and access permissions to blocks of memory, and translates virtual addresses to physical addresses.

Memory Protection Unit (MPU)
Hardware that controls access permissions to blocks of memory. Unlike an MMU, an MPU does not translate virtual addresses to physical addresses.

Microprocessor *See* Processor.

Miss *See* Cache miss.

MMU *See* Memory Management Unit.

Model Manager A software control manager that handles the event transactions between the model and simulator.

Modified Virtual Address (MVA)
A Virtual Address produced by the ARM processor can be changed by the current Process ID to provide a *Modified Virtual Address* (MVA) for the MMUs and caches.

See also Fast Context Switch Extension.

Monitor debug-mode
One of two mutually exclusive debug modes. In Monitor debug-mode the processor enables a software abort handler provided by the debug monitor or operating system debug task. When a breakpoint or watchpoint is encountered, this enables vital system interrupts to continue to be serviced while normal program execution is suspended.

See also Halt mode.

Mono_DAP *See* Mono Debug Access Port.

Mono Debug Access Port (Mono-DAP)
A JTAG to APB bridge that supports the JTAG programming model for access to the ETM or ETB using the existing scan chain.

Mono-TPIU *See* Mono Trace Port Interface Unit.

Mono Trace Port Interface Unit (Mono-TPIU)
A cut down TPIU for tracing a single source that supports backwards-compatible programming of PortSize.

MPU *See* Memory Protection Unit.

Multi-ICE A JTAG-based tool for debugging embedded systems.

Multi-layer An interconnect scheme similar to a cross-bar switch. Each master on the interconnect has a direct link to each slave, The link is not shared with other masters. This enables each master to process transfers in parallel with other masters. Contention only occurs in a multi-layer interconnect at a payload destination, typically the slave.

Multi-master AHB Typically a shared, not multi-layer, AHB interconnect scheme. More than one master connects to a single AMBA AHB link. In this case, the bus is implemented with a set of full AMBA AHB master interfaces. Masters that use the AMBA AHB-Lite protocol must connect through a wrapper to supply full AMBA AHB master signals to support multi-master operation.

MVA *See* Modified Virtual Address.

NaN Not a number. A symbolic entity encoded in a floating-point format that has the maximum exponent field and a nonzero fraction. An SNaN causes an invalid operand exception if used as an operand and a most significant fraction bit of zero. A QNaN propagates through almost every arithmetic operation without signaling exceptions and has a most significant fraction bit of one.

PA *See* Physical Address.

Penalty The number of cycles in which no useful Execute stage pipeline activity can occur because an instruction flow is different from that assumed or predicted.

Potentially exceptional instruction
An instruction that is determined, based on the exponents of the operands and the sign bits, to have the potential to produce an overflow, underflow, or invalid condition. After this determination is made, the instruction that has the potential to cause an exception causes the VFP11 coprocessor to enter the exceptional state and bounce the next trigger instruction issued.

See also Bounce, Trigger instruction, and Exceptional state.

Power-on reset *See* Cold reset.

Prefetching In pipelined processors, the process of fetching instructions from memory to fill up the pipeline before the preceding instructions have finished executing. Prefetching an instruction does not mean that the instruction has to be executed.

Prefetch Abort An indication from a memory system to the core that an instruction has been fetched from an illegal memory location. An exception must be taken if the processor attempts to execute the instruction. A Prefetch Abort can be caused by the external or internal memory system as a result of attempting to access invalid instruction memory.

See also Data Abort, External Abort and Abort.

Processor A processor is the circuitry in a computer system required to process data using the computer instructions. It is an abbreviation of microprocessor. A clock source, power supplies, and main memory are also required to create a minimum complete working computer system.

Programming Language Interface (PLI)
For Verilog simulators, an interface by which so-called foreign code (code written in a different language) can be included in a simulation.

Physical Address (PA)
The MMU performs a translation on *Modified Virtual Addresses* (MVA) to produce the *Physical Address* (PA) that is given to the AMBA bus to perform an external access. The PA is also stored in the data cache to avoid the necessity for address translation when data is cast out of the cache.

See also Fast Context Switch Extension.

Read Reads are defined as memory operations that have the semantics of a load. That is, the ARM instructions LDM, LDRD, LDC, LDR, LDRT, LDRSH, LDRH, LDRSB, LDRB, LDRBT, LDREX, RFE, STREX, SWP, and SWPB, and the Thumb instructions LDM, LDR, LDRSH, LDRH, LDRSB, LDRB, and POP.

Java instructions that are accelerated by hardware can cause a number of reads to occur, according to the state of the Java stack and the implementation of the Java hardware acceleration.

RealView ICE A system for debugging embedded processor cores using a JTAG interface.

Region A partition of instruction or data memory space.

Remapping Changing the address of physical memory or devices after the application has started executing. This is typically done to permit RAM to replace ROM when the initialization has been completed.

Replicator A replicator enables two trace sinks to be wired together and to operate independently on the same incoming trace stream. The input trace stream is output onto two (independent) ATB ports.

Reserved A field in a control register or instruction format is reserved if the field is to be defined by the implementation, or produces Unpredictable results if the contents of the field are not zero. These fields are reserved for use in future extensions of the architecture or are implementation-specific. All reserved bits not used by the implementation must be written as 0 and read as 0.

Rounding mode The IEEE 754 standard requires all calculations to be performed as if to an infinite precision. For example, a multiply of two single-precision values must accurately calculate the significand to twice the number of bits of the significand. To represent this value in the destination precision, rounding of the significand is often required. The IEEE 754 standard specifies four rounding modes.

In round-to-nearest mode, the result is rounded at the halfway point, with the tie case rounding up if it would clear the least significant bit of the significand, making it even.

Round-towards-zero mode chops any bits to the right of the significand, always rounding down, and is used by the C, C++, and Java languages in integer conversions.

Round-towards-plus-infinity mode and round-towards-minus-infinity mode are used in interval arithmetic.

RunFast mode In RunFast mode, hardware handles exceptional conditions and special operands. RunFast mode is enabled by enabling default NaN and flush-to-zero modes and disabling all exceptions. In RunFast mode, the VFP11 coprocessor does not bounce to the support code for any legal operation or any operand, but supplies a result to the destination. For all inexact and overflow results and all invalid operations that result from operations not involving NaNs, the result is as specified by the IEEE 754 standard. For operations involving NaNs, the result is the default NaN.

Saved Program Status Register (SPSR)
The register that holds the CPSR of the task immediately before the exception occurred that caused the switch to the current mode.

SBO *See* Should Be One.

SBZ *See* Should Be Zero.

SBZP *See* Should Be Zero or Preserved.

Scalar operation A VFP coprocessor operation involving a single source register and a single destination register.

See also Vector operation.

Scan Access Port (Scan-AP)
An optional component of the DAP that provides test access to on-chip scan chains.

Scan-AP *See* Scan Access Port.

Scan chain A scan chain is made up of serially-connected devices that implement boundary scan technology using a standard JTAG TAP interface. Each device contains at least one TAP controller containing shift registers that form the chain connected between **TDI** and **TDO**, through which test data is shifted. Processors can contain several shift registers to enable you to access selected parts of the device.

SCREG The currently selected scan chain number in an ARM TAP controller.

SDF *See* Standard Delay Format.

Set *See* Cache set.

Set-associative cache

In a set-associative cache, lines can only be placed in the cache in locations that correspond to the modulo division of the memory address by the number of sets. If there are *n* ways in a cache, the cache is termed *n*-way set-associative. The set-associativity can be any number greater than or equal to 1 and is not restricted to being a power of two.

Short vector operation

A VFP coprocessor operation involving more than one destination register and perhaps more than one source register in the generation of the result for each destination.

Should Be One (SBO)

Should be written as 1 (or all 1s for bit fields) by software. Writing a 0 produces Unpredictable results.

Should Be Zero (SBZ)

Should be written as 0 (or all 0s for bit fields) by software. Writing a 1 produces Unpredictable results.

Should Be Zero or Preserved (SBZP)

Should be written as 0 (or all 0s for bit fields) by software, or preserved by writing the same value back that has been previously read from the same field on the same processor.

Significand The component of a binary floating-point number that consists of an explicit or implicit leading bit to the left of the implied binary point and a fraction field to the right.

Sign-Off Model (SOM)

An opaque, compiled simulation model generated from a technology specific netlist of an ARM core, derived after gate level synthesis and timing annotation, that you can use in back-annotated gate-level simulations to prove the function and timing behavior of the device. It enables accurate timing simulation of SoCs and simulation using production test vectors from Automatic Test Pattern Generation (ATPG) tool such as Synopsys TetraMAX. It only supports back-annotation using SDF files. The SOM includes timing information but provides slower simulation than a DSM.

Single-Wire Debug Port
: An optional external interface for the DAP that provides a single-wire bidirectional debug interface.

Single-Wire JTAG (SWJ)
: A model whereby a run-control emulator (based on RVI-ME) is placed in the chip and communicated with using a single pin scheme (compared to the four to six for JTAG). This not only reduces pins, but SWJ provides power from the run-control emulator (through the pin). It also provides additional access and a unique ID. The use of this DBT model enables this mode to run very fast for download.

SOM
: *See* Sign-Off Model.

SPICE
: Simulation Program with Integrated Circuit Emphasis. An accurate transistor-level electronic circuit simulation tool that can predict how an equivalent real circuit behaves for given circuit conditions.

SPSR
: *See* Saved Program Status Register

Standard Delay Format (SDF)
: The format of a file that contains timing information to the level of individual bits of buses and is used in SDF back-annotation. An SDF file can be generated in a number of ways, but most commonly from a delay calculator.

Stride
: The stride field, FPSCR[21:20], specifies the increment applied to register addresses in short vector operations. A stride of 00, specifying an increment of +1, causes a short vector operation to increment each vector register by +1 for each iteration, while a stride of 11 specifies an increment of +2.

Subnormal value
: A value in the range ($-2^{Emin} < x < 2^{Emin}$), except for 0. In the IEEE 754 standard format for single-precision and double-precision operands, a subnormal value has a zero exponent and a nonzero fraction field. The IEEE 754 standard requires that the generation and manipulation of subnormal operands be performed with the same precision as normal operands.

Support code
: Software that must be used to complement the hardware to provide compatibility with the IEEE 754 standard. The support code has a library of routines that performs supported functions, such as divide with unsupported inputs or inputs that might generate an exception in addition to operations beyond the scope of the hardware. The support code has a set of exception handlers to process exceptional conditions in compliance with the IEEE 754 standard.

SW-DP
: *See* Single-Wire Debug Port.

SWJ
: *See* Single-Wire JTAG.

Synchronization primitive
The memory synchronization primitive instructions are those instructions that are used to ensure memory synchronization. That is, the LDREX, STREX, SWP, and SWPB instructions.

Tag The upper portion of a block address used to identify a cache line within a cache. The block address from the CPU is compared with each tag in a set in parallel to determine if the corresponding line is in the cache. If it is, it is said to be a cache hit and the line can be fetched from cache. If the block address does not correspond to any of the tags, it is said to be a cache miss and the line must be fetched from the next level of memory.

See also Cache terminology diagram on the last page of this glossary.

TAP *See* Test access port.

TCD *See* Trace Capture Device.

TCM *See* Tightly coupled memory.

Test Access Port (TAP)
The collection of four mandatory and one optional terminals that form the input/output and control interface to a JTAG boundary-scan architecture. The mandatory terminals are **TDI**, **TDO**, **TMS**, and **TCK**. The optional terminal is **TRST**. This signal is mandatory in ARM cores because it is used to reset the debug logic.

Thumb instruction A halfword that specifies an operation for an ARM processor in Thumb state to perform. Thumb instructions must be halfword-aligned.

Thumb state A processor that is executing Thumb (16-bit) halfword aligned instructions is operating in Thumb state.

Tightly coupled memory (TCM)
An area of low latency memory that provides predictable instruction execution or data load timing in cases where deterministic performance is required. TCMs are suited to holding:

- critical routines such as for interrupt handling
- scratchpad data
- data types whose locality is not suited to caching
- critical data structures, such as interrupt stacks.

Tiny A nonzero result or value that is between the positive and negative minimum normal values for the destination precision.

TLB *See* Translation Look-aside Buffer.

TPA See *Trace Port Analyzer*.

TPIU *See* Trace Port Interface Unit.

Trace Capture Device (TCD)
A generic term to describe Trace Port Analyzers, logic analyzers, and on-chip trace buffers.

Trace driver A Remote Debug Interface target that controls a piece of trace hardware. That is, the trigger macrocell, trace macrocell, and trace capture tool.

Trace funnel A device that combines multiple trace sources onto a single bus.

Trace hardware A term for a device that contains an Embedded Trace Macrocell.

Trace port A port on a device, such as a processor or ASIC, used to output trace information.

Trace Port Analyzer (TPA)
A hardware device that captures trace information output on a trace port. This can be a low-cost product designed specifically for trace acquisition, or a logic analyzer.

Trace Port Interface Unit (TPIU)
Drains trace data and acts as a bridge between the on-chip trace data and the data stream captured by a TPA.

Translation Lookaside Buffer (TLB)
A cache of recently used page table entries that avoid the overhead of page table walking on every memory access. Part of the Memory Management Unit.

Translation table A table, held in memory, that contains data that defines the properties of memory areas of various fixed sizes.

Translation table walk
The process of doing a full translation table lookup. It is performed automatically by hardware.

Trap An exceptional condition in a VFP coprocessor that has the respective exception enable bit set in the FPSCR register. The user trap handler is executed.

Trigger instruction The VFP coprocessor instruction that causes a bounce at the time it is issued. A potentially exceptional instruction causes the VFP11 coprocessor to enter the exceptional state. A subsequent instruction, unless it is an FMXR or FMRX instruction accessing the FPEXC, FPINST, or FPSID register, causes a bounce, beginning exception processing. The trigger instruction is not necessarily exceptional, and no processing of it is performed. It is retried at the return from exception processing of the potentially exceptional instruction.

See also Bounce, Potentially exceptional instruction, and Exceptional state.

Unaligned A data item stored at an address that is not divisible by the number of bytes that defines the data size is said to be unaligned. For example, a word stored at an address that is not divisible by four.

Undefined Indicates an instruction that generates an Undefined instruction trap. See the *ARM Architecture Reference Manual* for more details on ARM exceptions.

UNP *See* Unpredictable.

Unpredictable Means that the behavior of the ETM cannot be relied on. Such conditions have not been validated. When applied to the programming of an event resource, only the output of that event resource is Unpredictable. Unpredictable behavior can affect the behavior of the entire system, because the ETM is capable of causing the core to enter debug state, and external outputs can be used for other purposes.

Unpredictable For reads, the data returned when reading from this location is unpredictable. It can have any value. For writes, writing to this location causes unpredictable behavior, or an unpredictable change in device configuration. Unpredictable instructions must not halt or hang the processor, or any part of the system.

Unsupported values

Specific data values that are not processed by the VFP coprocessor hardware but bounced to the support code for completion. These data can include infinities, NaNs, subnormal values, and zeros. An implementation is free to select which of these values is supported in hardware fully or partially, or requires assistance from support code to complete the operation. Any exception resulting from processing unsupported data is trapped to user code if the corresponding exception enable bit for the exception is set.

VA *See* Virtual Address.

Vector operation A VFP coprocessor operation involving more than one destination register, perhaps involving different source registers in the generation of the result for each destination.

See also Scalar operation.

Victim A cache line, selected to be discarded to make room for a replacement cache line that is required because of a cache miss. The way that the victim is selected for eviction is processor-specific. A victim is also known as a cast out.

Virtual Address (VA)

The MMU uses its page tables to translate a Virtual Address into a Physical Address. The processor executes code at the Virtual Address, that might be located elsewhere in physical memory.

See also Fast Context Switch Extension, Modified Virtual Address, and Physical Address.

Warm reset Also known as a core reset. Initializes the majority of the processor excluding the debug controller and debug logic. This type of reset is useful if you are using the debugging features of a processor.

Watchpoint A watchpoint is a mechanism provided by debuggers to halt program execution when the data contained by a particular memory address is changed. Watchpoints are inserted by the programmer to enable inspection of register contents, memory locations, and variable values when memory is written to test that the program is operating correctly. Watchpoints are removed after the program is successfully tested. *See also* Breakpoint.

Way *See* Cache way.

WB *See* Write-back.

Word A 32-bit data item.

Word-invariant In a word-invariant system, the address of each byte of memory changes when switching between little-endian and big-endian operation, in such a way that the byte with address A in one endianness has address A EOR 3 in the other endianness. As a result, each aligned word of memory always consists of the same four bytes of memory in the same order, regardless of endianness. The change of endianness occurs because of the change to the byte addresses, not because the bytes are rearranged. The ARM architecture supports word-invariant systems in ARMv3 and later versions. When word-invariant support is selected, the behavior of load or store instructions that are given unaligned addresses is instruction-specific, and is in general not the expected behavior for an unaligned access. It is recommended that word-invariant systems use the endianness that produces the desired byte addresses at all times, apart possibly from very early in their reset handlers before they have set up the endianness, and that this early part of the reset handler must use only aligned word memory accesses.

See also Byte-invariant.

Write Writes are defined as operations that have the semantics of a store. That is, the ARM instructions SRS, STM, STRD, STC, STRT, STRH, STRB, STRBT, STREX, SWP, and SWPB, and the Thumb instructions STM, STR, STRH, STRB, and PUSH.

Java instructions that are accelerated by hardware can cause a number of writes to occur, according to the state of the Java stack and the implementation of the Java hardware acceleration.

Write-back (WB) In a write-back cache, data is only written to main memory when it is forced out of the cache on line replacement following a cache miss. Otherwise, writes by the processor only update the cache. This is also known as copyback.

Write buffer A block of high-speed memory, arranged as a FIFO buffer, between the data cache and main memory, whose purpose is to optimize stores to main memory.

Write completion The memory system indicates to the processor that a write has been completed at a point in the transaction where the memory system is able to guarantee that the effect of the write is visible to all processors in the system. This is not the case if the write is associated with a memory synchronization primitive, or is to a Device or Strongly

Ordered region. In these cases the memory system might only indicate completion of the write when the access has affected the state of the target, unless it is impossible to distinguish between having the effect of the write visible and having the state of target updated.

This stricter requirement for some types of memory ensures that any side-effects of the memory access can be guaranteed by the processor to have taken place. You can use this to prevent the starting of a subsequent operation in the program order until the side-effects are visible.

Write-through (WT) In a write-through cache, data is written to main memory at the same time as the cache is updated.

WT *See* Write-through.

XVC *See* eXtensible Verification Component.

XVC Test Scenario Manager
This co-ordinates the operation of multiple XVCs.

See also eXtensible Verification Component

XTSM *See* XVC Test Scenario Manager.

Cache terminology diagram
The diagram illustrates the following cache terminology:

- block address
- cache line
- cache set
- cache way
- index

- tag.