Fixed Virtual Platforms

Version 1.4

FVP Reference Guide

ARM

ARM DUI0837H

Fixed Virtual Platforms

FVP Reference Guide

Release Information

Document History

Issue	Date	Confidentiality	Change
A	31 May 2014	Non-Confidential	New document for Fast Models v9.0, from DUI0575H for v8.3.
B	30 November 2014	Non-Confidential	Update for v9.1.
C	28 February 2015	Non-Confidential	Update for v9.2.
D	31 May 2015	Non-Confidential	Update for v9.3.
E	31 August 2015	Non-Confidential	Update for v9.4.
F	30 November 2015	Non-Confidential	Update for v9.5.
G	29 February 2016	Non-Confidential	Update for v9.6.
H	31 May 2016	Non-Confidential	Update for v10.0.

ARM Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

LES-PRE-20349

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.

Unrestricted Access is an ARM internal classification.

Product Status

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

Web Address

http://www.arm.com

Contents
Fixed Virtual Platforms FVP Reference Guide

Preface

This preface introduces the *Fixed Virtual Platforms FVP Reference Guide*.

It contains the following:

About this book

ARM Fixed Virtual Platform Reference. This manual introduces the Fixed Virtual Platforms, and describes how you can use them with other tools.

Using this book

This book is organized into the following chapters:

Chapter 1 Introduction

This chapter introduces the document.

Chapter 2 Getting Started with Fixed Virtual Platforms

This chapter describes how to use FVPs.

Chapter 3 Programming Reference for Base FVPs

This chapter describes the memory map and the parameters for the peripheral and system component models.

Chapter 4 Programming Reference for MPS2 FVPs

This chapter describes the model of the hardware platform.

Chapter 5 Programming Reference for VE FVPs

This chapter describes the memory map and the parameters for the peripheral and system component models.

Glossary

The ARM Glossary is a list of terms used in ARM documentation, together with definitions for those terms. The ARM Glossary does not contain terms that are industry standard unless the ARM meaning differs from the generally accepted meaning.

See the *ARM Glossary* for more information.

Typographic conventions

italic

Introduces special terminology, denotes 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.

`monospace`

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>`

Encloses replaceable terms for assembler syntax where they appear in code or code fragments. For example:

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

SMALL CAPITALS

Used in body text for a few terms that have specific technical meanings, that are defined in the *ARM glossary*. For example, IMPLEMENTATION DEFINED, IMPLEMENTATION SPECIFIC, UNKNOWN, and UNPREDICTABLE.

Feedback

Feedback on this product

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

- The product name.
- The product revision or version.
- An explanation with as much information as you can provide. Include symptoms and diagnostic procedures if appropriate.

Feedback on content

If you have comments on content then send an e-mail to *errata@arm.com*. Give:

- The title *Fixed Virtual Platforms FVP Reference Guide*.
- The number ARM DUI0837H.
- If applicable, the page number(s) to which your comments refer.
- A concise explanation of your comments.

ARM also welcomes general suggestions for additions and improvements.

Note

ARM tests the PDF only in Adobe Acrobat and Acrobat Reader, and cannot guarantee the quality of the represented document when used with any other PDF reader.

Other information

- *ARM Information Center*.
- *ARM Technical Support Knowledge Articles*.
- *Support and Maintenance*.
- *ARM Glossary*.

Chapter 1
Introduction

This chapter introduces the document.

It contains the following sections:

1.1 About FVPs

Fixed Virtual Platforms (FVPs), or system models, enable development of software without the requirement for actual hardware.

The software models provide *Programmer's View* (PV) models of processors and devices. The functional behavior of a model is equivalent to real hardware. PV models sacrifice absolute timing accuracy to achieve fast simulated execution speed. This means that you can use the PV models for confirming software functionality, but you must not rely on the accuracy of cycle counts, low-level component interactions, or other hardware-specific behavior.

ARM supplies some PV models, Fast Models, as *Component Architecture Debug Interface* (CADI) shared libraries. Any environment compatible with the CADI API can load them, like Model Debugger, Model Shell, and System Canvas.

FVPs are non-customizable PV models suitable for software development, which ARM supplies as executable files.

Chapter 2
Getting Started with Fixed Virtual Platforms

This chapter describes how to use FVPs.

It contains the following sections:

2.1 FVP Debug

This section describes how to debug an FVP.

FVP Debug options

To debug an FVP, you can:

- Start the FVP from within a debugger.
- Connect a debugger to a model that is already running.

You can use your own debugger if it has a CADI interface to connect to an FVP. For information about using your debugger in this way, see your debugger documentation.

Semihosting support

Semi-hosting enables code running on a platform model to directly access the I/O facilities on a host computer. Examples of these facilities include console I/O and file I/O.

The simulator handles semihosting by intercepting HLT `0xF000`, or SVC `0x123456` or `0xAB`, depending on whether the processor is in A64, A32 or T32 state. It handles all other HLTs and SVCs as normal.

If the operating system does not use HLT `0xF000`, SVC `0x123456` or `0xAB` for its own purposes, it is not necessary to disable semihosting support to boot an operating system.

To temporarily or permanently disable semihosting support for a current debug connection, see your debugger documentation.

Related information

ARM Compiler toolchain Developing Software for ARM Processors.

2.2 Using the VE CLCD window

When an FVP starts, the FVP CLCD window opens, representing the contents of the simulated color LCD framebuffer. It automatically resizes to match the horizontal and vertical resolution that are set in the CLCD peripheral registers.

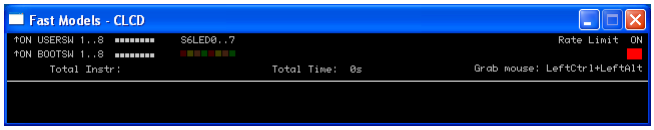

Figure 2-1 CLCD window in its default state at startup

The top section of the CLCD window displays the status information.

USERSW

Eight white boxes show the state of the VE User DIP switches:

These represent switch S6 on the VE hardware, USERSW[8:1], which is mapped to bits [7:0] of the SYS_SW register at address `0x1C010004`.

The switches are in the off position by default. To change its state, click in the area above or below a white box.

BOOTSW

Eight white boxes show the state of the VE Boot DIP switches.

These represent switch S8 on the VE hardware, BOOTSEL[8:1], which is mapped to bits [15:8] of the SYS_SW register at address `0x1C010004`.

The switches are in the off position by default.

Note

ARM recommends that you configure the Boot DIP switches using the `boot_switch` model parameter instead of using the CLCD interface. Changing Boot DIP switch positions while the model is running can result in unpredictable behavior.

S6LED

Eight colored boxes indicate the state of the VE User LEDs.

These represent LEDs D[21:14] on the VE hardware, which are mapped to bits [7:0] of the SYS_LED register at address `0x1C010008`. The boxes correspond to the red/yellow/green LEDs on the VE hardware.

Total Instr

A counter showing the total number of instructions executed.
Because the FVP models provide a *Programmer's View* (PV) of the system, the CLCD displays total instructions rather than total processor cycles. Timing might differ substantially from the hardware because:

- Bus fabric is simplified.
- Memory latencies are minimized.
- Cycle approximate processor and peripheral models are used.

In general, bus transaction timing is consistent with the hardware, but the timing of operations within the model is not accurate.

Total Time

A counter showing the total elapsed time, in seconds.

This time is wall clock time, not simulated time.

Rate Limit

A feature that disables or enables fast simulation.

Because the system model is highly optimized, your code might run faster than it would on real hardware. This effect might cause timing issues.

Rate Limit is enabled by default. Simulation time is restricted so that it more closely matches real time.

To disable or enable Rate Limit, click the square button. When you disable Rate Limit, the text changes from ON to OFF and the colored box becomes darker. You can configure this option when instantiating the model with the `rate_limit-enable` visualization component parameter.

When you click the **Total Instr** or **Total Time** items in the CLCD, the display changes to show **Instr/sec** (instructions per second) and **Perf Index** (performance index).

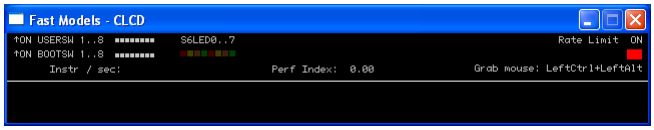

Figure 2-2 CLCD window with Rate Limit ON, showing Instr/sec and Perf Index

You can click the items again to toggle between the original and alternative displays.

Instr/sec

The number of instructions that execute per second of wall clock time.

Perf Index

The ratio of real time to simulation time. The larger the ratio, the faster the simulation runs. If you enable the Rate Limit feature, the Perf Index approaches unity.

You can reset the simulation counters by resetting the model.

The VE FVP CLCD displays the core run state for each core with a colored icon. The icons are to the left of the **Total Instr** (or **Inst/sec**) item. They appear when you start the simulation.

Figure 2-3 Core run state icons for a quad core model

Table 2-1 Core run state icon descriptions

Icon	State label	Description
	UNKNOWN	Run status unknown, that is, simulation has not started.
	RUNNING	Core running, is not idle, and is executing instructions.
	HALTED	External halt signal asserted.
	STANDBY_WFE	Last instruction executed was WFE and standby mode has been entered.
	STANDBY_WFI	Last instruction executed was WFI and standby mode has been entered.
	IN_RESET	External reset signal asserted.
	DORMANT	Partial core power down.
	SHUTDOWN	Complete core power down.

If the CLCD window has focus:

- Any keyboard input is translated to PS/2 keyboard data.
- Any mouse activity over the window is translated into PS/2 relative mouse motion data. The data is then streamed to the KMI peripheral model FIFOs.

Note

The simulator only sends relative mouse motion events to the model. As a result, the host mouse pointer does not necessarily align with the target OS mouse pointer.

You can hide the host mouse pointer by pressing the **left Ctrl**+**left Alt** keys. Press the keys again to redisplay the host mouse pointer. Only the **left Ctrl** key is operational. The **right Ctrl** key does not have the same effect.

If you prefer to use a different key, configure it with the `trap_key` visualization component parameter.

Related references

5.5.6 VE - differences in timing on page 5-78.
VEVisualisation - parameters on page 5-65.

2.3 Ethernet with VE FVPs

This section describes how to use Ethernet with VE FVPs.

Using Ethernet with VE FVPs

The VE FVPs have a virtual Ethernet component. This component is a model of the SMSC 91C111 Ethernet controller, and uses a TAP device to communicate with the network. By default, the Ethernet component is disabled.

Host requirements

Before you can use the Ethernet capability of VE FVPs, set up your host computer.

Target requirements

This section describes the target requirements.

Target requirements - about

The VE FVPs include a software implementation of the SMSC 91C111 Ethernet controller. Your target OS must therefore include a driver for this specific device. To use the SMSC chip, configure the kernel. Linux supports the SMSC 91C111.
The configurable SMSC 91C111 component parameters are:

- `enabled`.
- `mac_address`.
- `promiscuous`.

enabled

When the device is disabled, the kernel cannot detect the device.

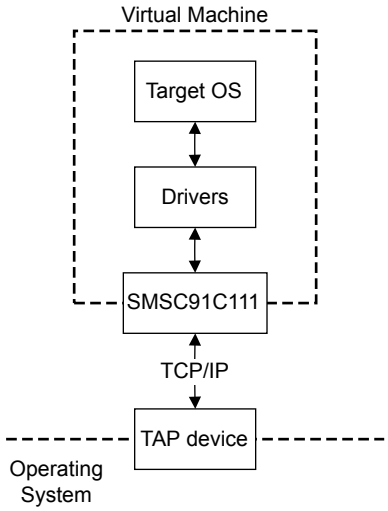

Figure 2-4 Model networking structure block diagram

To perform read and write operations on the TAP device, configure a HostBridge component. The HostBridge component is a virtual *Programmer's View* (PV) model. It acts as a networking gateway to exchange Ethernet packets with the TAP device on the host, and to forward packets to NIC models.

mac_address

There are two options for the `mac_address` parameter.

If a MAC address is not specified, when the simulator is run it takes the default MAC address, which is randomly generated. This random generation provides some degree of MAC address uniqueness when running models on multiple hosts on a local network.

promiscuous

The Ethernet component starts in promiscuous mode by default. In this mode, it receives all network traffic, even any not addressed to the device. Use this mode if you are using a single network device for multiple MAC addresses. Use this mode if, for example, you share the network card between your host OS and the VE FVP Ethernet component.

By default, the Ethernet device on the VE FVP has a randomly generated MAC address and starts in promiscuous mode.

2.4 Use a terminal with a system model

This section describes how to use a terminal with a system model.

Using a terminal with a system model

The Terminal component is a virtual component that enables UART data to be transferred between a TCP/IP socket on the host and a serial port on the target.

Note

To use the Terminal component with a Microsoft Windows 7 client, you must first install Telnet. The Telnet application is not installed on Microsoft Windows 7 by default.
Download the application by following the instructions on the Microsoft web site. Search for "Windows 7 Telnet" to find the Telnet FAQ page. To install Telnet:

1. Select **Start** > **Control Panel** > **Programs and Features**. This opens a window that enables you to uninstall or change programs.
2. Select **Turn Windows features on or off** on the left side of the bar. This opens the Microsoft Windows Features dialog. Select the **Telnet Client** check box.
3. Click **OK**. The installation of Telnet might take several minutes to complete.

The following figure shows a block diagram of one possible relationship between the target and host through the Terminal component. The TelnetTerminal block is what you configure when you define Terminal component parameters. The Virtual Machine is your FVP.

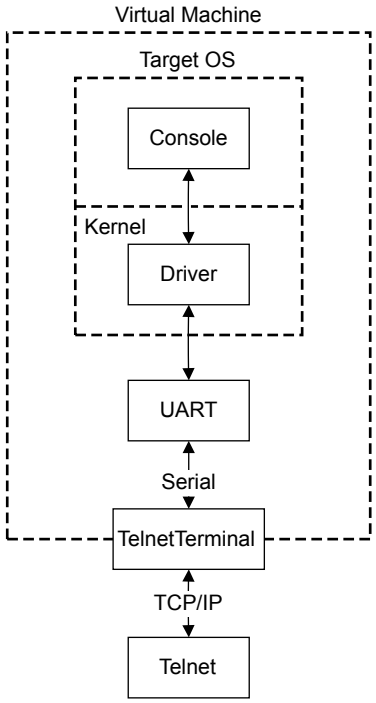

Figure 2-5 Terminal block diagram

On the target side, the console process invoked by your target OS relies on a suitable driver being present. Such drivers are normally part of the OS kernel. The driver passes serial data through a UART. The data is forwarded to the TelnetTerminal component, which exposes a TCP/IP port to the world outside of the FVP. This port can be connected to by, for example, a Telnet process on the host.

By default, the FVP starts four telnet Terminals when the model is initialized. You can change the startup behavior for each of the four Terminals by modifying the corresponding component parameters.

If the Terminal connection is broken, for example by closing a client telnet session, the port is re-opened on the host. This might have a different port number if the original one is no longer available. Before the first data access, you can connect a client of your choice to the network socket. If there is no existing connection when the first data access is made, and the `start_telnet` parameter is `true`, a host telnet session is started automatically.

The port number of a particular Terminal instance can be defined when the FVP starts. The actual value of the port used by each Terminal is declared when it starts or restarts, and might not be the value you specified if the port is already in use. If you are using Model Shell, the port numbers are displayed in the host window in which you started the model.

You can start the Terminal component in either telnet mode or raw mode.

Telnet mode

In telnet mode, the Terminal component supports a subset of the RFC 854 protocol. This means that the Terminal participates in negotiations between the host and client concerning what is and is not supported, but flow control is not implemented.

Raw mode

Raw mode enables the byte stream to pass unmodified between the host and the target. This means that the Terminal component does not participate in initial capability negotiations between the host and client. It acts as a TCP/IP port. You can use this feature to directly connect to your target through the Terminal component.

2.5 Virtual File System

The *Virtual File System* (VFS) enables your target to access parts of a host filesystem, through a target OS-specific driver and a memory-mapped device called the MessageBox. Access to the host filesystem is analogous to access to a shared network drive.

The VFS supports these filesystem operations:

getattr
Retrieve metadata for the file, directory or symbolic link.

mkdir
Create a new directory.

remove
Remove a file, directory or symbolic link.

rename
Rename a file, directory or symbolic link.

rmdir
Remove an empty directory.

setattr
Set metadata for the file, directory or symbolic link.

——— **Note** ———

The VFS does not implement `setattr`.

———————————

The VFS does not support symbolic links. The model cannot create hard links but those that the host operating system creates do function correctly.

The VFS supports these mount points:

closemounts
Free the iterator handle returned from `openmounts`.

openmounts
Retrieve an iterator handle for the list of available mounts.

readmounts
Read one entry from the mount iterator ID.

The VFS supports the following directory iterators:

closedir
Free a directory iterator handle retrieved by `opendir`.

opendir
Retrieve an iterator handle for the directory specified.

readdir
Read the next entry from the directory iterator.

——— **Note** ———

Datestamps returned are in milliseconds elapsed since the VFS epoch of January 01 1970 00:00 UTC and are host datestamps. The host datestamp might be in the future relative to the simulated OS datestamp.

———————————

The VFS supports these file operations:

closefile
Free a handle opened with `openfile`.

filesync
Force the host OS to flush all file data to persistent storage.

getfilesize
Return the size of a file, in bytes.

openfile
Return a handle to the file specified.

readfile
Read a block of data from a file.

setfilesize
Set the size of a file in bytes, either by truncating, or extending the file with zeroes.

writefile
Write a block of data to a file.

Chapter 3
Programming Reference for Base FVPs

This chapter describes the memory map and the parameters for the peripheral and system component models.

It contains the following sections:

3.1 Base - about

The Base Platform system model allows early development, distribution, and demonstration of software deliverables.

The standard peripheral set enables software development and porting. The platform is an evolution of the VE *Fixed Virtual Platforms* (FVPs), based on the ARM *Versatile™ Express* (VE) hardware development platform.

The Base Platform system model provides:

- Two configurable clusters of up to four core models that implement:
  - AArch64 at all exception levels.
  - Configurable AArch32 support at all exception levels.
  - Configurable support for little and big endian at all exception levels.
  - Generic timers.
  - Self-hosted debug.
  - CADI debug.
  - GICv3 memory-mapped processor interfaces and distributor.
- Peripherals for multimedia or networking environments.
- Four PL011 UARTs.
- A CoreLink™ CCI-400 Cache Coherent Interconnect.
- Architectural GICv3 model.
- High Definition LCD Display Controller, 1920×1080 resolution at 60fps, with single I2S and four stereo channels.
- 64MB NOR flash and board peripherals.
- CoreLink TZC-400 TrustZone® Address Space Controller.

ARM supplies these Base FVPs:

- FVP_Base_AEMv8A-AEMv8A.
- FVP_Base_AEMv8A-AEMv8A-CCN504.
- FVP_Base_AEMv8A-AEMv8A-MMU500+DMA330.
- FVP_Base_AEMv8A-AEMv8A-MMU500+DMA330-CCI500.
- FVP_Base_AEMv8A-AEMv8A-MMU500+DMA330-CCI550.
- FVP_Base_AEMv8A-AEMv8A-AEMv8A-AEMv8A-CCN502.
- FVP_Base_AEMv8A-AEMv8A-AEMv8A-AEMv8A-CCN512.
- FVP_Base_Cortex-A32x1, 2, 4.
- FVP_Base_Cortex-A35x1, 2, 4.
- FVP_Base_Cortex-A53x1, 2, 4.
- FVP_Base_Cortex-A57x1, 2, 4.
- FVP_Base_Cortex-A57x1, 2, 4-A32x1, 4, 4.
- FVP_Base_Cortex-A57x1, 2, 4-A35x1, 4, 4.
- FVP_Base_Cortex-A57x1, 2, 4-A53x1, 4, 4.
- FVP_Base_Cortex-A72x1, 2, 4.
- FVP_Base_Cortex-A72x1, 2, 4-A53x1, 4, 4.
- FVP_Base_Cortex-A72x4-A53x4-CCI500.
- FVP_Base_Cortex-A73x1, 2, 4.
- FVP_Base_Cortex-A73x1, 2, 4-A53x1, 4, 4.
- FVP_Base_Cortex-A73x4-A53x4-CCI500.

To run FVP_Base_AEMv8A-AEMv8A-MMU500+DMA330 or FVP_Base_AEMv8A-AEMv8A-MMU500+DMA330-CCI500, reconfigure the DMAC or SMMU ranges. The default ranges for these components overlap.

3.2 Base - memory

This section describes the memory of the Base Platform.

This section contains the following subsections:

3.2.1 Base - secure memory

Enable the access permissions with the `bp.secure_memory` parameter.

Table 3-1 Secure and Non-secure access permissions

Security	`bp.secure_memory = false`	`bp.secure_memory = true`
S	Secure and Non-secure access permitted.	Secure access is permitted, Non-secure access aborts.
S/NS	Secure and Non-secure access permitted.	Secure and Non-secure access are permitted.
P	Secure and Non-secure access permitted.	Access conditions are programmable by the TZC-400.

Note

The default state of the TZC-400 is to abort all accesses, even from Secure state.

Table 3-2 NSAIDs and filters that masters present to the TZC-400

Component	NSAID[a]	Filter
Cluster 0	9	0
Cluster 1	9	0
VirtIO	8	0
HDLCD0	2	2
CLCD	1	2

3.2.2 Base - memory map

The basis of this map is the Versatile Express RS2 memory map with extensions.

Table 3-3 Base Platform memory map

Peripheral	Start address	Size	End address	Security
Trusted Boot ROM, secure flash, IntelStrataFlashJ3	`0x00_0000_0000`	64MB	`0x00_03FF_FFFF`	S
Trusted SRAM	`0x00_0400_0000`	256KB	`0x00_0403_FFFF`	S
Trusted DRAM	`0x00_0600_0000`	32MB	`0x00_07FF_FFFF`	S
NOR flash, flash0, IntelStrataFlashJ3	`0x00_0800_0000`	64MB	`0x00_0BFF_FFFF`	S/NS
NOR flash, flash1, IntelStrataFlashJ3	`0x00_0C00_0000`	64MB	`0x00_0FFF_FFFF`	S/NS
PSRAM[b]	`0x00_1400_0000`	64MB	`0x00_17FF_FFFF`	S/NS
VRAM	`0x00_1800_0000`	32MB	`0x00_19FF_FFFF`	S/NS
Ethernet, SMSC 91C111	`0x00_1A00_0000`	16MB	`0x00_1AFF_FFFF`	S/NS

[a] *Non-Secure Access IDentity*.
[b] The device is implemented as RAM and is 8MB in size.

Table 3-3 Base Platform memory map (continued)

Peripheral	Start address	Size	End address	Security
USB, unimplemented	0x00_1B00_0000	16MB	0x00_1BFF_FFFF	S/NS
VE System Registers	0x00_1C01_0000	64KB	0x00_1C01_FFFF	S/NS
System Controller, SP810	0x00_1C02_0000	64KB	0x00_1C02_FFFF	S/NS
AACI, PL041	0x00_1C04_0000	64KB	0x00_1C04_FFFF	S/NS
MCI, PL180	0x00_1C05_0000	64KB	0x00_1C05_FFFF	S/NS
KMI - Keyboard, PL050	0x00_1C06_0000	64KB	0x00_1C06_FFFF	S/NS
KMI - Mouse, PL050	0x00_1C07_0000	64KB	0x00_1C07_FFFF	S/NS
UART0, PL011	0x00_1C09_0000	64KB	0x00_1C09_FFFF	S/NS
UART1, PL011	0x00_1C0A_0000	64KB	0x00_1C0A_FFFF	S/NS
UART2, PL011	0x00_1C0B_0000	64KB	0x00_1C0B_FFFF	S/NS
UART3, PL011	0x00_1C0C_0000	64KB	0x00_1C0C_FFFF	S/NS
VFS2	0x00_1C0D_0000	64KB	0x00_1C0D_FFFF	S/NS
Watchdog, SP805	0x00_1C0F_0000	64KB	0x00_1C0F_FFFF	S/NS
Base Platform Power Controller	0x00_1C10_0000	64KB	0x00_1C10_FFFF	S/NS
Dual-Timer 0, SP804	0x00_1C11_0000	64KB	0x00_1C11_FFFF	S/NS
Dual-Timer 1, SP804	0x00_1C12_0000	64KB	0x00_1C12_FFFF	S/NS
Virtio block device	0x00_1C13_0000	64KB	0x00_1C13_FFFF	S/NS
Real-time Clock, PL031	0x00_1C17_0000	64KB	0x00_1C17_FFFF	S/NS
CF Card, unimplemented	0x00_1C1A_0000	64KB	0x00_1C1A_FFFF	S/NS
Color LCD Controller, PL111	0x00_1C1F_0000	64KB	0x00_1C1F_FFFF	S/NS
Non-trusted ROM, nontrustedrom	0x00_1F00_0000	4KB	0x00_1F00_0FFF	S/NS
CoreSight and peripherals	0x00_2000_0000	128MB	0x00_27FF_FFFF	S/NS
REFCLK CNTControl, Generic Timer	0x00_2A43_0000	64KB	0x00_2A43_FFFF	S
EL2 Generic Watchdog Control	0x00_2A44_0000	64KB	0x00_2A44_FFFF	S/NS
EL2 Generic Watchdog Refresh	0x00_2A45_0000	64KB	0x00_2A45_FFFF	S/NS
Trusted Watchdog, SP805	0x00_2A49_0000	64KB	0x00_2A49_FFFF	S
TrustZone Address Space Controller, TZC-400	0x00_2A4A_0000	64KB	0x00_2A4A_FFFF	S
REFCLK CNTRead, Generic Timer	0x00_2A80_0000	64KB	0x00_2A80_FFFF	S/NS
AP_REFCLK CNTCTL, Generic Timer	0x00_2A81_0000	64KB	0x00_2A81_FFFF	S
AP_REFCLK CNTBase0, Generic Timer	0x00_2A82_0000	64KB	0x00_2A82_FFFF	S
AP_REFCLK CNTBase1, Generic Timer	0x00_2A83_0000	64KB	0x00_2A83_FFFF	S/NS
DMC-400 CFG, unimplemented	0x00_2B0A_0000	64KB	0x00_2B0A_FFFF	S/NS
GIC Physical CPU interface, GICC[c]	0x00_2C00_0000	8KB	0x00_2C00_1FFF	S/NS
GIC Virtual Interface Control, GICH[c]	0x00_2C01_0000	4KB	0x00_2C01_0FFF	S/NS

[c] You can configure the address of this region using parameters to the model. See *GICv3IRI - parameters*.

Table 3-3 Base Platform memory map (continued)

Peripheral	Start address	Size	End address	Security
GIC Virtual CPU Interface, GICV[c]	0x00_2C02_F000	8KB	0x00_2C03_0FFF	S/NS
CCI-400	0x00_2C09_0000	64KB	0x00_2C09_FFFF	S/NS
Non-trusted SRAM	0x00_2E00_0000	64KB	0x00_2E00_FFFF	S/NS
GICv3 IRI GICD[c]	0x00_2F00_0000	64KB	0x00_2F00_FFFF	S/NS
GICv3 IRI GITS[c]	0x00_2F02_0000	128KB	0x00_2F03_FFFF	S/NS
GICv3 IRI GICR[c]	0x00_2F10_0000	1MB	0x00_2F1F_FFFF	S/NS
Trusted Random Number Generator	0x00_7FE6_0000	4KB	0x00_7FE6_0FFF	S
Trusted Non-volatile counters	0x00_7FE7_0000	4KB	0x00_7FE7_0FFF	S
Trusted Root-Key Storage	0x00_7FE8_0000	4KB	0x00_7FE8_0FFF	S
DDR3 PHY, unimplemented	0x00_7FEF_0000	64KB	0x00_7FEF_FFFF	S/NS
HD LCD Controller, PL370	0x00_7FF6_0000	64KB	0x00_7FF6_FFFF	S/NS
DRAM, 0GB-2GB	0x00_8000_0000	2GB	0x00_FFFF_FFFF	P
DRAM, 2GB-32GB	0x08_8000_0000	30GB	0x0F_FFFF_FFFF	P
DRAM, 32GB-512GB	0x88_0000_0000	480GB	0xFF_FFFF_FFFF	P

3.2.3 Base - DRAM

The multiple DRAM regions do not alias each other and form a contiguous 512GB area. The total amount of DRAM on the Base Platform system model is configurable. This ability affects where usable DRAM appears.

If the Base Platform system model has `bp.dram_size=4`, the default, then 2GB of DRAM is accessible at `0x00_8000_0000` to `0x00_FFFF_FFFF`, and the remaining 2GB is accessible at `0x08_8000_0000` to `0x08_FFFF_FFFF`.

If, instead, the Base Platform system model has `bp.dram_size=8`, then 2GB of DRAM is accessible at `0x00_8000_0000` to `0x00_FFFF_FFFF` and the remaining 6GB is accessible at `0x08_8000_0000` to `0x09_FFFF_FFFF`.

The default contents of RAM not otherwise written by the simulation is a repeating sequence of the following 64-bit value: `0xCFDFDFDFDFDFDFCF`.

3.3 Base - interrupt assignments

The platform assigns the *Shared Peripheral Interrupts* (SPIs) and *Private Peripheral Interrupts* (PPIs) on the GIC.

Note

SPI and PPI numbers are mapped onto GIC interrupt IDs as the *ARM Generic Interrupt Controller Specification* describes.

Table 3-4 SPI GIC assignments

IRQ ID	SPI offset	Device
32	0	Watchdog, SP805
34	2	Dual-Timer 0, SP804
35	3	Dual-Timer 1, SP804
36	4	Real-time Clock, PL031
37	5	UART0, PL011
38	6	UART1, PL011
39	7	UART2, PL011
40	8	UART3, PL011
41	9	MCI, PL180, MCIINTR0
42	10	MCI, PL180, MCIINTR1
43	11	AACI, PL041
44	12	KMI - Keyboard, PL050
45	13	KMI - Mouse, PL050
46	14	Color LCD Controller, PL111
47	15	Ethernet, SMSC 91C111
56	24	Trusted Watchdog, SP085
57	25	AP_REFCLK, Generic Timer, CNTPSIRQ
58	26	AP_REFCLK, Generic Timer, CNTPSIRQ1
59	27	EL2 Generic Watchdog WS0
60	28	EL2 Generic Watchdog WS1
73	41	VFS2
74	42	Virtio block device
80	48	TZC-400 interrupt
92	60	cluster0.cpu0 PMUIRQ
93	61	cluster0.cpu1 PMUIRQ
94	62	cluster0.cpu2 PMUIRQ
95	63	cluster0.cpu3 PMUIRQ

Table 3-4 SPI GIC assignments (continued)

IRQ ID	SPI offset	Device
96	64	cluster1.cpu0 PMUIRQ
97	65	cluster1.cpu1 PMUIRQ
98	66	cluster1.cpu2 PMUIRQ
99	67	cluster1.cpu3 PMUIRQ
117	85	HD LCD Controller, PL370
139	107	Trusted Random Number Generator

Table 3-5 PPI GIC assignments

IRQ ID	PPI offset	Device
22	6	DCC, comms channel, interrupt
23	7	PMU, performance counter, overflow
24	8	CTI, Cross Trigger Interface, interrupt
25	9	Virtual CPU interface maintenance interrupt
26	10	Hypervisor timer interrupt
27	11	Virtual timer interrupt
29	13	Secure physical timer interrupt
30	14	Non-secure physical timer interrupt

3.4 Base - clocks

This section describes the clock frequencies of the Base Platform peripherals.

Table 3-6 Peripheral clock frequencies in the Base Platform

Device	Clock
Clusters	100MHz
REFCLK CNTControl, Generic Timer	100MHz
AP_REFCLK CNTCTL, Generic Timer	100MHz
Dual-Timer 0-1, SP804	35MHz
VE system registers	24MHz
UART 0-3, PL011	24MHz
KMI 0-1, PL050	24MHz
MCI, PL180	24MHz
AACI, PL041	24MHz
Ethernet, SMSC 91C111	24MHz
Watchdog, SP805	24MHz
Color LCD Controller, PL111	23.75MHz
HD LCD Controller, PL370	10MHz
Trusted Watchdog, SP805	32.768kHz
Real-time Clock, PL031	1Hz

3.5 Base - parameters

This section describes the parameters.

Table 3-7 Base Platform parameters

Parameter	Type	Allowed values	Default value	Description
`bp.dram_size`	`int`	`0x2-0x8`	`0x4`	Size of main memory in gigabytes: 2, 4, or 8.
`bp.proc_id`*n*[d]	`uint32_t`	`-`	-[e]	Processor ID for VE_SysRegs SYS_PROCID*n*.
`bp.secure_memory`	`bool`	`true`, `false`	`true`	The security state of the processor limits access to peripherals and RAM.
`bp.variant`[d]	`uint32_t`	`0x0-0xF`	-[e]	Board variant for VE_SysRegs SYS_ID.
`cache_state_modelled`	`bool`	`true`, `false`	`true`	Enable d-cache and i-cache state for all components.

Table 3-8 Base Platform debug parameters

Parameter	Type	Allowed values	Default value	Description
`dbgen`	`bool`	`true`, `false`	`true`	Debug authentication signal, **dbgen**.
`niden`	`bool`	`true`, `false`	`true`	Debug authentication signal, **niden**.
`spiden`	`bool`	`true`, `false`	`true`	Debug authentication signal, **spiden**.
`spniden`	`bool`	`true`, `false`	`true`	Debug authentication signal, **spniden**.

[d] Some platforms do not expose this parameter.
[e] Platform specific.

3.6 Base - components

This section describes the components.

This section contains the following subsections:

3.6.1 Base - components - about

These component models implement some of the functionality of the *Versatile Express* (VE) hardware.

A complete model implementation of a Base Platform system model includes both Base Platform-specific components and generic components such as buses and timers.

3.6.2 Base - Base_PowerController component

This section describes the Base_PowerController component.

Base_PowerController - control interface

The Base_PowerController provides a basic register interface for software to control the power-up and power-down of cores in the cluster.

Identify cores in the system to the Base_PowerController by writing 24 bits in MPIDR format, providing the following levels of affinity:

Bits [23:16]
Affinity level 2.

Bits [15:8]
Affinity level 1.

Bits [7:0]
Affinity level 0.

Examples of affinity usage are `not_applicable`/`cluster`/`processor` and `cluster`/`processor`/`thread`.

Base_PowerController - parameters

This section describes the parameters.

Table 3-9 Base_PowerController parameters

Parameter	Allowed values	Default value	Description
`startup`	-	'0.0.0.*'	A comma-separated list of cores to power up at startup or system reset. Specify core affinities with a dotted-quad ('0.0.0.0' refers to cluster0.cpu0 and '0.0.1.1' refers to cluster1.cpu1). Use wildcards to indicate all cores at an affinity level ('0.0.0.*' indicates all cores in cluster 0 and is equivalent to '0.0.0.0,0.0.0.1,0.0.0.2,…,0.0.0.255').

Base_PowerController - registers

This section describes the registers.

Register summary

This section describes the power control registers in order of offset from the base memory address.

Table 3-10 Base_PowerController register summary

Offset	Name	Type	Reset	Width	Description
0x00	PPOFFR	RW	0x---	32	Power Control Processor Off Register
0x04	PPONR	RW	0x---	32	Power Control Processor On Register
0x08	PCOFFR	RW	0x---	32	Power Control Cluster Off Register
0x0C	PWKUPR	RW	0x---	32	Power Control Wakeup Register
0x10	PSYSR	RW	0x---	32	Power Control SYS Status Register

PPOFFR

The *Power Control Processor Off Register* (PPOFFR) characteristics are: purpose, usage constraints, configurations, and attributes.

Purpose

Processor SUSPEND command when PWKUPR and the GIC are programmed appropriately to provide wakeup events from IRQ and FIQ events to that processor.

Usage constraints

Processor must make power-off requests only for itself.

Configurations

Available in all configurations.

Attributes

See the register summary table.

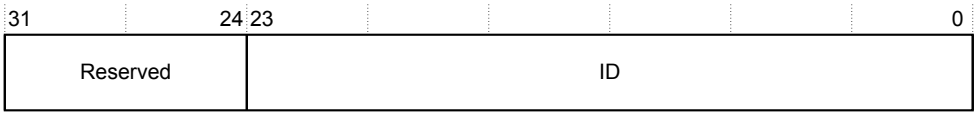

Figure 3-1 Power Control Processor Off Register bit assignments

Table 3-11 Power Control Processor Off Register bit assignments

Bits	Name	Function
[31:24]	-	Reserved.
[23:0]	ID	MPIDR format affinity value of the processor to be switched off. Programming error if MPIDR != self.

PPONR

The *Power Control Processor On Register* (PPONR) characteristics are: purpose, usage constraints, configurations, and attributes.

Purpose

Brings up a processor from low-power mode.

Usage constraints

Processor must make power-on requests only for other powered-off processors in the system.

Configurations

Available in all configurations.

Attributes

See the register summary table.

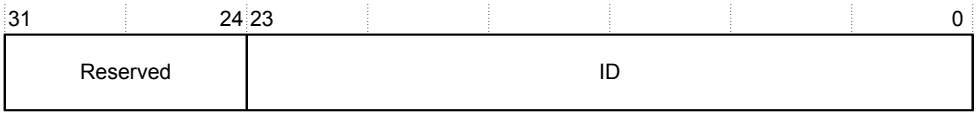

Figure 3-2 Power Control Processor On Register bit assignments

Table 3-12 Power Control Processor On Register bit assignments

Bits	Name	Function
[31:24]	-	Reserved.
[23:0]	ID	MPIDR format affinity value of the processor to be switched on. Programming error if MPIDR == self.

PCOFFR

The *Power Control Cluster Off Register* (PCOFFR) characteristics are: purpose, usage constraints, configurations, and attributes.

Purpose
Turns the cluster off.

Usage constraints
Cluster must make power-off requests only for itself.

Configurations
Available in all configurations.

Attributes
See the register summary table.

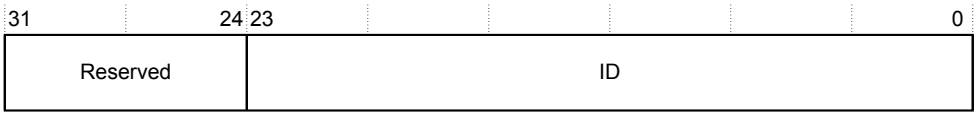

Figure 3-3 Power Control Cluster Off Register bit assignments

Table 3-13 Power Control Cluster Off Register bit assignments

Bits	Name	Function
[31:24]	-	Reserved.
[23:0]	ID	MPIDR format affinity value of powered-on processor in the cluster to be switched off. Programming error if MPIDR != self.

PWKUPR

The *Power Control Wakeup Register* (PWKUPR) characteristics are: purpose, usage constraints, configurations, and attributes.

Purpose
Configures whether wakeup requests from the GIC are enabled for this cluster.

Usage constraints
There are no usage constraints.

Configurations
Available in all configurations.

Attributes
See the register summary table.

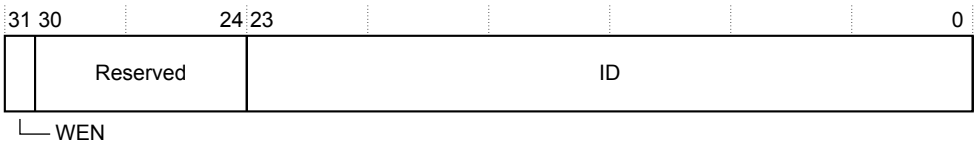

Figure 3-4 Power Control Wakeup Register bit assignments

Table 3-14 Power Control Wakeup Register bit assignments

Bits	Name	Function
[31]	WEN	If set, enables wakeup interrupts (return from SUSPEND) for this cluster.
[30:24]	-	Reserved.
[23:0]	ID	MPIDR format affinity value of processor whose Wakeup Enable bit is to be configured.

PSYSR

The *Power Control SYS Status Register* (PSYSR) characteristics are: purpose, usage constraints, configurations, and attributes.

Purpose

Provides information on the powered status of a given core. Software writes bits [23:0] for the required core and reads the value along with the associated status in bits [31:24].

Usage constraints

There are no usage constraints.

Configurations

Available in all configurations.

Attributes

See the register summary table.

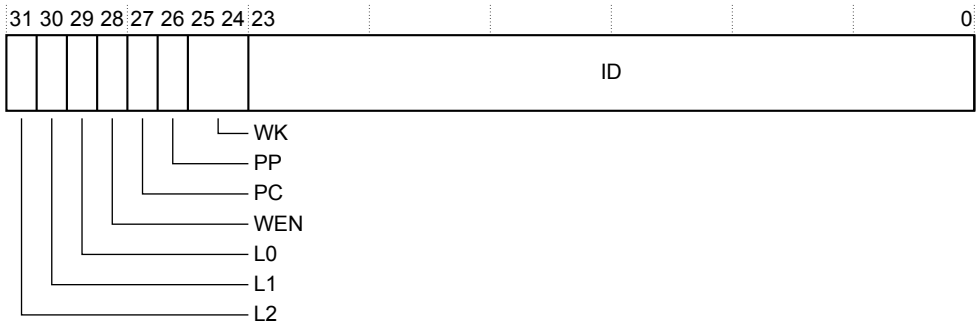

Figure 3-5 Power Control SYS Status Register bit assignments

Table 3-15 Power Control SYS Status Register bit assignments

Bits	Name	Function
[31]	L2	Read-only. A value of 1 indicates that affinity level 2 is active/on. If affinity level 2 is not implemented this bit is RAZ.
[30]	L1	Read-only. A value of 1 indicates that affinity level 1 is active/on. If affinity level 1 is not implemented this bit is RAZ.
[29]	L0	Read-only. A value of 1 indicates that affinity level 0 is active/on.
[28]	WEN	Read-only. A value of 1 indicates wakeup interrupts, return from SUSPEND, enabled for this processor. This is an alias of PWKUPR.WEN for this core.
[27]	PC	Read-only. A value of 1 indicates pending cluster off, the cluster enters low-power mode the next time it raises signal STANDBYWFIL2.

Table 3-15 Power Control SYS Status Register bit assignments (continued)

Bits	Name	Function
[26]	PP	Read-only. A value of 1 indicates pending processor off, the processor enters low-power mode the next time it raises signal STANDBYWFI.

[25:24]	WK	Read-only.
		Indicates the reason for LEVEL0 power on:
		`0b00`
		Cold power-on.
		`0b01`
		System reset pin.
		`0b10`
		Wake by PPONR.
		`0b11`
		Wake by GIC WakeRequest signal.

[23:0]	ID	MPIDR format affinity value.

3.6.3 Base - DebugAccessPort component

This section describes the DebugAccessPort component, a model of the *Debug Access Port* (DAP) for external debug connections.

DebugAccessPort - ports

This section describes the ports.

Table 3-16 DebugAccessPort ports

Name	Protocol	Type	Description
`ap_pvbus_m[2]`	PVBus	Master	Debug-access port to bus master channels 0 and 1.
`clock`	ClockSignal	Slave	Clock input.
`paddrdbg31`	Signal	Master	Output signal that indicates which master the access came from, AP0 or AP1. Configurable.

DebugAccessPort - parameters

This section describes the parameters.

Table 3-17 Base Platform DebugAccessPort parameters

Name	Type	Allowed values	Default value	Description
`ap0_rom_base_address`	`uint64_t`	`0x0-0xffffffffffffffff`	`0x0`	ROM base address for AP0.
`ap1_rom_base_address`	`uint64_t`	`0x0-0xffffffffffffffff`	`0x0`	ROM base address for AP1.
`ap0_has_debug_rom`	`bool`	`true`, `false`	`false`	AP0 has a Debug ROM.
`ap1_has_debug_rom`	`bool`	`true`, `false`	`false`	AP1 has a Debug ROM.
`ap0_set_paddrdbg31`	`bool`	`true`, `false`	`false`	Set `paddrdbg31` signal during accesses on AP0.
`ap1_set_paddrdbg31`	`bool`	`true`, `false`	`false`	Set `paddrdbg31` signal during accesses on AP1.

3.6.4 Base - simulator visualization component

This section describes the simulator visualization component.

Simulator visualization - parameters

This section describes the parameters.

Table 3-18 Simulator visualization parameters

Parameter	Allowed values	Default value	Description
`cluster0_name`	-	"Cluster0"	Cluster0 name.
`cluster1_name`	-	"Cluster1"	Cluster1 name.
`cpu_name`	-	""	Window title displays core name.
`disable_visualisation`	`true`, `false`	`false`	Enables or disables visualization.
`rate_limit-enable`	`true`, `false`	`true`	Rate limit simulation.
`recorder.checkInstructionCount`	`true`, `false`	`true`	Checks instruction count in recording file against actual instruction count during playback.
`recorder.playbackFileName`	-	""	Playback filename. An empty string disables playback.
`recorder.recordingFileName`	-	""	Recording filename. An empty string disables recording.
`recorder.recordingTimeBase`	-	`0x5F5E100`	Timebase in 1/s relative to the master clock. For example, 100 000 000 means 10 nanoseconds resolution simulated time for a 1Hz master clock. Used for recording. Higher values give a higher time resolution. Playback time base is always taken from the playback file.
`recorder.verbose`	-	`0x0`	Enables verbose messages. `0x0`, no messages. `0x1`, normal. `0x2`, more detail.
`trap_key`	-	`0x4a`	Trap key that works with left **`Ctrl`** to toggle mouse display.
`window_title`	-	"Fast Models - CLCD %cpu%"	`cpu_name` replaces window title, `%cpu%`.

3.6.5 Base - VE_SysRegs component

This section describes the VE system registers component.

VE_SysRegs - parameters

This section describes the parameters.

Table 3-19 Base Platform VE_SysRegs parameters

Name	Type	Allowed values	Default value	Description
exit_on_shutdown	bool	true, false	false	true: if software uses the SYS_CFGCTRL function SYS_CFG_SHUTDOWN, then the simulator shuts down and exits.[f]
mmbSiteDefault	uint8_t	0x0-0x2	0x1	Default MMB source. 0x0, MB. 0x1, DB1. 0x2, DB2.
tilePresent	bool	true, false	true	Tile fitted.
user_switches_value	uint32_t	-	0x0	User switches.

VE_SysRegs - registers

This section describes the configuration registers.

Table 3-20 Base Platform VE_SysRegs registers

Name	Offset	Access	Description
SYS_ID	0x00	Read/write	System identity.
SYS_SW	0x04	Read/write	Bits[7:0] map to switch S6.
SYS_LED	0x08	Read/write	Bits[7:0] map to user LEDs.
SYS_100HZ	0x24	Read only	100Hz counter.
SYS_FLAGS	0x30	Read/write	General-purpose flags.
SYS_FLAGSCLR	0x34	Write only	Clear bits in general-purpose flags.
SYS_NVFLAGS	0x38	Read/write	General-purpose non-volatile flags.
SYS_NVFLAGSCLR	0x3C	Write only	Clear bits in general-purpose non-volatile flags.
SYS_MCI	0x48	Read only	MCI.
SYS_FLASH	0x4C	Read/write	Flash control.
SYS_CFGSW	0x58	Read/write	Boot-select switch.
SYS_24MHZ	0x5C	Read only	24MHz counter.
SYS_MISC	0x60	Read/write	Miscellaneous control flags.
SYS_DMA	0x64	Read/write	DMA peripheral map.
SYS_PROCID0	0x84	Read/write	Processor ID.
SYS_PROCID1	0x88	Read/write	Processor ID.
SYS_PROCID2	0x8C	Read/write	Processor ID.
SYS_PROCID3	0x90	Read/write	Processor ID.
SYS_CFGDATA	0xA0	Read/write	Data to read/write from & to motherboard controller.
SYS_CFGCTRL	0xA4	Read/write	Control data transfer to motherboard controller.
SYS_CFGSTAT	0xA8	Read/write	Status of data transfer to motherboard controller.

[f] For more information on SYS_CFGCTRL, see the *Motherboard Express µATX V2M-P1 Technical Reference Manual*.

3.7 Base - differences between the AEMv8-A FVP and core FVPs

This section describes implementation features of the core models that the AEMv8-A model does not implement, or implements with significant differences.

- The default value of `cache_state_modelled` is 0.
- Components `cluster0` and `cluster1` are implementation cores, not AEMs. All parameters in these components are the parameters of the core implementation, not the parameters of the AEM. The values of `bp.proc_id0` and `bp.proc_id1` have fixed values consistent with the cores and are not configurable.
- The core defines the memory map of register banks within the GIC region, and the map is therefore not configurable. The parameter `bp.variant` communicates the nature of the memory map to target software. The parameter has a fixed value consistent with the memory map of the core, and is not configurable. Because the core implementations contain a specific version of the GIC, the parameter `gicv3.gicv2-only` is not available. The following registers in the GIC distributor are given fixed values to match the implementation, and are not configurable: `gic_distributor.reg-base`, `gic_distributor.reg-base-per-distributor`, `gic_distributor.GICD-alias`, `gic_distributor.ITS0-base`.
- The GICv2 CPU IDs are contiguous in implementation platform models, because the number of cores in each cluster is fixed. They can be noncontiguous in the AEMv8-A Base Platform FVP because there is space for four cores in the cluster. You can configure fewer than four cores.

3.8 Base - VE compatibility

ARM expects software that ran on the previous VE model to be compatible with this system model, but you might need to apply some configuration options.

This section contains the following subsections:

3.8.1 Base - VE compatibility - GICv2

This system model uses GICv3 by default. You can configure it to support GICv2 or GICv2m.

To configure the model as GICv2m, set the following:

```
-C gicv3.gicv2-only=1                          \
-C cluster0.gic.GICD-offset=0x10000            \
-C cluster0.gic.GICC-offset=0x2F000            \
-C cluster0.gic.GICH-offset=0x4F000            \
-C cluster0.gic.GICH-other-CPU-offset=0x50000  \
-C cluster0.gic.GICV-offset=0x6F000            \
-C cluster0.gic.PERIPH-size=0x80000            \
-C cluster1.gic.GICD-offset=0x10000            \
-C cluster1.gic.GICC-offset=0x2F000            \
-C cluster1.gic.GICH-offset=0x4F000            \
-C cluster1.gic.GICH-other-CPU-offset=0x50000  \
-C cluster1.gic.GICV-offset=0x6F000            \
-C cluster1.gic.PERIPH-size=0x80000            \
-C gic_distributor.GICD-alias=0x2c010000
```

To configure the model as GICv2, set the following:

```
-C gicv3.gicv2-only=1                         \
-C cluster0.gic.GICD-offset=0x1000            \
-C cluster0.gic.GICC-offset=0x2000            \
-C cluster0.gic.GICH-offset=0x4000            \
-C cluster0.gic.GICH-other-CPU-offset=0x5000  \
-C cluster0.gic.GICV-offset=0x6000            \
-C cluster0.gic.PERIPH-size=0x8000            \
-C cluster1.gic.GICD-offset=0x1000            \
-C cluster1.gic.GICC-offset=0x2000            \
-C cluster1.gic.GICH-offset=0x4000            \
-C cluster1.gic.GICH-other-CPU-offset=0x5000  \
-C cluster1.gic.GICV-offset=0x6000            \
-C cluster1.gic.PERIPH-size=0x8000            \
-C gic_distributor.GICD-alias=0x2c010000
```

To configure MSI frames for GICv2m, parameters are available to set the base address and configuration of each of 16 possible frames. Eight frames are Secure and eight frames are Non-secure:

```
-C gic_distributor.MSI_S-frame0-base=ADDRESS  \
-C gic_distributor.MSI_S-frame0-min-SPI=NUM   \
-C gic_distributor.MSI_S-frame0-max-SPI=NUM
```

In this example, you can replace `MSI_S` with MSI_NS, for NS frames, and you can replace `frame0` with `frame1` to `frame7` for each of the possible 16 frames. If the base address is not specified for a given frame, or the SPI numbers are out of range, the corresponding frame is not instantiated.

3.8.2 Base - VE compatibility - GICv3

If a Base Platform includes an implementation of the GICv3 system registers, it is enabled by default.

The GIC distributor and CPU (core) interface have parameters that allow configuration of the model to match different implementation options. Use `--list-params` to get a full list. Configuration options for the GIC model must be available under:

- `cluster[0-`*n*`].gic.*`
- `cluster[0-`*n*`].gicv3.*`
- `gic_distributor.*`

3.8.3 Base - VE compatibility - system global counter

The Generic Timer registers of the cores do not operate by default.

The model provides a memory-mapped interface to the system global counter, and enables the free-running timer from reset. However, the architectural requirement is that such a counter is not enabled at reset. As a result, the Generic Timer registers of the cores do not operate unless either:

- Software enables the counter peripheral by writing the FCREQ[0] and EN bits in CNTCR at `0x2a43000`. ARM recommends this approach.
- The `-C bp.refcounter.non_arch_start_at_default=1` parameter is set. This approach provides compatibility with older software.

3.8.4 Base - VE compatibility - disable security

Base Platform FVPs have an enhanced security map for peripherals. By default, it restricts access to some peripherals.

Software must program the TZC-400 to make any accesses to DRAM, because the reset configuration blocks all accesses.

For backward compatibility with software that cannot program the TZC-400, this parameter setting permits all accesses regardless of security state:

```
-C bp.secure_memory=false
```

3.9 Base - unsupported VE features

This system model does not support software that relies on some features of the VE model.

This section contains the following subsections:

3.9.1 Base - unsupported VE features - memory aliasing at 0x08_00000000

The VE model permits an alias of the 2GB region of DRAM between addresses `0x80000000` and `0xFFFFFFFF` with addresses `0x08_00000000` to `0x08_7FFFFFFF`. The Base Platform does not have this alias and the region `0x08_00000000` to `0x08_7FFFFFFF` is Reserved.

3.9.2 Base - unsupported VE features - boot ROM alias at 0x00_0800_0000

In the VE model, the region at `0x00_0800_0000` was an alias of the trusted boot ROM at `0x00_0000_0000`. It is now an independent region of NOR flash.

3.9.3 Base - unsupported VE features - change of older parameters

Most parameter names have been simplified between the VE model and the Base Platform system model.

Components that were previously in *motherboard* or *daughterboard* groups are now in a *bp* group. The model does not recognize the previous parameter names.

In a change to the previous default, the Base Platform models the core cache state by default. You can disable this using a single parameter for all cores in the simulation, using the `cache_state_modelled` parameter.

```
-C cache_state_modelled=0
```

Note

Cortex Base Platforms do not model the cache state by default.

Chapter 4
Programming Reference for MPS2 FVPs

This chapter describes the model of the hardware platform.

It contains the following sections:

4.1 MPS2 - about

The *Microcontroller Prototyping System 2* (MPS2) *Fixed Virtual Platform* (FVP) model implements a subset of the functionality of the MPS2 hardware.

MPS2 model platforms include MPS2 components and generic ones, such as buses and timers.

MPS2 platforms are sufficiently accurate to boot the Keil® RTX RTOS and run the Blinky application.

To list the model parameters and their types, allowed values, default values, and descriptions, run the model, passing in the `--list-params` argument.

On Windows, the MPS2 components are in the `%PVLIB_HOME%\examples\FVP_MPS2\LISA` directory.

On Linux, the MPS2 components are in the `$PVLIB_HOME/examples/FVP_MPS2/LISA` directory.

Related information

AN400 - ARM Cortex-M7 SMM on V2M-MPS2.

4.2 MPS2 - memory maps

This section describes the MPS2 memory maps.

This section contains the following subsections:

4.2.1 MPS2 - memory map for models without the ARMv8-M additions

This section describes the MPS2 memory map for older cores, without the ARMv8-M additions.

For standard ARM peripherals, see the TRM for that device.

Note

- A bus error is generated for accesses to memory areas not shown in this table.
- Any memory device that does not occupy the total region is aliased within that region.

Table 4-1 Overview of MPS2 memory map

Description	Modeled	Address range
Ethernet[g]	Partial	`0xA0000000-0xA000FFFF`
PSRAM (16MB)	Yes	`0x60000000-0x60FFFFFF`
VGA Image (512x128) (AHB)	Yes	`0x41100000-0x4110FFFF`
VGA Console (AHB)	Yes	`0x41000000-0x4100FFFF`
Block RAM (boot time)[h]	Yes	`0x40200000-0x402FFFFF`
Reserved	N/A	`0x40030000-0x401FFFFF`
SCC register	Yes	`0x4002F000-0x4002FFFF`
Reserved	N/A	`0x40029000-0x4002EFFF`
FPGA System Control & I/O, APB	Yes	`0x40028000-0x40028FFF`
Reserved	N/A	`0x40025000-0x40027FFF`
Audio I2S, APB	Partial	`0x40024000-0x40024FFF`
SBCon (Audio Configuration), APB	Yes	`0x40023000-0x40023FFF`
SBCon (Touch for LCD module), APB	Partial	`0x40022000-0x40022FFF`
PL022 (SPI for LCD module), APB	Partial	`0x40021000-0x40021FFF`
PL022 (SPI), APB	Yes	`0x40020000-0x40020FFF`
CMSDK system controller	Yes	`0x4001F000-0x4001FFFF`
Reserved for extra GPIO & other AHB peripherals	N/A	`0x40012000-0x4001EFFF`
CMSDK AHB GPIO #1	Yes	`0x40011000-0x40011FFF`
CMSDK AHB GPIO #0	Yes	`0x40010000-0x40010FFF`
CMSDK APB subsystem	Yes	`0x40000000-0x4000FFFF`
Reserved	N/A	`0x20800000-0x20FFFFFF`

[g] Through ahb_to_extmem16. Offset `0x0-0x0FE` for CSRs, `0x100-0x1FE` for FIFO.
[h] Reserved 64KB, 16K implemented. This memory is wrapped through the region.

Table 4-1 Overview of MPS2 memory map (continued)

Description	Modeled	Address range
ZBTSRAM 2 & 3 (2x32-bit)[i]	Yes	0x20000000-0x207FFFFF
Reserved	N/A	0x01010000-0x1FFFFFFF
Reserved	N/A	0x00800000-0x00FFFFFF
ZBTSRAM 1 (64-bit)[j]	Yes	0x00400000-0x007FFFFF
ZBTSRAM 1 (64-bit)	Yes	0x00004000-0x003FFFFF
Mappable memory[k]	Yes	0x00000000-0x00003FFF

4.2.2 MPS2 - memory map for models with the ARMv8-M additions

This section describes the MPS2 memory map for newer cores, with the ARMv8-M additions.

For standard ARM peripherals, see the TRM for that device.

Note

- A bus error is generated for accesses to memory areas not shown in this table.
- Any memory device that does not occupy the total region is aliased within that region.

Table 4-2 Overview of MPS2 memory map

Description	IDAU	Modeled	Address range
ZBTSRAM 1 (4MB) in *Non-secure* (NS) world. Reserved 8MB, only 4MB implemented. VTOR initialization value to be configurable in LAC). Second half (4MB) aliased to first half (4MB).	NS	Yes	0x00000000-0x007FFFFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0x00800000-0x0FFFFFFF
ZBTSRAM 1 (4MB) in *Secure* (S) world. Reserved 8MB, only 4MB implemented. Second half (4MB) aliased to first half (4MB).	S	Yes	0x10000000-0x107FFFFF
Not used. Default expansion port: bus error.	S	N/A	0x10800000-0x1FFFFFFF
ZBTSRAM 2&3 (4MB) in NS world. Reserved 8MB, only 4MB implemented. For IoT subsystems, different cores have different memory sizes. Second half (4MB) aliased to first half (4MB).	NS	Yes	0x20000000-0x207FFFFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0x20800000-0x20FFFFFF
PSRAM (16MB)	NS	Yes	0x21000000-0x21FFFFFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0x22000000-0x23FFFFFF
MTB SRAM. Reserved 64KB, only 16KB implemented. Aliased to 0x0 for booting in RTL simulation.	NS	Yes	0x24000000-0x2400FFFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0x24010000-0x2FFFFFFF
ZBTSRAM 2&3 (4MB) in S world. Reserved 8MB, only 4MB implemented. Second half (4MB) aliased to first half (4MB).	S	Yes	0x30000000-0x307FFFFF
Not used. Default expansion port: bus error.	S	N/A	0x30800000-0x30FFFFFF

[i] Reserved 8MB, 4MB available. The two SRAM blocks are interleaved.
[j] Wrapped. Only 4MB ZBTSRAM fitted.
[k] When zbt_boot_ctrl = 0, ZBTSRAM 1 is mapped to this region. Otherwise, Remap_ctrl = 0 maps Block RAM and Remap_ctrl = 1 maps ZBTSRAM 1. The V2M-MPS2 board microcontroller controls the zbt_boot_ctrl signal. The zbt_boot_ctrl signal overrides the boot option to enable use of the ZBT RAM.

Table 4-2 Overview of MPS2 memory map (continued)

Description	IDAU	Modeled	Address range
Not used. No memory gating unit on PSRAM (16MB) path because it is shared with Ethernet control. Default expansion port: bus error.	S	N/A	0x31000000-0x31FFFFFF
Not used. Default expansion port: bus error.	S	N/A	0x30800000-0x3FFFFFFF
CMSDK APB subsystem:	NS	-	0x40000000-0x4000FFFF
Timer 0.	NS	Yes	0x40000000-0x40000FFF
Timer 1.	NS	Yes	0x40001000-0x40001FFF
Dual Timer.	NS	Yes	0x40002000-0x40002FFF
Not used.	NS	N/A	0x40003000-0x40003FFF
UART #0.	NS	Yes	0x40004000-0x40004FFF
UART #1.	NS	Yes	0x40005000-0x40005FFF
UART #2.	NS	Yes	0x40006000-0x40006FFF
Not used.	NS	N/A	0x40007000-0x40007FFF
Watchdog.	NS	Yes	0x40008000-0x40008FFF
Not used.	NS	N/A	0x40009000-0x4000F000
GPIO #0.	NS	Yes	0x40010000-0x40010FFF
GPIO #1.	NS	Yes	0x40011000-0x40011FFF
GPIO #2.	NS	Yes	0x40012000-0x40012FFF
GPIO #3.	NS	Yes	0x40013000-0x40013FFF
Default slave inside AHB peripheral subsystem. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	Yes	0x40014000-0x40017FFF
DMA Controller #0.	NS	Yes	0x40018000-0x40018FFF
DMA Controller #1.	NS	Yes	0x40019000-0x40019FFF
Default slave inside AHB peripheral subsystem. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	Yes	0x4001A000-0x40017FFF
CMSDK system controller. PMU control and remap registers unused. Only reset option (lockup reset) and rest info available.	NS	Yes	0x4001F000-0x4001FFFF
FPGA APB Subsystem (unused APB space: RAZ/WI):	NS	-	0x40020000-0x4002FFFF
PL022 (SPI).	NS	Yes	0x40020000-0x40020FFF
PL022 (SPI for LCD).	NS	Partial	0x40021000-0x40021FFF
SBCon I2C (Touch for LCD).	NS	Partial	0x40022000-0x40022FFF
SBCon I2C (Audio configuration).	NS	Yes	0x40023000-0x40023FFF
Audio I2S.	NS	Partial	0x40024000-0x40024FFF
Not used.	NS	N/A	0x40025000-0x40027FFF

Table 4-2 Overview of MPS2 memory map (continued)

Description	IDAU	Modeled	Address range
FPGA system control & I/O (LEDs, buttons...).	NS	Yes	0x40028000-0x40028FFF
Not used.	NS	N/A	0x40029000-0x4002EFFF
SCC registers.	NS	Yes	0x4002F000-0x401FFFFF
Ethernet.	NS	Partial	0x40200000-0x402FFFFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0x40300000-0x40FFFFFF
VGA console.	NS	Yes	0x41000000-0x4100FFFF
VGA image.	NS	Yes	0x41100000-0x4113FFFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0x41140000-0x4FFFFFFF
CMSDK APB subsystem:	S	-	0x50000000-0x5000FFFF
Timer 0.	S	Yes	0x50000000-0x50000FFF
Timer 1.	S	Yes	0x50001000-0x50001FFF
Dual Timer.	S	Yes	0x50002000-0x50002FFF
Not used.	S	N/A	0x50003000-0x50003FFF
UART #0.	S	Yes	0x50004000-0x50004FFF
UART #1.	S	Yes	0x50005000-0x50005FFF
UART #2.	S	Yes	0x50006000-0x50006FFF
Not used.	S	N/A	0x50007000-0x50007FFF
Watchdog.	S	Yes	0x50008000-0x50008FFF
Not used.	S	N/A	0x50009000-0x5000F000
GPIO #0	S	Yes	0x50010000-0x50010FFF
GPIO #1	S	Yes	0x50011000-0x50011FFF
GPIO #2	S	Yes	0x50012000-0x50012FFF
GPIO #3	S	Yes	0x50013000-0x50013FFF
Default slave. Default expansion port: bus error.	S	Yes	0x50014000-0x50017FFF
DMA Controller #0.	S	Yes	0x50018000-0x50018FFF
DMA Controller #1.	S	Yes	0x50019000-0x50019FFF
Default slave. Default expansion port: bus error.	S	Yes	0x5001A000-0x5001EFFF
CMSDK system controller. PMU control and remap registers unused. Only reset option (lockup reset) and rest info available.	S	Yes	0x5001F000-0x5001FFFF
FPGA APB subsystem:	S	-	0x50020000-0x5002FFFF
PL022 (SPI).	S	Yes	0x50020000-0x50020FFF
PL022 (SPI for LCD).	S	Partial	0x50021000-0x50021FFF

Table 4-2 Overview of MPS2 memory map (continued)

Description	IDAU	Modeled	Address range
SBCon I2C (touch for LCD).	S	Partial	0x50022000-0x50022FFF
SBCon I2C (audio configuration).	S	Yes	0x50023000-0x50023FFF
Audio I2S.	S	Partial	0x50024000-0x50024FFF
Not used.	S	N/A	0x50025000-0x50027FFF
FPGA system control & I/O (LEDs, buttons...).	S	Yes	0x50028000-0x50028FFF
Not used.	S	N/A	0x50029000-0x5002EFFF
SCC registers.	S	Yes	0x5002F000-0x501FFFFF
Ethernet.	S	Partial	0x50200000-0x502FFFFF
Not used. Default expansion port: bus error.	S	N/A	0x50300000-0x50FFFFFF
VGA console.	S	Yes	0x51000000-0x5100FFFF
VGA image.	S	Yes	0x51100000-0x5113FFFF
Not used. Default expansion port: bus error.	S	N/A	0x51140000-0x57FFFFFS
Secure APB subsystem:	S	-	0x58000000-0x5800FFFF
TRNG / PRNG.	S	Yes	0x58000000-0x58000FFF
Unique ID or Secure storage.	S	Yes	0x58001000-0x58006FFF
Secure Control Registers.	S	Yes	0x58007000-0x58007FFF
Flash memory gating unit configuration (mapped to AHB port for CODE region in the bus matrix, not APB).	S	Yes	0x58008000-0x58009FFF
SRAM memory gating unit configuration (mapped to AHB port for SRAM region in the bus matrix, not APB).	S	Yes	0x5800A000-0x5800DFFF
Reserved.	S	N/A	0x5800E000-0x5800EFFF
Reserved.	S	N/A	0x5800F000-0x5800FFFF
Not used. Default expansion port: bus error.	S	N/A	0x50010000-0x5FFFFFFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0x60000000-0x6FFFFFFF
Not used. Default expansion port: bus error.	S	N/A	0x70000000-0x7FFFFFFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0x800000000-0x8FFFFFFF
Not used. Default expansion port: bus error.	S	N/A	0x90000000-0x9FFFFFFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0xA0000000-0xAFFFFFFF
Not used. Default expansion port: bus error.	S	N/A	0xB0000000-0xBFFFFFFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0xC0000000-0xCFFFFFFF
Not used. Default expansion port: bus error.	S	N/A	0xD0000000-0xDFFFFFFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0xE0000000-0xEFFFFFFF

Table 4-2 Overview of MPS2 memory map (continued)

Description	IDAU	Modeled	Address range
System ROM table. Exempted from checking.	Exempt	Yes	0xF0000000-0xF0000FFF
Not used. Default expansion port: bus error.	Exempt	N/A	0xF0001000-0xF00FFFFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0xF0100000-0xF01FFFFF
MTB SFR address space.	NS	Yes	0xF0200000-0xF0200FFF
Reserved. This region is nonexecutable.	NS	N/A	0xF0210000-0xF0213FFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0xF0214000-0xFFFFFFFF

4.3 MPS2 - interrupt assignments

This section describes the interrupt assignments.

Table 4-3 Interrupt assignments

Number	Interrupt
NMI	Watchdog.
0	UART 0 receive interrupt.
1	UART 0 transmit interrupt.
2	UART 1 receive interrupt.
3	UART 1 transmit interrupt.
4	UART 2 receive interrupt.
5	UART 2 transmit interrupt.
6	GPIO 0, 2 combined interrupt.
7	GPIO 1, 3 combined interrupt.
8	Timer 0.
9	Timer 1.
10	Dual Timer.
11	SPI #1 (LCD). The LCD had shared SPI #0 and SPI #1.
12	UART overflow (0, 1, 2).
13	Ethernet.
14	Audio I2S.
15	Touch screen.
16-31	GPIO 0 individual interrupts.
32-47	GPIO 1 individual interrupts. ARMv8-M additions.
48	SPI #0. ARMv8-M addition.
49	Reserved.
50	TRNG (Secure). ARMv8-M addition.
51	Unique ID and Secure storage (Secure). ARMv8-M addition.
52	DMA controller #0.
53	DMA controller #1.
54	SecureErrorIRQ. ARMv8-M addition.[1]

[1] Detection of Non-secure access to Secure address spaces (including other bus masters). Generated by Memory Gating unit, Peripheral Gating units, bus gasket for legacy bus masters.

4.4 MPS2 - differences between models and hardware

This section describes the features of the hardware that the models do not implement, or implement with significant differences.

MPS2 implements most devices. Some peripherals have minimal implementations:

- The Ethernet module in the model is a LAN91C111. The hardware documents specify a LAN9220.
- The Audio module is RAZ/WI.
- The STMPE811 touchscreen module only reports touch positions.
- The model of the Ampire LCD module supports a subset of the graphics modes.

You can display images and text on an emulated VGA output, images on the LCD, and text on the UART.

ARMv8-M

The model implements an *Implementation Defined Attribution Unit* (IDAU) inside each core. The top-level component coordinates the IDAU implementations by passing down parameters. In contrast, the MPS2 specification [ARMv8-M MPS2 System Specification (ARM-ECM-0468897), v0.8] has a common, system level IDAU, which all cores and various devices use.

You cannot reprogram the IDAUs of the model. The model only reads the Secure Controller Register Block register, NSC_CFG. Set it using the top-level parameters, NSC_CFG_0 and NSC_CFG_1 (corresponding to bits 0 and 1 of NSC_CFG, respectively).

The model does not have the random number generator or unique ID/secure storage of the MPS2 specification.

The model does not support MTB, ETM, and TPIU. MTB RAM is absent.

In the Memory Gating Unit, the model provides a configurable block size. For performance reasons, the minimum block size in the model is 4096 bytes. Hardware and later models might allow smaller block sizes. Software must use the BLK_CFG register to determine block size.

Timing

FVPs enable software applications to run in a functionally accurate simulation. However, because of the relative balance of fast simulation speed over timing accuracy, there are situations where the models might behave unexpectedly.

If your code interacts with real world devices such as timers and keyboards, data arrives in the modeled device in real world, or wall clock, time. However, simulation time can run faster than the wall clock. So, a single key press might be interpreted as several repeated key presses, or a single mouse click might be interpreted as a double click.

To avoid this mismatch, the FVPs provide the Rate Limit feature. Enabling Rate Limit forces the model to run at wall clock time. For interactive applications, ARM recommends enabling Rate Limit. Use the Rate Limit button in the CLCD display or the `rate_limit-enable` model instantiation parameter.

Chapter 5
Programming Reference for VE FVPs

This chapter describes the memory map and the parameters for the peripheral and system component models.

It contains the following sections:

5.1 VE - about

The VE FVPs are software system models. They provide functionally accurate models for software execution.

ARM produces the *Versatile Express* (VE) hardware development platform. The Motherboard Express *µAdvanced Technology Extended* (ATX) V2M-P1 is the basis for an integrated software and hardware development system. This system is also based on the ARM® *Symmetric MultiProcessor* (SMP) system architecture.

The motherboard provides:

- Peripherals for multimedia or networking environments.
- Access to motherboard peripherals and functions through a static memory bus to simplify access from daughterboards.
- High-performance PCI-Express slots for expansion cards.
- Consistent memory maps with different processor daughterboards that simplify software development and porting.
- Automatic detection and configuration of attached CoreTile Express and LogicTile Express daughterboards.
- Automatic shutdown for over-temperature or power supply failure.
- No system power-on for unconfigurable daughterboards.
- Power sequencing of system.
- Drag and drop file updating of configuration files.
- Support of either a 12V power-supply unit or an external ATX power supply.
- Support of FPGA and processor daughterboards to provide custom peripherals, early access to processor designs, or production test chips.

The VE FVPs are software system models. They contain:

- Virtual implementations of a motherboard.
- Single daughterboards each containing an ARM processor.
- Associated interconnections.

Note

ARM bases the models on the VE platform memory map, but does not intend them to be accurate representations of a specific VE hardware revision. The VE FVPs support selected peripherals. The models are sufficiently complete and accurate to boot the same operating system images as the VE hardware.

VE FVPs provide functionally accurate models for software execution. However, the models sacrifice timing accuracy to increase simulation speed. Key deviations from actual hardware are:

- Approximate timing.
- Simplified buses.
- No implementations for processor caches and the related write buffers.

ARM supplies these VE FVPs:

- FVP_VE Cortex-A5x1, 2, 4.
- FVP_VE Cortex-A7x1, 2, 3, 4.
- FVP_VE Cortex-A9x1, 2, 4.
- FVP_VE Cortex-A12x1, 2, 3, 4.
- FVP_VE Cortex-A15x1, 2, 4.
- FVP_VE Cortex-A15x1, 4-A7x1, 4.
- FVP_VE Cortex-A15x1, 4-A7x1, 4-MMU400-DMA330
- FVP_VE Cortex-A15x1-A7x1-MMU500-DMA330
- FVP_VE Cortex-A15x1-DP500
- FVP_VE Cortex-A15x1-DP550

- FVP_VE Cortex-A17x1, 2, 3, 4.
- FVP_VE Cortex-A17x1, 4-A7x1, 4.
- FVP_VE Cortex-R4.
- FVP_VE Cortex-R5x1, 2
- FVP_VE Cortex-R7x1, 2
- FVP_VE Cortex-R8x1, 2, 3, 4.

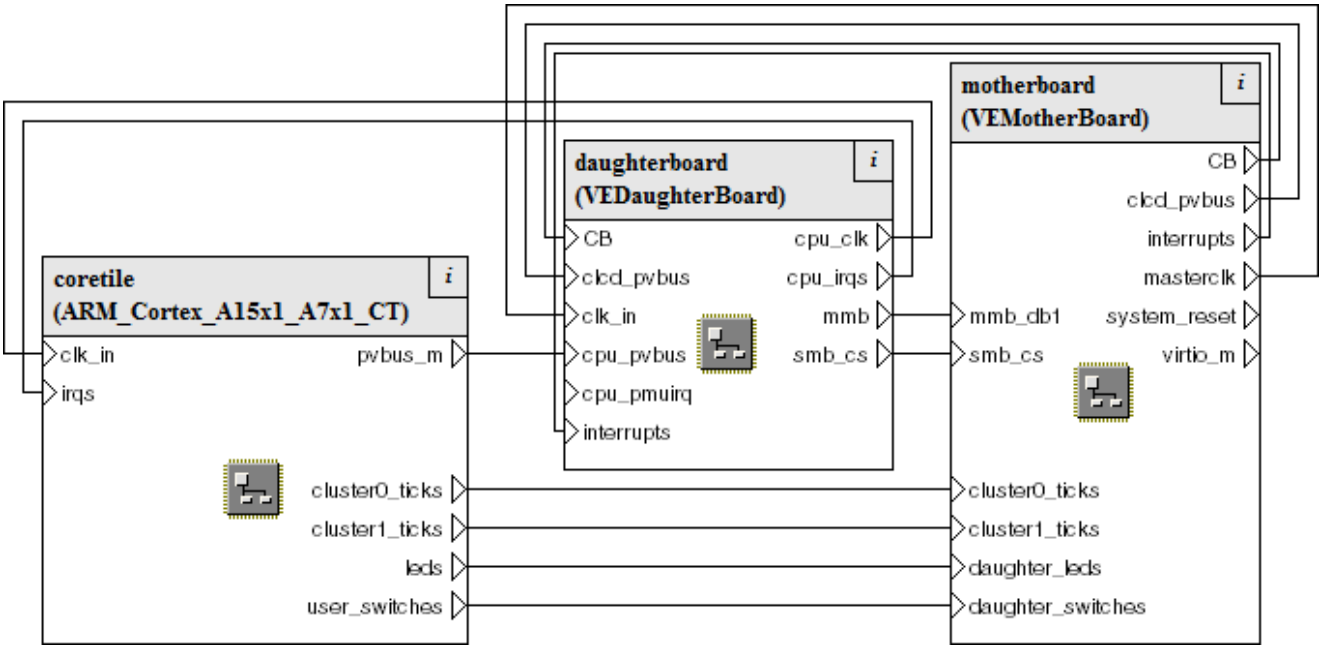

Figure 5-1 Block diagram of top-level VE model

Related references

Chapter 5 Programming Reference for VE FVPs on page 5-52.

Related information

Motherboard Express µATX V2M-P1 Technical Reference Manual.

5.2 VE - model memory map

This section describes the VE model memory map.

This section contains the following subsections:

5.2.1 VE - memory map - about

The global memory map for the VE model is based on the hardware Versatile Express RS1 memory map with the RS2 extensions.

Note

The VE FVP implementation of memory does not require programming the memory controller with the correct values. If you run applications on hardware, set up the memory controller properly. Otherwise, applications that run on an FVP might fail on hardware.

Related references

5.5 VE - differences between VE and CoreTile hardware and models on page 5-77.

5.2.2 VE - global memory map

This section describes the global memory map.

Table 5-1 VE global memory map

Peripheral	Modeled	Address range	Size
NOR FLASH0 (CS0)	Yes	`0x0000000000-0x0003FFFFFF`	64MB
Reserved	-	`0x0004000000-0x0007FFFFFF`	64MB
NOR FLASH0 alias (CS0)	Yes	`0x0008000000-0x000BFFFFFF`	64MB
NOR FLASH1 (CS4)	Yes	`0x000C000000-0x000FFFFFFF`	64MB
Unused (CS5)	-	`0x0010000000-0x0013FFFFFF`	-
PSRAM (CS1) - unused	No	`0x0014000000-0x0017FFFFFF`	-
Peripherals (CS2)	Yes	`0x0018000000-0x001BFFFFFF`	64MB
Peripherals (CS3)	Yes	`0x001C000000-0x001FFFFFFF`	64MB
CoreSight and peripherals	No	`0x0020000000-0x002CFFFFFF`[m]	-
Graphics space	No	`0x002D000000-0x002D00FFFF`	-
System SRAM	Yes	`0x002E000000-0x002EFFFFFF`	64KB
Ext AXI	No	`0x002F000000-0x007FFFFFFF`	-
4GB DRAM (in 32-bit address space)[n]	Yes	`0x0080000000-0x00FFFFFFFF`	2GB

[m] The private peripheral region address `0x2c000000` is mapped in this region. The parameter PERIPHBASE can be used to map the peripherals to a different address.

[n] The model contains only 4GB of DRAM. The DRAM memory address space is aliased across the three different regions and where the mapped address space is greater than 4GB.

Table 5-1 VE global memory map (continued)

Peripheral	Modeled	Address range	Size
Unused	-	0x0100000000-0x07FFFFFFFF	-
4GB DRAM (in 36-bit address space)[n]	Yes	0x0800000000-0x08FFFFFFFF	4GB
Unused	-	0x0900000000-0x7FFFFFFFFF	-
4GB DRAM (in 40-bit address space)[n]	Yes	0x8000000000-0xFFFFFFFFFF	4GB

5.2.3 VE - memory map for peripherals in the CS2 region

This section describes the CS2 region.

Table 5-2 Memory map for peripherals in the CS2 region

Peripheral	Modeled	Address range	Size	GIC Int
VRAM - aliased	Yes	0x0018000000-0x0019FFFFFF	32MB	-[o]
Ethernet (SMSC 91C111)	Yes	0x001A000000-0x001AFFFFFF	16MB	47
USB - unused	No	0x001B000000-0x001BFFFFFF	16MB	-

5.2.4 VE - memory map for peripherals in the CS3 region

This section describes the CS3 region.

Table 5-3 Memory map for peripherals in the CS3 region

Peripheral	Modeled	Address range	Size	GIC Int
Local DAP ROM	No	0x001C000000-0x001C00FFFF	64KB	-[p]
VE System Registers	Yes	0x001C010000-0x001C01FFFF	64KB	-
System Controller (SP810)	Yes	0x001C020000-0x001C02FFFF	64KB	-
TwoWire serial interface (PCIe)	No	0x001C030000-0x001C03FFFF	64KB	-
AACI (PL041)	Yes	0x001C040000-0x001C04FFFF	64KB	43
MCI (PL180)	Yes	0x001C050000-0x001C05FFFF	64KB	41, 42
KMI - keyboard (PL050)	Yes	0x001C060000-0x001C06FFFF	64KB	44
KMI - mouse (PL050)	Yes	0x001C070000-0x001C07FFFF	64KB	45
Reserved	-	0x001C080000-0x001C08FFFF	64KB	-
UART0 (PL011)	Yes	0x001C090000-0x001C09FFFF	64KB	37
UART1 (PL011)	Yes	0x001C0A0000-0x001C0AFFFF	64KB	38
UART2 (PL011)	Yes	0x001C0B0000-0x001C0BFFFF	64KB	39
UART3 (PL011)	Yes	0x001C0C0000-0x001C0CFFFF	64KB	40
VFS2	Yes	0x001C0D0000-0x001C0DFFFF	64KB	73
Reserved	-	0x001C0E0000-0x001C0EFFFF	64KB	-

[o] The Interrupt signal column lists the values to use to program your interrupt controller. These values are after mapping the SPI number by adding 32. The interrupt numbers from the peripherals are modified by adding 32 to form the interrupt number that the GIC sees. GIC interrupts 0-31 are for internal use.

[p] The Interrupt signal column lists the values to use to program your interrupt controller. These values are after mapping the SPI number by adding 32. The interrupt numbers from the peripherals are modified by adding 32 to form the interrupt number that the GIC sees. GIC interrupts 0-31 are for internal use.

Table 5-3 Memory map for peripherals in the CS3 region (continued)

Peripheral	Modeled	Address range	Size	GIC Int
Watchdog (SP805)	Yes	0x001C0F0000-0x001C0FFFFF	64KB	32
Reserved	-	0x001C100000-0x001C10FFFF	64KB	-
Timer-0 (SP804)	Yes	0x001C110000-0x001C11FFFF	64KB	34
Timer-1 (SP804)	Yes	0x001C120000-0x001C12FFFF	64KB	35
Reserved	-	0x001C130000-0x001C15FFFF	192KB	-
TwoWire serial interface (DVI) - unused	No	0x001C160000-0x001C16FFFF	64KB	-
Real-time Clock (PL031)	Yes	0x001C170000-0x001C17FFFF	64KB	36
Reserved	-	0x001C180000-0x001C19FFFF	128KB	-
CF Card - unused	No	0x001C1A0000-0x001C1AFFFF	64KB	
Reserved	-	0x001C1B0000-0x001C1EFFFF	256KB	-
Color LCD Controller (PL111)	Yes	0x001C1F0000-0x001C1FFFFF	64KB	46
Reserved	-	0x001C200000-0x001FFFFFFF	62KB	-

5.2.5 VE - global memory map for secure_memory option

The VE model has a `secure_memory` option. Enabling this option changes the memory map for several peripherals.

The peripherals are at the same location, but hidden behind a TZSwitch.

Table 5-4 Memory map for secure_memory option

Peripheral	Address range	Functionality with secure_memory enabled
SecureRO Flash (CS0)	0x0000000000-0x0003FFFFFF	Aborts on Non-secure accesses.
Secure SRAM	0x0004000000-0x000401FFFF	Aborts on Non-secure accesses.
Peripherals	0x0008000000-0x007DFFFFFF	Aborts on Secure accesses.
Secure DRAM	0x007e000000-0x007FFFFFFF	Aborts on Non-secure accesses.
4GB DRAM (in 32-bit address space)	0x0080000000-0xFFFFFFFFFF	Normal memory MPA, aborts on Secure accesses.

5.3 VE - clock and timer

This section describes the frequencies of the clock and timer.

Cluster `clk_in` frequency parameter
: 100MHz.

GenericTimer `base_frequency` parameter
: 100MHz.

5.4 VE - components

This section describes the VE components.

This section contains the following subsections:

5.4.1 VE - components - about

These component models implement some of the functionality of the *Versatile Express* (VE) hardware.

A complete model implementation of the VE platform includes both VE-specific components and generic ones such as buses and timers. See the hardware documentation for functionality information.

The parameter sections for these components describe the parameters that you can define at runtime, and the parameters that you can only modify at build time. They do not describe the parameters that you would not normally modify in the hardware.

5.4.2 VE - components - instance names

This section describes the names of the internal components.

Parameter syntax:

```
motherboard.component.parameter=value
```

Table 5-5 VE FVP instances

Name	Type
`cluster.cpu0`	Core
`motherboard`	VEMotherBoard
`motherboard.sp805_wdog`	SP805_Watchdog
`motherboard.pl050_kmi1`	PL050_KMI
`motherboard.pl050_kmi0`	PL050_KMI
`motherboard.Timer_2_3`	SP804_Timer
`motherboard.Timer_2_3.counter0`	CounterModule
`motherboard.Timer_2_3.counter1`	CounterModule
`motherboard.Timer_0_1`	SP804_Timer
`motherboard.Timer_0_1.counter0`	CounterModule
`motherboard.Timer_0_1.counter1`	CounterModule
`motherboard.sp810_sysctrl`	SP810_SysCtrl
`motherboard.pl111_clcd`	PL111_CLCD
`motherboard.pl031_rtc`	PL031_RTC
`motherboard.pl011_uart3`	PL011_Uart
`motherboard.pl011_uart2`	PL011_Uart

Table 5-5 VE FVP instances (continued)

Name	Type
motherboard.pl011_uart1	PL011_Uart
motherboard.pl011_uart0	PL011_Uart
motherboard.pl180_mci	PL180_MCI
motherboard.pl041_aaci	PL041_AACI
motherboard.ve_sysregs	VE_SysRegs
motherboard.smsc_91c111	SMSC_91C111
motherboard.ps2mouse	PS2Mouse
motherboard.ps2keyboard	PS2Keyboard
motherboard.mmc	MMC
motherboard.dummy_CF	WarningMemory
motherboard.dummy_usb	RAMDevice
motherboard.dummy_ram	RAMDevice
motherboard.dummy_local_dap_rom	RAMDevice
motherboard.psram	RAMDevice
motherboard.vram	RAMDevice
motherboard.flash1	IntelStrataFlashJ3
motherboard.flash0	IntelStrataFlashJ4
motherboard.flashloader1	FlashLoader
motherboard.flashloader0	FlashLoader
motherboard.terminal_0	TelnetTerminal
motherboard.terminal_1	TelnetTerminal
motherboard.terminal_2	TelnetTerminal
motherboard.terminal_3	TelnetTerminal
motherboard.audioout	AudioOut_SDL
motherboard.vis	VEVisualisation
motherboard.vis.recorder	VisEventRecorder
motherboard.virtualethernetcrossover	VirtualEthernetCrossover
motherboard.hostbridge	HostBridge
motherboard.vfs2	VFS2
motherboard.vfs2.messagebox	MessageBox
motherboard.virtioblockdevice	VirtioBlockDevice
daughterboard	VEDaughterBoard
daughterboard.sram	RAMDevice
daughterboard.vedcc	VEDCC
daughterboard.dram	RAMDevice

Table 5-5 VE FVP instances (continued)

Name	Type
daughterboard.dram_aliased	TZSwitch
daughterboard.dram_limit_4	TZSwitch
daughterboard.dram_limit_8	TZSwitch
daughterboard.introuter	VEInterruptMapper
daughterboard.secure_region	TZSwitch
daughterboard.nonsecure_region	TZSwitch
daughterboard.secureRO	IntelStrataFlashJ3
daughterboard.secureROloader	FlashLoader
daughterboard.secureSRAM	RAMDevice
daughterboard.secureDRAM	RAMDevice
daughterboard.hdlcd	PL370_HDLCD
daughterboard.dmc	RAMDevice
daughterboard.dmc_phy	RAMDevice

5.4.3 VE - peripheral components

This section describes the peripheral components on the motherboard.

Color LCD controller - parameters

This section describes the parameters.

The syntax to use in a configuration file or on the command line is:

`motherboard.pl111_clcd.`*`parameter=value`*

Table 5-6 Color LCD controller parameters

Parameter	Type	Allowed values	Default value	Description
pixel_double_limit	int	-	0x12C	The threshold in horizontal pixels below which pixels sent to the frame-buffer are doubled in size in both dimensions.

Elfloader - parameters

Configure this alternative method of loading elf files into the system.

Table 5-7 Elfloader parameters

Parameter	Type	Allowed values	Default value	Description
elf	string	-	""	ELF file name.
lfile	string	-	""	Load file for large address mapping.
ns_copy	bool	true, false	true	Copy whole file to NS memory space.

Ethernet - parameters

You can change these parameters after the model starts.

The syntax to use in a configuration file or on the command line is:

`motherboard.smsc_91c111.parameter=value`

Table 5-8 Ethernet parameters

Parameter	Type	Allowed values	Default value	Description
`enabled`	`bool`	`true`, `false`	`false`	Host interface connection enabled.
`mac_address`	`string`	See below.	`00:02:f7:ef:31:11`	Host/model MAC address.
`promiscuous`	`bool`	`true`, `false`	`true`	Put host into promiscuous mode, for example when sharing the Ethernet controller with the host OS.

`mac_address`

- If you do not specify a MAC address, when the simulator is run it takes the default MAC address and changes its bottom two bytes from `00:02` to the bottom two bytes of the MAC address of one of the adaptors on the host PC. This increases the chance of the MAC address being unique when running models on multiple hosts on a local network.
- If you specify the MAC address as auto, this generates a completely random local MAC address each time the simulator is run. The address has bit 1 set and bit 0 clear in the first byte to indicate a locally-administered unicast MAC address.

Note

DHCP servers are used to allocate IP addresses, but because they sometimes do this based on the MAC address provided to them, using random MAC addresses might interact with some DHCP servers.

System controller - parameters

You can change these parameters after the model starts.

The syntax to use in a configuration file or on the command line is:

`motherboard.sp810_sysctrl.parameter=value`

Table 5-9 System controller parameters

Parameter	Type	Allowed values	Default value	Description
`sysid`	`int`	-	`0x00000000`	Value for system identification register. `0`, SYS_ID register value = `0x0225f500`, corresponding to REV_A.`1`, SYS_ID register value = `0x12257500`, corresponding to REV_B. `2`, SYS_ID register value = `0x22252500`, corresponding to REV_C. For other values: SYS_ID register value of `0x0`.
`use_s8`	`bool`	`true`, `false`	`false`	Select whether switch S8 is enabled.

System register block - parameters

You can change these parameters after the model starts.

The syntax to use in a configuration file or on the command line is:

`motherboard.ve_sysregs.parameter=value`

Table 5-10 System register parameters

Parameter	Type	Allowed values	Default value	Description
exit_on_shutdown	bool	true, false	false	SYS_CFG_SHUTDOWN exits simulation.
mmbSiteDefault	int	0x0-0x2	0x1	Default MMB source (0=MB, 1=DB1, 2=DB2).
tilePresent	bool	true, false	true	CoreTile fitted status.
user_switches_value	int	-	0x00	User switches.

UART - parameters

You can change these parameters after the model starts.

The syntax to use in a configuration file or on the command line is:

`motherboard.pl011_uart`*`x`*`.`*`parameter`*`=`*`value`*

where *x* is the UART identifier 0, 1, 2, or 3.

Table 5-11 UART parameters

Parameter	Type	Allowed values	Default value	Description
baud_rate	int	-	0x9600	Baud rate.
clock_rate	int	-	0xE10000	Clock rate for PL011.
generic_uart	bool	true, false	false	Permit access only to the subset of registers defined as *Generic Uart* in the SBSA specification.
in_file	string	-	""	Input file for the UART to read.
out_file	string	-	""	Output file (use "-" to send all output to stdout)
in_file_escape_sequence	string	-	"##"	Input file escape sequence.
revision	string	-	"r1p4"	Revision to simulate (affects ID register and FIFO capacity).
shutdown_on_eot	bool	true, false	false	Shut down simulation when an EOT (ASCII 4) char is transmitted. Useful for regression tests when semihosting is not available.
shutdown_tag	string	-	""	String that causes a shutdown simulation when transmitted.
unbufferred_output	bool	true, false	false	Unbuffered output
untimed_fifos	bool	true, false	true	Ignore the clock rate and transmit/receive serial data immediately.
uart_enable	bool	true, false	false	Enable the UART when the system starts, to make baud_rate and clock_rate valid.

v8EmbeddedCrossTrigger_Matrix - parameters

This section describes the parameters.

Table 5-12 v8EmbeddedCrossTrigger_Matrix parameters

Name	Type	Allowed values	Default value	Description
has_CTIAUTHSTATUS	bool	true, false	true	Enables the CTIAUTHSTATUS register.
number-of-channels	uint32_t	0x3-0x20, 3-32	0x4, 4	Number of channels in the CTM.

Watchdog - parameters

You can change these parameters after the model starts.

The syntax to use in a configuration file or on the command line is:

`motherboard.sp805_wdog.`*`parameter`*`=`*`value`*

Table 5-13 Watchdog parameters

Parameter	Type	Allowed values	Default value	Description
simhalt	bool	true, false	false	Halt on reset.

5.4.4 VE - virtual components

This section describes the virtual components on the motherboard.

FLASH loader - parameters

This section describes the parameters.

The syntax to use in a configuration file or on the command line is:

`motherboard.flashloader`*`x`*`.`*`parameter`*`=`*`value`*

where *x* is the FLASH identifier 0 or 1.

Table 5-14 FLASH loader parameters

Parameter	Type	Allowed values	Default value	Description
fname	string	Valid strings	""	Path to the host file used to initialize FLASH contents when the model starts. The file can be gzip compressed.
fnameWrite	string	Valid strings	""	Path to the host file used to save FLASH contents when the model exits.

Hostbridge - parameters

This section describes the parameters.

The syntax to use in a configuration file or on the command line is:

`motherboard.hostbridge.`*`parameter`*`=`*`value`*

Table 5-15 Hostbridge parameters

Parameter	Type	Allowed values	Default value	Description
interfaceName	string	-	"ARM0"	Host interface identifier.
userNetPorts	string	-	""	Listening ports to expose in user-mode networking.
userNetSubnet	string	-	"172.20.51.0/24"	Virtual subnet for user-mode networking.
userNetworking	bool	true, false	false	Enable user-mode networking.

Multimedia card - parameters

This section describes the parameters.

The syntax to use in a configuration file or on the command line is:

`motherboard.mmc.`*`parameter=value`*

Table 5-16 *MultiMedia Card* (MMC) parameters

Parameter	Type	Allowed values	Default value	Description
`card_type`	`string`	-	`"SD"`	Card type: `"SD"` or `"eMMC"`.
`force_sector_addressing`	`bool`	`true`, `false`	`false`	Use sector addressing, even on small cards.
`p_fast_access`	`bool`	`true`, `false`	`true`	Do not simulate MMC block access delays.
`p_manid`	`int`	-	`0x2`	Card ID manufacturer ID.
`p_mmc_file`	`string`	-	`"mmc.dat"`	Backing store file.
`p_OEMid`	`int`	-	`0xCA4D0001`	Card ID OEM ID.
`p_prodName`	`string`	Six-character string	`"ARMmmc"`	Card ID product name.
`p_prodRev`	`int`	-	`0x1`	Card ID product revision.
`p_sernum`	`int`	-	`0xCA4D0001`	Card serial number.

TelnetTerminal - parameters

This section describes the parameters.

Table 5-17 TelnetTerminal parameters

Name	Type	Allowed values	Default value	Description
`mode`	`string`	`telnet`[q], `raw`[r]	`telnet`	Terminal operation mode.
`quiet`	`bool`	`true`, `false`	`false`	Avoid output on `stdout` or `stderr`.
`start_port`	`int`	-	5000	Telnet TCP port number, of the port for the terminal when the system starts. If this port is not free, the port value is incremented by 1 until a free port is found.
`start_telnet`	`bool`	`true`, `false`	`true`	Enable terminal when the system starts.

VFS2 - parameters

This section describes the parameters.

Table 5-18 VFS2 parameters

Name	Type	Allowed values	Default value	Description
`mount`	`string`	-	`"`	Path to host folder to make accessible inside the model.

VEVisualisation - parameters

This section describes the configuration parameters.

[q] In telnet mode, this component supports a subset of the telnet protocol defined in RFC 854.
[r] In raw mode, this component does not interpret or modify the byte stream contents. This permits a debugger connection, for example, to connect a gdb client to a gdbserver running on the target operating system.

Note

Setting the `rate_limit-enable` parameter to `true` (the default) prevents the simulation from running too fast on fast workstations and enables timing loops and mouse actions to work correctly. However, it reduces the overall simulation speed. If your priority is high simulation speed, set `rate_limit-enable` to `false`.

Table 5-19 VEVisualisation parameters

Name	Type	Allowed values	Default value	Description
`cluster0_name`	`string`	-	Cluster0	Label for cluster 0 performance values.
`cluster1_name`	`string`	-	Cluster1	Label for cluster 1 performance values.
`cpu_name`	`string`	-		Processor name displayed in window title.
`daughter_led_count`	`int`	`0-32`	`0`	Set to nonzero to display up to 32 LEDs. See `daughter_leds` port.
`daughter_user_switch_count`	`int`	`0-32`	`0`	Set this parameter to display up to 32 switches. See `daughter_user_switches` port.
`disable_visualisation`	`bool`	`true`, `false`	`false`	Disable the VEVisualisation component on model startup.
`rate_limit-enable`	`bool`	`true`, `false`	`true`	Restrict simulation speed so that simulation time more closely matches real time rather than running as fast as possible.
`recorder.checkInstructionCount`	`bool`	`true`, `false`	`true`	Check instruction count in recording file against actual instruction count during playback.
`recorder.playbackFileName`	`string`	-	`""`	Playback filename (empty string disables playback).
`recorder.recordingFileName`	`string`	-	`""`	Recording filename (empty string disables recording).
`recorder.recordingTimeBase`	`int`	-	`0x5F5E100`	Timebase in 1/s (relative to the master clock (where 100000000 means 10 nanoseconds resolution simulated time for a 1Hz master clock)) for recording (higher values give higher time resolution, playback timebase is always taken from the playback file).
`recorder.verbose`	`int`	-	`0x0`	Enable verbose messages (1=normal, 2=even more).
`trap_key`	`int`	Valid ATKeyCode key value	`74`, `0x4A`[s]	Trap key that works with left **Ctrl** key to toggle mouse display.
`window_title`	`string`	-	`"Fast Models - CLCD %cpu%"`	Window title (`cpu_name` replaces `%cpu%`).

[s] This is equivalent to the left **Alt** key, so pressing Left Alt and Left Ctrl simultaneously toggles the mouse display.

5.4.5 FVP_VE_Cortex-A15xn CoreTile component

This section describes the FVP_VE_Cortex-A15xn CoreTile component.

FVP_VE_Cortex-A15xn CoreTile - parameters

These components have instantiation-time parameters, which you can change when you start the models, where *x* = 1, 2, 4.

This CoreTile FVP is based on revision 2, patch 0 (r2p0) of the Cortex-A15 cluster.

The syntax to use in a configuration file is:

`cluster.`*`parameter`*`=`*`value`*

Table 5-20 FVP_VE_Cortex-A15xn CoreTile parameters

Parameter	Type	Allowed values	Default value	Description
`CFGSDISABLE`	`bool`	`true`, `false`	`false`	Disable some accesses to DIC registers.
`CLUSTER_ID`	`int`	0-15	0	Cluster ID value.
`IMINLN`	`bool`	`true`, `false`	`true`	Instruction cache minimum line size: `false` = 32 bytes, `true` = 64 bytes.
`PERIPHBASE`	`int`	-	`0x13080000`[t]	Base address of peripheral memory space.
`dic-spi_count`	`int`	0-224, in increments of 32	64	Number of shared peripheral interrupts implemented.
`internal_vgic`	`bool`	`true`, `false`	`true`	Configures whether the model of the cluster contains a *Virtual Generic Interrupt Controller* (VGIC).
`l1_dcache-state_modelled`	`bool`	`true`, `false`	`false`	Set whether L1 D-cache has stateful implementation.
`l1_icache-state_modelled`	`bool`	`true`, `false`	`false`	Set whether L1 I-cache has stateful implementation.
`l2_cache-size`	`int`	`0x080000`, `0x100000`, `0x200000`, `0x400000`.	`0x400000`	Set L2 cache size in bytes.
`l2_cache-state_modelled`	`bool`	`true`, `false`	`false`	Set whether L2 cache has stateful implementation.
`l2-data-slice`	`int`	0, 1, 2	0	L2 data RAM slice.
`l2-tag-slice`	`int`	0, 1	0	L2 tag RAM slice.

The FVP_VE_Cortex-A15MPx1 has the `PERIPHBASE` parameter set to `0x1F000000`, which is the base address of peripheral memory space on VE hardware.

The parameters for each Cortex-A15 core are set individually. Each core has its own timer and watchdog.

The syntax to use in a configuration file is:

`cluster.cpu[`*`n`*`].`*`parameter`*`=`*`value`*

where *n* is the core number, from 0-3 inclusive.

[t] If you are using the ARMCortexA15x*n*CT component on a VE model platform, this parameter is set automatically to `0x1F000000` and is not visible in the parameter list.

Table 5-21 FVP_VE_Cortex-A15xn CoreTile parameters - individual cores

Parameter	Type	Allowed values	Default value	Description
CFGEND	bool	true, false	false	Initialize to BE8 endianness.
CP15SDISABLE	bool	true, false	false	Initialize to disable access to some CP15 registers.
DBGROMADDR	int	0x12000003	0x12000003	This value is used to initialize the CP15 **DBGDRAR** register. Bits[39:12] of this register specify the ROM table physical address.
DBGROMADDRV	bool	true, false	true	true sets bits[1:0] of the CP15 **DBGDRAR** to indicate that the address is valid.
DBGSELFADDR	int	0x00010003	0x00010003	This value is used to initialize the CP15 **DBGDSAR** register. Bits[39:17] of this register specify the ROM table physical address.
DBGSELFADDRV	bool	true, false	true	true sets bits[1:0] of the CP15 **DBGDSAR** to indicate that the address is valid.
TEINIT	bool	true, false	false	T32 exception enable. The default has exceptions including reset handled in A32 state.
VINITHI	bool	true, false	false	Initialize with high vectors enabled.
ase-present[u]	bool	true, false	true	Set whether core model has been built with NEON™ support.
min_sync_level	int	0-3	0	Controls the minimum syncLevel by the CADI parameter interface.
semihosting-cmd_line	string	No limit except memory	"	Command line available to semihosting SVC calls.
semihosting-cwd	string	-	-	Virtual address of CWD.
semihosting-enable	bool	true, false	true	Enable semihosting SVC traps.
semihosting-ARM_SVC	int	0x000000-0xFFFFFF	0x123456	A32 SVC number for semihosting.
semihosting-Thumb_SVC	int	0x00-0xFF	0xAB	T32 SVC number for semihosting.
semihosting-heap_base	int	0x00000000-0xFFFFFFFF	0x0	Virtual address of heap base.
semihosting-heap_limit	int	0x00000000-0xFFFFFFFF	0x0F000000	Virtual address of top of heap.
semihosting-stack_base	int	0x00000000-0xFFFFFFFF	0x10000000	Virtual address of base of descending stack.

[u] The ase-present and vfp-present parameters configure the synthesis options for the Cortex-A15 model. The options are:

vfp present and ase present
NEON and VFPv3-D32 supported.
vfp present and ase not present
VFPv3-D16 supported.
vfp not present and ase present
Illegal. Forces vfp-present to true so model has NEON and VFPv3-D32 support.
vfp not present and ase not present
NEON and VFPv3-D32 not supported.

Table 5-21 FVP_VE_Cortex-A15xn CoreTile parameters - individual cores (continued)

Parameter	Type	Allowed values	Default value	Description
semihosting-stack_limit	int	0x00000000-0xFFFFFFFF	0x0F000000	Virtual address of stack limit.
vfp-enable_at_reset[v]	bool	true, false	false	Enable coprocessor access and VFP at reset.
vfp-present[u]	bool	true, false	true	Set whether processor model has been built with VFP support.

5.4.6 ARMAEMv8AMPCT component

This section describes the ARMv8-A *Architecture Envelope Model* (AEM) component.

ARMAEMv8AMPCT - parameters

This section describes the parameters that configure the behavior of the AEM ARMv8-A processor model.

ARMAEMv8AMPCT - cluster parameters

This section describes the parameters.

Note

- Terms such as `cluster` might replace `cpu` on some systems.
- The parameter `PERIPHBASE` is locked down in the VE FVP.

Table 5-22 ARMAEMv8AMPCT cluster parameters

Parameter	Type	Allowed values	Default value	Description
apsr_read_restrict	bool	true, false	false	At EL0, UNKNOWN bits of APSR are RAZ.
auxilliary_feature_register0	int	0x0-0xFFFFFFFF	0x0	Value for *Auxiliary Feature Register 0* (ID_AFR0).
BPIMVA_causes_translation_lookup	bool	true, false	false	Do a translation when BPIMVA instruction is executed. This translation might cause a translation fault.
clear_reg_top_eret	int	0x0-0x2	0x1	Clear top 32 bits of general purpose registers on exception return. `0x0` = preserve, `0x1` = clear to zero, `0x2` = random choice of preserve or clear to zero.
delay_serror	int	0x0-0xFFFFFFFF	0x0	Minimum propagation delay of the *System Error* (SERR) signal into the cluster. Accurate in low-latency mode (`-C cpu.scheduler_mode=1`), but otherwise any delay might be larger.
el0_el1_only_non_secure	bool	true, false	false	Controls security state of EL0 and EL1 if EL2 and EL3 are not implemented. `true` means non-secure.

[v] This model-specific behavior has no hardware equivalent.

Table 5-22 ARMAEMv8AMPCT cluster parameters (continued)

Parameter	Type	Allowed values	Default value	Description
exercise_stxr_fail	bool	true, false	false	When true, return a pseudorandom majority of *Store Exclusive Register* (STXR) instructions as Failed.
has_16bit_asids	bool	true, false	true	Enable 16-bit *Address Space IDentifiers* (ASIDs).
has_el2	bool	true, false	true	Enable EL2.
has_el3	bool	true, false	true	Enable EL3.
has_delayed_sysreg	bool	true, false	false	Delay the functional effect of system register writes until ISB or implicit barrier.
has_writebuffer	bool	true, false	false	Implement write access buffering before L1 cache. Might affect ext_abort behavior.
hcr_swio_res1	bool	true, false	false	Whether either HCR.SWIO or HCR_EL2.SWIO, or both are RES1.
is_uniprocessor	bool	true, false	false	Value for the U bit in MPIDR. true disables L1 cache coherency protocols.
max_32bit_el	int	-0x1-0x3	0x3	Maximum exception level supporting AArch32 modes. -0x1 means no support.
MIDR	int	0x0-0xFFFFFFFF	0x410FD0F0	Value for *Main ID Register* (MIDR).
mixed_endian	int	0x0-0x2	0x1	Enable the processor to change the endianness at runtime. 0x0 = not supported, 0x1 = supported at all exception levels, 0x2 = supported at EL0 only.
NUM_CORES	int	0x0-0x4	0x4	Number of cores implemented.
PA_SIZE	int	0x20-0x30	0x28	Physical address size, in bits.
register_reset_data	int	-	0x0	Fill data for register bits when they become UNKNOWN at reset.
scramble_unknowns_at_reset	bool	true, false	true	Fill in UNKNOWN bits in registers at reset with register_reset_data.
take_ccfail_undef	bool	true, false	true	In AArch32, take Undefined Instruction exception even if the instruction fails its condition-codes check.
tidcp_traps_el0_undef_imp_def	bool	true, false	true	The TIDCP bit traps, in EL0, undefined IMPLEMENTATION DEFINED instructions accessing coprocessor registers.
unpredictable_hvc_behaviour	int	0x0-0x1	0x0	Define HVC UNPREDICTABLE behavior in HYP mode, EL2, when the SCR.HCE bit is clear. 0x0 = Undefined Instruction, 0x1 = NOP instruction.

Table 5-22 ARMAEMv8AMPCT cluster parameters (continued)

Parameter	Type	Allowed values	Default value	Description
unpredictable_smc_behaviour	int	0x0-0x1	0x0	Define SMC UNPREDICTABLE behavior in Secure mode, EL3, when the SCR.SCD bit is clear. 0x0 = Undefined Instruction, 0x1 = NOP instruction.
warn_unpredictable_in_v7	bool	true, false	true	Warn of UNPREDICTABLE behavior in ARMv7.

ARMAEMv8AMPCT - core parameters

This section describes the parameters.

Each core in a cluster has its own parameters. The models use the parameters for cores in sequence, from cpu0 onwards. The models ignore parameters for uninstantiated cores.

Table 5-23 ARMAEMv8AMPCT core parameters

Parameter	Type	Allowed values	Default value	Description
cpu[*n*].CONFIG64	bool	true, false	true	Enable AArch64.
cpu[*n*].SMPnAMP	bool	true, false	true	This core is in the inner shared domain, and uses its cache coherency protocol.
cpu[*n*].CFGEND	bool	true, false	false	Use big-endian order.
cpu[*n*].CP15SDISABLE	bool	true, false	false	Disable access to some CP15 registers in AArch32.
cpu[*n*].ase-present	bool	true, false	true	Enable NEON.
cpu[*n*].VINITHI	bool	true, false	false	Enable high vectors. Base address 0xFFFF0000.
cpu[*n*].RVBAR	int	0x0-0xFFFFFFFFFFFC	0x0	Reset Vector Base Address when resetting into AArch64.
cpu[*n*].vfp-present	bool	true, false	true	Enable floating-point arithmetic.
cpu[*n*].vfp-enable_at_reset	bool	true, false	false	Enable coprocessor access and VFP at reset.[w]
cpu[*n*].force-fpsid	bool	true, false	false	Override the FPSID value.
cpu[*n*].force-fpsid-value	int	0x0-0xFFFFFFFF	0x0	Value for the overridden FPSID.
cpu[*n*].TEINIT	bool	true, false	false	Controls the initial state of SCTLR.TE in AArch32. When set, causes AArch32 exceptions (including reset) to be taken in T32 mode.
cpu[*n*].etm-present	bool	true, false	true	Enable *Embedded Trace Macrocell* (ETM).
cpu[*n*].min_sync_level	int	0x0-0x3	0x0	Minimum CADI syncLevel. 0 = off, 1 = syncState, 2 = postInsnIO, 3 = postInsnAll.

ARMAEMv8AMPCT - cache parameters

This section describes the parameters.

[w] This behavior is model-specific, with no hardware equivalent.

Table 5-24 ARMAEMv8AMPCT cache parameters

Parameter	Type	Allowed values	Default value	Description
`cache_maintenance_hits_watchpoints`	`bool`	`true`, `false`	`false`	Enable AArch32 cache maintenance by `DCIMVAC` to trigger watchpoints.[x]
`dcache-state_modelled`	`bool`	`true`, `false`	`false`	Stateful implementation, with line allocation in D-caches at all levels.[y]
`icache-state_modelled`	`bool`	`true`, `false`	`false`	Stateful implementation, with line allocation in I-caches at all levels.[y]
`memory.l2_cache.is_inner_cacheable`	`bool`	`true`, `false`	`true`	L2 cache is inner cacheable, not outer cacheable.
`memory.l2_cache.is_inner_shareable`	`bool`	`true`, `false`	`true`	L2 cache is inner shareable, not outer shareable.
`cache-log2linelen`	`int`	`0x4-0x8`	`0x6`	Log_2(cache-line length, in bytes).
`icache-log2linelen`	`int`	`0x0-0x8`	`0x0`	If non-zero, Log_2(instruction cache line length in bytes). Otherwise, use `cache-log2linelen`.
`cpu[n].DCZID-log2-block-size`	`int`	`0x0-0x9`	`0x8`	Log_2(block size) cleared by DC ZVA instruction.[z]
`dcache-size`	`int`	`0x4000-0x100000`	`0x8000`	L1 D-cache size, in bytes.
`dcache-ways`	`int`	`0x1-0x40`	`0x2`	Number of L1 D-cache ways.[aa]
`icache-size`	`int`	`0x4000-0x100000`	`0x8000`	L1 I-cache size, in bytes.
`icache-ways`	`int`	`0x1-0x40`	`0x2`	Number of L1 I-cache ways.[aa]
`l2cache-size`	`int`	`0x0-0x1000000`	`0x80000`	L2 cache size, in bytes.
`l2cache-ways`	`int`	`0x1-0x40`	`0x10`	Number of L2 cache ways.[aa]

ARMAEMv8AMPCT - TLB parameters

This section describes the parameters.

Table 5-25 ARMAEMv8AMPCT *Translation Lookaside Buffer* (TLB) parameters

Parameter	Type	Allowed values	Default value	Description
`stage12_tlb_size`	`int`	`0x1-0xFFFFFFFF`	`0x80`	Number of stage 1 and stage 2 TLB entries.
`stage1_tlb_size`	`int`	`0x0-0xFFFFFFFF`	`0x0`	Number of stage 1 TLB entries.
`stage2_tlb_size`	`int`	`0x0-0xFFFFFFFF`	`0x0`	Number of stage 2 TLB entries.
`stage1_walkcache_size`	`int`	`0x0-0xFFFFFFFF`	`0x0`	Number of stage 1 TLB walk cache entries.
`stage2_walkcache_size`	`int`	`0x0-0xFFFFFFFF`	`0x0`	Number of stage 2 TLB walk cache entries.
`instruction_tlb_size`	`int`	`0x0-0xFFFFFFFF`	`0x0`	Number of stage 1 and stage 2 ITLB entries. `0x0` for unified ITLB + DTLB.

[x] UNPREDICTABLE.
[y] Unified caches allocate lines only if these parameters are enabled at both I-side and D-side.
[z] As read from DCZID_EL0.
[aa] Sets are implicit from size.

Table 5-25 ARMAEMv8AMPCT *Translation Lookaside Buffer* (TLB) parameters (continued)

Parameter	Type	Allowed values	Default value	Description
enable_tlb_contig_check	bool	true, false	true	Check consistency of TLB entries in regions with the Contiguous Bit set.
has_tlb_conflict_abort	bool	true, false	false	Inconsistent TLB content generates aborts.
use_tlb_contig_hint	bool	true, false	false	Pagetable entries with the Contiguous Bit set generate large TLB entries.

ARMAEMv8AMPCT - cryptography parameters

This section describes the parameters.

Table 5-26 ARMAEMv8AMPCT cryptography parameters

Parameter	Type	Allowed values	Default value	Description
cpu[*n*].crypto_aes	int	0x0-0x2	0x2	AES hash level. 0 = AES-128, 1 = AES-192, 2 = AES-256.
cpu[*n*].crypto_sha1	int	0x0-0x1	0x1	Enable SHA1.
cpu[*n*].crypto_sha256	int	0x0-0x1	0x1	Enable SHA256.

ARMAEMv8AMPCT - GIC parameters

This section describes the parameters.

Table 5-27 ARMAEMv8AMPCT *Generic Interrupt Controller* (GIC) parameters

Parameter	Type	Allowed values	Default value	Description
dic-spi_count	int	0x0-0xE0	0x40	Number of *Shared Peripheral Interrupts* (SPIs) supported.
GICDISABLE	bool	true, false	true	Disable the GICv3 interface in each core. Leave enabled unless the platform contains a GICv3 distributor.
gicv3.BPR-min	int	0x0-0x3	0x2	Minimum value for GICC_BPR. Non-secure copy will be this value + 1.
gicv3.IIDR_base	int	0x0-0xFFFFFFFF	0x43B	Base value for calculating GICC_IIDR value.
gicv3.STATUSR-implemented	bool	true, false	true	If GICv3 core interface enabled, enable STATUS registers.
internal_vgic	bool	true, false	true	Enable VGIC peripheral. Enable unless a shared VGIC is present.
non_secure_vgic_alias_when_ns_only	int	0x0-0xFFFFFFFFFFFF	0x0	If no EL3 and no Secure state, the VGIC has a Secure alias. If this parameter is nonzero, the model forms a Non-secure alias from its value for the VGIC, aligned to 32KiB.

ARMAEMv8AMPCT - abort parameters

This section describes the parameters.

Table 5-28 ARMAEMv8AMPCT abort parameters

Parameter	Type	Allowed values	Default value	Description
abort_execution_from_device_memory	bool	true, false	false	Abort on execution from device memory.
ext_abort_normal_cacheable_read_is_sync	bool	true, false	true	Synchronous reporting of normal cacheable-read external aborts.
ext_abort_normal_noncacheable_read_is_sync	bool	true, false	true	Synchronous reporting of normal noncacheable-read external aborts.
ext_abort_device_read_is_sync	bool	true, false	true	Synchronous reporting of device read external aborts.
ext_abort_so_read_is_sync	bool	true, false	true	Synchronous reporting of strongly ordered read external aborts.
ext_abort_normal_cacheable_write_is_sync	bool	true, false	false	Synchronous reporting of normal cacheable write external aborts.
ext_abort_normal_noncacheable_write_is_sync	bool	true, false	false	Synchronous reporting of normal noncacheable write external aborts.
ext_abort_device_write_is_sync	bool	true, false	false	Synchronous reporting of device write external aborts.
ext_abort_so_write_is_sync	bool	true, false	true	Synchronous reporting of strongly ordered write external aborts.
ext_abort_ttw_cacheable_read_is_sync	bool	true, false	true	Synchronous reporting of TTW cacheable read external aborts.
ext_abort_ttw_noncacheable_read_is_sync	bool	true, false	true	Synchronous reporting of TTW noncacheable read external aborts.
ext_abort_prefetch_is_sync	bool	true, false	true	Synchronous reporting of instruction fetch external aborts.

Table 5-28 ARMAEMv8AMPCT abort parameters (continued)

Parameter	Type	Allowed values	Default value	Description
ext_abort_fill_data	int	-	0xFDFDFDFCFCFDFDFD	Returned data, if external aborts are asynchronous.
unpredictable_exclusive_abort_memtype	int	0x0-0x2	0x0	MMU abort if exclusive access is not supported. 0 = none, exclusives allowed in all memory, 1 = exclusives abort in Device memory, 2 = exclusives abort in any memory type other than WB inner cacheable.

ARMAEMv8AMPCT - debug architecture parameters

This section describes the parameters.

Table 5-29 ARMAEMv8AMPCT debug architecture parameters

Parameter	Type	Allowed values	Default value	Description
DBGPIDR	int	0x0- 0xFFFFFFFFFF	0x0	If zero, build a value for the *DeBuG Peripheral Identification Register* (DBGPIDR). If nonzero, override DBGPIDR with this value.
cpu[*n*].number-of-breakpoints	int	0x2-0x10	0x10	Number of breakpoints.
cpu[*n*].number-of-watchpoints	int	0x2-0x10	0x10	Number of watchpoints.
cpu[*n*].number-of-context-breakpoints	int	0x0-0x10	0x10	Number of context-aware breakpoints.
cpu[*n*].unpredictable_WPMASKANDBAS	int	0x0-0x3	0x1	Constrained unpredictable handling of watchpoints when mask and BAS fields specified. 0 = IGNOREMASK, 1 = IGNOREBAS, 2 = REPEATBAS8, 3 = REPEATBAS.
cpu[*n*].unpredictable_non-contigous_BAS	bool	true, false	true	Treat noncontiguous BAS field in watchpoint control register as all ones.
cpu[*n*].cti-number_of_triggers	int	0x0-0x8	0x8	Number of CTI event triggers.
cpu[*n*].cti-intack_mask	int	0x0-0xFF	0x1	Set bits mean the corresponding triggers need software acknowledgment through CTIINTACK. One bit per trigger.
watchpoint-log2secondary_restriction	int	0x0-0x3F	0x0	Log_2(secondary restriction of FAR/EDWAR) on watchpoint hit for load/store operations.

ARMAEMv8AMPCT - simulator parameters

This section describes the parameters.

Table 5-30 ARMAEMv8AMPCT simulator parameters

Parameter	Type	Default value	Description
`scheduler_mode`	`int`	`0x0-0x2`	Control instruction interleaving. `0x0` = default long quantum, `0x1` = low latency mode, short quantum and signal checking, `0x2` = lock-breaking mode, long quantum with additional context switches near load-exclusive instructions.
`cpu[n].max_code_cache`	`int`	`-`	Maximum cache size for code translations, in bytes.

ARMAEMv8AMPCT - semihosting parameters

This section describes the parameters.

Semihosting is a method of running your target software on the model to communicate with the host environment. The AEMs permit the target C library to access the I/O facilities of the host computer, such as the filesystem, keyboard input, and clock.

The semihosting parameters are repeated in groups for each core in the cluster, from `cpu0` onwards.

Table 5-31 ARMAEMv8AMPCT semihosting parameters

Parameter	Type	Allowed values	Default value	Description
`cpu[n].semihosting-ARM_SVC`	`int`	`0x0-0xFFFFFFFF`	`0x123456`	A32 SVC number for semihosted calls.
`cpu[n].semihosting-Thumb_SVC`	`int`	`0x0-0xFFFFFFFF`	`0xAB`	T32 SVC number for semihosted calls.
`cpu[n].semihosting-cmd_line`	`string`	`-`	`-`	Program name and arguments, `argc`, `argv`, for target programs using the semihosted C library.
`cpu[n].semihosting-cwd`	`string`	`-`	`-`	Virtual address of CWD.
`cpu[n].semihosting-enable`	`bool`	`true`, `false`	`true`	Enable semihosting of SVC instructions.
`cpu[n].semihosting-heap_base`	`int`	`-`	`0x00000000`	Virtual address of heap base.
`cpu[n].semihosting-heap_limit`	`int`	`-`	`0x0F000000`	Virtual address of top of heap.
`cpu[n].semihosting-stack_base`	`int`	`-`	`0x10000000`	Virtual address of base of descending stack.
`cpu[n].semihosting-stack_limit`	`int`	`-`	`0x0F000000`	Virtual address of stack limit.

ARMAEMv8AMPCT - boundary features and architectural checkers

Boundary features and architectural checkers are model capabilities that help your development and testing process by exposing latent problems in the target code.

Certain boundary features or architectural checkers, however, might have an adverse effect on the overall running speed of target code.

ARMAEMv8AMPCT - IMPLEMENTATION DEFINED features

Some aspects of the behavior of the processor are IMPLEMENTATION DEFINED in the ARM architecture, meaning that they can legally vary between different implementations.

Take care with code that uses these facilities if you intend to run it across multiple ARM implementations, because they might or might not be present.

5.5 VE - differences between VE and CoreTile hardware and models

This section describes features of the VE hardware that the models do not implement, or that have significant differences in implementation.

This section contains the following subsections:

5.5.1 VE - differences in memory maps

The model is based on the memory map of the hardware VE platform. ARM does not intend the model memory map to accurately represent a specific VE hardware revision. It is sufficiently complete and accurate to boot the same operating system images as the VE hardware.

In the memory map, memory regions that peripherals or memory do not explicitly occupy are unmapped. This omission includes regions that belong to an unimplemented peripheral, and areas that are documented as reserved. Accessing these regions from the host processor results in the model presenting a warning.

5.5.2 VE - differences in memory aliasing

The model implements address space aliasing of the DRAM. The same physical memory locations are visible at different addresses.

The lower 2GB of the DRAM are accessible at `0x00_80000000`. The full 8GB of DRAM are accessible at `0x08_00000000` and again at `0x80_00000000`.

You can configure memory aliasing with the `daughterboard.dram_alias` parameter.

Table 5-32 AEMv8-A simulator parameters

Parameter	Type	Default value	Description
`daughterboard.dram_alias`	`bool`	`true`	Alias the bottom 2GB region in upper memory.

5.5.3 VE - features not present in models

The system models do not implement some features of the hardware version of the motherboard.

- Two-wire serial bus interfaces.
- USB interfaces.
- PCI Express interfaces.
- Compact flash.
- *Digital Visual Interface* (DVI).
- Debug and test interfaces.
- *Dynamic Memory Controller* (DMC).
- *Static Memory Controller* (SMC).

5.5.4 VE - features partially implemented in models

The *Fixed Virtual Platforms* (FVPs) implement some of the Sound feature of the hardware VE motherboard.

For the Sound feature, the VE FVPs implement the PL041 AACI PrimeCell and the audio CODEC as in the VE hardware. However, it has a limited number of sample rates.

5.5.5 VE - restrictions on processor models

General restrictions apply to the *Fixed Virtual Platform* (FVP) implementations of ARM processors.

- The simulator does not model cycle timing. In aggregate, all instructions execute in one processor master clock cycle, except for Wait For Interrupt.
- Write buffers are not modeled, except in AEMs.
- Most aspects of TLB behavior are implemented in the models. In ARMv7 models, and later ones, the TLB memory attribute settings are used when stateful cache is enabled.
- No device-accurate MicroTLB is implemented.
- A single memory access port is implemented. The port combines accesses for instruction, data, DMA, and peripherals. Configuration of the peripheral port memory map register is ignored.
- All memory accesses are atomic and are performed in *Programmer's View* (PV) order. All transactions on the PVBus are a maximum of 64 bits wide. Unaligned accesses are always performed as byte transfers.
- Some instruction sequences are executed atomically, ahead of the component master clock, so that system time does advance during their execution. This change can affect sequential access of device registers where devices are expecting time to move on between each access.
- Interrupts are not taken at every instruction boundary.
- Integration and test registers are not implemented.
- Not all CP14 debug registers are implemented on all processors.
- Breakpoint types that the model supports directly are:
  - Single address unconditional instruction breakpoints.
  - Single address unconditional data breakpoints.
  - Unconditional instruction address range breakpoints.
- Pseudoregisters in the debugger support processor exception breakpoints. Setting an exception register to a nonzero value stops execution on entry to the associated exception vector.
- Performance counters are not implemented on all models.

The following restrictions apply to the FVP implementation of a Cortex-A9 MPCore cluster:

- The Cortex-A9MPCore cluster contains some memory-mapped peripherals. The FVP models them.
- The model cluster sees two 4GB address spaces, one as seen from Secure mode and one as seen from Normal mode. The address spaces contain zero-wait state memory and peripherals, but much of the space is unmapped.
- The RR bit in the SCTLR is ignored.
- The Power Control Register in the system control coprocessor is implemented but writing to it does not change the behavior of the model.
- The SCU is only partially modeled:
  - The SCU enable bit is ignored. The SCU is always enabled.
  - The SCU ignores the invalidate-all register.
  - A memory write followed by a read to refill from memory represents coherency operations, rather than using cache-to-cache transfers.
  - There is no address filtering within the SCU. The enable bit for this feature is ignored.

5.5.6 VE - differences in timing

Fixed Virtual Platforms (FVPs) allow software to run in a functionally accurate simulation. However, because of the balance of fast simulation speed and timing accuracy, in some situations the models might behave unexpectedly.

When code interacts with real world devices like timers and keyboards, data arrives in the modeled device in real-world, or wall-clock, time. However, simulation time can be running much faster than the wall clock. This difference means that a single keypress might be interpreted as several repeated key presses, or a single mouse click incorrectly becomes a double click.

The VE FVPs provide the **Rate Limit** feature to match simulation time to wall-clock time. Enabling the rate limit prevents the model from running faster than wall-clock time. Enable it with either the **Rate Limit** button in the CLCD display, or the `rate_limit-enable` model instantiation parameter. This

precaution avoids issues with two clocks running at different rates. For interactive applications, ARM recommends enabling the rate limit.

5.6 VE - Architecture Message plug-in - parameters

This section describes the parameters.

Note

To use these parameters, load the ArchMsgTrace plug-in into a model.

`TRACE.ArchMsg.`*`parameter=value`*

Table 5-33 Architecture Message plug-in error and warning message parameters

Parameter	Type	Allowed values	Default value	Description
`suppress_repeated`	`bool`	`true`, `false`	`true`	Suppress repeated messages from similar call sites.
`suppress_sources`	`string`	`-`	`-`	Space-separated list of components or events to not print.
`trace-file`	`string`	`-`	`-`	ArchMsg output file.