Imperas Guide to using Virtual Platforms

Platform / Module Specific Information for xilinx.ovpworld.org / Zynq_PS

Imperas Software Limited

Imperas Buildings, North Weston
Thame, Oxfordshire, OX9 2HA, U.K.
docs@imperas.com.

Author	Imperas Software Limited
Version	20211118.0
Filename	Imperas_Platform_User_Guide_Zynq_PS.pdf
Created	31 December 2021
Status	OVP Standard Release

Model Release Status

This model is released as part of OVP releases and is included in OVPworld packages. Please visit OVPworld.org.

Table Of Contents

1.0 Platform / Module: Zynq_PS

This document provides the details of the usage of an Imperas OVP Virtual Platform / Module. The first half of the document covers specifics of this particular component. For more information about Imperas OVP virtual platforms, how they are built and used, please see the later sections in this document.

1.1 Virtual Platform / Module Type

Hardware described using OVP can either be a platform, module, processor, or peripheral.
This hardware component is described as being a module. A module is a component that is used in other modules, platforms, or test harnesses. It is normally used to encapsulate a layer in a hierarchical system.

1.2 Licensing

Open Source Apache 2.0

1.3 Description

This module implements the Zynq 7000 Processing Sub-System (PS).
The PS integrates two ARM Cortex-A9 MPCore application processors, memories and peripherals.

1.4 Limitations

This module provides the peripheral behavior required to boot a Linux Kernel or XtratuM hypervisor.

Some of the peripherals are register-only, non-functional models. See the individual peripheral model documentation for details.

Snoop Control Unit is not implemented, Trust Zone memory protection is not implemented

1.5 Reference

ZC702 Board user Guide UG850 (v1.5) September 4,2015 (ug850-zc702-eval-bd.pdf) and
ZC706 Evaluation Board for the Zynq-7000 XC7Z045 All Programmable SoC (v1.6) March 29, 2016 (ug954-zc706-eval-bd-xc7z04-ap-soc.pdf) and
Zynq-7000 AP SoC Technical Reference Manual UG585 (v1.10) February 23, 2015 (https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

1.6 Description

Peripheral can0 (xilinx.ovpworld.org/peripheral/zynq_7000-can/1.0) is a dummy peripheral that implements register set only

Peripheral can1 (xilinx.ovpworld.org/peripheral/zynq_7000-can/1.0) is a dummy peripheral that implements register set only

Peripheral dummySMC is a dummy which traps accesses at address 0xe000e000 size 0x1000

Peripheral dummyUnknown1 is a dummy which traps accesses at address 0xe000f000 size 0x1000

Peripheral sdio0 (xilinx.ovpworld.org/peripheral/zynq_7000-sdio/1.0) is a dummy peripheral that implements register set only

Peripheral sdio1 (xilinx.ovpworld.org/peripheral/zynq_7000-sdio/1.0) is a dummy peripheral that implements register set only

Peripheral trustzone_security (xilinx.ovpworld.org/peripheral/zone_security/1.0) is a dummy peripheral that implements register set only

Peripheral DMAC (xilinx.ovpworld.org/peripheral/zynq_7000-dmac) is a dummy peripheral that implements register set only

Peripheral DDRC (xilinx.ovpworld.org/peripheral/zynq_7000-ddrc) is a dummy peripheral that implements register set only

Peripheral devcfg (xilinx.ovpworld.org/peripheral/zynq_7000-devcfg/1.0) provides XADC access for power monitoring. Connected memory contains the temperature, voltage and current values provided by the XADC

Peripheral dummyAXI_HP0 is a dummy which traps accesses at address 0xf8008000 size 0x1000

Peripheral dummyAXI_HP1 is a dummy which traps accesses at address 0xf8009000 size 0x1000

Peripheral dummyAXI_HP2 is a dummy which traps accesses at address 0xf800a000 size 0x1000

Peripheral dummyAXI_HP3 is a dummy which traps accesses at address 0xf800b000 size 0x1000

Peripheral dummyeFuse is a dummy which traps accesses at address 0xf800d000 size 0x1000

Peripheral dummyUnknown2 is a dummy which traps accesses at address 0xf800e000 size 0x1000

Peripheral trustzone_GPVsecurity (xilinx.ovpworld.org/peripheral/zynq_7000-tz_GPVsecurity/1.0) is a dummy peripheral that implements register set only

Peripheral GPV_qos301_cpu (xilinx.ovpworld.org/peripheral/zynq_7000-qos301/1.0) is a dummy peripheral that implements register set only

Peripheral GPV_qos301_dmac (xilinx.ovpworld.org/peripheral/zynq_7000-qos301/1.0) is a dummy peripheral that implements register set only

Peripheral GPV_qos301_iou (xilinx.ovpworld.org/peripheral/zynq_7000-qos301/1.0) is a dummy peripheral that implements register set only

1.7 Location

The Zynq_PS virtual platform / module is located in an Imperas/OVP installation at the VLNV: xilinx.ovpworld.org / module / Zynq_PS / 1.0.

2.0 External Ports for Module Zynq_PS

Table 1. External Ports

Port Type	Port Name	Internal Connection
busport	extPort	extPortBus
busport	extPortI2C	i2cBus
netport	irqf2p0_inP	irqf2p0
netport	irqf2p1_inP	irqf2p1
netport	irqf2p2_inP	irqf2p2
netport	irqf2p3_inP	irqf2p3
netport	irqf2p4_inP	irqf2p4
netport	irqf2p5_inP	irqf2p5
netport	irqf2p6_inP	irqf2p6
netport	irqf2p7_inP	irqf2p7
netport	irqf2p8_inP	irqf2p8
netport	irqf2p9_inP	irqf2p9
netport	irqf2p10_inP	irqf2p10
netport	irqf2p11_inP	irqf2p11
netport	irqf2p12_inP	irqf2p12
netport	irqf2p13_inP	irqf2p13
netport	irqf2p14_inP	irqf2p14
netport	irqf2p15_inP	irqf2p15
netport	irqf2p16_inP	irqf2p16
netport	irqf2p17_inP	irqf2p17
netport	irqf2p18_inP	irqf2p18
netport	irqf2p19_inP	irqf2p19
netport	irqp2f0_outP	can1_spi83
netport	irqp2f1_outP	uart1_spi82
netport	irqp2f2_outP	spi1_spi81
netport	irqp2f3_outP	i2c1_spi80
netport	irqp2f4_outP	sdio1_spi79
netport	irqp2f5_outP	eth1wu_spi78
netport	irqp2f6_outP	eth1_spi77
netport	irqp2f7_outP	usb1_spi76
netport	irqp2f8_outP	can0_spi60
netport	irqp2f9_outP	uart0_spi59
netport	irqp2f10_outP	spi0_spi58
netport	irqp2f11_outP	i2c0_spi57
netport	irqp2f12_outP	sdio0_spi56

netport	irqp2f13_outP	eth0wu_spi55
netport	irqp2f14_outP	eth0_spi54
netport	irqp2f15_outP	usb0_spi53
netport	irqp2f16_outP	gpio_spi52
netport	irqp2f17_outP	irqp2f17
netport	irqp2f18_outP	qspi_spi51
netport	irqp2f19_outP	smc_spi50
netport	irqp2f20_outP	dmac0_spi46
netport	irqp2f21_outP	dmac1_spi47
netport	irqp2f22_outP	dmac2_spi48
netport	irqp2f23_outP	dmac3_spi49
netport	irqp2f24_outP	dmac4_spi72
netport	irqp2f25_outP	dmac5_spi73
netport	irqp2f26_outP	dmac6_spi74
netport	irqp2f27_outP	dmac7_spi75
netport	irqp2f28_outP	dmaca_spi45
netport	extPortXADCMux	xadcmux
netport	gpio_bank0_outP	gpio_bank0_out
netport	gpio_bank0_inP	gpio_bank0_in
netport	gpio_bank1_outP	gpio_bank1_out
netport	gpio_bank1_inP	gpio_bank1_in
netport	gpio_bank2_outP	gpio_bank2_out
netport	gpio_bank2_oen_outP	gpio_bank2_oen_out
netport	gpio_bank2_inP	gpio_bank2_in
netport	gpio_bank3_outP	gpio_bank3_out
netport	gpio_bank3_oen_outP	gpio_bank3_oen_out
netport	gpio_bank3_inP	gpio_bank3_in

3.0 Formal Parameters declared for Module Zynq_PS

Table 2. Formal Parameters

Name	Type	Min	Max	Default
armmips	Uns32			
flashtype	string			micron
psclock	Uns32			
board	String			

4.0 Processor [arm.ovpworld.org/processor/arm/1.0] instance: cpu

4.1 Processor model type: 'arm' variant 'Cortex-A9MPx2' definition

Imperas OVP processor models support multiple variants and details of the variants implemented in this model can be found in:

- the Imperas installation located at ImperasLib/source/arm.ovpworld.org/processor/arm/1.0/doc
- the OVP website: OVP_Model_Specific_Information_arm_Cortex-A9MPx2.pdf

4.1.1 Description

ARM Processor Model

4.1.2 Licensing

Usage of binary model under license governing simulator usage.
Note that for models of ARM CPUs the license includes the following terms:
Licensee is granted a non-exclusive, worldwide, non-transferable, revocable licence to:
If no source is being provided to the Licensee: use and copy only (no modifications rights are granted) the model for the sole purpose of designing, developing, analyzing, debugging, testing, verifying, validating and optimizing software which: (a) (i) is for ARM based systems; and (ii) does not incorporate the ARM Models or any part thereof; and (b) such ARM Models may not be used to emulate an ARM based system to run application software in a production or live environment.
If source code is being provided to the Licensee: use, copy and modify the model for the sole purpose of designing, developing, analyzing, debugging, testing, verifying, validating and optimizing software which: (a) (i) is for ARM based systems; and (ii) does not incorporate the ARM Models or any part thereof; and (b) such ARM Models may not be used to emulate an ARM based system to run application software in a production or live environment.
In the case of any Licensee who is either or both an academic or educational institution the purposes shall be limited to internal use.
Except to the extent that such activity is permitted by applicable law, Licensee shall not reverse engineer, decompile, or disassemble this model. If this model was provided to Licensee in Europe, Licensee shall not reverse engineer, decompile or disassemble the Model for the purposes of error correction.
The License agreement does not entitle Licensee to manufacture in silicon any product based on this model.
The License agreement does not entitle Licensee to use this model for evaluating the validity of any ARM patent.
Source of model available under separate Imperas Software License Agreement.

4.1.3 Limitations

Instruction pipelines are not modeled in any way. All instructions are assumed to complete immediately. This means that instruction barrier instructions (e.g. ISB, CP15ISB) are treated as NOPs, with the exception of any undefined instruction behavior, which is modeled. The model does not implement speculative fetch behavior. The branch cache is not modeled.
Caches and write buffers are not modeled in any way. All loads, fetches and stores complete immediately and in order, and are fully synchronous (as if the memory was of Strongly Ordered or Device-nGnRnE type). Data barrier instructions (e.g. DSB, CP15DSB) are treated as NOPs, with the exception of any undefined instruction behavior, which is modeled. Cache manipulation instructions are implemented as NOPs, with the exception of any undefined instruction behavior, which is modeled.
Real-world timing effects are not modeled: all instructions are assumed to complete in a single cycle.
Performance Monitors are implemented as a register interface only except for the cycle counter, which is implemented assuming one instruction per cycle.
TLBs are architecturally-accurate but not device accurate. This means that all TLB maintenance and address translation operations are fully implemented but the cache is larger than in the real device.

4.1.4 Verification

Models have been extensively tested by Imperas. ARM Cortex-A models have been successfully used by customers to simulate SMP Linux, Ubuntu Desktop, VxWorks and ThreadX on Xilinx Zynq virtual platforms.

4.1.5 Core Features

Thumb-2 instructions are supported.
Trivial Jazelle extension is implemented.

4.1.6 Memory System

Security extensions are implemented (also known as TrustZone). Non-secure accesses can be made visible externally by connecting the processor to a 41-bit physical bus, in which case bits 39..0 give the true physical address and bit 40 is the NS bit.
VMSA secure and non-secure address translation is implemented.
TLB behavior is controlled by parameter ASIDCacheSize. If this parameter is 0, then an unlimited number of TLB entries will be maintained concurrently. If this parameter is non-zero, then only TLB entries for up to ASIDCacheSize different ASIDs will be maintained concurrently initially; as new ASIDs are used, TLB entries for less-recently used ASIDs are deleted, which improves model performance in some cases (especially when 16-bit ASIDs are in use). If the model detects that the TLB entry cache is too small (entry ejections are very frequent), it will increase the cache size automatically. In this variant, ASIDCacheSize is 8

4.1.7 Advanced SIMD and Floating-Point Features

SIMD and VFP instructions are implemented.
The model implements trapped exceptions if FPTrap is set to 1 in MVFR0 (for AArch32) or MVFR0_EL1 (for AArch64). When floating point exception traps are taken, cumulative exception flags are not updated (in other words, cumulative flag state is always the same as prior to instruction execution, even for SIMD instructions). When multiple enabled exceptions are raised by a single floating point operation, the exception reported is the one in least-significant bit position in FPSCR (for AArch32) or FPCR (for AArch64). When multiple enabled exceptions are raised by different SIMD element computations, the exception reported is selected from the lowest-index-number SIMD operation. Contact Imperas if requirements for exception reporting differ from these.
Trapped exceptions not are implemented in this variant (FPTrap=0)

4.1.8 Generic Interrupt Controller

GIC block is implemented (GICv1, including security extensions). Accesses to GIC registers can be viewed externally by connecting to the 32-bit GICRegisters bus port. Secure register accesses are at offset 0x0 on this bus; for example, a secure access to GIC register ICDDCR can be observed by monitoring address 0x00001000. Non-secure accesses are at offset 0x80000000 on this bus; for example, a non-secure access to GIC register ICDDCR can be observed by monitoring address 0x80001000

4.1.9 Debug Mask

It is possible to enable model debug features in various categories. This can be done statically using the "override_debugMask" parameter, or dynamically using the "debugflags" command. Enabled debug

features are specified using a bitmask value, as follows:
Value 0x004: enable debugging of MMU/MPU mappings.
Value 0x020: enable debugging of reads and writes of GIC block registers.
Value 0x040: enable debugging of exception routing via the GIC model component.
Value 0x080: enable debugging of all system register accesses.
Value 0x100: enable debugging of all traps of system register accesses.
Value 0x200: enable verbose debugging of other miscellaneous behavior (for example, the reason why a particular instruction is undefined).
Value 0x400: enable debugging of Performance Monitor timers
Value 0x800: enable dynamic validation of TLB entries against in-memory page table contents (finds some classes of error where page table entries are updated without a subsequent flush of affected TLB entries).
All other bits in the debug bitmask are reserved and must not be set to non-zero values.

4.1.10 AArch32 Unpredictable Behavior

Many AArch32 instruction behaviors are described in the ARM ARM as CONSTRAINED UNPREDICTABLE. This section describes how such situations are handled by this model.

4.1.11 Equal Target Registers

Some instructions allow the specification of two target registers (for example, double-width SMULL, or some VMOV variants), and such instructions are CONSTRAINED UNPREDICTABLE if the same target register is specified in both positions. In this model, such instructions are treated as UNDEFINED.

4.1.12 Floating Point Load/Store Multiple Lists

Instructions that load or store a list of floating point registers (e.g. VSTM, VLDM, VPUSH, VPOP) are CONSTRAINED UNPREDICTABLE if either the uppermost register in the specified range is greater than 32 or (for 64-bit registers) if more than 16 registers are specified. In this model, such instructions are treated as UNDEFINED.

4.1.13 Floating Point VLD[2-4]/VST[2-4] Range Overflow

Instructions that load or store a fixed number of floating point registers (e.g. VST2, VLD2) are CONSTRAINED UNPREDICTABLE if the upper register bound exceeds the number of implemented floating point registers. In this model, these instructions load and store using modulo 32 indexing (consistent with AArch64 instructions with similar behavior).

4.1.14 If-Then (IT) Block Constraints

Where the behavior of an instruction in an if-then (IT) block is described as CONSTRAINED UNPREDICTABLE, this model treats that instruction as UNDEFINED.

4.1.15 Use of R13

In architecture variants before ARMv8, use of R13 was described as CONSTRAINED UNPREDICTABLE in many circumstances. From ARMv8, most of these situations are no longer considered unpredictable. This model allows R13 to be used like any other GPR, consistent with the ARMv8 specification.

4.1.16 Use of R15

Use of R15 is described as CONSTRAINED UNPREDICTABLE in many circumstances. This model allows such use to be configured using the parameter "unpredictableR15" as follows:
Value "undefined": any reference to R15 in such a situation is treated as UNDEFINED;
Value "nop": any reference to R15 in such a situation causes the instruction to be treated as a NOP;
Value "raz_wi": any reference to R15 in such a situation causes the instruction to be treated as a RAZ/WI (that is, R15 is read as zero and write-ignored);
Value "execute": any reference to R15 in such a situation is executed using the current value of R15 on read, and writes to R15 are allowed (but are not interworking).
Value "assert": any reference to R15 in such a situation causes the simulation to halt with an assertion message (allowing any such unpredictable uses to be easily identified).
In this variant, the default value of "unpredictableR15" is "undefined".

4.1.17 Unpredictable Instructions in Some Modes

Some instructions are described as CONSTRAINED UNPREDICTABLE in some modes only (for example, MSR accessing SPSR is CONSTRAINED UNPREDICTABLE in User and System modes). This model allows such use to be configured using the parameter "unpredictableModal", which can have values "undefined" or "nop". See the previous section for more information about the meaning of these values.
In this variant, the default value of "unpredictableModal" is "nop".

4.1.18 Integration Support

This model implements a number of non-architectural pseudo-registers and other features to facilitate integration.

4.1.19 Memory Transaction Query

Two registers are intended for use within memory callback functions to provide additional information about the current memory access. Register transactPL indicates the processor execution level of the current access (0-3). Note that for load/store translate instructions (e.g. LDRT, STRT) the reported execution level will be 0, indicating an EL0 access. Register transactAT indicates the type of memory access: 0 for a normal read or write; and 1 for a physical access resulting from a page table walk.

4.1.20 Page Table Walk Query

A banked set of registers provides information about the most recently completed page table walk. There are up to six banks of registers: bank 0 is for stage 1 walks, bank 1 is for stage 2 walks, and banks 2-5 are for stage 2 walks initiated by stage 1 level 0-3 entry lookups, respectively. Banks 1-5 are present only for processors with virtualization extensions. The currently active bank can be set using register PTWBankSelect. Register PTWBankValid is a bitmask indicating which banks contain valid data: for example, the value 0xb indicates that banks 0, 1 and 3 contain valid data.
Within each bank, there are registers that record addresses and values read during that page table walk. Register PTWBase records the table base address, register PTWInput contains the input address that starts a walk, register PTWOutput contains the result address and register PTWPgSize contains the page size (PTWOutput and PTWPgSize are valid only if the page table walk completes). Registers PTWAddressL0-PTWAddressL3 record the addresses of level 0 to level 3 entries read, respectively.

Register PTWAddressValid is a bitmask indicating which address registers contain valid data: bits 0-3 indicate PTWAddressL0-PTWAddressL3, respectively, bit 4 indicates PTWBase, bit 5 indicates PTWInput, bit 6 indicates both PTWOutput and PTWPgSize. For example, the value 0x73 indicates that PTWBase, PTWInput, PTWOutput, PTWPgSize and PTWAddressL0-L1 are valid but PTWAddressL2-L3 are not. Register PTWAddressNS is a bitmask indicating whether an address is in non-secure memory: bits 0-3 indicate PTWAddressL0-PTWAddressL3, respectively, bit 4 indicates PTWBase, bit 6 indicates PTWOutput (PTWInput is a VA and thus has no secure/non-secure info). Registers PTWValueL0-PTWValueL3 contain page table entry values read at level 0 to level 3. Register PTWValueValid is a bitmask indicating which value registers contain valid data: bits 0-3 indicate PTWValueL0-PTWValueL3, respectively.

4.1.21 Artifact Page Table Walks

Registers are also available to enable a simulation environment to initiate an artifact page table walk (for example, to determine the ultimate PA corresponding to a given VA). Register PTWI_EL1S initiates a secure EL1 table walk for a fetch. Register PTWD_EL1S initiates a secure EL1 table walk for a load or store (note that current ARM processors have unified TLBs, so these registers are synonymous). Registers PTW[ID]_EL1NS initiate walks for non-secure EL1 accesses. Registers PTW[ID]_EL2 initiate EL2 walks. Registers PTW[ID]_S2 initiate stage 2 walks. Registers PTW[ID]_EL3 initiate AArch64 EL3 walks. Finally, registers PTW[ID]_current initiate current-mode walks (useful in a memory callback context). Each walk fills the query registers described above.

4.1.22 MMU and Page Table Walk Events

Two events are available that allow a simulation environment to be notified on MMU and page table walk actions. Event mmuEnable triggers when any MMU is enabled or disabled. Event pageTableWalk triggers on completion of any page table walk (including artifact walks).

4.1.23 Artifact Address Translations

A simulation environment can trigger an artifact address translation operation by writing to the architectural address translation registers (e.g. ATS1CPR). The results of such translations are written to an integration support register artifactPAR, instead of the architectural PAR register. This means that such artifact writes will not perturb architectural state.

4.1.24 TLB Invalidation

A simulation environment can cause TLB state for one or more address translation regimes in the processor to be flushed by writing to the artifact register ResetTLBs. The argument is a bitmask value, in which non-zero bits select the TLBs to be flushed, as follows:
Bit 0: EL0/EL1 stage 1 secure TLB
Bit 1: EL0/EL1 stage 1 non-secure TLB

4.1.25 Halt Reason Introspection

An artifact register HaltReason can be read to determine the reason or reasons that a processor is halted. This register is a bitfield, with the following encoding: bit 0 indicates the processor has executed a wait-for-event (WFE) instruction; bit 1 indicates the processor has executed a wait-for-interrupt (WFI) instruction;

and bit 2 indicates the processor is held in reset.

4.1.26 System Register Access Monitor

If parameter "enableSystemMonitorBus" is True, an artifact 32-bit bus "SystemMonitor" is enabled for each PE. Every system register read or write by that PE is then visible as a read or write on this artifact bus, and can therefore be monitored using callbacks installed in the client environment (use opBusReadMonitorAdd/opBusWriteMonitorAdd or icmAddBusReadCallback/icmAddBusWriteCallback, depending on the client API). The format of the address on the bus is as follows:
bits 31:26 - zero
bit 25 - 1 if AArch64 access, 0 if AArch32 access
bit 24 - 1 if non-secure access, 0 if secure access
bits 23:20 - CRm value
bits 19:16 - CRn value
bits 15:12 - op2 value
bits 11:8 - op1 value
bits 7:4 - op0 value (AArch64) or coprocessor number (AArch32)
bits 3:0 - zero
As an example, to view non-secure writes to writes to CNTFRQ_EL0 in AArch64 state, install a write monitor on address range 0x020e0330:0x020e0333.

4.1.27 System Register Implementation

If parameter "enableSystemBus" is True, an artifact 32-bit bus "System" is enabled for each PE. Slave callbacks installed on this bus can be used to implement modified system register behavior (use opBusSlaveNew or icmMapExternalMemory, depending on the client API). The format of the address on the bus is the same as for the system monitor bus, described above.

4.2 Instance Parameters

Several parameters can be specified when a processor is instanced in a platform. For this processor instance 'cpu' it has been instanced with the following parameters:

Table 3. Processor Instance 'cpu' Parameters (Configurations)

Parameter	Value	Description
endian	little	Select processor endian (big or little)
simulateexceptions	simulateexceptions	Causes the processor simulate exceptions instead of halting
mips	armmips	The nominal MIPS for the processor

Table 4. Processor Instance 'cpu' Parameters (Attributes)

Parameter Name	Value	Type
variant	Cortex-A9MPx2	enum
compatibility	ISA	enum
UAL	1	boolean
override_CBAR	0xf8f00000	Uns32
override_MIDR	0x413fc090	Uns32

4.3 Memory Map for processor 'cpu' bus: 'pBus'

Processor instance 'cpu' is connected to bus 'pBus' using master port 'INSTRUCTION'.
Processor instance 'cpu' is connected to bus 'pBus' using master port 'DATA'.

Table 5. Memory Map ('cpu' / 'pBus' [width: 32])

Lo Address	Hi Address	Instance	Component
0x0	0xFFFFF	brPtoDDR	bridge
remappable	remappable	dummyAXI_HP0	trap
remappable	remappable	dummyAXI_HP1	trap
remappable	remappable	dummyAXI_HP2	trap
remappable	remappable	dummyAXI_HP3	trap
remappable	remappable	dummyUnknown2	trap
remappable	remappable	dummyeFuse	trap
remappable	remappable	slcr	zynq_7000-slcr
0x100000	0x3FFFFFFF	brPtoDDRS	bridge
0x40000000	0xBFFFFFFF	pBustoExtPort	bridge
0xE0000000	0xE0101FFF	pBustoAPB	bridge
0xE0200000	0xE020001F	trustzone_security	zynq_7000-tz_security
0xF8000000	0xF8000BFF	slcr	zynq_7000-slcr
0xF8001000	0xF8001FFF	ttc0	zynq_7000-ttc
0xF8002000	0xF8002FFF	ttc1	zynq_7000-ttc
0xF8003000	0xF8003FFF	DMAC	zynq_7000-dmac
0xF8004000	0xF8004FFF	DMAC	zynq_7000-dmac
0xF8005000	0xF8005FFF	swdt	zynq_7000-swdt
0xF8006000	0xF8006FFF	DDRC	zynq_7000-ddrc
0xF8007000	0xF8007FFF	devcfg	zynq_7000-devcfg
0xF800C000	0xF800CFFF	OCM	zynq_7000-ocm
0xF8900000	0xF890003F	trustzone_GPVsecurity	zynq_7000-tz_GPVsecurity
0xF8946000	0xF8946FFF	GPV_qos301_cpu	zynq_7000-qos301
0xF8947000	0xF8947FFF	GPV_qos301_dmac	zynq_7000-qos301
0xF8948000	0xF8948FFF	GPV_qos301_iou	zynq_7000-qos301
0xF8F02000	0xF8F02FFF	l2cache	L2CachePL310

Table 6. Bridged Memory Map ('cpu' / 'brPtoDDR' / 'ddrBus' [width: 32])

Lo Address	Hi Address	Instance	Component
0x0	0x3FFFF	DDR0	ram
0x40000	0x7FFFF	DDR1	ram
0x80000	0xFFFFF	DDR2	ram

Table 7. Bridged Memory Map ('cpu' / 'brPtoDDRS' / 'ddrSBus' [width: 32])

Lo Address	Hi Address	Instance	Component
0x100000	0x3FFFFFFF	DDR3	ram

Table 8. Bridged Memory Map ('cpu' / 'pBustoExtPort' / 'extPortBus' [width: 32])

Lo Address	Hi Address	Instance	Component

Table 9. Bridged Memory Map ('cpu' / 'pBustoAPB' / 'apbBus' [width: 32])

Lo Address	Hi Address	Instance	Component
remappable	remappable	dummySMC	trap
remappable	remappable	dummyUnknown1	trap
0xE0000000	0xE0000FFF	uart0	uart
0xE0001000	0xE0001FFF	uart1	uart
0xE0002000	0xE0002FFF	usb0	zynq_7000-usb
0xE0003000	0xE0003FFF	usb1	zynq_7000-usb
0xE0004000	0xE0004FFF	i2c0	zynq_7000-iic
0xE0005000	0xE0005FFF	i2c1	zynq_7000-iic
0xE0006000	0xE0006FFF	spi0	zynq_7000-spi
0xE0007000	0xE0007FFF	spi1	zynq_7000-spi
0xE0008000	0xE0008FFF	can0	zynq_7000-can
0xE0009000	0xE0009FFF	can1	zynq_7000-can
0xE000A000	0xE000AFFF	GPIO	zynq_7000-gpio
0xE000B000	0xE000BFFF	eth0	gem
0xE000C000	0xE000CFFF	eth1	gem
0xE000D000	0xE000DFFF	qspi	zynq_7000-qspi
0xE0100000	0xE0100FFF	sdio0	zynq_7000-sdio
0xE0101000	0xE0101FFF	sdio1	zynq_7000-sdio

4.4 Net Connections to processor: 'cpu'

Table 10. Processor Net Connections ('cpu')

Net Port	Net	Instance	Component
SPI34	l2cache_spi34	l2cache	L2CachePL310
SPI41	swdt_spi41	swdt	zynq_7000-swdt
SPI42	ttc0_spi42	ttc0	zynq_7000-ttc
SPI43	ttc0_spi43	ttc0	zynq_7000-ttc
SPI44	ttc0_spi44	ttc0	zynq_7000-ttc
SPI45	dmaca_spi45		unknown
SPI45	dmaca_spi45	DMAC	zynq_7000-dmac
SPI46	dmac0_spi46		unknown
SPI47	dmac1_spi47		unknown
SPI48	dmac2_spi48		unknown
SPI49	dmac3_spi49		unknown
SPI50	smc_spi50		unknown
SPI51	qspi_spi51		unknown
SPI51	qspi_spi51	qspi	zynq_7000-qspi
SPI52	gpio_spi52		unknown
SPI52	gpio_spi52	GPIO	zynq_7000-gpio
SPI53	usb0_spi53		unknown
SPI54	eth0_spi54		unknown
SPI54	eth0_spi54	eth0	gem

SPI55	eth0wu_spi55		unknown
SPI56	sdio0_spi56		unknown
SPI56	sdio0_spi56	sdio0	zynq_7000-sdio
SPI57	i2c0_spi57		unknown
SPI57	i2c0_spi57	i2c0	zynq_7000-iic
SPI58	spi0_spi58		unknown
SPI58	spi0_spi58	spi0	zynq_7000-spi
SPI59	uart0_spi59		unknown
SPI59	uart0_spi59	uart0	uart
SPI60	can0_spi60		unknown
SPI60	can0_spi60	can0	zynq_7000-can
SPI69	ttc1_spi69	ttc1	zynq_7000-ttc
SPI70	ttc1_spi70	ttc1	zynq_7000-ttc
SPI71	ttc1_spi71	ttc1	zynq_7000-ttc
SPI72	dmac4_spi72		unknown
SPI73	dmac5_spi73		unknown
SPI74	dmac6_spi74		unknown
SPI75	dmac7_spi75		unknown
SPI76	usb1_spi76		unknown
SPI77	eth1_spi77		unknown
SPI77	eth1_spi77	eth1	gem
SPI78	eth1wu_spi78		unknown
SPI79	sdio1_spi79		unknown
SPI79	sdio1_spi79	sdio1	zynq_7000-sdio
SPI80	i2c1_spi80		unknown
SPI80	i2c1_spi80	i2c1	zynq_7000-iic
SPI81	spi1_spi81		unknown
SPI81	spi1_spi81	spi1	zynq_7000-spi
SPI82	uart1_spi82		unknown
SPI82	uart1_spi82	uart1	uart
SPI83	can1_spi83		unknown
SPI83	can1_spi83	can1	zynq_7000-can
SPI61	irqf2p0		unknown
SPI62	irqf2p1		unknown
SPI63	irqf2p2		unknown
SPI64	irqf2p3		unknown
SPI65	irqf2p4		unknown
SPI66	irqf2p5		unknown
SPI67	irqf2p6		unknown
SPI68	irqf2p7		unknown
SPI84	irqf2p8		unknown
SPI85	irqf2p9		unknown
SPI86	irqf2p10		unknown
SPI87	irqf2p11		unknown
SPI88	irqf2p12		unknown
SPI89	irqf2p13		unknown
SPI90	irqf2p14		unknown
SPI91	irqf2p15		unknown

irq_CPU0	irqf2p16		unknown
fiq_CPU0	irqf2p17		unknown
irq_CPU1	irqf2p18		unknown
fiq_CPU1	irqf2p19		unknown
reset_CPU0	reset_A9_CPU0	slcr	zynq_7000-slcr
reset_CPU1	reset_A9_CPU1	slcr	zynq_7000-slcr

5.0 Peripheral Instances

5.1 Peripheral [cadence.ovpworld.org/peripheral/uart/1.0] instance: uart0

5.1.1 Description

Cadence UART (Xilinx Zync Platform)

5.1.2 Licensing

Open Source Apache 2.0

5.1.3 Limitations

This is an incomplete model of the Cadence UART (uartps) as used on the Xilinx Zync devices.
It has basic functionality to support Linux boot on the Xilinx Zync platform.
There is no modeling of physical aspects of the UART, such as baud rates etc.

5.1.4 Reference

Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

Table 11. Configuration options (attributes) set for instance 'uart0'

Attribute	Value	Type	Expression
outfile	uart0.log	string	
finishOnDisconnect	1	boolean	
console	1	boolean	
xchars	130	uns32	
ychars	30	uns32	

5.2 Peripheral [cadence.ovpworld.org/peripheral/uart/1.0] instance: uart1

5.2.1 Description

Cadence UART (Xilinx Zync Platform)

5.2.2 Licensing

Open Source Apache 2.0

5.2.3 Limitations

This is an incomplete model of the Cadence UART (uartps) as used on the Xilinx Zync devices.

It has basic functionality to support Linux boot on the Xilinx Zync platform.
There is no modeling of physical aspects of the UART, such as baud rates etc.

5.2.4 Reference

Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

Table 12. Configuration options (attributes) set for instance 'uart1'

Attribute	Value	Type	Expression
outfile	uart1.log	string	
finishOnDisconnect	1	boolean	
console	1	boolean	
xchars	130	uns32	
ychars	30	uns32	

5.3 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-usb/1.0] instance: usb0

5.3.1 Description

Zynq 7000 USB Registers

5.3.2 Licensing

Open Source Apache 2.0

5.3.3 Limitations

This model implements the full set of registers but no behavior.

5.3.4 Reference

Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

There are no configuration options set for this peripheral instance.

5.4 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-usb/1.0] instance: usb1

5.4.1 Description

Zynq 7000 USB Registers

5.4.2 Licensing

Open Source Apache 2.0

5.4.3 Limitations

This model implements the full set of registers but no behavior.

5.4.4 Reference

Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

There are no configuration options set for this peripheral instance.

5.5 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-iic/1.0] instance: i2c0

5.5.1 Description

Zynq 7000 I2C Registers. This model also includes the behaviour for PCA9548 I2C Bus Switch

5.5.2 Licensing

Open Source Apache 2.0

5.5.3 Limitations

This model implements the full set of registers and behaviour to read and write the I2C address space.

5.5.4 Reference

Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf).
Evaluation Board ZC706 (ug954-zc706-eval-board-xc7z045-ap-soc.pdf)
Evaluation Board ZC702 (ug850-zc702-eval-board.pdf)

There are no configuration options set for this peripheral instance.

5.6 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-iic/1.0] instance: i2c1

5.6.1 Description

Zynq 7000 I2C Registers. This model also includes the behaviour for PCA9548 I2C Bus Switch

5.6.2 Licensing

Open Source Apache 2.0

5.6.3 Limitations

This model implements the full set of registers and behaviour to read and write the I2C address space.

5.6.4 Reference

Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf).
Evaluation Board ZC706 (ug954-zc706-eval-board-xc7z045-ap-soc.pdf)
Evaluation Board ZC702 (ug850-zc702-eval-board.pdf)

There are no configuration options set for this peripheral instance.

5.7 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-spi/1.0] instance: spi0

5.7.1 Description

Zynq 7000 SPI Registers

5.7.2 Licensing

Open Source Apache 2.0

5.7.3 Limitations

This model implements the full set of registers but no behavior.

5.7.4 Reference

Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

There are no configuration options set for this peripheral instance.

5.8 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-spi/1.0] instance: spi1

5.8.1 Description

Zynq 7000 SPI Registers

5.8.2 Licensing

Open Source Apache 2.0

5.8.3 Limitations

This model implements the full set of registers but no behavior.

5.8.4 Reference

Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

There are no configuration options set for this peripheral instance.

5.9 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-can/1.0] instance: can0

5.9.1 Description

Zynq 7000 CAN Registers

5.9.2 Licensing

Open Source Apache 2.0

5.9.3 Limitations

This model implements the full set of registers but no behavior.

5.9.4 Reference
Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

There are no configuration options set for this peripheral instance.

5.10 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-can/1.0] instance: can1

5.10.1 Description
Zynq 7000 CAN Registers

5.10.2 Licensing
Open Source Apache 2.0

5.10.3 Limitations
This model implements the full set of registers but no behavior.

5.10.4 Reference
Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

There are no configuration options set for this peripheral instance.

5.11 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-gpio/1.0] instance: GPIO

5.11.1 Description
Zynq 7000 Platform GPIO Registers (gpio)
Included is the visualization of LED and SW connectivity for the ZC702/ZC706 devices.

5.11.2 Licensing
Open Source Apache 2.0

5.11.3 Limitations
This model implements only the registers for generation of input or output data values.
No interrupt generation is currently included in the model.

5.11.4 Reference
Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

There are no configuration options set for this peripheral instance.

5.12 Peripheral [cadence.ovpworld.org/peripheral/gem/1.0] instance: eth0

5.12.1 Description

Model of Cadence Gigabit Ethernet Controller (GEM). For further details please consult README-EMAC.txt
This model is based upon the data and use in the Xilinx Zynq
Basic network Tx/Rx functionality tested using Xilinx Linux Kernel using wget and other similar tools
Tested with Xilinx SDK Example driver.

5.12.2 Licensing

Open Source Apache 2.0

5.12.3 Limitations

This model is based upon the data from the Xilinx Zynq platform, other registers may not be included.
Does not implement: VLAN, pause frames, filtering or timestamps.

5.12.4 Reference

Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

There are no configuration options set for this peripheral instance.

5.13 Peripheral [cadence.ovpworld.org/peripheral/gem/1.0] instance: eth1

5.13.1 Description

Model of Cadence Gigabit Ethernet Controller (GEM). For further details please consult README-EMAC.txt
This model is based upon the data and use in the Xilinx Zynq
Basic network Tx/Rx functionality tested using Xilinx Linux Kernel using wget and other similar tools
Tested with Xilinx SDK Example driver.

5.13.2 Licensing

Open Source Apache 2.0

5.13.3 Limitations

This model is based upon the data from the Xilinx Zynq platform, other registers may not be included.
Does not implement: VLAN, pause frames, filtering or timestamps.

5.13.4 Reference

Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

There are no configuration options set for this peripheral instance.

5.14 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-qspi/1.0] instance: qspi

5.14.1 Description

Zynq 7000 Quad-SPI Registers and incorporates Flash Memory (Spansion and Micron) for Zync zc702/zc706 boards

5.14.2 Licensing

Open Source Apache 2.0

5.14.3 Limitations

This model implements the full set of registers but not all flash memory accesses are supported.
The model is tested using Xilinx Example Project for R/W a QPSI memory on ZC702 platform using Polled and Interrupt driven Transfers.
https://github.com/Xilinx/embeddedsw/tree/master/XilinxProcessorIPLib/drivers/qspips/examples
The AXI mode of operation is not tested. There is no write protection implemented for memory access when in AXI mode.

5.14.4 Reference

Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)
https://xilinx.github.io/embeddedsw.github.io/qspips/doc/html/api/index.html

Table 13. Configuration options (attributes) set for instance 'qspi'

Attribute	Value	Type	Expression
flash	flashtype	string	

5.15 Peripheral [ovpworld.org/peripheral/trap/1.0] instance: dummySMC

5.15.1 Description

Open a port and allocate a region that is defined by parameters.
The region can be configured to act as standard memory or can report read/write accesses.

5.15.2 Licensing

Open Source Apache 2.0

5.15.3 Limitations

This peripheral cannot be used in a hardware description used to generate a TLM platform.

5.15.4 Reference

This is not based upon the operation of a real device but is intended to be used for bring up and development of new virtual platforms.

Table 14. Configuration options (attributes) set for instance 'dummySMC'

Attribute	Value	Type	Expression
portAddress	0xe000e000	uns32	
portSize	0x1000	uns32	

5.16 Peripheral [ovpworld.org/peripheral/trap/1.0] instance: dummyUnknown1

5.16.1 Description

Open a port and allocate a region that is defined by parameters.
The region can be configured to act as standard memory or can report read/write accesses.

5.16.2 Licensing

Open Source Apache 2.0

5.16.3 Limitations

This peripheral cannot be used in a hardware description used to generate a TLM platform.

5.16.4 Reference

This is not based upon the operation of a real device but is intended to be used for bring up and development of new virtual platforms.

Table 15. Configuration options (attributes) set for instance 'dummyUnknown1'

Attribute	Value	Type	Expression
portAddress	0xe000f000	uns32	
portSize	0x1000	uns32	

5.17 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-sdio/1.0] instance: sdio0

5.17.1 Description

Zynq 7000 SD/SDIO Registers

5.17.2 Licensing

Open Source Apache 2.0

5.17.3 Limitations

This model implements the full set of registers but no behavior.

5.17.4 Reference

Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

There are no configuration options set for this peripheral instance.

5.18 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-sdio/1.0] instance: sdio1

5.18.1 Description

Zynq 7000 SD/SDIO Registers

5.18.2 Licensing

Open Source Apache 2.0

5.18.3 Limitations

This model implements the full set of registers but no behavior.

5.18.4 Reference

Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

There are no configuration options set for this peripheral instance.

5.19 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-tz_security/1.0] instance: trustzone_security

5.19.1 Description

Zynq 7000 Trust Zone Security Registers

5.19.2 Licensing

Open Source Apache 2.0

5.19.3 Limitations

This model implements the set of registers but no behavior.

5.19.4 Reference

Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf) and
ug1019-zynq-trustzone

There are no configuration options set for this peripheral instance.

5.20 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-slcr/1.0] instance: slcr

5.20.1 Description

Zynq 7000 Platform System Level Control Registers (SLCR)

5.20.2 Licensing

Open Source Apache 2.0

5.20.3 Limitations

This model implements the full set of registers. Only behavior required for processor reset control is included.

5.20.4 Reference

Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

Table 16. Configuration options (attributes) set for instance 'slcr'

Attribute	Value	Type	Expression
deviceid	0x11	Uns32	
psclock	psclock	Uns32	
armmips	armmips	Uns32	

5.21 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-ttc/1.0] instance: ttc0

5.21.1 Description

Zynq 7000 Triple Timer Counter Registers

5.21.2 Licensing

Open Source Apache 2.0

5.21.3 Limitations

This model implements the full set of registers and basic behavior. It is not yet completed.

5.21.4 Reference

Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

There are no configuration options set for this peripheral instance.

5.22 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-ttc/1.0] instance: ttc1

5.22.1 Description

Zynq 7000 Triple Timer Counter Registers

5.22.2 Licensing

Open Source Apache 2.0

5.22.3 Limitations

This model implements the full set of registers and basic behavior. It is not yet completed.

5.22.4 Reference

Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

There are no configuration options set for this peripheral instance.

5.23 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-dmac/1.0] instance: DMAC

5.23.1 Description
Zynq 7000 Platform DMA Controller (DMAC)

5.23.2 Licensing
Open Source Apache 2.0

5.23.3 Limitations
This model implements the full set of registers. There is no behavior included.

5.23.4 Reference
Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

There are no configuration options set for this peripheral instance.

5.24 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-swdt/1.0] instance: swdt

5.24.1 Description
Zynq 7000 System Watchdog Timer Registers

5.24.2 Licensing
Open Source Apache 2.0

5.24.3 Limitations
This model implements the full set of registers but no behavior.

5.24.4 Reference
Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

There are no configuration options set for this peripheral instance.

5.25 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-ddrc/1.0] instance: DDRC

5.25.1 Description
Zynq 7000 Platform DDR Memory Controller (DDRC)

5.25.2 Licensing
Open Source Apache 2.0

5.25.3 Limitations
This model implements the full set of registers. There is no behavior included.

5.25.4 Reference
Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

There are no configuration options set for this peripheral instance.

5.26 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-devcfg/1.0] instance: devcfg

5.26.1 Description
Zynq 7000 Platform Device Configuration Registers (devcfg)

5.26.2 Licensing
Open Source Apache 2.0

5.26.3 Limitations
This is mainly a register only interface model. It provides behavior to access the power rails using the XADC interface. The power rail data is provided by values stored in memory which can be updated externally. It provides the ability to lock and un-lock registers.

5.26.4 Reference
Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

Table 17. Configuration options (attributes) set for instance 'devcfg'

Attribute	Value	Type	Expression
board	board	String	

5.27 Peripheral [ovpworld.org/peripheral/trap/1.0] instance: dummyAXI_HP0

5.27.1 Description
Open a port and allocate a region that is defined by parameters.
The region can be configured to act as standard memory or can report read/write accesses.

5.27.2 Licensing
Open Source Apache 2.0

5.27.3 Limitations

This peripheral cannot be used in a hardware description used to generate a TLM platform.

5.27.4 Reference

This is not based upon the operation of a real device but is intended to be used for bring up and development of new virtual platforms.

Table 18. Configuration options (attributes) set for instance 'dummyAXI_HP0'

Attribute	Value	Type	Expression
portAddress	0xf8008000	uns32	
portSize	0x1000	uns32	

5.28 Peripheral [ovpworld.org/peripheral/trap/1.0] instance: dummyAXI_HP1

5.28.1 Description

Open a port and allocate a region that is defined by parameters.
The region can be configured to act as standard memory or can report read/write accesses.

5.28.2 Licensing

Open Source Apache 2.0

5.28.3 Limitations

This peripheral cannot be used in a hardware description used to generate a TLM platform.

5.28.4 Reference

This is not based upon the operation of a real device but is intended to be used for bring up and development of new virtual platforms.

Table 19. Configuration options (attributes) set for instance 'dummyAXI_HP1'

Attribute	Value	Type	Expression
portAddress	0xf8009000	uns32	
portSize	0x1000	uns32	

5.29 Peripheral [ovpworld.org/peripheral/trap/1.0] instance: dummyAXI_HP2

5.29.1 Description

Open a port and allocate a region that is defined by parameters.
The region can be configured to act as standard memory or can report read/write accesses.

5.29.2 Licensing

Open Source Apache 2.0

5.29.3 Limitations

This peripheral cannot be used in a hardware description used to generate a TLM platform.

5.29.4 Reference

This is not based upon the operation of a real device but is intended to be used for bring up and development of new virtual platforms.

Table 20. Configuration options (attributes) set for instance 'dummyAXI_HP2'

Attribute	Value	Type	Expression
portAddress	0xf800a000	uns32	
portSize	0x1000	uns32	

5.30 Peripheral [ovpworld.org/peripheral/trap/1.0] instance: dummyAXI_HP3

5.30.1 Description

Open a port and allocate a region that is defined by parameters.
The region can be configured to act as standard memory or can report read/write accesses.

5.30.2 Licensing

Open Source Apache 2.0

5.30.3 Limitations

This peripheral cannot be used in a hardware description used to generate a TLM platform.

5.30.4 Reference

This is not based upon the operation of a real device but is intended to be used for bring up and development of new virtual platforms.

Table 21. Configuration options (attributes) set for instance 'dummyAXI_HP3'

Attribute	Value	Type	Expression
portAddress	0xf800b000	uns32	
portSize	0x1000	uns32	

5.31 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-ocm/1.0] instance: OCM

5.31.1 Description

Zynq 7000 Platform On Chip Memory Controller Registers (OCM)

5.31.2 Licensing

Open Source Apache 2.0

5.31.3 Limitations

This model implements the full set of registers. There is no behavior included.

5.31.4 Reference
Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

There are no configuration options set for this peripheral instance.

5.32 Peripheral [ovpworld.org/peripheral/trap/1.0] instance: dummyeFuse

5.32.1 Description
Open a port and allocate a region that is defined by parameters.
The region can be configured to act as standard memory or can report read/write accesses.

5.32.2 Licensing
Open Source Apache 2.0

5.32.3 Limitations
This peripheral cannot be used in a hardware description used to generate a TLM platform.

5.32.4 Reference
This is not based upon the operation of a real device but is intended to be used for bring up and development of new virtual platforms.

Table 22. Configuration options (attributes) set for instance 'dummyeFuse'

Attribute	Value	Type	Expression
portAddress	0xf800d000	uns32	
portSize	0x1000	uns32	

5.33 Peripheral [ovpworld.org/peripheral/trap/1.0] instance: dummyUnknown2

5.33.1 Description
Open a port and allocate a region that is defined by parameters.
The region can be configured to act as standard memory or can report read/write accesses.

5.33.2 Licensing
Open Source Apache 2.0

5.33.3 Limitations
This peripheral cannot be used in a hardware description used to generate a TLM platform.

5.33.4 Reference
This is not based upon the operation of a real device but is intended to be used for bring up and development of new virtual platforms.

Table 23. Configuration options (attributes) set for instance 'dummyUnknown2'

Attribute	Value	Type	Expression
portAddress	0xf800e000	uns32	
portSize	0x1000	uns32	

5.34 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-tz_GPVsecurity/1.0] instance: trustzone_GPVsecurity

5.34.1 Description

Zynq 7000 Trust Zone GPV Security Registers

5.34.2 Licensing

Open Source Apache 2.0

5.34.3 Limitations

This model implements the set of registers but no behavior.

5.34.4 Reference

Zynq-7000 TRM (https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf) and ug1019-zynq-trustzone

There are no configuration options set for this peripheral instance.

5.35 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-qos301/1.0] instance: GPV_qos301_cpu

5.35.1 Description

Zynq 7000 Platform Interconnect QoS (qos301)

5.35.2 Licensing

Open Source Apache 2.0

5.35.3 Limitations

This model implements the full set of registers. There is no behavior included.

5.35.4 Reference

Zynq-7000 TRM (https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

There are no configuration options set for this peripheral instance.

5.36 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-qos301/1.0] instance: GPV_qos301_dmac

5.36.1 Description
Zynq 7000 Platform Interconnect QoS (qos301)

5.36.2 Licensing
Open Source Apache 2.0

5.36.3 Limitations
This model implements the full set of registers. There is no behavior included.

5.36.4 Reference
Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

There are no configuration options set for this peripheral instance.

5.37 Peripheral [xilinx.ovpworld.org/peripheral/zynq_7000-qos301/1.0] instance: GPV_qos301_iou

5.37.1 Description
Zynq 7000 Platform Interconnect QoS (qos301)

5.37.2 Licensing
Open Source Apache 2.0

5.37.3 Limitations
This model implements the full set of registers. There is no behavior included.

5.37.4 Reference
Zynq-7000 TRM
(https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf)

There are no configuration options set for this peripheral instance.

5.38 Peripheral [arm.ovpworld.org/peripheral/L2CachePL310/1.0] instance: l2cache

5.38.1 Description
ARM PL310 L2 Cache Control Registers

5.38.2 Licensing
Open Source Apache 2.0

5.38.3 Limitations
Programmers View, register model only. Does NOT model functionality, just provides registers to allow code to run.

5.38.4 Reference

ARM PrimeCell Level 2 Cache Controller (PL310) Technical Reference Manual (ARM DDI 0246)

There are no configuration options set for this peripheral instance.

6.0 Overview of Imperas OVP Virtual Platforms

This document provides the details of the usage of an Imperas OVP Virtual Platform / Module. The first half of the document covers specifics of this particular virtual platform / module.

This second part of the document, includes information about Imperas OVP virtual platforms and modules, how they are built and used.

The Imperas virtual platforms are designed to provide a base for you to run high-speed software simulations of CPU-based SoCs and platforms on any suitable PC. They are typically based on the functionality of vendors fixed or evaluation platforms, enabling you to simulate software on these reference platforms. Typically virtual platforms are fixed and require the vendor to modify or extend them. Imperas virtual platforms are different in that they enable you to extend the functionality of the virtual platform, to closer reflect your own platform, by adding more component models, running different operating systems or adding additional applications.

Imperas virtual platforms are created using the Imperas iGen technology, allowing them to be used with Imperas OVP based simulators and also with Accellera/OSCI compliant SystemC simulators and commercial EDA System Design environments that use SystemC.

Virtual platforms include simulation models of the target devices, including the processor model(s) for the target device plus enough peripheral models to boot an operating system or run bare metal applications. The platform and the peripheral models used in most of the virtual platforms are open source, so that you can easily add new models to the platform as well as modify the existing models. Some models are only provided as binary, normally because the IP owner has restricted the release of the model source. In this case, please contact Imperas for more information.

There are typically several generic flavors of the virtual platforms for specific processor families, some targeting full operating systems, such as Linux, and some which focus on Real Time Operating Systems (RTOS) such as Mentor Nucleus or freeRTOS. OVP models of the processor cores are included in the virtual platforms, and for those processors which support mulitple cores SMP Linux is often supported for that virtual platform. For all of these virtual platforms, many of the peripheral components of the platform are modeled, often including the Ethernet and USB components. The semi-hosting capability of the Imperas virtual platform simulator products enables connection via the Ethernet and USB components from the virtual platform to the real world via the x86 host machine.

The Imperas OVP CPU models are written using the OVP Virtual Machine Interface (VMI) API that defines the behavior of the processor. The VMI API makes a clear line between model and simulator allowing very good optimization and world class high speed performance. The processor models are Instruction Accurate and do not model the detailed cycle timing of the processor and they implement functionality at the level of a Programmers View of the processor and peripherals and the software running on them does not know it is not running on hardware. Many models are provided as a binary shared object and also as source. This allows the download and use of the model binary or the use of the source to explore and modify the model. The models are run through an extensive QA and regression testing process and most processor model families are validated using technology provided by the processor IP owners. All the models in this platform are developed with the Open Virtual Platforms APIs and are implemented in C. A platform can be modeled as different levels of hierarchy using separately describable and compilable modules.

More information on modeling and APIs can be found on the www.OVPworld.org site.

7.0 Getting Started with Imperas OVP Virtual Platforms

Virtual platforms are downloadable from the OVPworld website OVPworld.org/downloads. You need to browse and look for '<platform processor name> Examples'. You do need to be registered and logged in on the OVP site to download. OVPworld currently provides 32 bit host versions of packages containing virtual platforms.

When downloading, choose, Linux or Windows host. 32 bit packages can be installed and executed on 32 bit or 64 bit hosts. If you require a 64 bit host version please contact Imperas.

For example, for the ARM Versatile Express platform booting Linux on Cortex-A15MP Single, Dual, and Quad core procesors, you would want the download package: 'OVPsim_demo_Linux_ArmVersatileExpress_arm_Cortex-A15MP'.

Most virtual platform packages contain the platform and all the processor and peripheral models needed. You will need to download a simulator to run the platform. You can use OVPsim, downloadable from OVPworld.org/downloads, or you can use one of the Imperas simulators (imperas.com/products) available commercially from Imperas.

8.0 Simulating Software

8.1 Getting a license key to run

After you have downloaded you will need a runtime license key before the simulators will run. For OVPsim please visit OVPworld.org/likey and provide the required information and an evaluation/demo license key will be automatically sent to you. If you are using Imperas, then please contact Imperas for a license key.

8.2 Normal runs

To run a platform, read the section below on command line control of the platform and the section on setting command line arguments.

8.3 Loading Software

For most virtual platforms the platform is already configured to run the default software application/program and there is normally a script to run that sets some arguments. You can then copy/edit this script to select your own applications etc.

The example application programs are typically .elf format files and are provided pre-compiled. There are normally makefiles and associated scripts to recompile the example applications.

To find more information about compiling and loading software, the following document should be looked at: Imperas_Installation_and_Getting_Started.pdf.

8.4 Semihosting

In a virtual platform, semihosting is not normally used as there is normally hardware that implements the appropriate functionality - for example I/O will be handled by UARTs etc.

8.5 Using a terminal (UART)

If the platform includes one or more UARTs you will need to connect a terminal program to it so that you can see output and type into the simulated program. Review the list of peripherals below and see what configuration options it has been set with. In most cases there is an option to set to instruct the simulator to 'pop up' a terminal window connected to the simulated UART.

8.6 Interacting with the simulation (keyboard and mouse)

If the platform has a simulated UART you can normally set a command to get the simulator to pop up a terminal window allowing you to see output from the simulated UART and also allowing you to type characters into the UART that can be processed by the simulated software.

If your simulated platform has an LCD device then you can often configure it to recognize mouse movements and mouse clicks - allowing full interaction.

To see these interactions in action, have a look at some of the available videos available at OVPworld.org/demosandvideos.

8.7 More Information (Documentation) on Simulation

To find more information about running simulations and more of the options the simulators provide, the following documents should be looked at:
Imperas_Installation_and_Getting_Started.pdf
Simulation_Control_of_Platforms_and_Modules_User_Guide.pdf
Advanced_Simulation_Control_of_Platforms_and_Modules_User_Guide.pdf
OVP_Control_File_User_Guide.pdf

A full list of the currently available OVP documentation is available: OVPworld.org/documentation.

9.0 Debugging Software running on an Imperas OVP Virtual Platform

The Imperas and OVP simulators have several different interfaces to debuggers. These include several proprietary formats and also the standard GNU RSP format is supported allowing many compatible debuggers to be used. Below are some examples that Imperas directly support.

9.1 Debugging with GDB

A GNU debugger (GDB) can be connected to a processor in a platform using the RSP protocol. This allows the application program running on a processor to be debugged using a specific GDB for the processor selected. When using the Imperas Professional products many connections can be made allowing a GDB to be connected to all the processors in the platform.

The use of GDB is documented: OVPsim_Debugging_Applications_with_GDB_User_Guide.pdf.

*9.2 Debugging with Imperas M*DBG*

The Imperas multi-processor debugger can be connected to a platform and through this connection you can debug application programs running on all of the processors instanced within the platform. It is also capable, within this single unified environment, to debug peripheral model behavioral code in conjunction with the processor application programs.

For more information please see the Imperas M*DBG user guide.

The Imperas multi-processor debugger is also capable of controlling the Imperas Verification Analysis aand Profiling (VAP) tools in real time, making them invaluable to application program development, debugging and analysis.

For more information please see the Imperas VAP tools user guide.

9.3 Debugging with the Imperas eGui and GDB

Imperas eGui gives a GUI front end to the use of the GDB debugger. It allows use of all the features of GDB including source level application program debugging on processors.

*9.4 Debugging with the Imperas eGui and M*DBG*

Imperas eGui gives a GUI front end to the Imperas multi-processor debugger. It provides all the features of this debugger but does so with source level application program debugging on processors and source level debugging of the behavioral code on peripheral components in the platform. A context view shows all the processor and peripheral components within the platform and allows switching between them to examine the state of each at the event at which the simulation was stopped

Imperas eGui provides a menu from which the Imperas VAP tools can be controlled.

9.5 Debugging with Imperas eGui and Eclipse

Imperas provide a GUI based on Eclipse called eGui. This provides a GUI front end to use with a standard GDB or the Imperas MPD (Multi-Processor Debugger).

The use of eGui is documented: eGui_Eclipse_User_Guide.pdf.

A standard Eclipse CDT development environment can be connected to one or more processors in a platform (multiple processors require an Imperas professional product). The simulation platform is started remotely or using the external tool feature in Eclipse, opens a debug port and awaits the connection with Eclipse. All features provided by the Eclipse CDT development environment are available to be used to debug software applications executing on the processors in the platform.

The use of Eclipse is documented: OVPsim_Debugging_Applications_with_Eclipse_User_Guide.pdf.

9.6 Debugging applications running under a simulated operating system

If the simulated platform is running an Operating System and the platform has a UART or Ethernet etc connection then it is often possible to connect an external debugger and debug the applications running under the simulated operating system.

An example would be a simulated platform running the Linux operating system, such as the MIPS Malta, or ARM Versatile Express. Within the simulated Linux you can start a gdbserver that connects from within the simulation through a UART out to the host PC via a port. Within the host PC you start a terminal program and connect to the port with a debugger such as GDB and can then debug the simulated user application.

10.0 Modifying the Platform / Module

10.1 Platforms / Modules use C/C++ and OVP APIs

The Imperas and OVP simulators execute a platform / module that is written in C/C++ and that makes function calls into the simulator's APIs. Thus the virtual platform / module is compiled from C/C++ into a binary shared object that the simulator loads and runs. OVP provides the definition and documentation that defines the C APIs for modeling the platforms, modules, the peripherals, and the processors. You can find more information about these APIs on the OVP website and in the OVP API documentation.

10.2 Platforms/Modules/Peripherals can be easily built with iGen from Imperas

Imperas provides a product 'iGen' that takes an input script file and creates the C/C++ files needed for platforms, modules, and peripherals - it creates the C/C++ file that is compiled into the platform, module or peripheral that is needed as an object file by the simulator. iGen creates the C/C++ files, you then need to add any necessary behaviors or further details etc. For platforms iGen creates either a C platform or a C++ SystemC TLM2 platform. For peripherals or modules iGen creates the C files and also provides a native C++ SystemC TLM2 interface to allow the peripheral/module to be instantiated in SystemC TLM2 platforms.

Information on iGen is available from: imperas.com/products.

10.3 Re-configuring the platform

There will nornmally be several configuration options that you can set when running the platform without the neeed to change any source. Refer to the section above on command line arguments. If these do not allow you to make the changes you need, then you may need to edit and recompile the source of the platform.

The source of the platform, modules, and the source of the peripherals will be installed as part of the packages you are using. The sources are located in the Imperas/OVP installation VLNV source tree. The VLNV term refers to: Vendor (eg arm.ovpworld.org), Library (eg platform), Name, (eg ArmVersatileExpress-CA15), and Version (eg 1.0). To modify the platform, locate the platform source files.

If you are an Imperas user and have access to iGen, we recommend you modify the source script files and regenerate and recompile the C that makes up the platform. Refer to the Imperas iGen model generator guide and the Imperas platform generator guide.

If you are using the C or SystemC TLM2 platforms with OVPsim, then you can edit the C/C++ files, recompile the source directly using the supplied makefiles, and the run the simulator directly with the resultant shared object.

10.4 Replacing peripherals components

If you need to replace peripherals, find the appropriate place in the source of the platform, make the change you need, and recompile etc. Look in the library for documentation on available peripherals and their configuration options.

10.5 Adding new peripherals components

If you need to add peripherals, find the appropriate place in the source, make the additions you need, and recompile etc. Look in the library for documentation on available peripherals and their configuration options.

If you need to create new peripheral components then use iGen to very quickly create the necessary C/C++ files that get you started. With iGen you can create peripherals with register/memory state in a few lines of iGen source. When adding behavior to the peripherals refer to the OVP API documentation.

11.0 Available Virtual Platforms

Table 24. Imperas / OVP Extendable Platform Kits (13 available)

Name	Vendor
AlteraCycloneIII_3c120	altera.ovpworld.org
AlteraCycloneV_HPS	altera.ovpworld.org
ArmIntegratorCP	arm.ovpworld.org
ArmVersatileExpress	arm.ovpworld.org
ArmVersatileExpress-CA15	arm.ovpworld.org
ArmVersatileExpress-CA9	arm.ovpworld.org
AtmelAT91SAM7	atmel.ovpworld.org
FreescaleKinetis60	freescale.ovpworld.org
FreescaleKinetis64	freescale.ovpworld.org
FreescaleVybridVFxx	freescale.ovpworld.org
MipsMalta	mips.ovpworld.org
RenesasUPD70F3441	renesas.ovpworld.org
XilinxML505	xilinx.ovpworld.org

Table 25. Imperas General Virtual Platforms (6 available)

Name	Vendor
arm-ti-eabi	arm.imperas.com
armm-ti-coff	arm.imperas.com
armm-ti-eabi	arm.imperas.com
HeteroAlteraCycloneV_HPS_CycloneIII_3c120	imperas.ovpworld.org
HeteroArmNucleusMIPSLinux	imperas.ovpworld.org
SiFiveFU540	imperas.ovpworld.org

Table 26. Imperas Modules (component of other platforms) (55 available)

Name	Vendor
AlteraCycloneIII_3c120	altera.ovpworld.org
AlteraCycloneV_HPS	altera.ovpworld.org
AE350	andes.ovpworld.org
ARMv8-A-FMv1	arm.ovpworld.org
ArmIntegratorCP	arm.ovpworld.org
ArmVersatileExpress	arm.ovpworld.org
ArmVersatileExpress-CA15	arm.ovpworld.org
ArmVersatileExpress-CA9	arm.ovpworld.org
AtmelAT91SAM7	atmel.ovpworld.org
ArmCortexMFreeRTOS	imperas.ovpworld.org
ArmCortexMuCOS-II	imperas.ovpworld.org
ArmuKernel	imperas.ovpworld.org
ArmuKernelDual	imperas.ovpworld.org
BareMetalMIPS	imperas.ovpworld.org
Dual_ARMv8-A-FMv1_VLAN	imperas.ovpworld.org
Hetero_1xArm_3xMips32	imperas.ovpworld.org
Hetero_ARM_RISCV_NeuralNetwork	imperas.ovpworld.org

Hetero_ARMv8-A-FMv1_Cortex-M3	imperas.ovpworld.org
Hetero_ARMv8-A-FMv1_MIPS_microAptiv	imperas.ovpworld.org
Hetero_AlteraCycloneV_HPS_AlteraCycloneIII_3c120	imperas.ovpworld.org
Hetero_ArmIntegratorCP_XilinxMicroBlaze	imperas.ovpworld.org
Hetero_ArmVersatileExpress_MipsMalta	imperas.ovpworld.org
Hetero_ArmVersatileExpress_XilinxMicroBlaze	imperas.ovpworld.org
Quad_ArmVersatileExpress-CA15	imperas.ovpworld.org
RiscvRV32FreeRTOS	imperas.ovpworld.org
MipsMalta	mips.ovpworld.org
iMX6S	nxp.ovpworld.org
RenesasUPD70F3441	renesas.ovpworld.org
ghs-multi	renesas.ovpworld.org
virtio	riscv.ovpworld.org
FaultInjection	safepower.ovpworld.org
PublicDemonstrator	safepower.ovpworld.org
Zynq_PL_DualMicroblaze	safepower.ovpworld.org
Zynq_PL_NoC	safepower.ovpworld.org
Zynq_PL_NoC_node	safepower.ovpworld.org
Zynq_PL_NostrumNoC	safepower.ovpworld.org
Zynq_PL_NostrumNoC_node	safepower.ovpworld.org
Zynq_PL_RO	safepower.ovpworld.org
Zynq_PL_SingleMicroblaze	safepower.ovpworld.org
Zynq_PL_TTELNoC	safepower.ovpworld.org
Zynq_PL_TTELNoC_node	safepower.ovpworld.org
Zynq_PL_TTELNoC_processing_node_public_demonstrator	safepower.ovpworld.org
Zynq_PL_TTELNoC_public_demonstrator	safepower.ovpworld.org
Zynq_PL_TTELNoC_sensor_actor_node_public_demonstrator	safepower.ovpworld.org
FU540	sifive.ovpworld.org
S51CC	sifive.ovpworld.org
coreip-s51-arty	sifive.ovpworld.org
coreip-s51-rtl	sifive.ovpworld.org
dualFifo	vendor.com
XilinxML505	xilinx.ovpworld.org
Zynq	xilinx.ovpworld.org
Zynq_PL_Default	xilinx.ovpworld.org
Zynq_PS	xilinx.ovpworld.org
zc702	xilinx.ovpworld.org
zc706	xilinx.ovpworld.org

Table 27. Imperas / OVP Bare Metal Virtual Platforms (22 available)

Name	Vendor
BareMetalNios_IISingle	altera.ovpworld.org
BareMetalArcSingle	arc.ovpworld.org
BareMetalArm7Single	arm.ovpworld.org
BareMetalArmCortexADual	arm.ovpworld.org
BareMetalArmCortexASingle	arm.ovpworld.org
BareMetalArmCortexASingleAngelTrap	arm.ovpworld.org
BareMetalArmCortexMSingle	arm.ovpworld.org

ArmCortexMFreeRTOS	imperas.ovpworld.org
ArmCortexMuCOS-II	imperas.ovpworld.org
BareMetalArmx1Mips32x3	imperas.ovpworld.org
Or1kUclinux	imperas.ovpworld.org
BareMetalM14KSingle	mips.ovpworld.org
BareMetalMips32Dual	mips.ovpworld.org
BareMetalMips32Single	mips.ovpworld.org
BareMetalMips64Single	mips.ovpworld.org
BareMetalMipsDual	mips.ovpworld.org
BareMetalMipsSingle	mips.ovpworld.org
BareMetalOr1kSingle	ovpworld.org
BareMetalM16cSingle	posedgesoft.ovpworld.org
BareMetalPowerPc32Single	power.ovpworld.org
BareMetalV850Single	renesas.ovpworld.org
ghs-multi	renesas.ovpworld.org

#