OVP Guide to Using Processor Models

Model Specific Information for variant renesas_RL78_RL78-S1

Imperas Software Limited
Imperas Buildings, North Weston
Thame, Oxfordshire, OX9 2HA, UK
docs@imperas.com

Author	Imperas Software Limited
Version	0.5
Filename	OVP_Model_Specific_Information_rl78_RL78-S1.pdf
Created	23 February 2018
Status	OVP Standard Release

Copyright Notice

Right to Copy Documentation

Destination Control Statement

Disclaimer

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 Overview

This document provides the details of an OVP Fast Processor Model variant.

OVP Fast Processor Models are written in C and provide a C API for use in C based platforms. The models also provide a native interface for use in SystemC TLM2 platforms.

The models are written using the OVP VMI API that provides a Virtual Machine Interface 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.

Most 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 model families are validated using technology provided by the processor IP owners.

There is a companion document (OVP Guide to Using Processor Models) which explains the general concepts of OVP Fast Processor Models and their use. It is downloadable from the OVPworld website documentation pages.

1.1 Description

RL78 Family Processor Model.

1.2 Licensing

Open Source Apache 2.0

1.3 Reference

RL78 User Manual: Software, Single-Chip microcontrollers, http://documentation.renesas.com/doc/products/mpumcu/doc/rl78/r01us0015ej0220_rl78.pdf

1.4 Limitations

All instructions are supported except the MULU, MULHU, MULH, DIVHU, MACHU and MACH instructions that are not implemented.

Banked registers are not supported

The PMC (Processor Model Control) register behavior is not modeled.

This processor model requires that RAM is available at the address range of the memory mapped registers

Address ranges 0xFFEE0 to 0xFFEFF for General purpose registers (e.g. X, A)

Address ranges 0xFFFF0 to 0xFFFFF for special function registers (e.g. SP)

This processor model should be started with a reset signal. The processor reads from the reset vector 0x0000 on reset and uses this value for the initial PC

1.5 Verification

Models have been tested by eSOL TRINITY and Imperas

1.6 Features

External exceptions are supported

The BRK instruction (internal trap) is supported

Memory mirroring is supported
Memory mapped registers is supported

2 Configuration

2.1 Location

The model source and object file is found in the VLNV tree at:
renesas.ovpworld.org/processor/rl78/1.0

2.2 GDB Path

The default GDB for this model is found at:
$IMPERAS_HOME/lib/$IMPERAS_ARCH/gdb/rl78-elf-gdb

2.3 Semi-Host Library

The default semi-host library file is found in the VLNV tree at :
renesas.ovpworld.org/semihosting/rl78Newlib/1.0

2.4 Processor Endian-ness

This is a LITTLE endian model.

2.5 QuantumLeap Support

A simulator using this processor will not be able to use QuantumLeap.

2.6 Processor ELF Code

The ELF code supported by this model is: 0xc5

3 Other Variants in this Model

Table 1. All variants in this model

Variant	Description
RL78-S1	RL78-S1
RL78-S2	RL78-S2
RL78-S3	RL78-S3

4 Bus Ports

Table 2. Bus Ports

Type	Name	min	max
master (initiator)	INSTRUCTION	20	32
master (initiator)	DATA	20	32
slave	GPRSP	-	-

slave	SFRSP	-	-

5 Net Ports

Table 3. Net Ports

Name	Type
reset	input
extint	input
intAck	output

6 FIFO Ports

No FIFO Ports in this model.

7 Parameters

Table 4. Parameters that can be set in the model, type:

Name	Type	Description
verbose	Boolean	Verbose mode
sim_ac_flag	Boolean	simulate PSW.AC flag
exit_on_halt	Boolean	simulation will exit on HALT instruction
mirror_rom_addr	Uns32	mirror rom addr
mirror_start_addr	Uns32	mirror start addr
mirror_end_addr	Uns32	mirror end addr

8 Execution Modes

Table 5. CPU modes implemented in the model, type:

Name	Code
RB0	0
RB1	1
RB2	2
RB3	3

9 Exceptions

Table 6. Exceptions handled by the model, type:

Name	Code
RST	0
TRP	0

IAW	0
BRK	126
IRQ	65535

10 Hierarchy of the model

A CPU core may allow the user to configure it to instance many processors of a Symmetrical Multi Processor (SMP). A CPU core may also have sub elements within a processor, for example hardware threading blocks.
OVP processor models can be written to include SMP blocks and to have many levels of hierarchy.

Some OVP CPU models may have a fixed hierarchy, and some may be configured by settings in a configuration register. Please see the register definitions of this model.

This model documentation shows the settings and hierarchy of the default settings for this model variant.

10.1 Level 1:

This level in the model hierarchy has 2 commands.

This level in the model hierarchy has 6 register groups:

Table 7. Register groups

Group name	Registers
GPR	8
Bank1	8
Bank2	8
Bank3	8
Bank4	8
System	7

This level in the model hierarchy has no children.

11 Model Commands

11.1 Level 1:

11.1.1 isync

specify instruction address range for synchronous execution

Table 8. isync command arguments

Argument	Type	Description
-addresshi	Uns64	end address of synchronous execution range
-addresslo	Uns64	start address of synchronous execution range

11.1.2 itrace

enable or disable instruction tracing

Table 9. itrace command arguments

Argument	Type	Description
-after	Uns64	apply after this many instructions
-enable	Boolean	enable instruction tracing
-instructioncount	Boolean	include the instruction number in each trace
-off	Boolean	disable instruction tracing
-on	Boolean	enable instruction tracing
-registerchange	Boolean	show registers changed by this instruction
-registers	Boolean	show registers after each trace

12 Registers

12.1 Level 1:

12.1.1 GPR

Table 10. Registers at level 1, type: , register group: 'GPR'

Name	Bits	Initial value (Hex)		Description
X	8	0	rw	X
A	8	0	rw	A
C	8	0	rw	C
B	8	0	rw	B
E	8	0	rw	E
D	8	0	rw	D
L	8	0	rw	L
H	8	0	rw	H

12.1.2 Bank1

Table 11. Registers at level 1, type: , register group: 'Bank1'

Name	Bits	Initial value (Hex)		Description
X[RB0]	8	0	rw	R00
A[RB0]	8	0	rw	R01
C[RB0]	8	0	rw	R02
B[RB0]	8	0	rw	R03
E[RB0]	8	0	rw	R04
D[RB0]	8	0	rw	R05
L[RB0]	8	0	rw	R06
H[RB0]	8	0	rw	R07

12.1.3 Bank2

Table 12. Registers at level 1, type: , register group: 'Bank2'

Name	Bits	Initial value (Hex)		Description
X[RB1]	8	0	rw	R08
A[RB1]	8	0	rw	R09
C[RB1]	8	0	rw	R10
B[RB1]	8	0	rw	R11
E[RB1]	8	0	rw	R12
D[RB1]	8	0	rw	R13
L[RB1]	8	0	rw	R14
H[RB1]	8	0	rw	R15

12.1.4 Bank3

Table 13. Registers at level 1, type: , register group: 'Bank3'

Name	Bits	Initial value (Hex)		Description
X[RB2]	8	0	rw	R16
A[RB2]	8	0	rw	R17
C[RB2]	8	0	rw	R18
B[RB2]	8	0	rw	R19
E[RB2]	8	0	rw	R20
D[RB2]	8	0	rw	R21
L[RB2]	8	0	rw	R22
H[RB2]	8	0	rw	R23

12.1.5 Bank4

Table 14. Registers at level 1, type: , register group: 'Bank4'

Name	Bits	Initial value (Hex)		Description
X[RB3]	8	0	rw	R24
A[RB3]	8	0	rw	R25
C[RB3]	8	0	rw	R26
B[RB3]	8	0	rw	R27
E[RB3]	8	0	rw	R28
D[RB3]	8	0	rw	R29
L[RB3]	8	0	rw	R30
H[RB3]	8	0	rw	R31

12.1.6 System

Table 15. Registers at level 1, type: , register group: 'System'

Name	Bits	Initial value (Hex)		Description
PSW	8	6	r-	status register
ES	8	f	rw	ES
CS	8	0	rw	CS
PC	20	0	rw	program counter
PMC	8	0	r-	processor mode control
MEM	8	0	r-	MEM
SP	16	0	rw	stack pointer

#

