仕様書

SC-HEAP E3

Platform functional specification

THIS IS A TENTATIVE DOCUMENT CORRESPOINDING TO THE JAPANESE DOCUMENT : MSS-SG-12-0061-02.

【原紙査閲承認済み】

LLWEB-00009484 _E		
MSS-SG-12-0061-02 _E		
2012/8/31		
MCU Software Tool Department MCU Software Division		
Approve	Inspect	issue
Sato	---	Sato Yoshinaga Arai Otsuka

本書の位置付け

本書は、SC-HEAP E3 プラットフォームの仕様書です。

おことわり

本書に記載されている会社名・製品名等は各社の商標又は登録商標です。
本書の内容は、後日変更される場合があります。
本書に記載の技術について特許調査を行っておりません。

Table of Contents

Table of figures

Table of tables

1. Overview

1.1. Corresponded Version

This document describes the specification for SC_HEAP E3 V1.00.

1.2. 本書の位置付け

本書の対象読者は、以下に示すマイコンハードウェア設計者およびマイコンソフトウェア開発者です。

A) リリース物件の SC-HEAP E3 シミュレータの動作確認を行ないたいだけの方
(リリース物件検収者)

B) リリースされた SC-HEAP E3 アーキテクチャを変更せずに、その上で動く RH850 マイコンソフトウェアの動作特性を知りたい方
(マイコンソフトウェア開発者)

C) リリースされた SC-HEAP E3 アーキテクチャの構造は変更しないが、あるモデル IP の機能を修正・追加したい方
(マイコンハードウェア設計者)

D) リリースされた SC-HEAP E3 アーキテクチャの構造を変更し、新規モデル IP を追加したい方
(マイコンハードウェア設計者)

本書を読むことによって、以下のことがわかります。

1) V850 E3(RH850)アーキテクチャをもつ SystemC シミュレータの作成方法(1.7章)
2) シミュレーション実行のためのコンフィギュレーション設定およびアドレスマップの設定方法 (Error: Reference source not found 章)
3) シミュレーション実行する V850 E3(RH850)ターゲットプログラムの作成方法(Error: Reference source not found 章)
4) シミュレーションの実行方法(Error: Reference source not found章)
5) 解析結果(シミュレーションログ)の見方(6章)
6) シミュレーション実行に関する注意事項(Error: Reference source not found章)
7) プラットフォーム構築ファイル群の構造(8章)
8) 新規にモデル IP を追加する場合の手順(9章)

対象読者毎に読んでいただきたい章は以下の通りです。

	1 章	2 章	3 章	4 章	5 章	6 章	7 章	8 章	9 章	10 章以降
読者 A	○	○	○	○	○	--	--	--	--	○
読者 B	○	--	○	○	○	○	○	--	--	○
読者 C	○	○	○	○	○	○	○	○	--	○
読者 D	○	○	○	○	○	○	○	○	○	○

本書を読むにあたっては、以下の前提知識が必要です。

1) Linux マシン上でのプログラム実行に関する知識
2) V850 E3(RH850)ターゲットプログラム作成の一般知識(アドレスマップ、RH850 アーキテクチャなどの知識)
3) トランザクションレベルのハードウェア動作の一般知識(レイテンシなどの知識)
4) SystemC の一般知識（モジュール宣言・接続などの知識）【新規 SystemC IP を追加する場合】

なおシミュレーションの実行で何か問題が生じた場合には、まずは Error: Reference source not found 章の「注意事項」を参照して下さい。

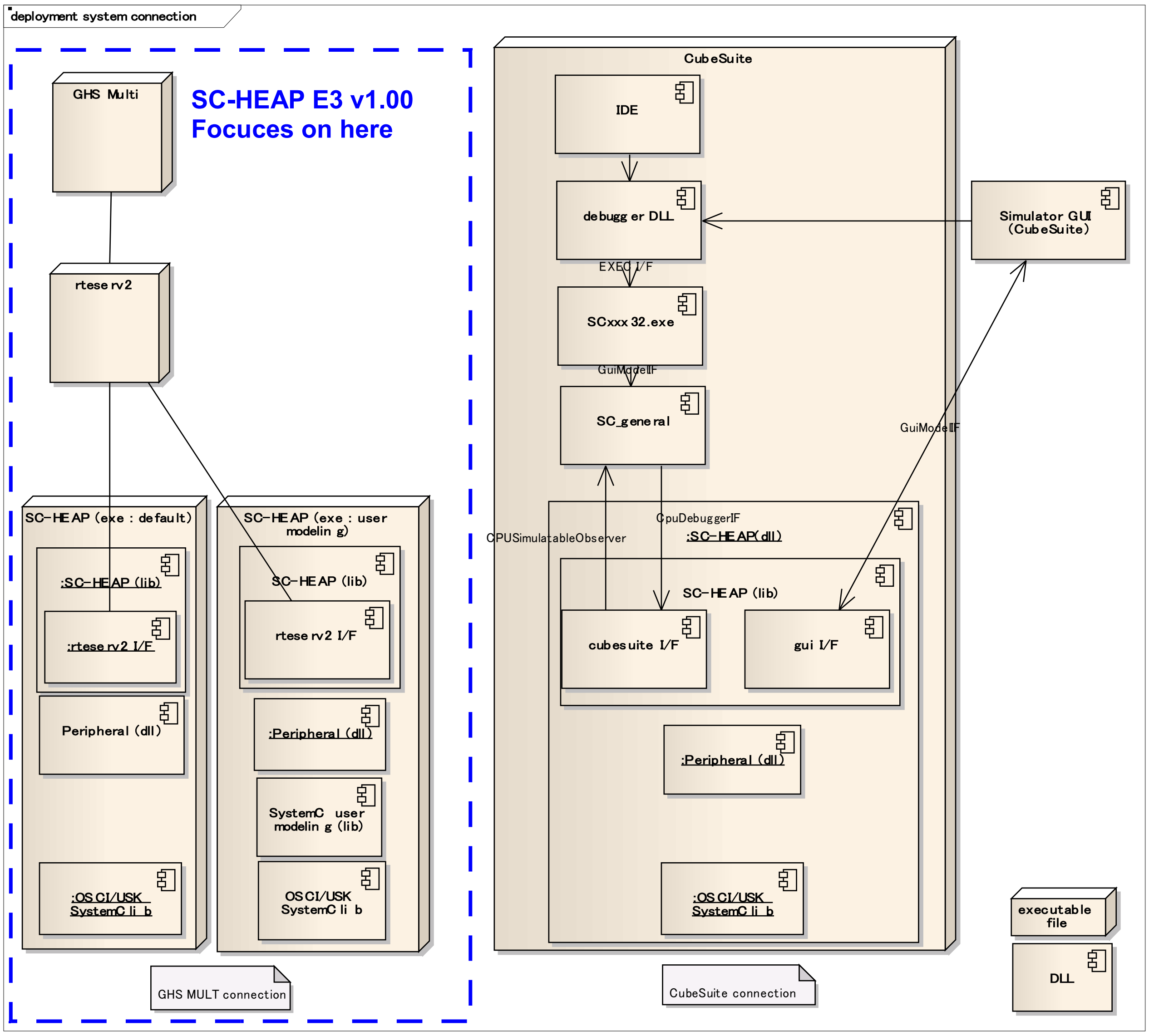
1.3. Usage of the model

This model (sometimes called "platform" in the document) targets RH850 MCU (Generation5) which include V850E3 (RH850) CPU subsystem and the peripherals especially for Autmotive application.
In this model, CPU hierarchy is called "PE hierarchy", CPU subsystem including "PE hierarchy" is called "CPUSS hierarchy". The peripherals are connected outsinde of "CPUSS hierarchy".
This model will be used for
"SW development e.g. for Multicore SW"
"performance estimation before preparing RTL".
This platform consists of V850 E3(RH850) ISS (CForest, ASTC Fast ISS), INTC1/2, PE2PEINT, MEV, bus, bridge, gurad, DMA and peripherals.
User own peripheral can be also connected to CPUSS hierarchy in the platform. （refer to "User manual for SC-HEAP E3 User modeling environment", currently in Japanse only）
This platform can be co-simuate with MATLAB/Simulin.

This platform is executed under ACCELLERA SystemC kernel and can be executed with GHS MULTI debugger for V850.

Regarding peripheral macros, please refere to the related document[1].
Regarding co-simulation with MATLAB/Simulink, please refere to the related document[2].

[1] ASTC が作成するユーザガイドをここに示します
[2] 今後作成予定の「SC-HEAP E3 SMPILS 接続機能仕様書」をここに示します



1.4. Features

This platform includes the following features.

1) Simulate the RH850 target SW on SC-HEAP E3.
2) Change the attribute for each IP with the configuration file or python command
3) Develop driver software for peripheral macros
4) Execute the model with batch mode
5) Debug the SW with GHS MULTI debugger
6) Output trace files
7) Simulate by connecting with user own IP
8) Co-simulate with MATLAB/Simulinlk

1.5. Runtime environment

The runtime environment is shown in Table 1.

Table 1 runtime environment

【Linux ACCELLERA/USK 版 SystemC】

64bit マシン上でテストするが、シミュレータは 64bit 化しない

OS	Red Hat Enterprise Linux release 5.5 32bit/64bit
Compiler	g++ 4.1.2
S ystemC environment	ACCELLERA SystemC 2.2.0, TLM 2.0.1 USK バージョン未定
Python	Python2.7
debugger	Multi+rteserv2 6.x

【Windows ACCELLERA/USK 版 SystemC】

OS	Microsoft Windows 7 32bit/64bit
Compiler	Visual Studio 2010
SystemC environment	ACCELLERA SystemC 2.2.0, TLM 2.0.1 USK バージョン未定
Python	Python2.7
debugger	Multi+rteserv2 6.x

1.6. Directory

Directory structure for SC-HEAP E3 platform is shown in from Figure 1 to Figure 11. In this document, CPUSS hierarchy Figure 3 is sometimes called “SC-HEAP E3 platform” narrowly.
Regarding User modeling environment, please refer to the related document[3].

```
scheapCompile
 ├─build ......................... build / runtime environment
 ├─lib ........................... library for models_astc.(*.so)
 ├─lib-models .................... IP library in CPUSS(debug)
 ├─lib-models03 .................. IP library in CPUSS
 ├─lib-usk-models03 .............. USK SLL for models (CPUSS)
 ├─lib-models_astc  ...............  ACCELLERA  SLL  for  models  (ASTC’ s
peripheral)
 ├─lib-usk-models03_astc ......... USK SLL for models_astc (ASTC’ s peripheral)
 ├─lib-models_rvc ................ ACCELLERA SLL for models (RVC’ s peripheral)
 ├─lib-usk-models03_rvc .......... USK SLL for models_astc (RVC’ s peripheral)
 ├─multi ......................... Multi debug server
 ├─models ........................ model source in CPUSS
 ├─models_astc ................... ASTC’ s model source
 ├─models_rvc .................... RVC’ s model source
 ├─include_astc .................. ASTC’ s model source
 ├─include_rvc ................... RVC’ s header
 └─soft .......................... sample program
pltfrmCompile .................... user modeling environment
 ├・・・
 :
```

Figure 1 Top structure (Linux ACCELLERA/USK edition SystemC)

[3]「SC-HEAP E3 ユーザモデリング環境使用説明書」を今後準備予定

```
scheapCompile
├─build ......................... build/ runtime environment
│   └─TOPV01
│       └─msvc100
│            ├──Release .................. ACCELLERA Windows sim.exe & SCHEAP-E3-G5.lib : Release
mode
│           ├─Debug .................... ACCELLERA Windows sim.exe & SCHEAP-E3-G5.lib: Debug mode
│            ├──Release_astc ............. ACCELLERA Windows ASTC’ s peripheral library: Release
mode
│           ├─Debug_astc ............... ACCELLERA Windows ASTC’ s peripheral library: Debug mode
│            ├──Release_rvc .............. ACCELLERA Windows RVC’ s peripheral library : Release
mode
│           ├─Debug_rvc ................ ACCELLERA Windows RVC’ s peripheral library : Debug mode
│            ├─Release_USK .............. USK  Windows sim.exe & SCHEAP-E3-G5.lib        : Release
mode
│           ├─Debug_USK ................ USK  Windows sim.exe & SCHEAP-E3-G5.lib     : Debug mode
│           ├─Release_USK .............. USK  Windows SCHEAP-E3-G5-USK.lib: Release mode
│           ├─Debug_USK................. USK  Windows SCHEAPD-E3-G5-USK.lib: Debug mode
│           ├─Release_USK .............. USK  ASTC’ s peripheral library  : Release mode
│           ├─Debug_astc_USK ........... USK  ASTC’ s peripheral library  : Debug mode
│           ├─Release_rvc_USK .......... USK  RVC’ s peripheral library   : Release mode
│           └─Debug_rvc_USK ............ USK  RVC’ s peripheral library   : Debug mode
├─lib ................................. library for models_astc.(*.dll)
├─multi ............................... Multi debug server
├─models .............................. REL model IP (Stool, CPU Dev.2, ACCELLERA)
├─models_astc ......................... ASTC’ s models.
├─models_rvc .......................... RVC’ s models.
├─include_astc ........................ ASTC’ s Header
├─include_rvc ......................... RVC’ s Header
└─soft ................................ sample program
pltfrmCompile ......................... user modeling environment
├…
:
```

Figure 2 Top structure (Windows ACCELLERA/USK edition SystemC)

```
└─build
    ├─TOPV01 ....................... SC-HEAP E3build environment
    ├─PFV01 ........................ peripheral connection class.
    ├─SMPILS ....................... MATLAB/Simulink connection function.
    ├─ITxxx......................... xxxxxx の実行環境
```

review
sample
evnrioment

Figure 3 Runtime environment (Linux ACCELLERA/USK edition SystemC)

```
└─build
    ├─ITxxxxxx ......................  xxxxxx の実行環境
    :   HISTORY .....................  変更履歴
    :   sim.x .......................  シミュレータ(へのシンボリックリンク)
    :   heap.cfg ....................  コンフィグレーションファイル
        iLB_core1.map ...............  iLB_13 用マップファイル
        iLB_core2.map ...............  iLB_23 用マップファイル
        dLB_core1.map ...............  dLB_14 用マップファイル
        dLB_core2.map ...............  dLB_24 用マップファイル
        top.IOB_33.map ..............  IOB_33 用マップファイル
        top.AHBSIF_3A.map ...........  AHBSIF_3A 用マップファイル  ※1
        top.APB32IF_3B.map ..........  APB32IF_3B 用マップファイル  ※1
        top.ELBIF_3C.map ............  ELBIF_3C 用マップファイル  ※1
        top.MEMIF_3D.map ............  MEMIF_3D 用マップファイル  ※1
        run_core.csh ................  シミュレーション起動スクリプト
        run_multi.csh ...............  シミュレーション起動スクリプト(Multi 使用時)

 ※1  設定値の内容は空
```

review
sample
evnrioment

Figure 4 Example of runtime environment：ITintv1m (Linux ACCELLERA/USK edition SystemC)

```
└─build
    ├─TOPV01 ........................  SC-HEAP E3 build enviroment
    │   └─msvc100 ...................  VisualStudio build environment
    ├─PFV01 .........................  peripheral connection class.
    ├─SMPILS ........................  MATLAB/Simulink connection function.
    ├─ITxxx..........................  xxxxxx の実行環境
```

review
sample
evnrioment

Figure 5 Runtime environment (Windows ACCELLERA/USK edition SystemC)

```
└─build
    ├─ITintv1m ......................  割り込み(intv1m_v4_heap)の実行環境
    :   HISTORY .....................  変更履歴
    :   sim.x .......................  シミュレータ(sim.exe へのシンボリックリンク)
    :   heap.cfg ....................  コンフィグレーションファイル
        top.iLB_13.map ..............  iLB_13 用マップファイル
        top.iLB_23.map ..............  iLB_23 用マップファイル
        top.dLB_14.map ..............  dLB_14 用マップファイル
        top.dLB_24.map ..............  dLB_24 用マップファイル
        top.IOB_33.map ..............  IOB_33 用マップファイル
        top.AHBSIF_3A.map ...........  AHBSIF_3A 用マップファイル  ※1
        top.APB32IF_3B.map ..........  APB32IF_3B 用マップファイル  ※1
        top.ELBIF_3C.map ............  ELBIF_3C 用マップファイル  ※1
        top.MEMIF_3D.map ............  MEMIF_3D 用マップファイル  ※1
        run_core_win.bat ............  シミュレーション起動バッチファイル
        run_multi_win.bat ...........  シミュレーション起動バッチファイル(Multi 使用時)
 ※1  設定値の内容は空
```

review
sample
evnrioment

Figure 6 Example of runtime environment：ITintv1m (Windows ACCELLERA/USK edition SystemC)

```
└─multi ........................ debug server
    rteserv2 ...................
    spawn.py ................... python script
```

Figure 7 debug server(Linux ACCELLERA/USK edition SystemC)

```
└─multi ........................ debug server
    rteserv2.exe ...............
    spawn.py ................... python script
```

Figure 8 debug server (Windows ACCELLERA/USK edition SystemC)

```
└─models ......................... the model build environment
    ├─NSMVRH850V01 ............... RH850(top)hierarchy
    ├─NSMVG3MSSV01 ............... CPUSS hierarchy
    ├─NSMVG3MPEV01 ............... PE hierarchy
                                   (CPU hierarchy, INTC1/2,PE2PEINT,MEV etc.)
    ├─NSMVG3MCPUV01............... CPU hierarchy (CA ISS + Fast ISS)
    ├─NSMVINTC1V01................ INTC1
    ├─NSMVINTC2V01................ INTC2
    ├─ATLTLB32.................... 32 bit bus
    ├─ATLTLB64.................... 64 bit bus
    ├─ATLTSLAVE32................. 32 bit bus slave
    ├─ATLTSLAVE64................. 64 bit bus slave
    ├─VPI2APB..................... VPI2APB
    ├─VCI2AHB..................... VCI2AHB
    ├─AHB2VCI..................... AHB2VCI
    ├─common ..................... shared source
    ├─common_bus ................. sharre bus source
    ├─iss ........................ ISS engine
    │   ├─cforest_g3m . .......... CForestG3M (CA ISS)
    │   └─fastiss_astc............ ASTC Fast ISS
    └─tlm ........................ ACCELLERA TLM 2.0.1
```

Figure 9 build environment of model IP (Linux/Windows ACCELLERA/USK edition SystemC)

```
└─soft ..........................  smple program
   ├─caxi ......................  排他制御(caxi)ソフト
   ├─dma .......................  DMA(dma)ソフト
   ├─exsync ....................  排他・同期(exsync)ソフト
   ├─intv1m_v4_heap ............  割り込み(intv1m_v4_heap)ソフト
   ├─lbm .......................  バスモニタ用ソフト
   ├─lib .......................  ライブラリ(stdout 用)
   ├─mec .......................  相互排除制御レジスタ(mec)ソフト
   ├─mpu .......................  MPU(E2R)ソフト
   ├─mpu_e2rv3 .................  MPU(E2RV3)ソフト
   ├─ppu .......................  周辺保護ソフト
   ├─scrbench050930c ...........  Toyota ベンチ(シングルコア)ソフト
   ├─scrbench050930c.multi .....  Toyota ベンチ(マルチコア)ソフト
   ├─set1 ......................  排他制御(set1)ソフト
   ├─simddsp2cpu ...............
   ├─stdout ....................  標準出力(stdout)ソフト
   └─tsu .......................  タイミング監視ソフト
```

reiew later on

Figure 10 sample program

```
└─soft ..........................  sample program
   ├─caxi ......................  排他制御(caxi)ソフト
   :   |   HISTORY ...............  変更履歴
   :   |   README ................  readme
   :   |   Makefile ..............  core1/core2 の Makefile 呼び出し Makefile
       ├─common ..................  core1/core2 共通部分ソース
       |     boot.850 ............  asm ソースファイル
       |     lock.850 ............  asm ソースファイル
       |     share.c .............  C ソースファイル
       |     share.h .............  ヘッダファイル
       |     wait.c ..............  C ソースファイル
       ├─core1 ...................  core1 側ソース
       |     Makefile ............  メイクファイル
       |     data.c ..............  C ソースファイル
       |     local.h .............  ヘッダファイル
       |     main.c ..............  C ソースファイル
       |     start.850 ...........  asm ソースファイル
       |     tp.ld ...............  リンクディレクティブ
       └─core2 ...................  core2 側ソース
             Makefile ............  メイクファイル
             data.c ..............  C ソースファイル
             local.h .............  ヘッダファイル
             main.c ..............  C ソースファイル
             start.850 ...........  asm ソースファイル
             tp.ld ...............  リンクディレクティブ
```

review later on

Figure 11 sample program : caxi

1.7. Module composition

The module of which SC-HEAP E3 platform is composed is indicated on Figure 12～Error: Reference source not found.
It's the module composition when using ACCELLERA/USK edition SystemC, and indicates the

construction as of the 2012/12/E time (A phrase as 2012/12/E isn't kept in the figure title.) and 2012/8/E.
Correspondence with each module and a model IP is indicated in Table 2.

Table 2 module and model IP

【CPUSS hierarchy】(model IP name : NSMVG3MSSV01)

Module inside Figure 12～Figure 16	instance name in hierachy	model IP name
E3 3PE	G3MPE	NSMVG3MPEV01
INTC2	INTC2	NSMVINTC2V01
VCI2AHB	VCI2AHB_BRIDGE	VCI2AHB
AHB2VCI	AHB2VCI_BRIDGE	AHB2VCI
AHB BUS_ARB	AHB(AHB BUS_ARBis included in AHB)	ATLTLB64
AHB BUS_DECODE	AHB(AHB BUS_ARBis included in AHB)	ATLTLB64
PEx:VPI2APB(x is core number)	VPI2APB_BRIDGE_x (x is core number)	VPI2APB
VPI2APB_BRIDGE_DMA	VPI2APB_BRIDGE_DMA	VPI2APB
GAPB	GAPB	ATLTLB32

【PE hierarchy】(model IP name : NSMVG3MPEV01)

Module inside Figure 17、Error: Reference source not found	Instance name in hierarchy	Model IP name
ISS	G3MCPU	NSMVG3MCPUV01
INTC1	INTC1_x(x is PEID)	NSMVINTCV01
PEx:APB_router(x is core number)	LAPB_DECODER_x (x is PEID)	OSCI2DCDR
DUMMY_SLAVE_x (x is 7-number of cores)	DUMMY_SLAVE_x (x is 0 to (7-number of cores)-1)	DmySlv32

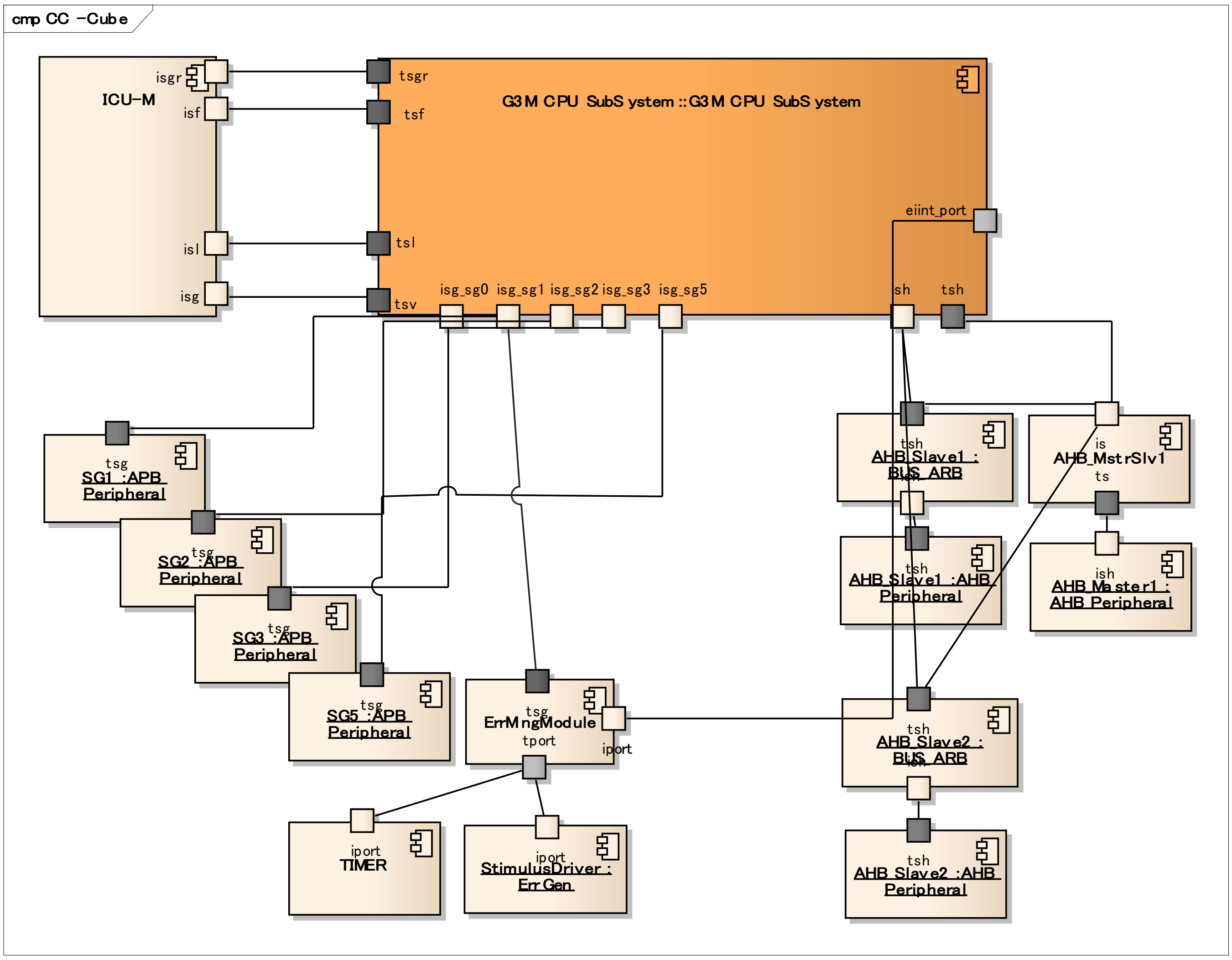


Figure 12 Platform(RH850 hierarchy) (Linux/Windows ACCELLERA/USK SystemC)

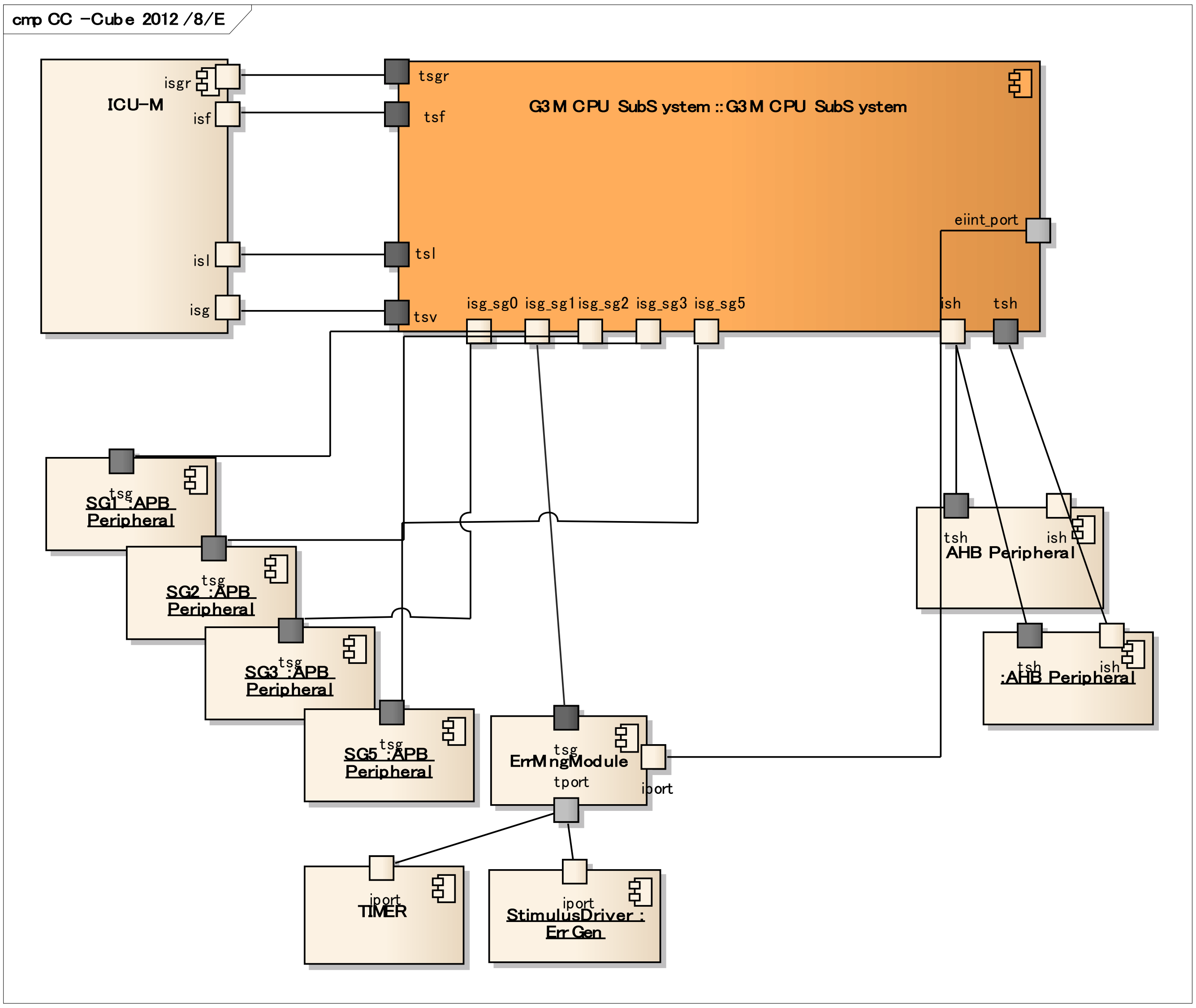


Figure 13 Platform(RH850 hierarchy) (Linux/Windows ACCELLERA/USK SystemC) on 2012/8/E

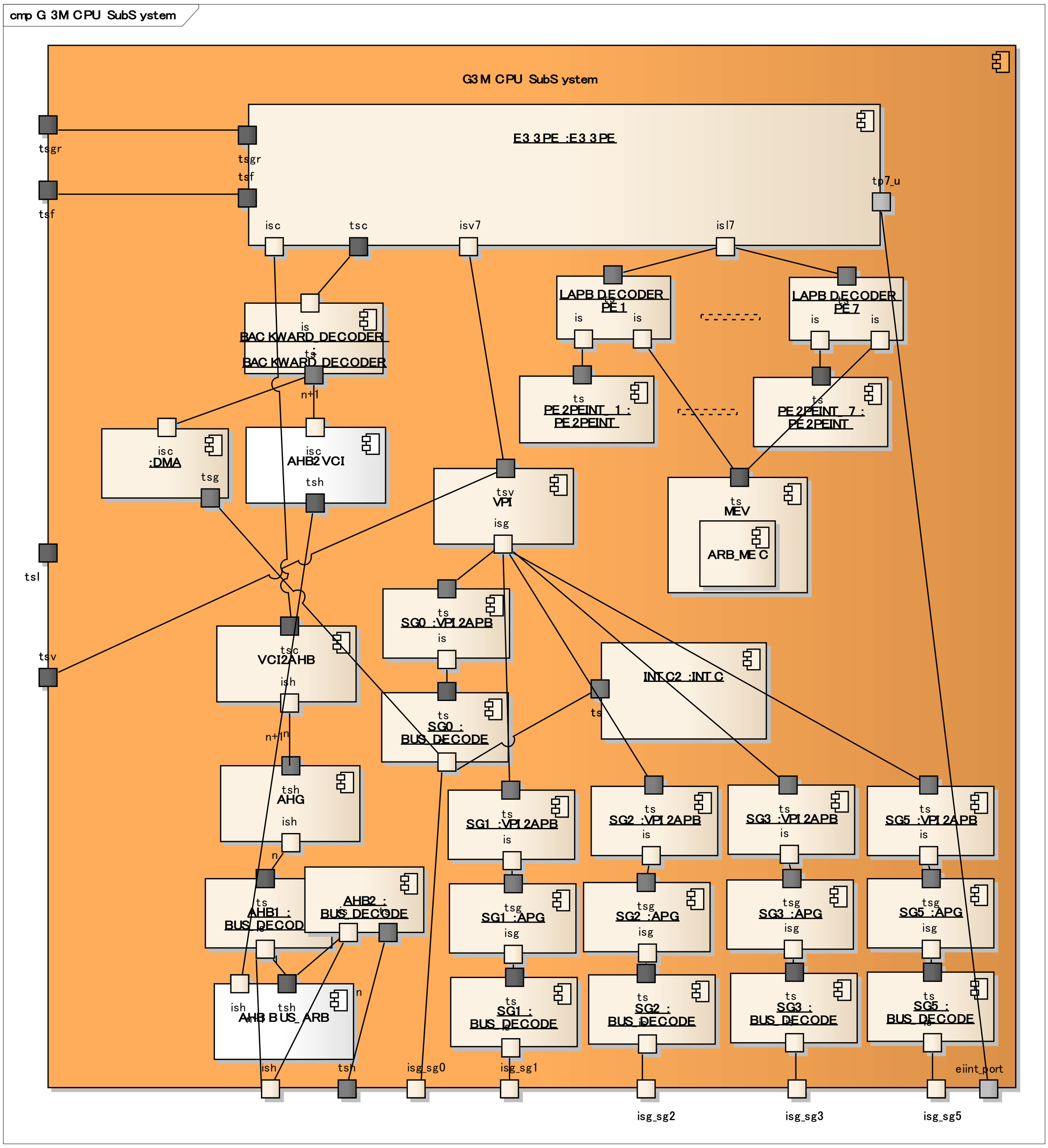


Figure 14 SC-HEAP E3 CPUSS hierarchy (Linux/Windows ACCELLERA/USK SystemC)

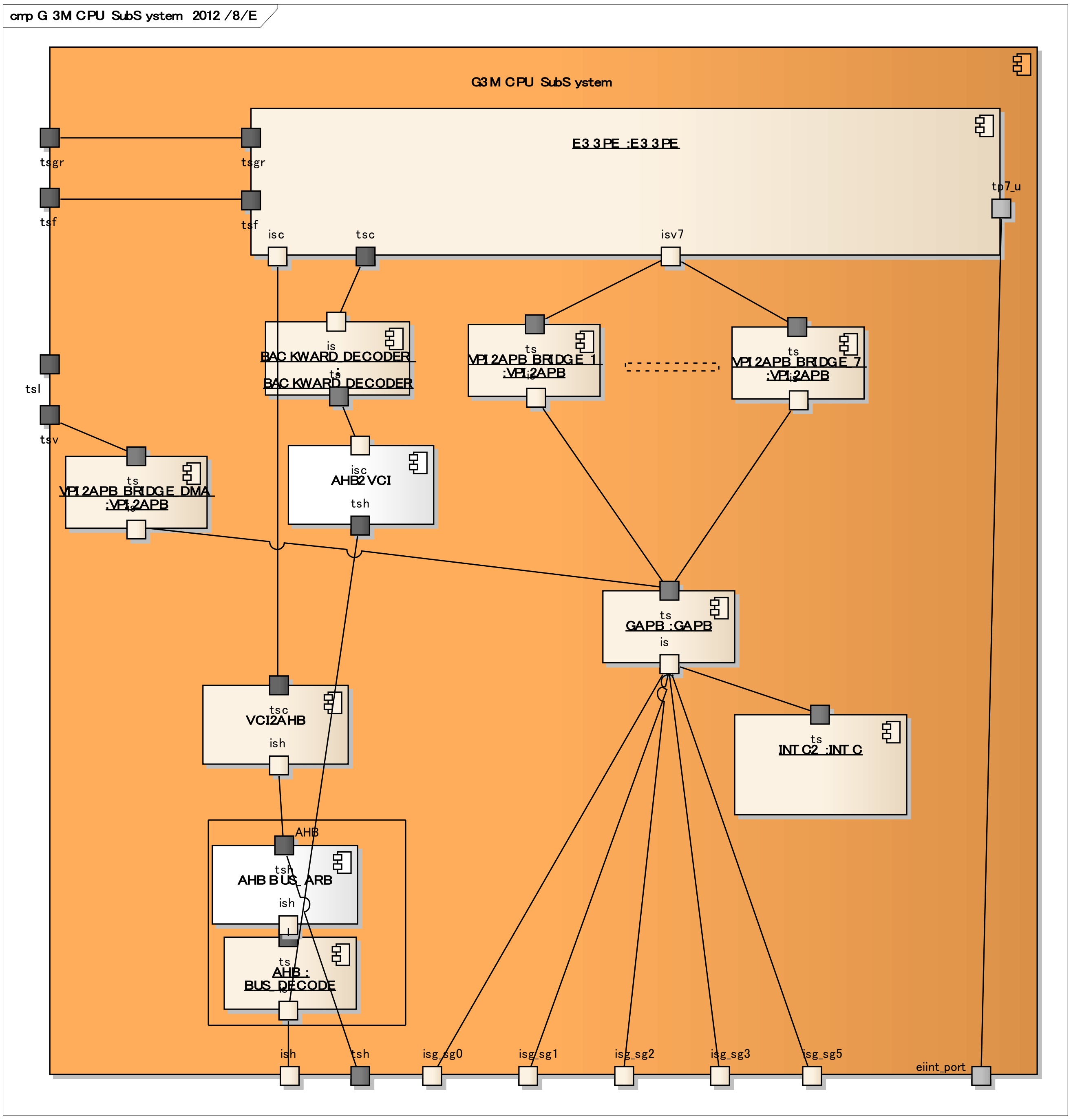


Figure 15 SC-HEAP E3 CPUSS hierarchy (Linux/Windows ACCELLERA/USK SystemC) on 2012/2/E

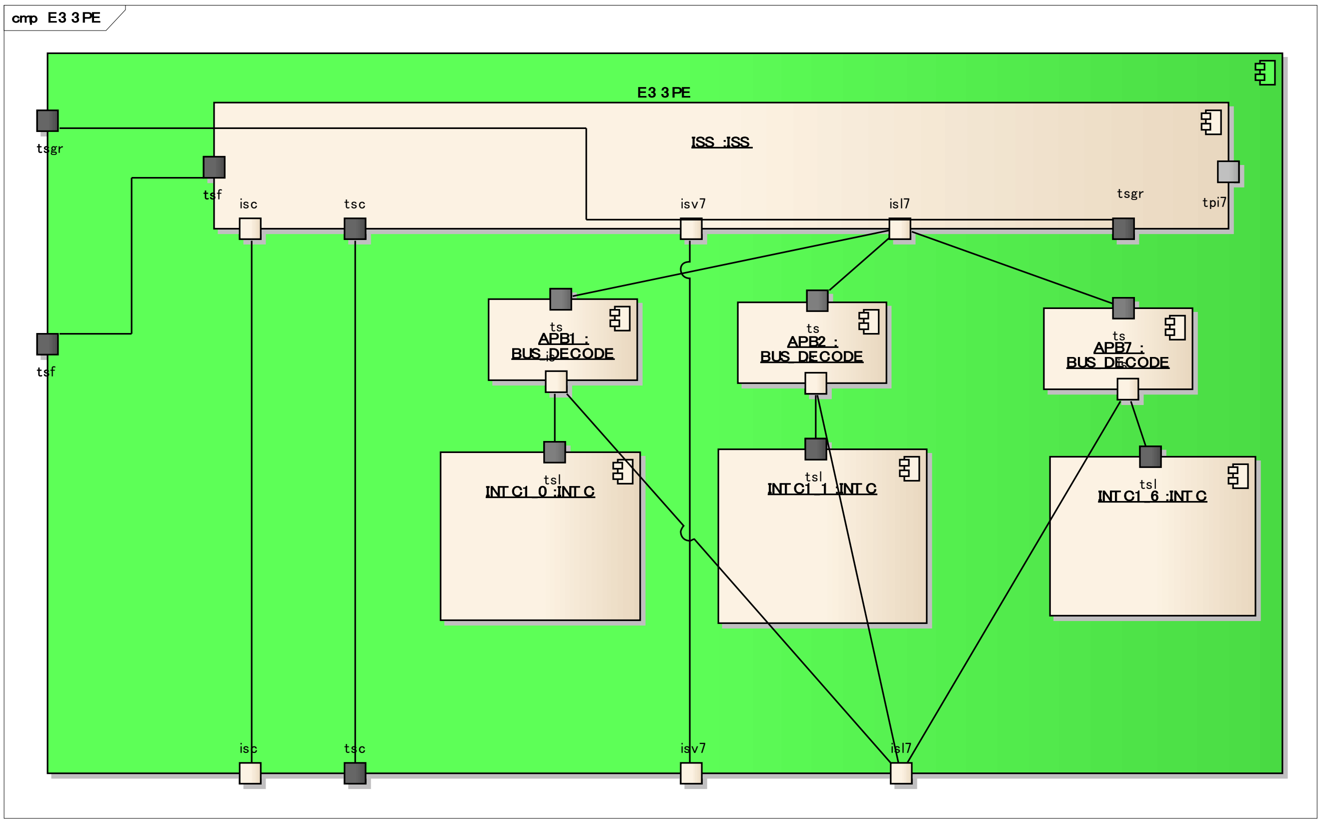


Figure 16 SC-HEAP E3 PE hierarchy (Linux/Windows ACCELLERA/USK SystemC)

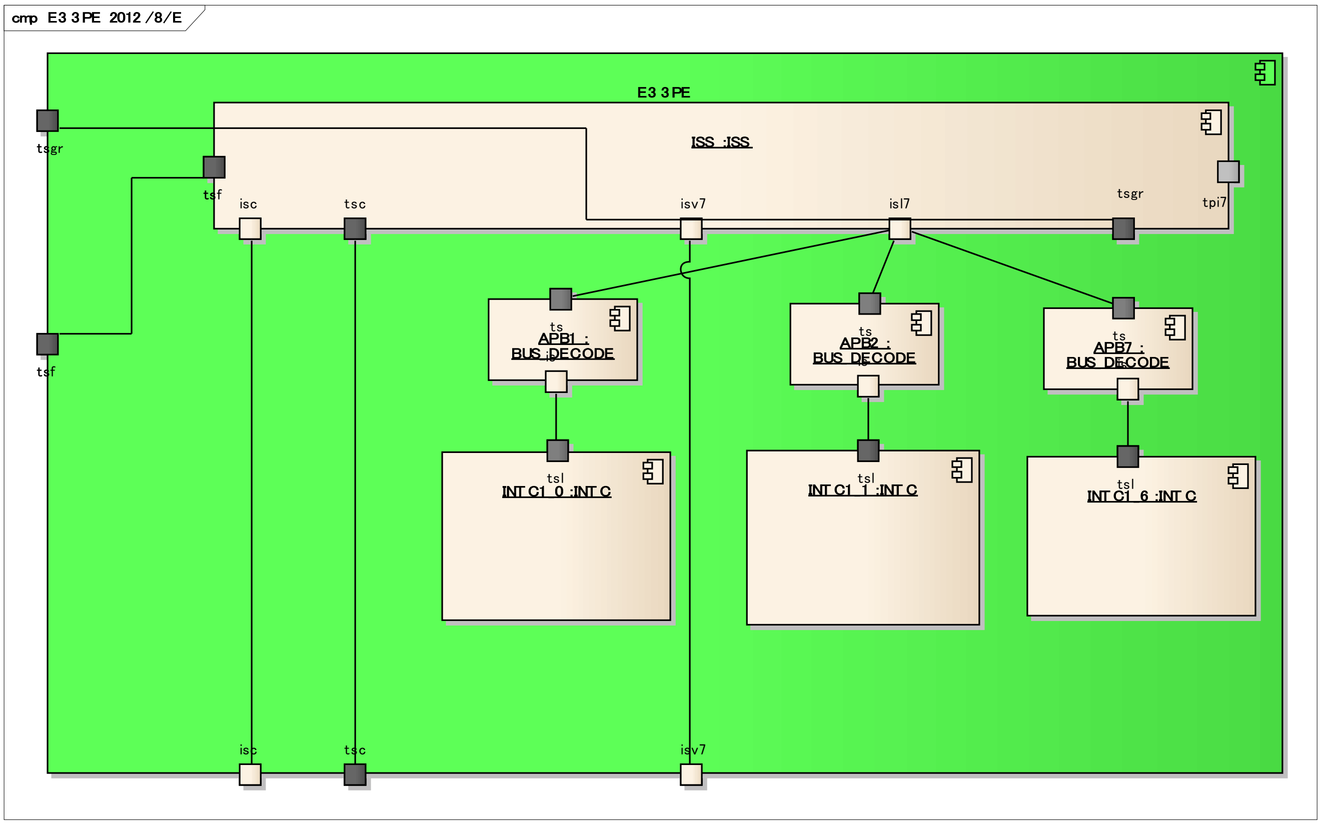


Figure 17 SC-HEAP E3 PE hierarchy (Linux/Windows ACCELLERA/USK SystemC) on 2012/2/E

2. Create simulator

The way to create a simulator from a source of a release package is indicated at this chapter. The simulator which is already made is included in a release package, but when you'd like to make a simulator from the source codes, please refer to the following procedure. It's possible to test the behavior this simulator using the simulator included in a release package or your own the simulator. We assume that the <PROJTOP> which shows in this chapter indicates the top hierarchy of the release package in Figure 1 and Figure 2.
Please refer to related document[4] about the user modeling environment, if required.

2.1. Machine environment

The machine environment in which a simulator is created on this platform is shown in Table 3.

Table 3 machine environment(create simulator)

【Linux ACCELLERA/USK 版 SystemC】

OS	Red Hat Enterprise Linux release 5.5 32bit/64bit
Compiler	g++ 4.1.2
Perl	v5.8.8
S ystemC environment	ACCELLERA SystemC 2.2.0, TLM 2.0.1 USK バージョン未定
Python	Python2.7

【Windows ACCELLERA/USK 版 SystemC】

OS	Microsoft Windows 7 32bit/64bit
Compiler	Visual Studio 2010
S ystemC environment	ACCELLERA SystemC 2.2.0, TLM 2.0.1 USK バージョン未定
Python	Python2.7

The environment when the release person in charge created a simulator on the sharing design environment,in REL is shown in Table 4 by reference.

Table 4 machine environment release person in charge uses(create simulator)(Linux ACCELLERA/USK SystemC)

Build machine	Bs command CPU sever (execute "bs -os RHEL5 -M 1000" command)
Compiler	/usr/bin/g++
SystemC environment	/proj/soft106/Heap/tools/systemc-2.2.0_lnxe5_gcc412
Python	/proj/soft109/HeapE3/tools/python/python2.7.3

Please refer to 8.4 if you check the Makefile to build the simulator.

[4] WEB-00289167-04.00J「SC-HEAP ユーザモデリング環境使用説明書」 TBD

2.2. Preliminary check point

When making a simulator, please check the following point.

【Linux ACCELLERA/USK SystemC】
Check the following make variables in <PROJTOP>/build/TOPV01/Makefile. If it’s wrong, please rewrite the Makefile.

Make variable name	meaning
SYSTEMC_HOME	Location of ACCELLERA SystemC 2.2.0
PYTHONHOME	Location of Python
PYTHON_VERSION	Python version. Use it to speficy Python Library name.
MAKE	Location of Gmake
CXX	Location of the C++ compiler

【Windows ACCELLERA/USK SystemC】
Check the following variable in the VCproject under <PROJTOP>/build/TOPV01/msvc100. Set the PATH to Microsoft Visual Studio.

Make variable name	meaning
SYSTEMC_HOME	Location of ACCELLERA SystemC 2.2.0
PYTHON_DIR	Location of Python

2.3. Procedure to create a simulator

【Linux ACCELLERA/USK SystemC】
A simulator is made by starting gmake under <PROJTOP>/build/TOPV01.
The build is done by the following procedure.

1. <PROJTOP>/build/TOPV01/Makefile calls Makefile_scheap and Makefile_scheap calls the following Makefiles.
   Makefile for each IP under <PROJTOP>/models (<PROJTOP>/models/*/Makefile)
   Makefile for ASTC IP (<PROJTOP>/models_astc/*/Makefile)
   Makefile for RVC IP (<PROJTOP>/models_rvc/*/Makefile)
   Makefile for SMPILS (<PROJTOP>/build/SMPILSV02/Makefile)
   Library for each built model IP is located under <PROJTOP>/lib-models03[5].
2. <PROJTOP>/models/iss/cforest_g3m/sim/Makefile is called.
   <PROJTOP>/build/TOPV01/Makefile_scheap calls <PROJTOP>/build/TOPV01/Makefile_scheap.tb, then the simulator is built.
3. <PROJTOP>/build/TOPV01/Makefile is called with the argument “usk” when using USK edition. A calling sequence of each Makefile is like the above.

【Windows ACCELLERA/USK SystemC】
Simulator is created with the project under <PROJTOP>/build/TOPV01/msvc100.
Simulator is created as “sim.exe” under <PROJTOP>/build/TOPV01/msvc100/Release(_usk) and

[5] Table 36に示す make 変数:LIBPATH_ROOT で変更可能。

Debug(_usk).

A build is performed by the following procedure.

1. start VisualStudio 2010 by double-click scheapE3.sln or scheapE3.vcproj under <PROJTOP>build/TOPV01/msvc100.
2. [Build]-> [batch build] is chosen from the menu.
3. If the dialogue opens, check [composition] which is set if the [build] is Debug or Release, and pushe down a [rebuilding] button.
4. When using USK edition, check [composition] which is set if the [build] is Debug_usk or Release_usk, and pushe down a [rebuilding] button.

【Notice】

- The following CForest generation file isn't generated automatically in Windows vcproj. The one compiled by Linux is copied and used in Winodws.

  inst_declaration.h, inst_id_list.h, sregfile.h, sreg.h, sreg_enum.h, trace_operand.h, cedar_arch_info.cpp, inst_func_table.cpp, sregfile_init.cpp, trace_operand_gen.cpp
- When compiling Windows, a line feed code should be corrected, if required. such as using unix2dos in cygwin,
- Even CForest is compiled with-m32 (Linux).

3. Setting for running simualtor

All setting necessary to simulator execution is explained at this chapter. It's possible to change the simulator condition by setting various attributes. Please refer to the sample in the release package to understand the concept.

3.1. Setting environmental variable

Please refer to the following environmental variables when simulator is executed..
【using USK version or RH850 peripheral macros】

variable name	set value
LD_LIBRARY_PATH （Linux）	・describe PATH of dynamic link library of peripheral macros Specify the PATH scheap/lib in Figure 1. In case of Accellera, specify the path "scheapCompile/lib/osci_release". In case of USK, specify the path "scheapCompile/lib/usk_release". ・describe the USK SystemC library path. Set /proj/soft108/Heap/tools/USK/xxxx.
PATH （Windows）	・describe PATH of dynamic link library of peripheral macros. Specify in PATH scheap/lib in Figure 1. In case of Accellera, specify the path "scheapCompile/lib/osci_release". In case of USK, specify the path "scheapCompile/lib/usk_release". ・describe in USK SystemC library path. E:¥SCHEAP¥HeapE3¥buildTest¥SC_HEAP_E3_v100¥scheapCompile¥lib
RLM_LICENSE	ASTC product license server. e.g. l5login11.design.necel.com@5035
PYTHONHOME	Path to Python e.g. /proj/soft109/HeapE3/tools/python/python2.7.3

3.2. Model configuration

担当：新井 S, 吉永 S, 大塚氏

It's necessary to set the attribute of each model IP before simulation run. The attribute value is written in a configuration file. A configuration file is analyzed by a simulator before simulation starting, and the attribute value is set for each model IP. A configuration file sample "heap.cfg" is relased as the sample under the location Figure 3. When not connecting with a Multi debugger and confirming to run the simulator by a batch mode, heap.cfg is used. It's possible to omit description of some attribute values indicated in the following table, but a configuration file must be specified at the time of a command start. A configuration file is the a text file. 1 line of the configuration file must be within 512 characters.

The attributes in Figure 12～Error: Reference source not found for each IP is shown in the tablesTable 5～Table 13.
When specifying the plural attributes by one identifier, the explained attribute is specified with the underline.

Table 5 system attribute

set attribute	identifier	setting example
The clock unit (frequency)	[FREQ]	[FREQ]=(200000000.0, SC_NS) possible specified value: The numerical value of double type
The clock unit (second)	[FREQ]	[FREQ]=(200000000.0, SC_NS) possible specified value: SC_PS, SC_NS, SC_US and SC_MS,SC_SEC

Table 6 output error message

set attribute	identifier	setting example
Error message output destination specification	[ERROR_MESSAGE_FILE]	[ERROR_MESSAGE_FILE]=ErrMsg.txt possible specified value: the file name

Table 7 ロギング機能

設定する属性	識別子	設定例
ロギング機能有効/無効 (※社外非公開)	[CM_REPORT_WORK]	[CM_REPORT_WORK]=WORK 指定可能値：WORK
ロギング機能デバッグモード (※社外非公開)	[CM_REPORT_DEBUG]	[CM_REPORT_DEBUG]=DEBUG 指定可能値：DEBUG
出力ファイル (※社外非公開)	[CM_REPORT_FILE]	[CM_REPORT_FILE]=CMREPORT.log 指定可能値：ファイル名

Table 8 端子vcd出力機能

設定する属性	識別子	設定例
vcd出力ファイル指定	[INTC VCD_FILE]	[INTC_VCD_FILE]=INTC_REQ 指定可能値：ファイル名（ファイル名指定で機能ONを兼ねる）

Table 9 Connect / unconnect peripheral macro (※use on Linux/Widnows ACCELLERA/USK edition SystemC)

set attribute	identifier	setting example
Connect / unconnect peripheral macro	[PERIPHERAL]	[PERIPHERAL]=NONE possible value: NONE、PFRH850、SMPILS、PFRH850&SMPILS

Table 10 attribute of G3MCPU

set attribute	identifier	setting example
Address mask ※ E1.00 is undealt with.	[G3MCPU_MASK]	[G3MCPU_MASK]=0xffffffff possible value: The numerical value of unsigned int type
Multi start client IP address	[V850E2R_MULTI]	[V850E2R_MULTI]=127.0.0.1 possible value: IP address of a Multi start machine
From Multi, portnumber to ISS	[V850E2R_MULTI_PORT]	[V850E2R_MULTI_PORT]=9988 possible value: The numerical value of int type
Target program	[G3MCPU_PROGRAM]	[G3MCPU_PROGRAM]=core0.hex possible value: Program pathname
The kind of debuggers	[G3MCPU_DEBUG_MODE]	[G3MCPU_DEBUG_MODE]=NONE possible value: NONE, MULTI and CUBESUITE
The kind of ISS ※ E1.00 is undealt with.	[G3MCPU_SIM_MODE]	[G3MCPU_SIM_MODE]=CAISS possible value: CAISS and FASTISS
The number of PE	[G3MCPU_PE_NUM]	[G3MCPU_PE_NUM]=3 possible value: The numerical value of 1-7
PE classification ※ E1.00 is undealt with.	[G3MCPU_PE_TYPE]	[G3MCPU_PE_TYPE]=G3M possible value: G3M, G3P and G3K
The PE number	[G3MCPU_VM_HT_NUM]	[G3MCPU_VM_HT_NUM]=(1, 2, 2) possible value: 1-numerical value specified by [G3MCPU_PE_NUM]
The number of virtual machines	[G3MCPU_VM_HT_NUM]	[G3MCPU_VM_HT_NUM]=(1, 2, 2) possible value: The numerical value of 1-8
The number of hardware threads	[G3MCPU_VM_HT_NUM]	[G3MCPU_VM_HT_NUM]=(1, 2, 2) possible value: The numerical value of 1-64
Presence of FPU ※ E1.00 is undealt with.	[G3MCPU_FPU]	[G3MCPU_FPU]=true possible value: True and false
Presence of SIMD ※ E1.00 is undealt with.	[G3MCPU_SIMD]	[G3MCPU_SIMD]=true possible value: True and false
The number of TLB entry ※ E1.00 is undealt with.	[G3MCPU_TLB_ENTRY]	[G3MCPU_TLB_ENTRY]=1 possible value: Uncertain.

The TLB smallest paging size ※ E1.00 is undealt with.	[G3MCPU_TLB_SIZE]	[G3MCPU_TLB_SIZE]=3 possible value: Uncertain.
The number of MPU territory ※ E1.00 is undealt with.	[G3MCPU_MPU_ENTRY]	[G3MCPU_MPU_ENTRY]=12 possible value: The numerical value of 0-16
SNOOZE stop period ※ E1.00 is undealt with.	[G3MCPU_SNZ_TIME]	[G3MCPU_SNZ_TIME]=32 possible value: 32 or 64 or 128 or 256.
Interrupt handler extended specifications presence ※ E1.00 is undealt with.	[G3MCPU_INT_EXP]	[G3MCPU_INT_EXP]=true possible value: True and false
MA exception presence ※ E1.00 is undealt with.	[G3MCPU_MA]	[G3MCPU_MA]=true possible value: True and false
NC:RBASE value ※ E1.00 is undealt with.	[G3MCPU_NC_RBASE]	[G3MCPU_NC_RBASE]=0x00000000 possible value: The numerical value of unsigned int type
NC:SPID value ※ E1.00 is undealt with.	[G3MCPU_MC_SPID]	[G3MCPU_MC_SPID]=0 possible value: The numerical value of 0-3
The I-Cache size ※ E1.00 is undealt with.	[G3MCPU_ICACHE]	[G3MCPU_ICACHE]=8 possible value: The numerical value of 0 or 1 or 2 or 4 or 8 or 16
The number of I-Cache ways ※ E1.00 is undealt with.	[G3MCPU_ICACHE]	[G3MCPU_ICACHE]=4 possible value: The numerical value of 1 or 2 or 4
The D-Cache size ※ E1.00 is undealt with.	[G3MCPU_DCACHE]	[G3MCPU_DCACHE]=8 possible value: Uncertain.
The number of D-Cache ways ※ E1.00 is undealt with.	[G3MCPU_DCACHE]	[G3MCPU_DCACHE]=4 possible value: Uncertain.
Read latency (ROM) ※ E1.00 is undealt with.	[G3MCPU_LATENCY]	[G3MCPU_LATENCY]=0 possible value: The numerical value of unsigned int type
Write latency (ROM) ※ E1.00 is undealt with.	[G3MCPU_LATENCY]	[G3MCPU_LATENCY]=0 possible value: The numerical value of unsigned int type
Read latency (Global RAM) ※ E1.00 is undealt with.	[G3MCPU_GROM_LATENCY]	[G3MCPU_GROM_LATENCY]=0 possible value: The numerical value of unsigned int type
Write latency (Global RAM) ※ E1.00 is undealt with.	[G3MCPU_GROM_LATENCY]	[G3MCPU_GROM_LATENCY]=0 possible value: The numerical value of unsigned int type
Read latency (Local RAM of self-PE) ※ E1.00 is undealt with.	[G3MCPU_LRAM_LATENCY]	[G3MCPU_GROM_LATENCY]=0 possible value: The numerical value of unsigned int type
Write latency (Local RAM of self-PE) **※ E1.00 is undealt with.**	[G3MCPU_LRAM_LATENCY]	[G3MCPU_GROM_LATENCY]=0 possible value: The numerical value of unsigned int type
Read latency (Local RAM of other PE) ※ E1.00 is undealt with.	[G3MCPU_LRAM_LATENCY]	[G3MCPU_GROM_LATENCY]=0 possible value: The numerical value of unsigned int type
Write latency (Local RAM of other PE) ※ E1.00 is undealt with.	[G3MCPU_LRAM_LATENCY]	[G3MCPU_GROM_LATENCY]=0 possible value: The numerical value of unsigned int type
The PC trace report output file name	[G3MCPU_PROFILE_TRACE]	[G3MCPU_PROFILE_TRACE_12]=profile_12.log possible value: The file name

The PC trace report output format	[G3MCPU_PROFILE_TRACE_FORMAT]	[G3MCPU_PROFILE_TRACE_FORMAT]=PC possible value: PC (PIPE※ closure outside the company)
PC trace report output address area ※ E1.00 is undealt with.	[G3MCPU_PROFILE_TRACE_ADDR_RANGE]	[G3MCPU_PROFILE_TRACE_ADDR_RANGE]=(0x0, 0xffffffff) possible value: The numerical value of unsigned long type
PC trace report output time area ※ E1.00 is undealt with.	[G3MCPU_PROFILE_TRACE_TIME_RANGE]	[G3MCPU_PROFILE_TRACE_TIME_RANGE]=(0x0, 0xffffffffffffffff) possible value: The numerical value of unsigned long long type
The cash trace report output file name ※ E1.00 is undealt with.	[G3MCPU_PROFILE_MEMORY]	[G3MCPU_PROFILE_MEMORY_12]=cache.log possible value: The file name
Cash trace report output address area ※ E1.00 is undealt with.	[G3MCPU_PROFILE_MEMORY_ADDR_RANGE]	[G3MCPU_PROFILE_MEMORY_ADDR_RANGE]=(0x0, 0xffffffff) possible value: The numerical value of unsigned long type
Cash trace report output time area ※ E1.00 is undealt with.	[G3MCPU_PROFILE_MEMORY_TIME_RANGE]	[G3MCPU_PROFILE_MEMORY_TIME_RANGE]=(0x0,0xffffffffffffffff) possible value: The numerical value of unsigned long long type
Cash trace report event ※ E1.00 is undealt with.	[G3MCPU_PROFILE_MEMORY_EVENT]	[G3MCPU_PROFILE_MEMORY_EVENT]=ALL possible value: ALL, I, RW
Execution summary report ※ E1.00 is undealt with.	[G3MCPU_PROFILE_TRACE_SUMMARY]	[G3MCPU_PROFILE_TRACE_SUMMARY]=exesum.log possible value: The file name

Table 11 attribute of AHB bus(AHB)

set attribute	identifier	setting example
The address map file name	[AHB_MAPFILE]	[AHB_MAPFILE]=AHB.map possible value: Character string (The address map file name is shown.)

Table 12 attribute of GAPB bus(GAPB)

set attribute	identifier	setting example
The address map file name	[GAPB_MAPFILE]	[GAPB_MAPFILE]=GAPB.map possible value: Character string (The address map file name is shown.)

Table 13 attribute of LAPB decoder(PEx_LAPB_DECODER)(x is core number)

set attribute	identifier	setting example

The address map file name	[LAPB_MAPFILE]	[LAPB_MAPFILE]=LAPB.map possible value: Character string (The address map file name is shown.)

3.3. Configuration setting for Multi

担当：大塚氏

When connecting with a Multi debugger and simulating, configuration file gen-heap.cfg is generated by Multi debugging server (rteserv2) which assigns an IP address and a port and adds a parameter to configuration file gen-heap.cfg automatically based on the configuration file heap.cfg at the time of a simulation start. A simulator executes with a Multi debugger with this setting. The attribute value of the configuration setting added automatically is shown in Table 14.

Table 14 attribute of Multi configuration file

set attribute	identifier	setting example
Multi start client IP address	[V850E2R_MULTI]	[V850E2R_MULTI]=127.0.0.1
From Multi, portnumber to ISS	[V850E2R_MULTI_PORT]	[V850E2R_MULTI_PORT]=9988

When using Multi, it's necessary to specify the license server and the path to Multi. A setting example is shown below.

- Specify license server
  ```
  setenv GHS_LMWHICH ghs
  setenv GHS_LMHOST @172.21.28.49
  ```
- Specify the PATH to Multi
  ```
  setenv MULTI /proj/soft103/lang/GHS/v800-2000_v5/linux86
  set path=($MULTI $path)
  ```

3.4. Setting address map

担当：新井Ｓ

It's necessary to set one address map for each bus before simulation run. The following address map is necessary in SC-HEAP E3 E1.00.

Instance name of Bus/bus IF	e.g. Address map file name（spcify in heap.cfg)
AHB	AHB.map
GAPB	GAPB.map
PEx_LAPB_DECODER(x is core number)	LAPB_x.map

※Even if a connected slave doesn't exist in the above address mapfile, please prepare an empty file.

An address map is written in address mapfile. The format in the address mapfile is below.

```
The slave socket name< blank> starting address< blank>
size
```

"Slave socket name" consists of slave's target socket name and an instance name with a hierarchy of the SystemC model IP (a hierarchy identifier is dot (.)). For example when the hierarchy name is

"RH850.G3MSS.G3MPE", the slave instance name is "SLAVE" and the target socket name is "ts", the slave socket name is "RH850.G3MSS.G3MPE.SLAVE.ts".
The starting address and the size can be specified by both of a decimal number or a hexadecimal number. When spcifying it as a hexadecimal number, please add "0x" as a prefix.

Besides, e.g. if the module like VCI:BUS_DECODE (Figure 14 or Figure 15) have several slaves via VCI2AHB, their start addresses and sizes should be written for same slave port name in the bus map file. This sample is shown in below.

```
RH850.G3MSS.AXI2AHB.ts     0x00500000   0x00000044
RH850.G3MSS.AXI2AHB.ts     0x00600000   0x00008000
RH850.G3MSS.AXI2AHB.ts     0x00700000   0x00000100
```

Figure 18 example of plural start address / size

A sample example of the address map which is provided in the release package is shown in Figure 19. **TBD**

```
; Address map file
;
; [ Format ]
; slave_target_port_name <Tab or Space> StartAddress <Tab or Space> Size
;
; BusSlave_target_port_name is the target_port name of the bus slave.
; It is with hierarchy path like "top.slave.target_port".
;
; Address is specified with decimal or hexadecimal.
; In the case of hexadecimal, "0x" needs to be added at the
; head of address like "0x100000"
;
; Size is           TBD
; In the c
; head of
;
; Address mask can be set as follows.
; # 0xFFFFFFF
; In above case, the transaction address is masked with
; 0xFFFFFFF in decoding.
;
; for FlashCache_15
E2SPFP.NSMVHEAPV02.CACHE_15.target_port 0x00000000   0x00040000
; for EXctrlMEM_18
E2SPFP.NSMVHEAPV02.EXMEM_18.target_port 0x00300000   0x00100000
; for EXctrlMEM_28
E2SPFP.NSMVHEAPV02.EXMEM_28.target_port 0x00400000   0x00100000
; for INTC_11
;E2SPFP.INTC_11.target_port      0x03fff100  0x00000100
E2SPFP.NSMVHEAPV02.INTC_11.target_port  0x1fff6000   0x00000460
; for DMAC_32 via bridge_17
E2SPFP.NSMVHEAPV02.BRIDGE_17.target_port     0x00500000   0x00000044
; for Slave_34 via bridge_17
E2SPFP.NSMVHEAPV02.BRIDGE_17.target_port     0x00600000   0x00008000
; for GTM24_35 via bridge_17
E2SPFP.NSMVHEAPV02.BRIDGE_17.target_port     0x1ffffb00   0x00000100
```

Figure 19 example of address (bus) map file

4. Writing a target program

担当：吉永 S
本章では、シミュレーション時に使用する RH850 マイコンソフトウェア(ターゲットプログラム)を作成する方法を示します。ここで示す手順で作成したターゲットプログラムを、Error: Reference source not found 章で示す手順にしたがって実行用ファイルに設定することにより、シミュレーションを行なうことができます。

4.1. HEAP アーキテクチャにおけるターゲットプログラム作成の注意点

担当：吉永 S, 大塚氏
Figure 10に示したサンプルプログラムでは、V850E2R_12 モジュール用と V850E2R_22 モジュール用のプログラムをそれぞれ別に作成しています。共有部分はブート処理部のみで、そのブート処理部の中でプロセッサ ID を用いた判定処理を行ない、V850E2R_12(プロセッサ ID は 1)または V850E2R_22(プロセッサ ID は 2)のスタートアップルーチンおよびメインルーチンの処理を行なっています。新たにターゲットプログラムを作成する場合には、このサンプルプログラムを参照して下さい。

なおターゲットプログ _12 と V850E2R_22 の
処理をわけ、それ以 タック領域やヒープ領
域が重ならないように

TBD
Wirte after preparing TP

作成したターゲットフ ファイルおよび Multi
起動用スクリプト(Mu ます。コンフィギュレー
ションファイルにター グラム」の属性値とし
て hex フォーマットの ファイルにターゲットプログラムを指定する場合には、Table 62 に示す「PROGRAM_1」および「PROGRAM_2」の属性値として a.out フォーマットのターゲットプログラムファイル名を指定します。

5. Run simulator

How to run a simulator is explained at this chapter. It's possible to check the behavior of this simulator using a sample program. We assume that the <PROJTOP> in this chapter is the top hierarchy of the model in the release package shown in Figure 1.
Please refer to related document[6]about the user modeling environment, if required.

5.1. Run the SystemC model only

5.1.1. Machine environment

The machine environment which carries out a simulator on this platform inTable 15.

Table 15 machine environment(use only simulator)

【Linux ACCELLERA/USK SystemC】

OS	Red Hat Enterprise Linux release 5.5 32bit/64bit
Python	Python 2.7

【Windows ACCELLERA/USK SystemC】

OS	Microsoft Windows 7 32bit/64bit
Python	Python 2.7

5.1.2. Preliminary check point

Please check the following points when running the simulator only.

【Linux ACCELLERA/USK SystemC】

1. check if < PROJTOP> /build/TOPV01/sim.x exists. If not exist, Please refer to 1.7章 and create the simulator sim.x.
2. check the following variables in < PROJTOP> /build/"execution environment directory"/run_core.csh. If not exist, please rewrite run_core.csh.

SIML_EXE	PATH to a simulator (sim.x)
RSLT_LOG	The file name of start log
HEAP_CFG	Configuration file for models
CYCL_NUM	The number of simulation cycles

3. check if the mapfile is existed in the above configuration file in the item 2.
4. check if the target program is existed in the above configuration file in the item 2.
5. check the write permission for the location where the log is output.
6. check if the disk area where the log is output is sufficient.
7. check if the script file name is specified in start option-py_scr in simulator sim.x, when inputting th command from a Python script.

【Windows ACCELLERA/USK 版 SystemC 使用時】

1. check if < PROJTOP> /build/TOPV01/msvc100/Release/sim.x exists. If not exist, Please refer to 1.7章 and create the simulator sim.x.
2. check the following variables in < PROJTOP> /build/"execution environment directory"/run_core_win.bat. If not exist, please rewrite run_core_win.bat.

SIML_EXE	PATH to a simulator (sim.exe)

[6] WEB-00289167-03.00J「SC-HEAP ユーザモデリング環境使用説明書」

RSLT_LOG	The file name of start log
HEAP_CFG	Configuration file for models
CYCL_NUM	The number of simulation cycles

3. check if the mapfile is existed in the above configuration file in the item 2.
4. check if the target program is existed in the above configuration file in the item 2.
5. check the write permission for the location where the log is output.
6. check if the disk area where the log is output is sufficient.
7. check if the script file name is specified in start option-py_scr in simulator sim.exe, when inputting th command from a Python script.

5.1.3. Execution sequence

When carrying out a simulator only, a console or a script file can be chosen for a Python command input method. In case of specifying the script file, -py_scr for the booting option of the simulator (sim.x in Linux or sim.exe in Windows) is used. In case of not specifying it, Python conslose is used.
Execution sequence is as follows.

1. If specify -py_scr , the Python script file must be prepared. If not specify –py_scr, the Python script is unncessary.

   e.g. test.py

```
SCHEAP.setFreq( 1, "SC_NS" )          # Frequency setting
SCHEAP.sc_start(  10000  )              #  Do  simulation  during  10000
cycles
```

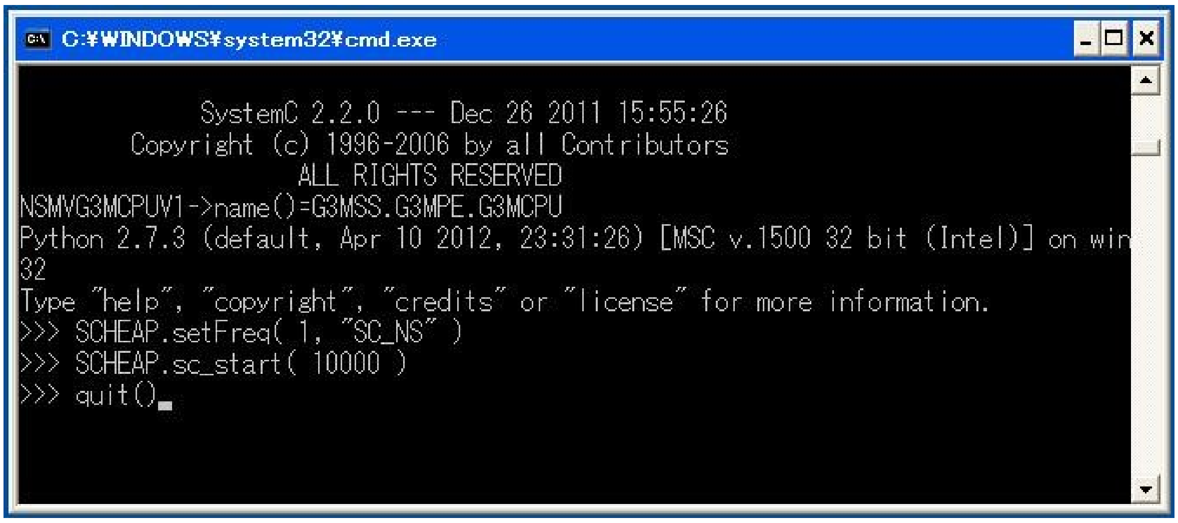
2. A script is started under each execution environment directory to carry out.

A script is started under each execution environment directory to carry out a simulator only.
A script name which corresponds to a directory of the execution environment below is as follows.

The environment	Directory	Script name
Linux ACCELLERA/USK edition SystemC	<PROJTOP>/build/"execution environment directory"	r un_core.csh
Windows ACCELLERA/USK edition SystemC	<PROJTOP>/build/"execution environment directory"	run_core_win.bat

3. A Python console boots after a simulator starts, then please input the command. ※ When specifying –py_scr, the console doesn't boot.

   Input example



```
          SystemC 2.2.0 --- Dec 26 2011 15:55:26
     Copyright (c) 1996-2006 by all Contributors
                  ALL RIGHTS RESERVED
NSMVG3MCPUV1->name()=G3MSS.G3MPE.G3MCPU
Python 2.7.3 (default, Apr 10 2012, 23:31:26) [MSC v.1500 32 bit (Intel)] on win
32
Type "help", "copyright", "credits" or "license" for more information.
>>> SCHEAP.setFreq( 1, "SC_NS" )
>>> SCHEAP.sc_start( 10000 )
>>> quit()_
```

4. After execution, the execution result is checked in a logfile.

After simulation, the start log is output in the logfile specified at section Error: Reference source not found. A result is output into the log files specified the model configuration. Please check the execution result in the those logfiles.

5.2. Run the SystemC model connected with MUTLI

5.2.1. Machine environment

A simulator on this platform is connected with Multi and the machine environment carried out is shown in Table 16.

Table 16 Machine environment (When it's connected with Multi and it's carried out.)

【Linux ACCELLERA/USK edition SystemC】

OS	Red Hat Enterprise Linux release 5.5 32bit/64bit
Python	Python 2.7
Debugger	Multi v6.1
Debugging server	rteserv2(MULTI バージョン未定 V800)

【Windows ACCELLERA/USK edition SystemC】

OS	Microsoft Windows 7 32bit/64bit
Python	Python 2.7
Debugger	Multi v6.1
Debugging server	rteserv2(MULTI バージョン未定 V800)

5.2.2. Preliminary check point

When connecting and carrying out a simulator with Multi, please check the following point.

【Linux ACCELLERA/USK edition SystemC】

1. Check if <PROJTOP>/build/TOPV01/sim.x exists. If not exist, please create the simulator sim.x referring to 1.7章.
2. Check if the following variables specified in <PROJTOP>/build/" each execution directory" /run_multi.csh is set correctly. If not correct, please rewrite run_multi.csh.

HEAP_CFG	Configuration file for models
RSLT_LOG	The file name for start log
MULT_DIR	Installed directory for Multi
SIMX_DIR	Directory of a simulator (sim.x) (※ isn't a path to sim.x.)
DSRV_DIR	Directory of the debugging server (rteserv2)
TIME_OUT	Time-out time
SOFT_PE1/PE2	Taget program(.out)
MAIN_PE1/PE2	Main function in SW on PE1/PE2 (use for displaying source code on debugger)

3. Check if the specified map file exists in the model configuration file specified in the above item 2.
4. Check if the specified target program exists in the model configuration file specified in the above item 2.
5. Check the write permission for the directory where the log is output.
6. Check if the disk area where the log is output is sufficient.

【Windows ACCELLERA/USK edition SystemC】

1. Check if <PROJTOP>/build/TOPV01/msvc100/Release/sim.exe exists. If not exist, please create the simulator sim.exe referring to 1.7章.
2. Check if the following variables specified in <PROJTOP>/build/" each execution directory" /run_multi_win.bat is set correctly. If not correct, please rewrite run_multi_win.bat.

HEAP_CFG	Configuration file for models
RSLT_LOG	The file name for start log
MULT_DIR	Installed directory for Multi
SIMX_DIR	Directory of a simulator (sim.exe) (※ isn't a path to sim.exe.)
DSRV_DIR	Directory of the debugging server (rteserv2)
TIME_OUT	Time-out time
SOFT_PE1/PE2	Taget program(.out)
MAIN_PE1/PE2	Main function in SW on PE1/PE2 (use for displaying source code on debugger)

3. Check if the specified map file exists in the model configuration file specified in the above item 2.
4. Check if the specified target program exists in the model configuration file specified in the above item 2.
5. Check the write permission for the directory where the log is output.
6. Check if the disk area where the log is output is sufficient.

5.2.3. Execution sequence

When connecting and carrying out a simulator with Multi, Python command input method are limited from a script file. The current state script file name will be also fixed as "scheap.py".
Execution sequence is shown on below.

1. Prepare the script file

   e.g. scheap.py

```
SCHEAP.setFreq( 1, "SC_NS" )          # Frequency setting
SCHEAP.sc_start(  10000  )              #  Do  simulation  during  10000
cycles
```

2. A script is started under each execution environment directory to carry out.

A script name which corresponds to a directory of the execution environment below is shown.

The environment	Directory	Script name
Linux ACCELLERA/USK edition SystemC .	<PROJTOP>/build/"execution environment directory"	r un_multi.csh
Windows ACCELLERA edition /USKSystemC	<PROJTOP>/build/"execution environment directory"	run_multi_win.bat

Useable Python methond on this platform

The useable Python method is described below.
A library is defined as “SCHEAP”. When using the following method, please be sure to add "SCHEAP." to a head.

Method name	meaning	Target class	Use example
sc_start	Simulation run	main	SCHEAP.sc_start (10000) Running 10000 cycles
setFreq	Clock setting	main	SCHEAP.setFreq (10,"SC_NS") Clock frequency is set to 10[ns] 2nd argument is used like the following. Hz: Hertz KHz:Kilo Hertz MHz:Mega Hertz GHz:Giga Hertz

6. How to check analysis result

Please refer to attached documents of ZSG-F31-11-0097-02 "LLWEB-00008479_SystemC RH850 ISS module function specification" and "specification for the trace function.ppt" for details.

7. Note

When unexpected issues have been found, check of the following setting, please.

7.1. Note about a platform

担当：新井Ｓ

none

7.2. Note about a MULTI connection

担当：大塚氏

■ Is license server setting improper?
It's necessary to establish a Multi license server to start Multi as it was indicated in 3.3. Whether this is established right (Can Multi be started independently?), please check it.

■ Inconsistency between the configuration for models and the configuration for Multi
It's necessary to adjust the attribute value set by a configuration file for models and the attribute value set by a configuration file for Multi as it was shown in 3.3. Please confirm whether these are consistent.

■ A different debugging server process is going
Please confirm if more than 2 Multi will start for same socket number.

7.3. Note about a target program

担当：吉永Ｓ

■ Inconsistency of processor ID
If the shared code for each IP is used, HTCFG0.PEID is used to judge on which processor the source code is executed. Please check the behavior on the model with the information at 6章.

■

■ RAM 領域の
本プラットフォ　V850E2R_12 モジュー
ルおよび V8　きます。自分とは別の

TBD

TP の注意事項は最後に記入予定

V850E2R モジュールによる意図せぬ RAM 領域の変更等が発生していないかを確認して下さい。特にスタック領域やヒープ領域は見落としがちなので注意して下さい。

■ 同一割込の使用

本プラットフォームでは、同一の割込アドレスを V850E2R_12 モジュールと V850E2R_22 モジュールの両方で使用することはできません。同一割込が使用されていないかを確認して下さい。

8. Explanation of platform building files

Please refer to related document[7] for details about the user modeling environment.

8.1. Explanation of NSMVRH850V01

The whole platform is bound up with the hierarchy called RH850. RH850 hierachy "NSMVRH850V01 " is instantiated in main.cpp. The following is done in RH850.

1. Declare configuration variables
2. Analyze configuration file
3. Instantiate G3MSS hierarchy instance
4. Select the connection of the peripheral macros or SMPILS and connect the user IP
5. delete G3MSS hierarchical instance

The respective contents are explained by the following item.

8.1.1. Declare configuration variables

The configuration variables used in RH850 hierarchy in SC-HEAP E3 platfrom is shown from Table 17. The simulation can be done without rebuilding the simulator by specifying values in the configuration file.

Table 17 Connect/Unconnect peripheral macros

type	Variable name	Initial value	Meaning
unsigned int	mPeripheral	none	Specify peripheral connection status

8.1.2. Analyze configuration file

The anlyzed part for the configuration file is defined. Please refer to 8.2.2 and Error: Reference source not found for the setting value and refer to Error: Reference source not found for the configuration identifier..The source code to anlzyze the configuration file is shown in Figure 20 source to analyze configuration file.

[7] WEB-00289167-03.00J「SC-HEAP ユーザモデリング環境使用説明書」

```
char config_word[512];
char config_seps[]=" ,=()¥t¥n¥r";
char *config_token;
ifstream configFile( filename );

// read the config file
while(1){
  configFile.getline(config_word, 512, '¥n');
  config_token = strtok(config_word, config_seps);
  if (configFile.eof()) {
    break;
  }
  if ((config_token == NULL) || (strncmp(config_token, "//", 2) == 0)) {
    continue;
  }
  if (strcmp(config_token, "[END]") == 0){
    break;
  }
  if (strcmp(config_token, "[PERIPHERAL]") == 0) {
    mPeripheral = get_cfg_PERIPHERAL(strtok(0, config_seps), this->name(), "read_config_file");
    continue;
  }
}
```

Figure 20 source to analyze configuration file

8.1.3. Instantiate G3MSS hierarchy

In RH850, G3M hierarchy is instatiated. Please refer to NSMVG3MSSV01 class for the hierarchy 8.2 for details.To instatiate G3MSS hierarchy, the configuration file at 8.3.1 is necessary.

A source code of instantiation for G3MSS hierarchy instance is shown on Figure 30.

```
G3MSS = new NSMVG3MSSV01("G3MSS", config_file);
```

Figure 21 source to instantiate G3MSS

8.1.4. Select the connection of the peripheral macros or SMPILS and connect the user IP

Before starting a simulation in RH850, connection of the peripheral macros or SMPILS are selected and the user IP of SMPILS are connected. It's specified by setting a value of [PERIPHERAL] of a configuration file to activate which connection (i.e. which function is called).
Each connection and the function which corresponds to those are as follows.

pltfrmPC()　　: Connection description for the user IP. The user IP, which is higher connection priority than PFC1 macros, is described.

pltfrmRH850()　: Connection description for RH850 peripheral macros.

pltfrmSmpils(): Connection description to connect Simulink

pltfrm()　　: Connection description for the user IP. After RH850 peripheral macro connection, a connection of the user IP to an open port is described. When the user describes a connection to RH850 peripheral macros, it's described here. If not used, an empty function is defined.

pltfrmPFC1GND(): It's called by default. Connect an open terminal after connections of all above.
These functions are called by the following order.
pltfrmFC() ⇒ pltfrmRH850() ⇒ pltfrm() ⇒ pltfrmSmpils() ⇒ pltfrmRH850GND()

8.1.5. Delete G3MSS hierarchical instance

Delete G3MSS hierachical instance by destructor in RH850.

The source code to generate G3MSS hierachical instance is in Figure 30.

```
delete G3MSS;
```

Figure 22 source to delete G3MSS instance

8.2. Explanation of NSMVG3MSSV01 and NSMVG3MPEV01

担当：吉永 S, 新井 S

The G3MSS hierarchy is NSMVG3MSSV01, and the G3MPE hierarchy is a class of NSMVG3MPEV01. The following declaration/definition is given in the class NSMVG3MSSV01 and NSMVG3MPEV01.

1. Declaration in port/channel
2. Declaration of configuration variable
3. Constructor definition
4. Declaration of destructor
5. Declaration of analysis process part in configuration

The respective contents are explained by the following item.

8.2.1. Declaration in port/channel

A port / channel in the model IP (i.e. same as the wire in the top hierarchy) is declared.

Ports/channels used in NSMVG3MSSV01 hierarchy of SC-HEAP E3 platform is shown in Table 18 to Table 23.

Table 18 external common port

type	Port name	meaning
slct_sc_in<sc_dt::uint64>*	PEN_sys_clock	Clock for PEN (N:1-7)
slct_sc_in<sc_dt::uint64>	VPI_clk	Clock for VPI buses
slct_sc_in<sc_dt::uint64>	VCI_clk	Clock for VCI
slct_sc_in<sc_dt::uint64>	AHB_clk	Clock fo AHB
slct_sc_in<sc_dt::uint64>	From_clk	Clock for Flash I/F
slct_sc_in<sc_dt::uint64>	Gram_clk	Clock for Global RAM
slct_sc_in<bool>	sys_reset	Reset

Table 19 external port to global APB

type	Socket name	meaning

TlmInitiatorSocket	isg_sg0	Initiator socket of SG0
TlmInitiatorSocket	isg_sg1	Initiator socket of SG1
TlmInitiatorSocket	isg_sg2	Initiator socket of SG2
TlmInitiatorSocket	isg_sg3	Initiator socket of SG3
TlmInitiatorSocket	isg_sg5	Initiator socket of SG5

Table 20 external port to global AHB slave

type	Socket name	meaning
TlmInitiatorSocket	ish	Initiator socket of AHB

Table 21 external port to global AHB master

type	Socket name	meaning
TlmTargetSocket	tsh	Target socket of AHB

Table 22 external port to INTC1

type	Port name	meaning
slct_sc_in<bool>*	feint	FE level interrupt request
slct_sc_in<bool>*	fenmi	FE level NMI request
slct_sc_in<bool>*	PEN_eiintM	PEN EI level interrupt request (N:1-7, M:0-31)
slct_sc_in<bool>*	PEN_eiint_typeM	The PEN EI level interrupt detection type designation (N:1-7, M:0-31)

Table 23 external port to INTC2

type	Port name	meaning
slct_sc_in<bool>*	eiintM	EI level interrupt request(M:32-511)
slct_sc_in<bool>*	eiint_typeM	The EI level interrupt detection type(M:32-511)

The port /channel used by NSMVG3MPEV01 hierarchy of SC-HEAP E3 platform is shown in Table 24.

Table 24 common port / channel

type	Port / channel name	meaning
TlmInitiatorSocket<64>	isc	VCI bus port ← G3MCPU
TlmInitiatorSocket<32>*	PEN_isv	VPI port ← G3MCPU
TlmInitiatorSocket<32>*	PEN_isl	Local APB port ← G3MCPU
TlmTargetSocket<128>	tsf	ICU-M → Flash I/F
TlmTargetSocket<64>	tsc	VCI bus port → G3MCPU
TlmTargetSocket<32>	tsgr	ICU-M, DMA→ Local GRAM I/F
sc_in<bool>*	feint	FE level interrupt request
sc_in<bool>*	fenmi	FE level NMI request
sc_in<bool>*	PEN_eiintlM	PEN EI level interrupt request(N:1-7, M:0-31)
sc_in<bool>*	PEN_eiint_typel	PEN EI level interrupt detection type(N:1-7, M:0-31)
sc_in<sc_dt::uint64>*	PEN_eiintlM	EI level interrupt request(M:32-511)
sc_in<sc_dt::uint64>	PEN_eiint_typeM	EI level interrupt detection type(M:32-511)
sc_in<sc_dt::uint64>	PEN_sys_clock	Clock for PEN(N:1-7)
sc_in<sc_dt::uint64>	VPI_clk	Clock for VPI
sc_in<sc_dt::uint64>	VCI_clk	Clock for VCI

sc_in<bool>	From_clk	Clock for Flash I/F
sc_in<sc_dt::uint64>	Gram_clk	Clock for Global RAM
sc_in<bool>	sys_reset	Reset

Table 25 channel between INT2 and G3MPE hierachy

type	channel name	meaning
sc_in<sc_dt::uint64>*	i2*X*_g*Y*_eiint	EI level interrupt request(*X*:0-3、*Y*:0-3)
sc_out<unsigned int>*	i1_p*N*_t*Z*_intack	EI level interrupt response(*N*:1-7、*Z*:0-3)

Table 26 MESINT-OR channel

E1.00 では無し。

type	channel name	meaning

Table 27 OR-INTC channel

E1.00 では無し。

type	channel name	meaning

Table 28 V850E3-TSU channel

E1.00 では無し。

type	channel name	meaning

Table 29 V850E3-PPU channel

E1.00 では無し。

type	channel name	meaning

Table 30 external IOB port

E1.00 では無し。

type	channel name	meaning

8.2.2. Declaration of configuration variable

The variables for the constructor arguments of each model IP used in G3MSS and G3MPE hierarchy is declared. Without doing a re-build of a simulator by setting the configuration value the user specifies for each model IP, it's possible to do the simulation with each changed value.

A variable for configurations used by the G3MSS hierarchy of SC-HEAP E3 platform is indicated from table 34.
The item which becomes the net account is a variable for configurations established besides the constructor.
"-a chisel, it's effective." and, a variable exists in itself by all except for except for in case of the condition of the designation, but anything it's is not used for the set value.

Table 31 variable for AHB bus(AHB)

type	Variable name	initial value	meaning
char[1024]	ahb_map_file	none	Address map file name

Table 32 variable for GAPB bus(GAPB)

type	Variable name	initial value	meaning
char[1024]	gapb_map_file	none	Address map file name

The configuration variables used in G3MPE hierarchy of SC-HEAP E3 platform is shown from Error: Reference source not found.
網掛けになっている項目は、コンストラクタ以外で設定されるコンフィグレーション用変数です。
なお「～のみ有効」となっているものは、指定の条件の場合以外では変数自体は存在しますが、設定値は使用されません。

Table 33 variable for LAPB decoder(Pex_LAPB_DECODER)

type	Variable name	initial value	meaning
char[1024]	lapb_map_file	none	Address map file name

8.2.3. Constructor definition

The constructor to generate an instance of class NSMVG3MSSV01 and NSMVG3MPEV01 are defined. The following sequence is done by a constructor.

1. Setting a fixed value to a variable for constructors of each model IP
2. Setting configuration value to the variable for constructors of each model IP
3. Instantiating each model IP
4. Setting the relation of connection of each model IP instance

The respective contents are explained by the following item.

Setting a fixed value to a variable for constructors of each model IP

A variable for the constructor explained at 8.2.2 are set by a fixed value.

A source in fixed value setting processing part of a variable for constructors of each model IP (excerpt of NSMVG3MSSV01) is shown in Figure 23.

```
NSMVG3MSSV01::
NSMVG3MSSV01(sc_module_name module_name, const char *config_file ):
  sc_module(module_name),
  GAPB_clk("GAPB_clk"),
  VPI_clk("VPI_clk"),
  VCI_clk("VCI_clk"),
  AHB_clk("AHB_clk"),
  From_clk("From_clk"),
  Gram_clk("Gram_clk"),
  sys_reset("sys_reset"),
  fenmi("fenmi"),
  feint("feint"),
  tsh("tsh"),
  tsgr("tsgr"),
  tsf("tsf"),
  tsv("tsv"),
  ish("ish"),
  isg_sg0("isg_sg0"),
  isg_sg1("isg_sg1"),
  isg_sg2("isg_sg2"),
  isg_sg3("isg_sg3"),
  isg_sg5("isg_sg5")
{
  char port_name[64];

  mConfigFile= config_file;
    //////// 以下省略 ////////

}
```

Figure 23 Constructor：source to set constructor variable(excerpt from NSMVG3MSSV01)

Setting configuration value to the variable for constructors of each model IP

The attribute specified by the model configuration file is set to the constructor variable explained in 8.2.2. To set the attribute value, the function read_config_file to analyze the configuration file, which is explained at 8.2.5, is used.

A source to set to the variable from the configuration value is described in Figure 24.

```
read_config_file(config_file);
```

Figure 24 Constructor：source to set configuration file

Instantiating each model IP

Each model IP which consists of SC-HEAP E3, which is explained in Error: Reference source not found, is instantiated. The arguments in the the constructor to be changed in the boot process is set with the configuration variable in 8.2.2.

A source to instantiate each model IP (excerpt of NSMVG3MPEV01) is shown in Figure 25.

```
// ----------------------------------------
mpG3MCPU2 = new NSMVG3MCPUV01("G3MCPU", mConfigFile);

//////// 以下省略 ////////
```

Figure 25 Constructor：source to instantiate each model IP(excerpt from NSMVG3MPEV01)

Setting the relation of connection of each model IP instance

The appropriate the connection is build with the instantiated IPs.

The source to connect each IP instance (excerpt of NSMVG3MPEV01) is shown in Figure 26.

```
// ----------------------------------------
  for(int i=0; i<PE_MAX_NUM; i++){
    // for isv
    sprintf(port_name, "PE%d_isv", i+1);
    PE_isv[i] = new TlmInitiatorSocket<32>(port_name);
    (*((mpG3MCPU->PE_isv)[i]))(*(PE_isv[i]));

    // for isl
    sprintf(port_name, "PE%d_isl", i+1);
    PE_isl[i] = new TlmInitiatorSocket<32>(port_name);
    (*((mpG3MCPU->PE_isl)[i]))(*(PE_isl[i]));

    // for PE's clock
    sprintf(port_name, "PE%d_sys_clk", i+1);
    PE_sys_clk[i] = new sc_in<sc_dt::uint64>(port_name);
    (*((mpG3MCPU->PE_sys_clk)[i]))(*(PE_sys_clk[i]));

    // for PE's fenmi
    sprintf(port_name, "PE%d_fenmi", i+1);
    PE_fenmi[i] = new sc_in<bool>(port_name);
    (*((mpG3MCPU->PE_fenmi)[i]))(*(PE_fenmi[i]));

    // for PE's feint
    : (省略)

    // for PE's eiint
    : (省略)
  }

  for(int i=0; i<(EI_ALL_CH_NUM - EI_INTC1_CH_NUM); i++){
    sprintf(port_name, "eiintl%d", i+EI_INTC1_CH_NUM);
    eiintl[i] = new sc_in<bool>(port_name);
    (*((mpG3MCPU->eiintl)[i]))(*(eiintl[i]));

    sprintf(port_name, "eiint_type%d", i+EI_INTC1_CH_NUM);
    eiint_type[i] = new sc_in<bool>(port_name);
     (*((mpG3MCPU->eiint_type)[i]))(*(eiint_type[i]));
  }

  (mpG3MCPU->isx)(isx);
  tsf(mpG3MCPU->tsf);
  tsx(mpG3MCPU->tsx);
  tsgr(mpG3MCPU->tsgr);

  mpG3MCPU->VPI_clk(VPI_clk);
  mpG3MCPU->Lapb_clk(Lapb_clk);
  mpG3MCPU->VCI_clk(AXI_clk);
  mpG3MCPU->From_clk(From_clk);
  mpG3MCPU->Gram_clk(Gram_clk);
  mpG3MCPU->sys_reset(sys_reset);
//////// 以下省略 ////////
```

Figure 26 Constructor : source to connect each model IP(excerpt from NSMVG3MPEV01)

8.2.4. Declaration of destructor

The destructor is defined to delete the instance created in class NSMVG3MSSV01.
The destructor deletes the instance which is created by the constructor.

The source to destruct SC-HEAP E3 platform（excerpt of NSMVG3MSSV01）is shown in Figure 27.

```
~NSMVG3MSSV01(void){
    if (G3MPE)              delete G3MPE;

//////// 以下省略 ////////
```

Figure 27 destructor source(excerpt from NSMVG3MSSV01)

8.2.5. Declaration of analysis process part in configuration

The analysis process part for the configuration file is defined for the variables which correspond to the constructor argument for each IP in SC-HEAP E3. Please refer to 8.2.2 for the set variable and to Error: Reference source not found for the identifier (attribute).

The source to analyze the configuration file (excerpt of NSMVG3MPEV01) is shown in Figure 28.

```
void read_config_file( const char *filename ){
    char      word[512];
    char      seps[] = " ,=();¥t¥n¥r";
    char      *token;
    bool      error_detected = false;
    int       reg_num;

    ifstream  configFile_(filename);

#ifdef CM_REPORT
    :
    : (省略)
    :
#endif

    // read the config file to decide the log file
    while(1) {
        configFile_.getline(word, 512, '¥n');
        token = strtok(word, seps);
        // ----------------------------------------
        // For EOF
        if (config_file.eof()) {
            break;
        }

        // ----------------------------------------
        // For comment
        if ((token == NULL) || (strncmp(token, "//", 2) == 0)) {
            continue;
        }

        // ----------------------------------------
        // detect end marker
        // NOTICE:You have to check whether token is NULL first.
        if (strcmp(token, "[END]") == 0) {
            break;
        }

        // ----------------------------------------
        // get token
        // ----------------------------------------
        // For miscellaneous
    }
```

Figure 28 source to analyze the configuration file(excerpt from NSMVG3MPEV01)

8.3. Explanation of main.cpp

担当：吉永 S, 新井 S

The SystemC simulation is start from the function sc_main. It is located in main.cpp and it is done as follows.

1. Analysis of a command argument
2. Analysis of a configuration file
3. Instantiating top hierarchy
4. 周辺マクロ／SMPILS 接続の選択、ユーザ IP の接続

5. Starting simulation
6. Calling function to finish simulation
7. Deletes top hierarchy

Each process is explained below.

8.3.1. Analysis of a command argument

In this platform, 2 arguments are received as the command arguments shown inTable 34.

Table 34 command argument

Command argument	meaning
-n <cycle number>	Specify the number of simulation cycle
-config <filename>	Specify the configuration file name
-v	Display SC-HEAP E3 model version
-help	Display how to use the comman
-py_scr <filename>	Use Python script to execute the simulator.

The source to analyze the command argument (partly excerpt) is shown in Figure 29.

```
if (argc > 1){
    for (int count = 1; count < argc; count++){
        if (strcmp(argv[count], "-n") == 0){
            cycle_number = token_to_int(argv[count + 1], "main", "-n cycle");
        }
        if (strcmp(argv[count], "-config") == 0){
            config_file = argv[count + 1];
        }
        ///// 以下省略 /////
```

Figure 29 source to analyze the command argument(excerpt)

8.3.2. Analysis of a configuration file

There is no analysis of the configuration file in sc_main function.

8.3.3. Instantiating top hierarchy

In this platform, the hierarchy RH850 should be instantiated to do the SystemC simulation. Please refer to 8.1 about class NSMVRH850V01. The configuration file name in 8.3.1 is necessay to instantiate the RH850 hierarchy.

The source to instantiate the top hierarchy is shown in Figure 30.

```
RH850 = new NSMVRH850V01( “RH850” ,config_file);
```

Figure 30 instantiate top instance

8.3.4. Starting simulation

In this platform sc_start is used to start the simulation. The function sc_start is called linking with the Python command[8]. sc_main function boots Python to receive the Python command.

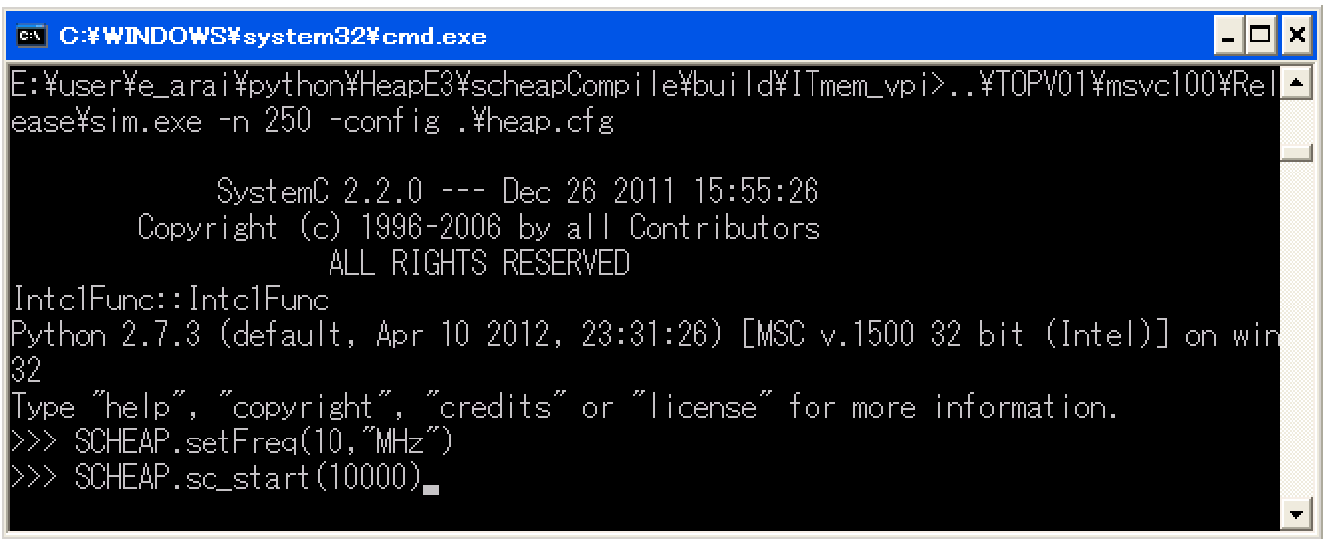
The source to call the booting process of the simulation is shown in Figure 31.

```
sc_start( cycle_number * glb_clock_period, glb_clock_time_unit );
```

Figure 31 source to start simulation

Before entering Python command "sc_start" to start the simulation, the clock frequency should be set. It is set by "setFreq" command.

The example to start simulation is in Figure 31.



```
C:¥WINDOWS¥system32¥cmd.exe
E:¥user¥e_arai¥python¥HeapE3¥scheapCompile¥build¥ITmem_vpi>..¥TOPV01¥msvc100¥Rel
ease¥sim.exe -n 250 -config .¥heap.cfg

             SystemC 2.2.0 --- Dec 26 2011 15:55:26
        Copyright (c) 1996-2006 by all Contributors
                    ALL RIGHTS RESERVED
Intc1Func::Intc1Func
Python 2.7.3 (default, Apr 10 2012, 23:31:26) [MSC v.1500 32 bit (Intel)] on win
32
Type "help", "copyright", "credits" or "license" for more information.
>>> SCHEAP.setFreq(10,"MHz")
>>> SCHEAP.sc_start(10000)_
```

Figure 32 example to input command when starting simulation

8.3.5. Finish simulation

In this platform, Python is finished after simulation.

The part to finish Python is as follows.

```
mShPythonAPI.DestructorPy();
```

Figure 33 Source code to finish Python

8.3.6. Delete top hierachy

RH850 hierachy as top hierarchy is deleted.

The code to delete RH850 hierachy is as follows.

```
delete RH850;
```

Figure 34 Soruce code to delete RH850 hierarchy

8.3.7. Calling function to finish simulation TBD

[8] Please refer to about Python command.

本プラットフォームでは、シミュレーションを通じて得られたキャッシュ統計情報を出力するためにインスタンスで定義した end_of_simulation 関数を利用します。

シミュレーション終了時処理の呼び出し処理部のソースを Figure 35 に示します。

```
E2SPFP->end_of_simulation();
```

Figure 35 source to finish simulation

8.4. Explanation of Makefile

担当：吉永 S

A simulator on SC-HEAP E3 platform is built with each model IP, each Makefile (in case of this release, Makefile_scheap : Makefile for Linux ACCELLERA/USK edition simulators) and top Makefile. In this chapter, top Makefile is explained.
Besides, this Makefile is used for Linux ACCELLERA/USK edition SystemC.

8.4.1. Linux ACCELLERA/USK edition SystemC

担当：吉永 S

.1. Environment variable needed before Makefile

担当：吉永 S

Please refer to the following environmental variables when Makefile is used.

【with ACCELLERA】

Variable name	The set value
SYSTEMC_HOME	Path to ACCELLERA SystemC library default value is set in Makefile_scheap.
OSCI_TLM_PATH	ACCELLERA TLM library path default value is set in Makefile_scheap.tb.
PYTHONHOME	Path to Python installed directory
PYTHON_VERSION	Python version
SCHEAP_HOME	Path to SCHEAP

【with USK】

Variable name	The set value
LD_LIBRARY_PATH	Specify path to USK SystemC library Default value is /proj/soft108/Heap/USK/lib/
USK_HOME	Path to USK SystemC library

【with peripheral macros】

Variable name	The set value
LD_LIBRARY_PATH	Sepcify path to dynamic link library of RH850 peripheral macros Specify path to scheap/lib in Figure 1

.2. make target

担当：吉永 S

The top Makefile has the make target in Table 35.

Table 35 make target in top Makefile

Target name	The function
default	Call Makefile to create simulator / SC-HEAP E3 library. ACCELLERA SystemC library is used. Optimization flag etc. is not specified and use the setting value in Makefile_scheap to be called later on.
release	Makefile for /SC-HEAP E3 library is used to build the simulator in a release mode. ACCELLERA SystemC library is used.
Debug	Call Makefile to create simulator / SC-HEAP E3 library under debug mode, Accellera SystemC library is used.
Clean	A clean target of Makefile for /SC-HEAP E3 library is started and diretories to locate the libraries are deleted.
release_usk	Run Makefile to build simulator and create SC-HEAP E3 library in release mode by specifying USK for SystemC.
debug_usk	Run Makefile to build simulator and create SC-HEAP E3 library in debug mode by specifying USK for SystemC.
Clean-usk	Run clean target for USK
Version	Output build environment information

.3. Structure of Makefile

担当：吉永 S

The top Makefile has the followings.

1. Setting Make variables
2. Target action

The respective contents are explained by the following item.

Setting Make variables

The make variable indicated in Table 36 is used in top Makefile. The make variable where the cycle is marked for "Environmental dependency" should be checked to create the simulator in the other environment not described in Error: Reference source not found. The make variables marked with the circle for “overwrite” is overwritten when Makefile for each model IP or Makefile_scheap for simulator is called.

Table 36 make variable in top Makefile

Make variable name	Environmental dependency	Over write	The meaning
PROJ_HOME		○	Location of project top
MODE			Release / debug
OPTFLAG			Optimized flg
DBGFLAG			Debug flag
SYSTEMC_HOME		○	location of SystemC
OSCI_TLM_PATH		○	Location of ACCELLERA TLM header file
USK_HOME		○	USK home directory

LIBPATH_ROOT			Location of model IP library objects
PROJ_HOME			Locatiojn of project
KERNEL_LIB_POSTFIX			Type of simulator kernel. Use as the string added to target name.
TARGET_ARCH			Target environment
SIMTARGET			Executed file name to build simulator
TARGET_A_SCMAIN			Library name of simulator contained with sc_main
TARGET_A			Simulator library name execluded with sc_main

Target action

A target action of top Makefile is shown on Figure 36 to Figure 43.

```
default:
         @printf "Building the CForest (RELEASE) ...¥n"
         @(cd ../../models/iss/cforest_g3m/sim; make scheap)
         @printf "Building the SCHEAP-E3-G5 (RELEASE) ...¥n"
         @(make -f Makefile_scheap)
```

Figure 36 default target action in top Makefile

```
release:
         @printf "Building the CForest (RELEASE) ...¥n"
         @(cd ../../models/iss/cforest_g3m/sim; make scheap)
         @printf "Building the SCHEAP-E3-G5 (RELEASE) ...¥n"
         @(make -f Makefile_scheap ¥
         MODE=release                ¥
         OPTFLAG='-m32 -O3'          ¥
         DBGFLAG='-DNDEBUG')
```

Figure 37 release target action in top Makefile file

```
debug:
         @printf "Building the CForest (DEBUG) ...¥n"
         @(cd ../../models/iss/cforest_g3m/sim; make scheap)
         @printf "Building the SCHEAP-E3-G5 (DEBUG) ...¥n"
         @(make -f Makefile_scheap ¥
         MODE=debug                  ¥
         OPTFLAG='-m32 -O0'          ¥
         DBGFLAG='-ggdb -DDEBUG')
```

Figure 38 debug target action in top Makefile

```
clean:
         @printf "Cleaning the CForest ...¥n"
         @(cd ../../models/iss/cforest_g3m/sim; make clean)
         @(pwd)
         @printf "Cleaning the SCHEAP-E3-G5 ...¥n"
         @(make -f Makefile_scheap clean)
```

Figure 39 clean target action in top Makefile

```
usk:     release-usk
release-usk:
         @printf "Building the CForest (RELEASE) ...¥n"
         @(cd ../../models/iss/cforest_g3m/sim; make scheap)
         @printf "Building the SCHEAP-E3-G5-USK (RELEASE) ...¥n"
         @make -f Makefile_scheap MODE=release SYSTEMC_LIB=usk   ¥
                  OPTFLAG='-m32 -O3' DBGFLAG='-DNDEBUG'          ¥
                  SYSTEMC_HOME=${USK_HOME}/systemc               ¥
                  OSCI_TLM_PATH=${USK_HOME}/TLM2/include/tlm     ¥
                  LIBPATH_ROOT=$(PROJ_HOME)/lib-usk-models03     ¥
                  KERNEL_LIB_POSTFIX=-USK TARGET_ARCH=CentOS4    ¥
                  SIMTARGET='sim-usk.x' TARGET_A='SCHEAP-E3-G5-USK.a'
```

Figure 40 usk and release-usk target action in top Makefile

```
debug-usk:
         @printf "Building the SCHEAP-G5-USK (DEBUG) ...¥n"
         @make -f Makefile_scheap MODE=release SYSTEMC_LIB=usk   ¥
                  OPTFLAG='-m32 -O0' DBGFLAG='-ggdb -DDEBUG'     ¥
                  SYSTEMC_HOME=${USK_HOME}/systemc               ¥
                  OSCI_TLM_PATH=${USK_HOME}/TLM2/include/tlm      ¥
                  LIBPATH_ROOT=$(PROJ_HOME)/lib-usk-models03     ¥
                  KERNEL_LIB_POSTFIX=-USK TARGET_ARCH=CentOS4    ¥
                  SIMTARGET='sim-usk.x'
```

Figure 41 default-usk target action in top Makefile

```
clean-usk:
         @printf "Cleaning the CForest ...¥n"
         @(cd ../../models/iss/cforest_g3m/sim; make clean)
         @printf "Cleaning the SCHEAP-E3-G5-USK ...¥n"
         @make -f Makefile_scheap SYSTEMC_LIB=usk                ¥
                  SYSTEMC_HOME=${USK_HOME}/systemc               ¥
                  OSCI_TLM_PATH=${USK_HOME}/TLM2/include/tlm     ¥
                  LIBPATH_ROOT=$(PROJ_HOME)/lib-usk-models03     ¥
                  KERNEL_LIB_POSTFIX=-USK TARGET_ARCH=CentOS4    ¥
                  SIMTARGET='sim-usk.x' clean
```

Figure 42 clean-usk target action in top Makefile

```
version:
        @make -f Makefile_scheap version
```

Figure 43 version target action in top Makefile

8.5. Explanation of Makefile_scheap

担当：吉永 S、修正後、新井 S, 大塚氏の加筆

Each Makefile to build each model IP by calling Makefile_scheap. The structure of Makefile_scheap is explained in this section.
Makefile_scheap is used for Linux ACCELLERA/USK edition SystemC.

8.5.1. make target

Makefile_scheap has make target in the table.

Table 37 make target in Makefile_scheap

Target name	The function
all	A directory for library is made and Makefile for each model IPs and simulator is called.
clean	Clean target for each model IP and Makefile for simulation builds are started and a directories for libraries are eliminated.
version	Display version

8.5.2. Structure of Makefile_scheap

Makefile_scheap has the following structures.

1. Setting make variables
2. Target action

The respective contents are explained by the following item.

Setting make variables

Makefile_scheap uses the make variable in Table 38. The make variable where the cycle is marked for "Environmental dependency" should be checked to create the simulator in the other environment not described in Error: Reference source not found.The make variables marked with the circle for “overwrite” row is overwritten when Makefile for each model IP or Makefile_scheap.tb for simulator is called.

Table 38 make variable in Makefile_scheap

Make variable name	Environmental dependence	address	The meaning
MODEL		○	Model name (It's used only in Makefile for model IPs.)
SIMTARGET		○	Simulator name (It's used only in Makefile for simulator builds.)
PROJ_HOME		○	The location of the project top
LIBPATH_ROOT		△	Library output place (It's used at the time of a LIBPATH address.)
MODELPATH_ROOT			Model IP source allocating place
MODEL_COMMON			Model name of a model IP sharing source
MODEL_COMMON_BUS			The bus related inclusion file stock folder name

MODEL_NSMVRH850V01			IP model name of RH850 model
MODEL_NSMVG3MSSV01			IP model name of G3MSS model
MODEL_NSMVG3MPEV01			IP model name of G3MPE model
MODEL_NSMVG3MCPUV01			IP model name of G3MCPU model
MODEL_FASTISS			IP model name of FASTISS model
MODEL_BUS32			Model name of ATLTLB32 model
MODEL_BUS64			Model name of ATLTLB64 model
MODEL_VPI2APB			Model name of VPI2APB model
MODEL_VCI2AHB			Model name of VCI2AHB model
MODEL_AHB2VCI			Model name of AHB2VCI model
MODEL_NSMVINTC1V01			Model name of INTC1
MODEL_NSMVINTC2V01			Model name of INTC2
LIBPATH_ASTC_ROOT			Location to ASTC library
MODEL_ASTC_HOME			Location of ASTC model IP srouce codes
BUILD_PF_HOME			Root directory of PFV01
BUILD_SMPILS_HOME			Root directory of SMPILSV01
INCLUDE_ASTC_PATH			Location of ASTC model include files
MODEL_ASTC_PATH			=MODEL_ASTC_HOME
MODEL_ASTC_DMA_PATH			Location of ASTC DMA model IP
BUILD_PF_PATH			Location of PFV01
PFRH850_MOD_PATH			Location of PFRH850
BUILD_SMPILS_PATH			Location of SMPILSV01
BUILD_PFV01			Path to PFV01
BUILD_SMPILS			Path to SMPILSV01
SYSTEMC_HOME	○	○	The location of Accellera or USK SystemC environment
TARGET_ARCH	○	○	The target environment
SYSTEMC_INCPATH			The location of ACCELLERA or USK SystemC Header
SYSTEMC_LIBPATH			The location of ACCELLERA or the USK SystemC library
SYSTEMC_LIB		○	Identifier in ACCELLERA or USK SystemC library (It's used only in Makefile for simulator builds.)
MAKE	○	○	Location of gmake
RM	○		RM name
RM_OPT	○		RM option
CXX	○	○	The location of the C++ compiler
OPTFLAG	○	○	Optimization flag
DEFFLAG	○	○	Defined macros
DBGFLAG	○	○	Debugging flag
INCPATH	○	○	Include path
INC_PFV01PATH			Include path to PFV01
INC_SMPILSV01PATH			Include path to SMPILSV01
PYTHONHOME	○	○	Python installed directory path
PYTHON_VERSION	○	○	Python version
PYTHON_INCPATH	○	○	Python include path
PYTHON_LIBPATH	○	○	Python library path

For your information, the setting value for make variables regaring compilation in the release package is explained.

Table 39 setting value of C++ compile option

Make variable name	The set value and its meaning
OPTFLAG	-m32 -g
	Specify 32bit execuation format on 64bit machine and creating debugging objects
DEFFLAG	-DLINUX_DEF
	LINUX : reused source uses, but the reason is unknown. Affect the build models/{common}.
DBGFLAG	(not specify)
	Not specify to output debugging information
INCPATH	-I$(SYSTEMC_INCPATH) -I. -I$(PYTHON_INCPATH) -I$(MODELPATH_ROOT)/$(TLM_INCLUDE_HEADER) -I$(MODELPATH_ROOT)/$(MODEL_COMMON) -I$(MODELPATH_ROOT)/$(MODEL_NSMVG3MSSV01) -I$

	(MODELPATH_ROOT)/$(MODEL_NSMVG3MPEV01) -I$(MODELPATH_ROOT)/$(MODEL_NSMVG3MCPUV01) -I$(MODELPATH_ROOT)/$(MODEL_ATLTLB32) -I$(MODELPATH_ROOT)/$(MODEL_ATLTLB64) -I$(MODELPATH_ROOT)/$(MODEL_VPI2APB) -I$(MODELPATH_ROOT)/$(MODEL_AXI2AHB) -I$(MODELPATH_ROOT)/$(MODEL_AHB2AXI) -I$(MODELPATH_ROOT)/$(MODEL_ATLTSLAVE32) -I$(MODELPATH_ROOT)/$(MODEL_ATLTSLAVE64) -I$(MODELPATH_ROOT)/$(MODEL_NSMVINTC1V01) -I$(MODELPATH_ROOT)/$(MODEL_NSMVINTC2V01) -I$(MODELPATH_ROOT)/$(MODEL_FASTISS)/core/rh850 -I$(MODELPATH_ROOT)/$(MODEL_FASTISS)/runtime -I$(MODELPATH_ROOT) -I$(OSCI_TLM_PATH) -I$(PFRH850_MOD_PSTH) -I$(MODEL_ASTC_DMA_PATH) -I$(MODEL_ASTC_HOME)
	Specify to add include path to Accellera / USK SystemC header, current directory, model coman header files, G3MSS header files, G3MPE header files and G3MCPU header files, or include path to Accellera / USK SystemC library path.
INC_PFV01PATH	-I$(INCPATH) -I$(PYTHON_INCPATH) -I$(MODELPATH_ROOT)/$(MODEL_NSMVG3MSSV01) -I$(MODEL_ASTC_PATH) -I$(MODEL_ASTC_DMA_PATH)
	Include path passed to PFV01 Makeifle
INC_SMPILSV01PATH	-I$(INCPATH) -I$(PYTHON_INCPATH) -I$(MODELPATH_ROOT)/$(MODEL_NSMVG3MSSV01)
	Include path passed to SMPILSV01 Makefile

Target action

Target action of Makefile_scheap is shown on Figure 44 to Figure 46.

```
all:
        if test ! -d $(LIBPATH_ROOT); then mkdir $(LIBPATH_ROOT); fi;
        (cd $(MODELPATH_ROOT)/$(MODEL_COMMON);                          ¥
         $(MAKE) -f Makefile                                            ¥
         MODEL="$(MODEL_COMMON)"                                        ¥
         PROJ_HOME="$(PROJ_HOME)"                                       ¥
         LIBPATH="$(LIBPATH_ROOT)/$(MODEL_COMMON)"                      ¥
         SYSTEMC_HOME="$(SYSTEMC_HOME)"                                 ¥
         TARGET_ARCH="$(TARGET_ARCH)"                                   ¥
         MAKE="$(MAKE)"                                                 ¥
         CXX="$(CXX)"                                                   ¥
         OPTFLAG="$(OPTFLAG)"                                           ¥
         DEFFLAG="$(DEFFLAG)"                                           ¥
         DBGFLAG="$(DBGFLAG)"                                           ¥
         INCPATH="$(INCPATH)"                                           ¥
        )
        //////// 以下省略 ////////
```

Figure 44 all target action in Makefile_scheap(excerpt)

```
clean:
        (cd $(MODELPATH_ROOT)/$(MODEL_COMMON);                          ¥
         $(MAKE) -f Makefile                                            ¥
         LIBPATH="$(LIBPATH_ROOT)/$(MODEL_COMMON)"                      ¥
        clean )
        (cd $(MODELPATH_ROOT)/$(MODEL_NSMVG3MSSV01);                    ¥
         $(MAKE) -f Makefile                                            ¥
         LIBPATH="$(LIBPATH_ROOT)/$(MODEL_NSMVG3MSS01)"                 ¥
        clean )
        //////// 以下省略 ////////
```

Figure 45 clean tatget action in Makefile_scheap(excerpt)

```
version:
        @echo "#### build machine and OS version"
        @uname -a
        @echo " "
        @echo "#### g++ version"
        @$(CXX) --version
        @echo " "
        @echo "#### gmake version"
        @$(MAKE) --version
        @echo " "
        @echo "#### OSCI version"
        @(strings $(SYSTEMC_LIBPATH)/lib$(SYSTEMC_LIB).a | grep "SystemC" )
        @echo " "
        @echo "#### IP component version"
          @(cd $(MODELPATH_ROOT); cvs status | egrep -e "========" -e "File:" -e
"revision" )
```

Figure 46 version target action in Makefile_scheap(excerpt)

8.6. Explanation of Makefile_scheap.tb

担当：吉永S、修正後、新井S, 大塚氏の加筆

Building a simulator on SC-HEAP E3 platform and a library of SC-HEAP E3 is done by calling Makefile_scheap.tb after calling each Makfile to build each model IP in Makefile_scheap. In the section, the structure of Makefile_scheap.tb to build the simulator is explained.

8.6.1. make target

Makefile_scheap.tb has the make target in Table 40.

Table 40 make target in Makefile_scheap.tb

Target name	The function
All	Target $(SIMTARGET) is started.
$(SIMTARGET)	Build the simulator specified by $(SIMTARGET).
Clean	Remove files compiled by build
$(LIBPATH)/main.o	Main object
$(LIBPATH)/pltfrm.o	Peripheral macros connection(※ dummy file).

8.6.2. Structure of Makefile_scheap.tb

Makefile_scheap.tb has the following structures.

1. Setting make variables
2. Setting build rule
3. Target action
4. ヘッダファイル／ライブラリファイルのコピー

The respective contents are explained by the following item.

Setting make variables

The make variable shown in Table 41 is used in Makefile_scheap.tb. "Environmental dependency" should be checked to create the simulator in the other environment not described in Error: Reference source not found.The make variables marked with the circle for “overwrite” row is overwritten by the upper level Makefile.

Table 41 make variable in Makefile_scheap.tb

Make variable name	Environmental dependence	Over write	The meaning
SIMTARGET		○	Simulator name
TARGET_A_SCMAIN		○	Library of SC-HEAP E3 including sc_main
TARGET_A		○	Library of SC-HEAP E3 exncluding sc_main
PROJ_HOME		○	The location of the project top
LIBPATH		○	Library output place
MODEL_HOME			Model IP source allocating place
MODEL_COMMON_PATH			The location of the model IP sharing source
NSMVRH850V01_PATH			Location of RH850 model IP source
NSMVG3MSSV01_PATH			Location of G3MSS model IP source
COMMON_A			Model IP shared library name
NSMVRH850V01_A			RH850 model IP library name
NSMVG3MSSV01_A			G3MSS model IP library name
NSMVG3MPEV01_A			G3MPE model IP library name
NSMVG3MCPUV01_A			G3MCPU model IP library name
ATLTLB32_A			32-bit bus model IP library name
ATLTLB64_A			64-bit bus model IP library name
VPI2APB_A			VPI2APB model IP library name
VCI2AHB_A			VCI2AHB model IP library name
AHB2VCI_A			AHB2VCI model IP library name
NSMVINTC1V01_A			INTC1 model IP library name
NSMVINTC2V01_A			INTC2 model IP library name
CFOREST_A			CFOREST library name
CFOREST_CEDARE3V5_A			CFOREST CEDARE3V5 library name
FASTISS_CORE_ASTC_A			FastISS core library name
FASTISS_RUNTIME_ASTC_A			FastISS runtime library name
FASTISS_ASTC_A			FastISS runtime library name
LIB_ASTC_DMA_A			ASTC DMA モデル IP ライブラリ名
LIB_ASTC_IOPORTS_A			ASTC IOPORTS モデル IP ライブラリ名
LIB_ASTC_RTC_A			ASTC RTC モデル IP ライブラリ名
LIB_ASTC_CSIH_A			ASTC CSIH モデル IP ライブラリ名
LIB_ASTC_CSIG_A			ASTC CSIG モデル IP ライブラリ名
LIB_ASTC_ADC_A			ASTC ADC モデル IP ライブラリ名
LIB_ASTC_TIMERJ_A			ASTC TIMERJ モデル IP ライブラリ名
LIB_ASTC_TIMERA_A			ASTC TIMERA モデル IP ライブラリ名
LIB_ASTC_WDTA_A			ASTC WDTA モデル IP ライブラリ名
LIB_ASTC_STIM_A			ASTC Stimulus Driver モデル IP ライブラリ名
LIB_ASTC_PFV01_A			ASTC PFV01 ライブラリ名
LIB_ASTC_PRCCM_A			ASTC PRCCM モデル IP ライブラリ名
LIB_ASTC_TAOPA_A			ASTC TAOPA モデル IP ライブラリ名
LIB_ASTC_LMA_A			ASTC LMA モデル IP ライブラリ名
LIB_ASTC_VCOMP_A			ASTC VCOMP モデル IP ライブラリ名

LIB_ASTC_RNG_A			ASTC RNG モデル IP ライブラリ名
LIB_ASTC_DCRC_A			ASTC DCRC モデル IP ライブラリ名
LIB_ASTC_OSTM_A			ASTC OSTM モデル IP ライブラリ名
LIB_ASTC_ERAY_A			ASTC ERAY モデル IP ライブラリ名
LIB_ASTC_CLMA_A			ASTC CLMA モデル IP ライブラリ名
LIB_ASTC_AFCAN_A			ASTC aFCAN モデル IP ライブラリ名
LIB_ASTC_CNTA_A			ASTC CNTA モデル IP ライブラリ名
LIB_ASTC_DNF_A			ASTC DNF モデル IP ライブラリ名
LIB_ASTC_ENC_A			ASTC ENC モデル IP ライブラリ名
LIB_ASTC_ETHERMAC_A			ASTC EtherMAC モデル IP ライブラリ名
LIB_ASTC_EVENTCTRL_A			ASTC Event Controller モデル IP ライブラリ名
LIB_ASTC_FCLA_A			ASTC FCLA モデル IP ライブラリ名
LIB_ASTC_IICB_A			ASTC IICB モデル IP ライブラリ名
LIB_ASTC_KR_A			ASTC KR モデル IP ライブラリ名
LIB_ASTC_MODE_A			ASTC MODE モデル IP ライブラリ名
LIB_ASTC_PIC_A			ASTC PIC モデル IP ライブラリ名
LIB_ASTC_SELA_A			ASTC SELA モデル IP ライブラリ名
LIB_ASTC_PFV01_BASE_A			ASTC PFV01 ベースライブラリ名
SYSTEMC_HOME	○	○	the location of ACCELLERA or USK SystemC environment
TARGET_ARCH	○	○	The target environment
SYSTEMC_LIB		○	Identifier in ACCELLERA or USK SystemC library
SYSTEMC_INCPATH			The location of ACCELLERA or USK SystemC Header
SYSTEMC_LIBPATH			The location of ACCELLERA or the USK SystemC library
PYTHONHOME	○	○	PYTHON installed directory path
PYTHON_VERSION	○	○	PYTHON version
PYTHON_INCPATH	○	○	PYTHON include path
PYTHON_LIBPATH	○	○	PYTHON library path
PLTFRM_COMPILE_LIB_HOME			Path to pltfrmCompile/lib
PLTFRM_COMPILE_INCLUDE_HOME			Path to pltfrmCompile/include
LIB_ASTC_PATH		○	Location of ASTC library
LIB_FASTISS_PATH			Location of FastISS library
BUILD_PF_HOME		○	Root directory of PFV01
MODEL_ASTC_MODEL		○	Location of ASTC model IP
MODEL_ASTC_DMA_PATH		○	Location of ASTC DMA model IP
BUILD_PF_PATH		○	Path to PFV01
BUILD_PF_PFRH850_PATH			Path to PFV01/PFRH850
INCLUDE_ASTC_PATH			Path to include_astc
CFOREST_LIB_PATH			Path to CFOREST library
MAKE	○	○	The location of make
RM	○		Rm name
RM_OPT	○		Rm option
AR			AR command
AR_APPEND			File addition by AR command
AR_DEK			File deletion by AR command
CXX	○	○	The location of the C++ compiler
OPTFLAG	○	○	Optimization flag
DEFFLAG	○	○	Defined macros flag
DBGFLAG	○	○	Debugging flag
INCPATH	○	○	Include path
DEPFLAG	○		Dependence output flag
DEFFLAG_MINE	○		Defined macros flag(for original expansion)
DBGFLAG_MINE	○		Debugging flag(for original expansion)
INCPATH_MINE	○		Include path(for original expansion)
CXXFLAGS	○		Compiler flag
LFLAGS	○		Link flag
SOURCES			Build source
OBJECTS			Build object
LIBS			Library

LIBS_BEFORE_ASTC			Library linked before ASTC library is linked
LIBS_ASTC			ASTC library

For your information, the setting value for make variables regaring compilation in the release package is explained.

Table 42 setting value of C++ compile option

Make variable	The set value and its meaning
OPTFLAG	-m32 -g
	Specify 32bit execuation format on 64bit machine and creating debugging objects
DEFFLAG	(無指定)
	Not specify the specific define macros. Overwrite top Makefile.
DBGFLAG	(無指定)
	Not specify to output debugging information. Overwritten by top Makefile.
INCPATH	-I$(SYSTEMC_INCPATH) -I. -I$(MODEL_COMMON_PATH)
	Specify to add ACCELLERA or USK SystemC Header files, current directories, model common header files. Overwritten by top Makefile.
DEPFLAG	-MM
	Specify to output dependency of a source file related to the user header file.
DEFFLAG_MINE	-DLINUX_DEF
	LINUX：use reused source. Detailes unknown. Affect models/{common}.
DBGFLAG_MINE	(無指定)
	Not specify to output debugging information
INCPATH_MINE	-I$(NSMVG3MSSV01_PATH) -I$(ATLTSLAVE32_PATH) -I$(ATLTSLAVE64_PATH)
	Specify to add include pathto each model header.
CXXFLAGS	$(OPTFLAG) $(DEFFLAG) $(DEFFLAG_MINE) $(DBGFLAG) $(DBGFLAG_MINE) $(INCPATH) $(INCPATH_MINE)
	Converge options specifed by each macro
LFLAGS	-ldl -lpthread -W1,-whole-archive -L$(SYSTEMC_LIBPATH) -l$(SYSTEMC_LIB) -W1,-no-whole-archive -rdynamic
	Specify library path to ACCELLERA / USK SystemC library, the kind of link library and static link.

Setting build rule

An implicit build rule of Makefile_scheap.tb is shown in Figure 47.

```
%.d:%.cpp
        $(CXX) $(DEPFLAG) $(CXXFLAGS) $< sed 's!$*.o:!$$(LIBPATH)/&!g' > $@
$(LIBPATH)/%.o:%.cpp
        $(CXX) -c $(CXXFLAGS) -o $@ $<
```

Figure 47 setting build rule in Makefile_scheap.tb

Target action

A target action of Makefile_scheap.tb is shown on Figure 48から Figure 51.

```
all:    $(SIMTARGET)
```

Figure 48 all target action in Makefile_scheap.tb

```
$(SIMTARGET): $(OBJECTS)
        $(CXX) $(CXXFLAGS) -o $@ $(OBJECTS) $(ASTC_HEAP_OBJECTS) $(LIBS_BEFORE_ASTC) $(LIBS_ASTC) $
(LIBS) $(LFLAGS) 2>&1 | c++filt
        @echo "sim.x was made!!"
        $(AR) $(TARGET_A_SCMAIN) $(shell find $(LIBPATH) -name '*.o' -print)
        @echo ".a was made!!"
        $(AR_APPEND) $(TARGET_A_SCMAIN) $(shell find $(LIB_ASTC_PATH) -name '*.o' -print)
        @echo ".a was made!!"
        $(AR_DEL) $(TARGET_A_SCMAIN) $(LIBPATH)/pltfrm.o
        @echo ".a was made!!"
        cp $(TARGET_A_SCMAIN) $(PLTFRM_COMPILE_LIB_HOME)
        $(AR_DEL) $(TARGET_A_SCMAIN) $(LIBPATH)/main.o
        mv $(TARGET_A_SCMAIN) $(TARGET_A)
        cp $(CFOREST_A) $(PLTFRM_COMPILE_LIB_HOME)
        cp $(CFOREST_CEDARE3V5_A) $(PLTFRM_COMPILE_LIB_HOME)
        cp $(FASTISS_CORE_ASTC_A) $(PLTFRM_COMPILE_LIB_HOME)
        cp $(FASTISS_RUNTIME_ASTC_A) $(PLTFRM_COMPILE_LIB_HOME)
        cp $(FASTISS_ASTC_A) $(PLTFRM_COMPILE_LIB_HOME)
        cp $(MODEL_COMMON_PATH)/global.h $(PLTFRM_COMPILE_INCLUDE_HOME)
        cp $(MODEL_COMMON_PATH)/CmErrMsg.h $(PLTFRM_COMPILE_INCLUDE_HOME)
        cp $(MODEL_COMMON_PATH)/slct_sc_signal_ports.h $(PLTFRM_COMPILE_INCLUDE_HOME)
        cp $(MODEL_COMMON_PATH)/OSCI2.h $(PLTFRM_COMPILE_INCLUDE_HOME)
        cp $(NSMVG3MSSV01_PATH)/NSMVG3MSSV01.h $(PLTFRM_COMPILE_INCLUDE_HOME)
        cp $(INCLUDE_ASTC_PATH)/PFRH850.h $(PLTFRM_COMPILE_INCLUDE_HOME)
#       cp $(INCLUDE_ASTC_PATH)/HEAPPlatform.h $(PLTFRM_COMPILE_INCLUDE_HOME)
        @echo "Done"
```

Figure 49 $(TARGET) target action in Makefile_scheap.tb

```
clean:
        @($(RM) $(RM_OPT) $(OBJECTS) core*);
        @($(RM) * d);
```

Figure 50 clean target action in Makefile_scheap.tb

```
$(LIBPATH)/main.o:  main.cpp main.d
```

Figure 51 $(LIBPATH)/main.o target action in Makefile_scheap.tb(use file dependency)

```
$(LIBPATH)/pltfrm.o:  pltfrm.cpp pltfrm.d
```

Figure 52 $(LIBPATH)/pltfrm.o target action in Makefile_scheap.tb(use file dependency)

```
$(LIBPATH)/ShPythonAPI.o:  ShPythonAPI.cpp ShPythonAPI.d
```

Figure 53 $(LIBPATH)/ShPythonAPI.o target action in Makefile_scheap.tb(use file dependency)

Copying header file / library
After executing this Makefile, simulator executable file and the library of G3MSS hierarchy are created. This library and the include files for it are copied to some directories in pltfrmCompile.

Table 43 copied file and destination of copy

file name	Destination of copy
Library file ※make variables : specifed by $(TARGET_A_SCMAIN, CFOREST_A, CFOREST_CEDARE3V5_A, FASTISS_CORE_ASTC_A, FASTISS_RUNTIME_ASTC_A, FASTISS_ASTC_A)	Directory specified by $(PLTFRM_COMPILE_LIB)
$(MODELS_COMMON_PATH)/global.h	Directory specified by $(PLTFRM_COMPILE_INCLUDE)
$(MODELS_COMMON_PATH)/CmErrMst.h	
$(MODELS_COMMON_PATH)/slct_sc_signal_ports.h	
$(MODELS_COMMON_PATH)/OSCI2.h	
$(NSMVG3MSSV01)/NSMVG3MSSV01.h	
$(INCLUDE_ASTC_PATH)/PFRH850.h	
$(INCLUDE_ASTC_PATH)/HEAPPlatform.h	

8.7. Explanation of scheapE3.sln

担当：佐藤、新井 S USK は残す？

To create the simulator for SC-HEAP E3 platform and CPUSS hierarchy library with VisualStudio 2010(abbreviated as VS) on Windows Platform, Visual Studio Solution file and Visual Studio Project file are used. In the section, Solution file scheapE3.sln and project file SCHEAP-E3-G5_EXE.vcproj/scheapE3_g5.vcxproj/RH850-G5.vcproj/scheapE3_models.vcproj to build the simulator are explained.

Besides, these files are used when using Windows Accellera/USK SystemC.

.1. Environment variable needed before using scheapE3.sln

Please refer to the following environmental variables when using scheapE3.sln.

【ACCELLERA】

Variable name	The set value
SCHEAP_HOME	Home directory to the project
SYSTEMC_HOME	Path to ACCELLERA SystemC library
CFOREST_DIR	Path to CForestG3M
TLM_INC_DIR	Path to TLM
PYTHON_DIR	Path to PYTHON

【USK】

Variable name	The set value
SCHEAP_HOME	プロジェクトのホームディレクトリ
CFOREST_DIR	Path to CForestG3M
USK_HOME	Path to USK SystemC libary
PYTHON_DIR	Path to PYTHON

8.7.2. Structure of scheapE3.sln

Please refer to MSDN document for detailed file formant or corresponded defined value.

This solution is consisted of the following projects.

Table 44 projects in scheapE3.sln

Project name	The file name	The explanation
SCHEAP-E3-G5_EXE	SCHEAP-E3-G5_EXE.vcxproj	Compile pltfrm.cpp and link
SCHEAP-E3-G5	scheapE3_g5.vcxproj	Compile main.cpp and create SCHEAP-E3-G5.lib
RH850-G5	RH850-G5.vcxproj	Compile pltfrmSmpils and create RH850-G5.lib
PFV01	PFV01.vcproj	Compile RH850 peripheral connection class
SCHEAP-E3-Models	scheapE3_models.vcproj	Compile model IP
cforestg3m	cforestg3m.vcproj	Compile CForestG3M

This solution is as follows.

Table 45 solution of scheapE3.sln

Solution	Target	The construction of each project
Release	ACCELLERA edition SC-HEAP E3 simulator. Release mode.	cforestg3m:Release SCHEAP-E3-Models:Release RH850-G5:Release SCHEAP-E3-G5:Release SCHEAP-E3-G5_EXE:Release
Release_usk	USK edition SC-HEAP E3 simultor. Release mode	cforestg3m:Debug SCHEAP-E3-Models:Debug RH850-G5:Debug SCHEAP-E3-G5:Debug SCHEAP-E3-G5_EXE:Deubg
Debug_usk	ACCELLERA edition SC-HEAP E3 simulator. Debug mode.	cforestg3m:Debug SCHEAP-E3-G5:Debug_usk SCHEAP-E3-Models:Debug_usk
Debug_usk	USK edition SC-HEAP E3 simultor. Debug mode	cforestg3m:Debug SCHEAP-E3-G5:Debug_usk SCHEAP-E3-Models:Debug_usk

8.7.3. Structure of SCHEAP-E3-G5_EXE.vcxproj

SCHEAP-E3-G5_EXE.vcxproj is the project file to compile pltfrm.cpp and link it.
Please refer to the MSDN document regarding detailed file format and corresponded defined value
The structure of the project is as follows.

Table 46 solution of SCHEAP-E3-G5_EXE.vcxproj

Composition	target	SystemC	Build mode	Property sheet
Release	Executable file	ACCELLERA	Release	heapE3-base.vsprops heapE3-release.vsprops osci.vsprops
Release_usk	Executable file	USK	Release	heapE3-base.vsprops heapE3-release.vsprops usk.vsprops
Debug	Executable file	ACCELLERA	Debug	heapE3-base.vsprops heapE3-debug.vsprops osci.vsprops
Debug_usk	Executable file	USK	Debug	heapE3-base.vsprops heapE3-debug.vsprops usk.vsprops

In this file、the files, model IP folders and the filter for the files which are located in them are shown in Table 50. In "Solution explorer" of VS Window, the setting values can be confirmed.

Table 47 filter in SCHEAP-E3-G5_EXE.vcxproj

filter name	Corresponded file / folder fo model IP
<Source file>¥pltfrm.cpp	<PROJTOP>/build/TOPV01/pltfrm.cpp

In this file, the contents in Table 51 are set.
The setting item which isn't below, isn't changed as default.
In "Property page" of VS Window, the setting values can be confirmed.

Table 48 setting value in SCHEAP-E3-G5_EXE.vcxproj

【[C++/C] Item】

Setting name	The set value
[General]Additional indlude file	※remarks: the following long thing is one setting value. Inherit from osci.props ※3 $(SYSTEMC_HOME)¥src; $(SYSTEMC_HOME)¥src¥sysc¥packages; $(TLM_INC_DIR); Inherit from usk.props ※4 $(USK_HOME)¥systemc¥include; $(USK_HOME)¥TLM2¥include¥tlm; Inherit from heapE3-base.props $(SCHEAP_HOME)¥scheapCompile¥models¥common; $(SCHEAP_HOME)¥scheapCompile¥models¥common¥smpils; $(SCHEAP_HOME)¥scheapCompile¥models¥NSMVG3MSSV01; $(SCHEAP_HOME)¥scheapCompile¥models¥NSMVINTC1V01; $(SCHEAP_HOME)¥scheapCompile¥include_astc; $(SCHEAP_HOME)¥scheapCompile¥models_astc; Set scheapE3_g5.vcxproj ..¥..¥..¥models¥NSMVRH850V01; ..¥..¥..¥models¥NSMVG3MSSV01; ..¥..¥..¥models¥common; ..¥..¥..¥models¥NSMVG3MPEV01; $(CFOREST_DIR)¥sim¥CedarE3V5; $(CFOREST_DIR)¥sim¥CedarE3V5¥fpu_soft; $(CFOREST_DIR)¥sim¥CForestG3M; ..¥..¥..¥models¥NSMVG3MCPUV01; ..¥..¥..¥models¥ATLTSLAVE32; ..¥..¥..¥models¥ATLTSLAVE64; $(PYTHON_DIR)¥include
[General] format of debug information	Not specify(=blank)※2
[General] Warning level	level 3
[Optimize] Optimize	invaliid※1 (inherit from heapE3-debug.props) Maximize performance(/O2)※2 (inherit from heapE3-release.props)

[Preprocessor] define preprocessor	※remarks: the following long thing is one setting value.
	Inherit from heapE3-release.props
	NDEBUG
	Inherit from usk.props
	SC_USE_KERNEL_DLL;
	Inherit from vsprops¥heapE3-base.props
	SUPPORT_SC_HEAP;
	MSVC;WIN32;
	WIN32_LEAN_AND_MEAN;
	_CRT_SECURE_NO_WARNINGS;
	_CRT_NONSTDC_NO_WARNINGS;
	SC_INCLUDE_DYNAMIC_PROCESSES;
	WINDOWS_DEF;
	_CONSOLE;
	_SYSTEMC_21_;
	SC_USE_SC_STRING_OLD;
	NSMVINTC711_DEF;
	CM_REPORT_OUT;
	_LARGEFILE64_SOURCE;
	_FILE_OFFSET_BITS=64;
	RUBYFPU;
	CVT_UNSIGNED;
	SRV800;
	FM850;
	OUTPUT_LOG;
	NOHALT_OPTION;
	SINGLE_STEP;
	PSW_US_0FIX;
	UCPOP;
	TLB64;
	NMULTIMEDIA;
	FPUAPPROXIMATION;
	SAVEEPC;
	_MBCS
	Set scheapE3_g5.vcxproj
	WIN32
	NDEBUG
	_CONSOLE

[Generate code] do minimum rebuild	No
[Generate code] valid for C++ exception	Yes
[Generate code] basic runtime check	Fixed valud
[Generate code] rutime library	Multi-thread DLL(/MD)
[Language] valid runtime information	Not specify(=blank)※2
[Precompiled header] make/use precompiled header	Not use precompiled header
[Command line] additional option	/vmg

※1 only Debug build
※2 only Release build
※3 only Accellera edition
※4 only USK build
※The "..\..\..\" included in an inclusion directory assumes a relative path to the location of <PROJTOP>\models.

【[Linker] Item】

Setting name	The set value
[General] Output file	$(OutDir)/sim.exe
[General] valid incremental link	Not specify(=blank)※2※4

[General] Additional library directory	Debug;※1 Debug_astc;※1 $(SYSTEMC_HOME)¥msvc100¥SystemC¥Debug;※1 Debug_usk;※3 Debug_astc_usk;※3 $(USK_HOME)¥systemc¥lib-msvc100※3 ..¥..¥..¥cforest_g3m¥cforestg3m¥cforestg3m¥Debug;※1※3 Release;※2 Release_astc;※2 $(SYSTEMC_HOME)¥msvc100¥SystemC¥Release;※2 Release_usk;※4 Release_astc_usk;※4 $(USK_HOME)¥systemc¥lib-msvc100※4 $ (SCHEAP_HOME)¥scheapCompile¥models¥iss¥fastiss_astc¥rh850 ¥iss¥lib¥osci-win32-msvc100¥rh850;※2 $(SCHEAP_HOME)¥scheapCompile¥lib $(PYTHON_DIR)¥libs
[Input] Additional dependency file	SystemC.lib※3 WS2_32.lib SCHEAP-E3-G5.lib; fastiss.lib rh850-rntime.lib rh850-core.lib python27.lib
[Debug] generate debug information	Not specify(=blank)※2※4
[Debug] generate program data base file	$(TargetDir)$(TargetName).pdb※2※4
[System] Subsystem	Console
[Optimize] Reference	Not specify(=blank)※2※4
[Optimize] Compress COMDAT	Not specify(=blank)※2※4
[Details] Target computer	MachineX86

※1 only Debug build
※2 only Debug_usk build
※3 only Release build
※4 only Release_usk build

8.7.4. Structure of scheapE3_g5.vcxproj

scheapE3_g5.vcxproj is the project file to compile CPUSS/PE hierarchy and link them.
Please refer to the MSDN document regarding detailed file format and corresponded defined value
The structure of the project is as follows.

Table 49 solution for scheapE3-g5.vcxproj

Composition	Target	SystemC.	Build mode	Inference of a property seat
Release	executable file	ACCELLERA	Release	heapE3-base.vsprops heapE3-release.vsprops osci.vsprops
Release_usk	executable file	USK	Release	heapE3-base.vsprops heapE3-release.vsprops usk.vsprops
Debug	executable file	ACCELLERA	Debug	heapE3-base.vsprops heapE3-debug.vsprops osci.vsprops
Debug_usk	executable file	USK	Debug	heapE3-base.vsprops heapE3-debug.vsprops usk.vsprops

In this file、the files, model IP folders and the filter for the files which are located in them are shown in

Table 50. In “Solution explorer” of VS Window, the setting values can be confirmed.

Table 50 filter in scheapE3-g5.vcxproj

filter name	Corresponded file / folder fo model IP
<Source file>¥TOPV01	<PROJTOP>/build/TOPV01/main.cpp <PROJTOP>/build/TOPV01/ShPythonAPI.cpp

In this file, the content in Table 51 is set.
The setting item which is not described below is not canged as default.
In “Property page” on VS Windows the setting value can be confirmed.

Table 51 setting value in scheapE3-g5.vcxproj

【[C++/C] Item】

Setting name	The set value
[General]Additional indlude file	※remarks: the following long thing is one setting value. Inherit from osci.props ※3 $(SYSTEMC_HOME)¥src; $(SYSTEMC_HOME)¥src¥sysc¥packages; $(TLM_INC_DIR); Inherit from usk.props ※4 $(USK_HOME)¥systemc¥include; $(USK_HOME)¥TLM2¥include¥tlm; Inherit form heapE3-base.props $(SCHEAP_HOME)¥scheapCompile¥models¥common; $(SCHEAP_HOME)¥scheapCompile¥models¥common¥smpils; $(SCHEAP_HOME)¥scheapCompile¥models¥NSMVG3MSSV01; $(SCHEAP_HOME)¥scheapCompile¥models¥NSMVINTC1V01; $(SCHEAP_HOME)¥scheapCompile¥include_astc; $(SCHEAP_HOME)¥scheapCompile¥models_astc; Set scheapE3_g5.vcxproj ..¥..¥..¥models¥NSMVRH850V01; ..¥..¥..¥models¥NSMVG3MSSV01; ..¥..¥..¥models¥common; ..¥..¥..¥models¥NSMVG3MPEV01; $(CFOREST_DIR)¥sim¥CedarE3V5; $(CFOREST_DIR)¥sim¥CedarE3V5¥fpu_soft; $(CFOREST_DIR)¥sim¥CForestG3M; ..¥..¥..¥models¥NSMVG3MCPUV01; ..¥..¥..¥models¥ATLTSLAVE32; ..¥..¥..¥models¥ATLTSLAVE64; $(PYTHON_DIR)¥include
[General] format of debug information	Not specify(=blank)※2
[General] Warning level	level 3
[Optimize] Optimize	invaliid※1 (inherit from heapE3-debug.props) Maximize performance(/O2)※2 (inherit from heapE3-release.props)

[Preprocessor] define preprocessor	※remarks: the following long thing is one setting value.
	Inherit from heapE3-release.props
	NDEBUG
	Inherit from usk.props
	SC_USE_KERNEL_DLL;
	Inherit from vsprops¥heapE3-base.props
	SUPPORT_SC_HEAP;
	MSVC;WIN32;
	WIN32_LEAN_AND_MEAN;
	_CRT_SECURE_NO_WARNINGS;
	_CRT_NONSTDC_NO_WARNINGS;
	SC_INCLUDE_DYNAMIC_PROCESSES;
	WINDOWS_DEF;
	_CONSOLE;
	_SYSTEMC_21_;
	SC_USE_SC_STRING_OLD;
	NSMVINTC711_DEF;
	CM_REPORT_OUT;
	_LARGEFILE64_SOURCE;
	_FILE_OFFSET_BITS=64;
	RUBYFPU;
	CVT_UNSIGNED;
	SRV800;
	FM850;
	OUTPUT_LOG;
	NOHALT_OPTION;
	SINGLE_STEP;
	PSW_US_0FIX;
	UCPOP;
	TLB64;
	NMULTIMEDIA;
	FPUAPPROXIMATION;
	SAVEEPC;
	_MBCS
	Set scheapE3_g5.vcxproj as follows
	ICACHE_ENABLE;
	_ENABLE_PIC_;
	DISABLE_SSC;
	DISABLE_GDB;
	REVISION="f4fe6b7";
	VERSION="v120321";
	_SECURE_SCL=0;
	SUPPORT_SC_HEAP;
	DISABLE_PYTHON;
	UNICODE;
	_UNICODE;

[Generate code] do minimum rebuild	No
[Generate code] valid for C++ exception	Yes
[Generate code] basic runtime check	Fixed valud
[Generate code] rutime library	Multi-thread DLL(/MD)
[Language] valid runtime information	Not specify(=blank)※2
[Precompiled header] make/use precompiled header	Not use precompiled header
[Command line] additional option	/vmg

※1 only Debug build
※2 only Release build
※3 only Accellera edition
※4 only USK build
※The "..\..\..\" included in an inclusion directory assumes a relative path to the location of <PROJTOP>\models.

【[library] Item 項目】

Setting name	The set value
Output file	\$(OutDir)\$(TargetName)\$TargetExt)
Additional dependency file	RH850-G5.lib
Additional library	Release※1 Release_usk※2

※1 only ACCELLERA build
※2 only USK edition build

【[Build event] item】

Setting name	The set value
ビルド後のイベント	scheapE3_postbuild.bat Osci-debug※1 scheapE3_postbuild.bat Osci-release※2 scheapE3_postbuild.bat usk-debug※3 scheapE3_postbuild.bat usk-release※4 (ターゲット及び pltfrmCompile 側でも使用するヘッダファイルをコピー)

※1 only Debug build
※2 only Release build
※3 only Accellera edition
※4 only USK build

8.7.5. RH850_G5.vcxproj の構造

RH850_G5.vcxproj is the project file to compile pltfrmSmpils.cpp and to create RH850-G5.lib for RH850 hierarchy.
Please refer to the MSDN document regarding detailed file format and corresponded defined value
The structure of the project is as follows.

Table 52 solution of RH850_G5.vcxproj

Composition	target	SystemC	Build mode	Property sheet
Release	Executable file	ACCELLERA	Release	heapE3-base.vsprops heapE3-release.vsprops osci.vsprops
Release_usk	Executable file	USK	Release	heapE3-base.vsprops heapE3-release.vsprops usk.vsprops
Debug	Executable file	ACCELLERA	Debug	heapE3-base.vsprops heapE3-debug.vsprops osci.vsprops
Debug_usk	Executable file	USK	Debug	heapE3-base.vsprops heapE3-debug.vsprops usk.vsprops

In this file、the files, model IP folders and the filter for the files which are located in them are shown in Table 50. In “Solution explorer” of VS Window, the setting values can be confirmed.

Table 53 filter in RH850_G5.vcxproj

filter name	Corresponded file / folder fo model IP
<Source file>¥SMPILSV01	<PROJTOP>/build/SMPILSV01/pltfrmSmpils.cpp

In this file, the contents in Table 51 are set.
The setting item which isn't below, isn’t changed as default.
In “Property page” of VS Window, the setting values can be confirmed.

Table 54 setting value in RH850_G5.vcxproj

【[C++/C] Item】

Setting name	The set value
[General]Additional indlude file	※remarks: the following long thing is one setting value. Inherit from osci.props ※3 $(SYSTEMC_HOME)¥src; $(SYSTEMC_HOME)¥src¥sysc¥packages; $(TLM_INC_DIR); Inherit from usk.props ※4 $(USK_HOME)¥systemc¥include; $(USK_HOME)¥TLM2¥include¥tlm; Inherit from heapE3-base.props $(SCHEAP_HOME)¥scheapCompile¥models¥common; $(SCHEAP_HOME)¥scheapCompile¥models¥common¥smpils; $(SCHEAP_HOME)¥scheapCompile¥models¥NSMVG3MSSV01; $(SCHEAP_HOME)¥scheapCompile¥models¥NSMVINTC1V01; $(SCHEAP_HOME)¥scheapCompile¥include_astc; $(SCHEAP_HOME)¥scheapCompile¥models_astc; Set RH850_G5.vcxproj ..¥..¥..¥models¥NSMVRH850V01; ..¥..¥..¥models¥NSMVG3MSSV01; ..¥..¥..¥models¥common; ..¥..¥..¥models¥NSMVG3MPEV01; $(CFOREST_DIR)¥sim¥CedarE3V5; $(CFOREST_DIR)¥sim¥CedarE3V5¥fpu_soft; $(CFOREST_DIR)¥sim¥CForestG3M; ..¥..¥..¥models¥NSMVG3MCPUV01; ..¥..¥..¥models¥ATLTSLAVE32; ..¥..¥..¥models¥ATLTSLAVE64; $(PYTHON_DIR)¥include
[General] format of debug information	Not specify(=blank)※2
[General] Warning level	level 3
[Optimize] Optimize	invaliid※1 (inherit from heapE3-debug.props) Maximize performance(/O2)※2 (inherit from heapE3-release.props)

[Preprocessor] define preprocessor	※remarks: the following long thing is one setting value.
	Inherit from heapE3-release.props
	NDEBUG
	Inherit from usk.props
	SC_USE_KERNEL_DLL;
	Inherit from vsprops¥heapE3-base.props
	SUPPORT_SC_HEAP;
	MSVC;WIN32;
	WIN32_LEAN_AND_MEAN;
	_CRT_SECURE_NO_WARNINGS;
	_CRT_NONSTDC_NO_WARNINGS;
	SC_INCLUDE_DYNAMIC_PROCESSES;
	WINDOWS_DEF;
	_CONSOLE;
	_SYSTEMC_21_;
	SC_USE_SC_STRING_OLD;
	NSMVINTC711_DEF;
	CM_REPORT_OUT;
	_LARGEFILE64_SOURCE;
	_FILE_OFFSET_BITS=64;
	RUBYFPU;
	CVT_UNSIGNED;
	SRV800;
	FM850;
	OUTPUT_LOG;
	NOHALT_OPTION;
	SINGLE_STEP;
	PSW_US_0FIX;
	UCPOP;
	TLB64;
	NMULTIMEDIA;
	FPUAPPROXIMATION;
	SAVEEPC;
	_MBCS
	Set RH850_g5.vcxproj
	ICACHE_ENABLE;
	_ENABLE_PIC_;
	DISABLE_SSC;
	DISABLE_GDB;
	REVISION="f4fe6b7";
	VERSION="v120321";
	_SECURE_SCL=0;
	SUPPORT_SC_HEAP;
	DISABLE_PYTHON;
	UNICODE;
	_UNICODE;

[Generate code] do minimum rebuild	No
[Generate code] valid for C++ exception	Yes
[Generate code] basic runtime check	Fixed valud
[Generate code] rutime library	Multi-thread DLL(/MD)
[Language] valid runtime information	Not specify(=blank)※2
[Precompiled header] make/use precompiled header	Not use precompiled header
[Command line] additional option	/vmg

※1 only Debug build
※2 only Release build
※3 only Accellera edition
※4 only USK build
※The "..\..\..\" included in an inclusion directory assumes a relative path to the location of <PROJTOP>\models.

【[Library] Item】

Setting name	The set value
Output file	$(OutDir)$(TargetName)$TargetExt)
Additional dependency file	SCHEAP-E3-Models.lib All obj files in $(CFOREST_DIR)¥cforestg3m¥Release
Additional library	Release※1 Release_usk※2

※1 only Accellera edition
※2 only USK build

8.7.6. Structure of scheapE3_models.vcproj

scheapE3_models.vcproj is the project file to compile model IPs.
Please refer to MSDN document about detailed file format or corresponded defined values.
This project structure is as follows.

Table 55 solution for scheapE3_models.vcproj

Composition	SystemC.	Build mode	The SC-HEAP form	Inference of a property seat
Release	ACCELLERA	Release	unit	heapE3-base.vsprops heapE3-release.vsprops osci.vsprops
Release_usk	USK	Release	unit	heapE3-base.vsprops heapE3-release.vsprops usk.vsprops
Debug	ACCELLERA	Debug	unit	heapE3-base.vsprops heapE3-debug.vsprops osci.vsprops
Debug_usk	USK	Debug	unit	heapE3-base.vsprops heapE3-debug.vsprops usk.vsprops

In this file、the files, model IP folders and the filter for the files which are located in them are shown in Table 56. In “Solution explorer” of VS Window, the setting values can be confirmed.

Table 56 filter in scheapE3_models.vcxproj

filter name	Corresponded file / folder fo model IP
AHB2VCI	<PROJTOP>/models/AHB2VCI
ATLTLB32	<PROJTOP>/models/ATLTLB32
ATLTLB64	<PROJTOP>/models/ATLTLB64
ATLTSLAVE32	<PROJTOP>/models/ATLTSLAVE32
ATLTSLAVE64	<PROJTOP>/models/ATLTSLAVE64
common	<PROJTOP>/models/common
NSMVRH850V01	<PROJTOP>/models/NSMVRH850V01
NSMVG3MSSV01	<PROJTOP>/models/NSMVG3MSSV01
NSMVG3MPEV01	<PROJTOP>/models/NSMVG3MPEV01
NSMVG3MCPUV01	<PROJTOP>/models/NSMVG3MCPUV01
NSMVINTC1V01	<PROJTOP>/models/NSMVINTC1V01
NSMVINTC2V01	<PROJTOP>/models/NSMVINTC2V01
VCI2AHB	<PROJTOP>/models/AXI2AHB
VPI2APB	<PROJTOP>/models/VPI2APB

In this file, the contents in Table 57 are set.
The setting item which isn't below, isn’t changed as default.

In “Property page” of VS Window, the setting values can be confirmed.

Table 57 setting value in scheapE3-models.vcxproj

【[C++/C] Item】

Setting name	The set value
[General]Additional indlude file	※remarks: the following long thing is one setting value. Inherit from osci.props ※3 $(SYSTEMC_HOME)¥src $(SYSTEMC_HOME)¥src¥sysc¥packages $(TLM_INC_DIR) Inherit from heapE3-base.props $(SCHEAP_HOME)¥scheapCompile¥models¥common $(SCHEAP_HOME)¥scheapCompile¥models¥common¥smpils $(SCHEAP_HOME)¥scheapCompile¥models¥NSMV85E2SPFPV01 $(SCHEAP_HOME)¥scheapCompile¥models¥NSSCHEAPLBV01 $(SCHEAP_HOME)¥scheapCompile¥include_astc $(SCHEAP_HOME)¥scheapCompile¥models_astc Set scheapE3-models.vcproj ..¥..¥..¥models¥NSMVG3MSSV01;. .¥..¥..¥models¥common; ..¥..¥..¥models¥NSMVG3MPEV01; $(CFOREST_DIR)¥sim¥CedarE3V5¥fpu_soft; $(CFOREST_DIR)¥sim¥CedarE3V5; $(CFOREST_DIR)¥sim¥CForestG3M; ..¥..¥..¥models¥NSMVG3MCPUV01; ..¥..¥..¥models¥NSMVG3MCPUV01¥CaISS; ..¥..¥..¥models¥NSMVG3MCPUV01¥Rteserv2; ..¥..¥..¥models¥ATLTLB32; ..¥..¥..¥models¥ATLTLB64; ..¥..¥..¥models¥AHB2AXI; ..¥..¥..¥models¥AXI2AHB; ..¥..¥..¥models¥VPI2APB; ..¥..¥..¥models¥ATLTSLAVE32; ..¥..¥..¥models¥ATLTSLAVE64; ..¥..¥..¥models¥common_bus;
[General] format of debug information	Not specify(=blank)※2
[General] Warning level	level 3
[Optimize] Optimize	invaliid※1 (inherit from heapE3-debug.props) Maximize performance (/O2) ※ 2 (inherit from heapE3-release.props)

[Preprocessor] define preprocessor	※remarks: the following long thing is one setting value. Inherit from heapE3-release.props NDEBUG Inherit from vsprops¥heapE3-base.props SUPPORT_SC_HEAP MSVC WIN32 WIN32_LEAN_AND_MEAN _CRT_SECURE_NO_WARNINGS _CRT_NONSTDC_NO_WARNINGS SC_INCLUDE_DYNAMIC_PROCESSES WINDOWS_DEF _CONSOLE _SYSTEMC_21_ SC_USE_SC_STRING_OLD NSMVINTC711_DEF CM_REPORT_OUT _LARGEFILE64_SOURCE _FILE_OFFSET_BITS=64 RUBYFPU CVT_UNSIGNED SRV800 FM850 OUTPUT_LOG NOHALT_OPTION SINGLE_STEP PSW_US_0FIX UCPOP TLB64 NMULTIMEDIA FPUAPPROXIMATION SAVEEPC _MBCS Set scheapE3-models.vcproj GCCSPARC ICACHE_ENABLE _ENABLE_PIC_ DISABLE_SSC DISABLE_GDB REVISION="f4fe6b7" VERSION="v120321" _SECURE_SCL=0 SUPPORT_SC_HEAP DISABLE_PYTHON UNICODE _UNICODE TEST_ISS
[Generate code] do minimum rebuild	No
[Generate code] valid for C++ exception	Yes
[Generate code] basic runtime check	Fixed value
[Generate code] rutime library	Multi-thread DLL(/MD)
[Language] valid runtime information	Not specify(=blank)※2
[Precompiled header] make/use precompiled header	Not use precompiled header
[Command line] additional option	/vmg

※1 only Debug build
※2 only Release build
※3 only Accellera edition
※4 only USK build
※The "..\..\..\" included in an inclusion directory assumes a relative path to the location of <PROJTOP>\models.

8.7.7. Structure of PFV01.vcxproj

PFV01.vcproj is the project file to compile the peripheral IPs.
Please refer to MSDN document about detailed file format or corresponded defined values.
This project structure is as follows.

Table 58 solution in PFV01.vcxproj

Composition	SystemC.	Build mode	Inference of a property seat
Osci-release	ACCELLERA	Release	heapE3-base.vsprops heapE3-release.vsprops Osci.vsprops
usk-release	USK	Release	heapE3-base.vsprops heapE3-debug.vsprops usk.vsprops
Osci-debug	ACCELLERA	Debug	heapE3-base.vsprops heapE3-release.vsprops Osci.vsprops
usk_debug	USK	Debug	heapE3-base.vsprops heapE3-debug.vsprops usk.vsprops

In this file、the files, model IP folders and the filter for the files which are located in them are shown in Table 59. In “Solution explorer” of VS Window, the setting values can be confirmed.

Table 59 filter in PFV01.vcxproj

filter name	Corresponded file / folder fo model IP
PFRH850	<PROJTOP>/build/PFV01/PFRH850.cpp
pltfrmRH850	<PROJTOP>/build/PFV01/pltfrmRH850.cpp

In this file, the contents in Table 60 are set.
The setting item which isn't below, isn’t changed as default.
In “Property page” of VS Window, the setting values can be confirmed.

Table 60 setting value in PFV01.vcxproj

【[C++/C] Item】

Setting name	The set value
[General]Additional indlude file	※remarks: the following long thing is one setting value. $(SYSTEMC_HOME)/src;※3 (inherit from Osci.vsprops) $(TLM_INC_DIR);※3 (inherit from Osci.vsprops) $(USK_HOME)¥systemc¥include:※4 (inherit from usk.vsprops) $(USK_HOME)¥TLM2¥include¥tlm;※4 (inherit from usk.vsprops)) ../../../models/common; (inherit from heap-base.vsprops) ../../../models/common_smpils; (inherit from heap-base.vsprops) ? ../../../models/NSMV85E2SPFPV01; (inherit from heap-base.vsprops)? ../../../include_astc; (inherit from heap-base.vsprops)TBC ../../../models_astc; (inherit from heap-base.vsprops) TBC
[General] format of debug information	program data base※1 Invalid ※2
[General] Warning level	Level 3
[General] Support migration to 64bit	No

[Optimize] Optimize	invalid ※1 (inherit from heap-debug.vsprops) Runtime performance※2 (inherit from heap-release.vsprops)
[Preprocessor] define preprocessor	※注:長いですが、以下で一つの設定値です WIN32; (heap-base.vsprops) WINDOWS_DEF; (heap-base.vsprops) NDEBUG;※2 (heap-base.vsprops) _CONSOLE; (heap-base.vsprops) _SYSTEMC_21_; (heap-base.vsprops) SC_USE_SC_STRING_OLD; (heap-base.vsprops) NSMVINTC711_DEF; (heap-base.vsprops) CM_REPORT_OUT; (heap-base.vsprops) _LARGEFILE64_SOURCE; (heap-base.vsprops) _FILE_OFFSET_BITS=64; (heap-base.vsprops) _V850E2R_LOCAL_BUS_; (heap-base.vsprops) _FLASH_CACHE_INTERNAL_; (heap-base.vsprops) V850E2; (heap-base.vsprops) RUBYFPU; (heap-base.vsprops) WIN32_LEAN_AND_MEAN; (heap-base.vsprops) CVT_UNSIGNED; (heap-base.vsprops) SRV800; (heap-base.vsprops) FM850; (heap-base.vsprops) OUTPUT_LOG; (heap-base.vsprops) NOHALT_OPTION; (heap-base.vsprops) INT_FOR_E2RTL_VERIFICATION; (heap-base.vsprops) SINGLE_STEP; (heap-base.vsprops) PSW_US_0FIX; (heap-base.vsprops) NA85E2R; (heap-base.vsprops) NA85E2RV3; (heap-base.vsprops) UCPOP; (heap-base.vsprops) TLB64; (heap-base.vsprops) NMULTIMEDIA; (heap-base.vsprops) FPUAPPROXIMATION; (heap-base.vsprops) SAVEEPC; (heap-base.vsprops); _CRT_SECURE_NO_WARNINGS; (heap-base.vsprops) _CRT_NONSTDC_NO_WARNINGS; (heap-base.vsprops) SC_INCLUDE_DYNAMIC_PROCESSES; (heap-base.vsprops) V3CLASS4=1; (heap-base.vsprops) HEAP_RESET_ACTIVE_LEVEL=false; (heap-base.vsprops) _MBCS; (Multi-byte Character Support) SC_USE_KERNEL_DLL;※4 (usk.vsprops)
[Generate code] do minimum rebuild	No
[Generate code] basic runtime check	Fixed value
[Generate code] rutime library	Multi-thread DLL(/MD)
[Language] valid runtime information	Yes
[Precompiled header] make/use precompiled header	Not use precompiled header
[Command line] additional option	No
[Command line] additional option	/vmg

※1 only Debug build
※2 only Release build
※3 only Accellera edition
※4 only USK build
※The "..\..\..\" included in an inclusion directory assumes a relative path to the location of <PROJTOP>\models.

8.8. Explanation of run_core* file

担当：吉永 S, 新井 S, 大塚氏

As explained at 5.1, it's the script file to carry out a simulator by a SystemC model element
The script file name and content are various depending on the execution environment.

【Linux ACCELLERA/USK 版 SystemC】
The script file name is run_core.csh.
The shell script variable shown in Table 61 is used in run_core.csh. The shell script variables for which the circle is marked in the row "Machine environmental dependency" should be confirmed when the other machine environment which is different from the machine environment shown at 5.1.1 is usd.

Table 61 shell script variable in run_core.csh

Shell script variable name	Machine Environmental dependency	The meaning
SIML_EXE		The location of the executable file of a simulator
RSLT_LOG		The preservation location of the log
HEAP_CFG		The location of the configuration file for models
CYCL_NUM		The number of simulation cycles

A source of run_core.csh is shown on Figure 54.

```
#!/bin/csh -f

# set variables
set SIML_EXE="./sim.x"
set RSLT_LOG="result.txt"
set HEAP_CFG="heap.cfg"
set CYCL_NUM="100000"

# invoke command
(echo -n "#### Created date was "; date +'%D %T %Z')  >& ${RSLT_LOG}
(echo -n "#### current directory was "; pwd)        >>& ${RSLT_LOG}
(echo    "#### check programs are as follows")      >>& ${RSLT_LOG}
(grep "G3MCPU_PROGRAM" ${HEAP_CFG})                 >>& ${RSLT_LOG}
${SIML_EXE} -n ${CYCL_NUM} -config ${HEAP_CFG}      >>& ${RSLT_LOG}
```

Figure 54 source of run_core.csh

【Windows ACCELLERA/USK edition SystemC】
The script file name is run_core_win.bat.
The contents of the variable / source used in run_core_win.bat is prepared based on the contents of the variable / source used in run_core.csh which is used for . Linux ACCELLERA/USK edition SystemC.

8.9. Explanation of run_multi* file

担当 :大塚氏

As explained at 5.2, it's the script file to connect with Multi and carry out a simulator.
The script file name and its contents are various dependent on each execution environment.

【Linux ACCELLERA/USK edition SystemC】

The script file name is run_multi.csh.
The shell script variable shown in Table 62 is used in run_multi.csh. The shell script variables for which the circle is marked in the row "Machine environmental dependency" should be confirmed when the other machine environment which is different from the machine environment shown at 5.1.1 is usd.

Table 62 shell script variable in run_multi.csh

Shell script variable name	Machine Environ mental dependen cy	The meaning
HEAP_CFG		The location of the configuration file for models
RSLT_LOG		The location of the start logfile
MULT_DIR	○	The location of Multi
SIMX_DIR		The location of the executable file of a simulator
DSRV_DIR		The location of the executable file of a debugging server (rteserv2)
TIME_OUT		Time-out time of the Multi debugging server
MULTIEXE		Multi executable file (with pass)
SPWN_DIR		The location of the pyhton script
RTESERV2		Executable file of a debugging server (rteserv2)
MULTI_PY		Multicfg.py script
MULTI_RC		The location of the startup file of Multi
SOFT_PEn		PEn(n:PEID)が使用するターゲットプログラム
MAIN_PEn		PEn(n:PEID)が使用するターゲットプログラムのメイン関数
CNCT_CMD		Connection parameter ※E3:-sc_heap_e3
LOAD_CMD		The load command of the target program
MOVE_CMD		Move command to the main function of the target program
RCON_CMD		reconnect command

A source of run_multi.csh for 2CPU is shown on Figure 55 as an example.

```
#!/bin/csh -f

# set variables
set SIMX_DIR="./"
set MULT_DIR="/proj/soft109/HeapE3/tools/GHS/v800-2000_v5/linux86"
set DSRV_DIR="../../multi"
set HEAP_CFG="./heap_multi.cfg"
set RSLT_LOG="./result_multi.txt"
set SOFT_PE1="./test_bp2/core1/core1"
set SOFT_PE2="./test_bp2/core2/core2"
set MAIN_PE1="main"
set MAIN_PE2="main"
set TIME_OUT="10"

set MULTIEXE="$MULT_DIR/multi"
set SPWN_DIR="$MULT_DIR/python/ghs_examples/spawn"
set RTESERV2="$DSRV_DIR/rteserv2"
set MULTI_PY="./multicfg.py"
set MULTI_RC="./multicfg.rc"

set CNCT_CMD="${RTESERV2} -D -cpu pe1 -cpu pe2 -sc_heap_config ${HEAP_CFG} -sc_heap_e3 ${SIMX_DIR}"
set LOAD_CMD="route my_program_1 prepare_target -load; route my_program_2 prepare_target -load;"
set MOVE_CMD="route my_program_1 e ${MAIN_PE1}; route my_program_2 e ${MAIN_PE2};"
set RCON_CMD="target reconnect; ${LOAD_CMD} ${MOVE_CMD}"

# multicfg.py
echo "dbw = self_dbw;"                         > ${MULTI_PY}
echo "dbw.RunCmd(" '"'${LOAD_CMD}'"' ");"       >> ${MULTI_PY}
echo "dbw.RunCmd(" '"'wait'"' ");"              >> ${MULTI_PY}
echo "dbw.RunCmd(" '"'${MOVE_CMD}'"' ");"       >> ${MULTI_PY}
echo "sys.path.append(" '"'${SPWN_DIR}'"' ");" >> ${MULTI_PY}
echo "import spawn;"                            >> ${MULTI_PY}
echo "spawn.spawn();"                           >> ${MULTI_PY}

# multi rc
echo debugbutton reconnect i=letter_r c='"'${RCON_CMD}'"'        > ${MULTI_RC}
echo "new -alias my_program_1 -bind debugger.pid.1 ${SOFT_PE1}" >> ${MULTI_RC}
echo "new -alias my_program_2 -bind debugger.pid.2 ${SOFT_PE2}" >> ${MULTI_RC}
echo wait                                                       >> ${MULTI_RC}
echo route debugger.pid.1 halt                                  >> ${MULTI_RC}
echo route debugger.pid.2 halt                                  >> ${MULTI_RC}
echo wait                                                       >> ${MULTI_RC}
echo "py -f ${MULTI_PY}"                                        >> ${MULTI_RC}

# invoke command
(echo -n "#### Created date was "; date +'%D %T %Z') >& ${RSLT_LOG}
(echo -n "#### current directory was "; pwd)       >>& ${RSLT_LOG}
(echo    "#### check programs are as follows")     >>& ${RSLT_LOG}
(grep "G3MCPU_PROGRAM" ${HEAP_CFG})                >>& ${RSLT_LOG}
(echo    "#### ${HEAP_CFG}")                       >>& ${RSLT_LOG}
cat ${HEAP_CFG}                                    >>  ${RSLT_LOG}
${MULTIEXE}  -p  ${MULTI_RC}  -servertimeout  $TIME_OUT  -connect="${CNCT_CMD}"  -cmd  taskwindow  >>&  $
{RSLT_LOG}
```

Figure 55 source of run_multi.csh for 2CPU

【Windows ACCELLERA/USK edition SystemC】
The script file name is run_multi_win.bat.
The contents of the variable / source used in run_multi_win.bat is prepared based on the contents of the variable / source used in run_multi.csh which is used for . Linux ACCELLERA/USK edition SystemC.

9. Procedure to add model IP

When adding a model IP to the platform, NSMVG3MSSV01 or NSMVG3MPEV01 which is changed to add model IP should be modified.

1. Add port / channel
2. Add configuration variable
3. Modify constructor
4. Modify destructor
5. Modify analysis procedure of configuration file
6. Modify procedure at simulation end

The respective contents are explained below.

9.1. Modify NSMVG3MSSV01 / NSMVG3MPEV01

Below is a procedure to modify G3MSS hierarchy and the G3MPE hierarchy.

9.1.1. Add port / channel

Please add ports / channels for new added model IP.

9.1.2. Add configuration variable

Please add configuration variables corresponded to the arguments (arguments in constructor of new added model IP) which may be able to be changed during running simulation.

9.1.3. Modify constructor

Please set the initial value for configuration variables added at 9.1.2, instantiate the new added model IP and set the connection of new added model IP instance.

9.1.4. Modify destructor

Please add the deletion procedure of the new added model IP in the class NSMVG3MSSV01 or NSMVG3MPEV01.

9.1.5. Modify analysis procedure of configuration file

Please add the analysis procedure of the configuration file to set added new configuration variable at 9.1.2 to the one of the prepared configuration file. For details, please define a configuration identifier and analyze it and set the value to the configuration variable

9.1.6. Modify procedure at simulation end

When you'd like to do some processing at the time of a simulation end by a new additional model IP, please add the procedure in the end_of_simulation function.

9.2. Modify NSMVRH850V01

There is no modification of RH850 hierarchy by adding new model IP basically.

9.3. Modify main.cpp

There is no modification of main.cpp by adding new model IP basically.

9.4. Modify Makefile

There is no modification of Makefile by adding new model IP basically.

9.5. Modify Makefile_scheap

Please add a model name of a new additional model IP to the make variable. Please add the procedure to call new mode IP's Makefile to build new model IP to all and clean target. In this case, please change the MODEL variable and LIBPATH variable with the appropriate variables which is passed to new additional model IP's Makefile. Please create the Makefile of new additional mode IP baed on the Makefile of existent model. In this acse, please be careful about overwriting the make variable in "Overwrite" row in Table 36 by the latest Makefile.

9.6. Modify Makefile_scheap.tb

Please add a path to a new additional model IP and the generated archive file name to the make variable.
Please add a specified path to INCLUDE_MINE and the archive file name to LIBS.

9.7. Modify configuration file

Please add the configuration ID added at 9.1.5 to the configuration file and set the appropriate value

9.8. Modify bus map file

When a new additional model IP has a target port of a local bus, please add an entry to the appropriate bus map file. Please add the model IP instance name specified as slave port name. Please add this port name referring to the new additional model IP specification.

9.9. Modify when adding Ptyhon command

When you'd like to do the parameter operation about an added model IP from Python, please refer to the following related sentences[9] and add the Python command.

[9] MSS-SG-12-00xx-01「Python I/F functional specification」

10. Others

10.1. Inevitable environmental variable to run simulator

担当：新井Ｓ
A necessary environment variable to run a simulator is below.
A setting file like setup.csh is usually prepared, so it's used to set it.

LD_LIBRARY_PATH （Linux）	When using the ASTC's USK SystemC library and peripheral connection class, the location of the library is added. The place of USK_SystemC library is usually /proj/soft108/Heap/tools/USK/usk-1.6.0. The location of the library is usually scheapCompile/lib.
PATH（Windows）	The location of the library should be specified. Usually, the location of the library is %SCHEP_HOME%¥scheapCompile¥lib and %SCHEAP_HOME%¥scheapCompile¥models¥iss¥fastiss_astc¥rh850¥iss¥lib¥osci-win32-msvc100¥rh850. When using the ASTC's USK SystemC library and peripheral connection class, the location of the library is added. The place of USK_SystemC library is usually E:¥SCHEAP¥toolsWin¥USK¥usk-1.6.0¥systemc¥lib-msvc80

10.2. Inevitable option to build SystemC library

A a necessary option is shown when building a SystemC library below.

/MD	It's necessary to share all objects with USK SystemC library using DLL. Hereinafter the explanation of /MD option: A run-time library of the version corresponding to the multi-threading and the version corresponding to DLL is used by application. **_MT** and **_DLL** are defined and MSVCRT.lib is inserted into object files by a compiler.

10.3. Error message

10.3.1. Error message

An error message of a simulator is shown below.
After an error message is output, a simulation will be finished immediately.

Message	The explanation	Hierarchy to ouput message
Error: cannot open config file	Output when A configuration file (heap.cfg) cannot be opened.	main
[FREQ] clock number(1st parameter) must be over 0, but specified [%f].	Output when the frequency specified by a parameter [FREQ] is smaller than 0.	main
Unexpected token was specified [%s] on [G3MCPU_DEBUG_MODE]	Output except when the debug mode specified by [G3MCPU_DEBUG_MODE] was not NONE, MULTI or CUBESUITE+.	NSMVG3MCPU

10.3.2. Warning message

A warning message of a simulator is shown below.
After a warning message is output, a simulation is continued.

Message	The explanation	replaced value(default value)	Hierarchy to ouput message
Unexpected token was specified [%s] on -n cycle. Treat 0 instead.	Ouput when the number of execution cycles isn't specified in the argument "-n" at a simulation start	0	main
Specified value exceed INT_MIN on -n cycle. Treat INT_MIN instead.	Ouput when the number of execution cycles is specified as a value smaller then minimum int value in the argument "-n" at a simulation start	INT_MIN	main
Specified value exceed INT_MAN on -n cycle. Treat INT_MAN instead.	Ouput when the number of execution cycles is specified as a value bigger then maximum int value in the argument "-n" at a simulation start	INT_MAN	main

11. Terminology

担当：吉永 S, 新井 S, 大塚氏

Word	meaning
アーキテクチャ	コンピュータのハードウェアおよびソフトウェアの設計仕様のこと。
ISS	Instruction Set Simulator の略で、マイコンの命令レベルシミュレータのこと。
IP	Intellectual Property の略で、システム LSI 上の機能ブロックのこと。
アドレスマップ	マイコン周辺機能がマイコンのどのアドレスに接続されているかを表す住所録のようなもの。
ACCELLERA	ハードウェア設計関連の技術および言語規格の標準化団体。OSCI と統合後、OSCI で開発された SystemC を現在管理している。OSCI とは、The Open System C Initiative の略。またこの組織が提供する SystemC リファレンスシミュレータなどのツール全般を指す場合もある。
SystemC	LSI 設計言語の一種で、回路動作を C++言語クラスライブラリによって行なう環境のこと。
デバッグサーバ	GUI デバッグ機能を提供するプログラムのこと。
PC	Program Counter の略で、マイコンで現在実行している命令の位置を示すレジスタのこと。
HEAP	Highe End Automotive Platform の略で、CPU 開発第三部が開発中の次世代自動車電装用マイコンのこと。
V850 E3(RH850)	CPU 開発第三部が開発中の V850 アーキテクチャの派生品のこと。
Multi	Green Hills Software Inc.の製品である組込マイコン向けプログラム開発環境のこと。
マルチコア	一つのシリコンの上に複数のプロセッサコアを搭載した状態のこと。
リンクディレクティブ	プログラムコードをどのメモリ位置に配置するかの指示情報のこと。
ASTC	Australian Semiconductor Technology Company の略。PFC1 周辺マクロの外注先
USK	ASTC 製 SystemC ライブラリ

12. Reference

担当：吉永 S, 新井 S, 大塚氏

Please refer to the following document if necessary.

document number	title
ZSG-F31-11-0097-02	SystemC RH850 ISS モジュール機能仕様書
ZSG-F31-11-0161-01	SC-HEAP_E3 バス I/F 概要書
ZSG-F31-11-0162-01	SC-HEAP_E3 バス I/F TLM タイミング仕様書
ZSG-F31-11-0163-01	SC-HEAP_E3 バス I/F 機能仕様書
ZSG-F31-11-0168-01	SC-HEAP_E3 バス機能仕様書
ZSG-F31-11-0122-01	SC-HEAP E3 MULTI 接続モジュール機能仕様書
MSS-SG-12-00xx-01	SC-HEAP_E3 Python I/F 機能仕様書

13. History

担当：吉永 S, 新井 S, 大塚氏

版　数	発行年月日	発　行　部　門	承　認	査　閲	担　当
初版	2012年3月30日	ソフトウェア統括部 ソフトツール部	佐藤	－	新井
	初版				
第２版	2012年7月31日	ソフトウェア統括部 ソフトツール部	佐藤	ー	新井
	全体 ・OSCIをACCELLERAに修正した。 表1、3、4、16 ・Pythonの情報を追加した。 図9 ・RH850階層及びINTC1,2を追加した。 表2 ・RH850階層及びINTC1,2を追加した。 図14、15 ・INTC2追加 図16,17 ・INTC1追加 図18,19 ・CPU階層図削除 2.3項 ・ASTC、RVCのMakefileの所在修正 5項 ・Pythonに対応した実行方法説明に変更 7.2項 ・“Multiデバッガでのボタンの連打”を削除 8.1項 ・NSMVRH850の説明追加 表26 ・割込み要求ポートを修正 表27 ・割込み信号についてのチャネル追加 表37 ・-py_scr追加 8.3.3項 ・RH850のインスタンスに変更 表41、42、44、54 ・MODEL_NSMVINTC1V01及びMODEL_NSMVINTC2V01追加 9.2項 ・NSMVRH850V01の修正追加 9.9項 ・Pythonコマンドを追加したい場合の修正追加				
第３版	2012年8月xx日	ソフトウェア統括部 ソフトツール部	佐藤	ー	新井

DMA

図2、8.7項
- Windowsビルド環境のソリューションRelease_pltfrm, Release_pltfrm_usk, Debug_pltfrm, Debug_pltfrm_uskは削除

図3、5
- scheapCompile/build/PRFRLV01→scheapCompile/build/PFV01に変更

表2
- コア間ルータがAXI→VCIに変更になったため8/Eリリース時点ではBACKWARD_DECODERは不要なので削除

表2
- LocalAPBバスはG3MSS階層からG3MPE階層へ移動

図12、13、14
- DMAはRH850階層からG3MSS階層へ移動

図15、16、17、18
- コア間ルータがAXI→VCIに変更

2.2、3.1、8.4、8.5、8.6、8.7項
- Pythonについての設定追加

表5
- [FREQ]削除

表11、31、3.4項
- AXIバスデコード削除

5.3項
- setFreqの引数変更

表18、図21、26
- コア間ルータがAXI→VCIに変更に伴うポート名変更

表20、21
- AXI用ソケット削除

8.3.2項
- main.cppにおけるパージング削除

8.3.4、8.3.5項
- Pythonに対応

8.7項
- SCHEAP-E3-G5.vcprojをscheap-e3-g5.lib作成専用のプロジェクトに変更
- sim.exeを作成するSCHEAP-E3-G5_EXE.vcprojの項目を8.7.1項に追加
- RH850-G5.libを作成するRH850-G5.vcprojの項目を8.7.5項に追加
- 周辺接続のlibを作成するPFV01.vcprojの項目を8.7.7項に追加

[EOF] VSB NPB 残り１サイクル 12