ERRATA
for
SCE Tutorial, Version 2.2.0 Beta

Mahesh Prabhu
Andreas Gerstlauer

This addendum to the *System-On-Chip Environment (SCE Version 2.2.0 Beta): Tutorial* (Technical Report CECS-TR-03-41, July 2003) discusses the differences and changes in the tutorial as compared to the latest version of SCE (Version 2.2.1).

In general, the primary differences are:

(a) SCE menus have changed and menu items have been added, moved or renamed.

(b) The top levels of the hierarchy of the vocoder design example now contain a proper testbench setup with *stimulus* and *monitor* behaviors and *speech* and *serial* I/O blocks as part of the actual *vocoder* design. However, the top-level *coder* design model has not changed and has the same hierarchy as referred to throughout the tutorial, i.e. the testbench changes and additions do not affect the flow.

(c) The profiler that is part of SCE has been rewritten and improved in later versions of SCE. As such, the various profiled numbers and estimated metrics (e.g. computation operations and hence corresponding graphs and charts) are different from the ones seen in the tutorial.

This will also affect the frame delay timing results obtained during simulations. In particular, it can be noticed that the pure software solution will already satisfy the given timing constraints. However, given that decoding and other applications will be added to the system at a later point (outside of the scope of this tutorial), for the purpose of designing the encoder as shown in the tutorial, we need to consider a significantly tighter constraint of about half the required final delays, i.e. 10ms per frame or 1.63s total (leaving enough room and engineering margin for additional tasks to be added later).

(d) The Communication Synthesis design step has been split into two parts: Network Exploration and Communication Synthesis. In general, SCE's communication synthesis capabilities have been vastly expanded to support advanced target architectures with complex bus networks and a wide array of bus communication and synchronization parameters.

The remainder of this document will list the relevant tutorial sections along with page numbers and the differences between the tutorial screen-shots and what is actually seen in SCE when following instructions on that page of the tutorial. Only the most significant differences will be discussed in so far as they are relevant to the tutorial flow. Other changes might be noticed but if they are not mentioned in this document, they will not affect the flow.

Note: this errata document only covers tutorial Section 1 through Section 3. Section 4 (Hardware Synthesis) and Section 5 (Software Synthesis) will be updated at a later point.

2.2.3. Open specification model

Page **17**: *Opening the testbench.sc file*

Use File→Import instead of File→Open, rest of the actions are the same. You will be opening the *testbench.sc* file with the Import window.

Page **22**: *Viewing the hierarchy*

To view the hierarchy, right click on *coder* and select Chart (instead of Hierarchy). This opens the graphical SpecC hierarchy window.

3.2.1. Try pure software implementation

Page **60**: *PE Allocation*

After pressing Ok to confirm the selection in the PE Selection dialog, a new dialog will pop up to allow entering the parameters of the allocated Motorola DSP. Use the default parameters, i.e. accept the dialog by pressing Ok.

3.2.5. Generate architecture model

Page **85**: *Architecture refinement*

Use the default settings in the hierarchy refinement options, i.e with Channel refinement enabled.

3.2.6. Browse architecture model

Page **87**: *Opening the hierarchy browser*

Select View→Chart to show the graphical hierarchy window.

3.4.1. Select bus protocols

Page **113-117**: *Network allocation*

For allocating the Network first select Synthesis→Allocate Network. In the Network Allocation window you will have 3 tabs: Busses, CEs and Connectivity. In the Busses tab, by default the "Motorola_DSP56600_PortA" bus will be present, which is the required bus. Go to the Connectivity tab and set the "Port0" of "HW" as "Slave" on "Bus0", which can be done using the drop down menu within the table at the intersection of the respective columns. In the end press Ok to complete the selection.

3.4.2. Map channels to busses

Page **118-119**: *Channel routing*

The step of mapping channels to busses is not necessary any more. Since there is only one bus in the system, the refinement tools will automatically find and select a default route to use. However, to look at channel mapping options, click into the Mapping column for any top-level channel and a Channel Routing dialog will open.

3.4.3. Generate communication model

As discussed in the introduction, the communication synthesis and refinement step has been divided into two parts. After network allocation and channel routing, first, network refinement has to be performed. Select Synthesis→Network refinement and in the Network Refinement window, use the default settings and press Start. Rename the generated model to "VocoderNet.sir" through the project window as in previous steps. Simulate the resulting network model of the design by selecting Validation→Compile and then Validation→Simulate to validate its correctness.

After network refinement, synthesis of communication in each bus segment of the system has to be done to complete the communication design process. For this, we start by setting the parameters for all communication links in the system. Select Synthesis→Assign Link Parameters, which will pop up the Link Parameter dialog with one tab for each bus in the system. In the Link Parameters dialog, address maps and synchronization mechanisms for every channel and every PE interface on the busses have to be defined. On "Bus0", select the Start address of "c_link_DSP__HW" to be a value between 0x8000-0xFFFF and the Interrupt to be "MasterIntA" from the drop down menu. Finally, click Ok to accept and confirm the bus parameter definition.

Page **120-121**: *Communication refinement*

Select Synthesis→Communication refinement to invoke the final communication refinement too. This pops a dialog where the user is given the open to either generate a Pin Accurate Model or a Transaction Level Model of the design. Accept the default option of generating a PAM and press Start to proceed.

3.4.4. Browse communication model

Page **124**: *Opening the hierarchy browser*

Select View→Chart to show the graphical hierarchy window.

Page **126**: *Viewing the processor hierarchy*

The first level of hierarchy inside the generated DSP model will contain the DSP core running next to its associated interrupt controller. To see the actual interrupt handling behavior inside the DSP core, select View→Add level once more to add a second level of hierarchy to be displayed.