EECS 222C: System-on-Chip Software Synthesis Lecture 5

Rainer Dömer

doemer@uci.edu

The Henry Samueli School of Engineering
Electrical Engineering and Computer Science
University of California, Irvine

Lecture 5: Overview

- Project Discussion
  - Assignment 2
  - Assignment 3
- Assignment 4
- System-on-Chip Environment (SCE)
  - SCE System Design Flow
  - SCE Demonstration
- Embedded Software
  - Execution Times
  - Embedded Operating Systems
    - Real-time Operating Systems (RTOS)
  - Scheduling
    - Aperiodic scheduling

Assignment 2

1. Practice SpecC Tools
   - Setup
     - `source /opt/sce-20080601/bin/setup.csh`
   - Examine simple examples
     - `mkdir simple_tests`
     - `cd simple_tests`
     - `cp $SPECC/examples/simple/* .`
     - `ls`
     - `vi HelloWorld.sc`
   - Practice the compiler
     - `man scc`
     - `scc HelloWorld –sc2out –vv -ww`
   - Practice the simulator
     - `./HelloWorld`
   - Practice the tools
     - `man sir_tree`
     - `scc Adder -sc2sir -o Adder.sir`
     - `sir_tree -bt Adder.sir FA`

DONE.

Assignment 2

2. Convert JPEG Encoder application into SpecC Model
   - Version 0
     - Compile JPEG Encoder with SpecC compiler
     - `scc jpegencoder.sc –vv -ww`

DONE.

   - Version 1
     - Introduce test bench
       - Stimulus behavior (`ReadBmp`)
       - Design-under-Test behavior (`JPEGencoder`)
         - » Seq. child behaviors (`DCT1`, `DCT2`, `Quantize`, `Zigzag`, `Huffman`)
         - » Communication through variables mapped to ports
       - Monitor behavior (`DiffGolden`)

DONE?

   - Version 1.1
     - Add timing to test bench
       - Print encoding time for each block (in Stimulus and/or Monitor)

DONE??

   - Version 2.0
     - Create a parallel model
       - Change DUT execution to '`par { }`'
       - Change communication to typed `double_handshake` channels

DONE??

   - Version 2.1
     - Create a pipelined model
       - Change communication to typed `queue` channels

DONE???

Assignment 3

1. Become familiar with the System-on-Chip Environment (SCE)
   - Setup
     - Note that we will use the 2004 version of SCE for the tutorial:
     - **`source /opt/sce-20041007/bin/setup.csh`**
     - **`rm –rf ~/.sce`**
     - **`mkdir demo`**
     - **`cd demo`**
     - **`setup_demo`**
   - Open the SCE Tutorial document
     - **`acroread SCE_Tutorial/sce-tutorial.pdf &`**
     - To protect the environment and save some trees, please *do not print* the tutorial document! It contains 250 pages and you will likely read it only once… ;-)
   - Follow the SCE Tutorial instructions
     - **`sce &`**
     - **`...`**

     Skipped!
   - Cleanup
     - When done (or to start over), clean up your demo directory
     - **`cd ..`**
     - **`rm –rf demo`**

Assignment 3

2. Simulate your JPEG Encoder model in SCE **Model "V2.1"?**
   - Setup
     - Note that we will use the 2008 version of SCE for the JPEG Encoder:
     - **`source /opt/sce-20080601/bin/setup.csh`**
     - **`rm –rf ~/.sce`**
     - **`cd jpegencoder`**
     - **`sce`**
   - Create a new project in SCE
     - **`Project->New`**
     - **`Project->Settings`**
       - Set verbosity level to 3 and warning level to 2
       - Adjust any other options the compiler may need to compile your model
     - **`Project->SaveAs "jpegencoder.sce"`**
   - Load your design model into SCE
     - **`File->Import "jpegencoder.sc"`**
     - **`Project->AddDesign`**
     - Right-click on **`jpegencoder.sir`** in the project window, and **`Rename`** the model to **`JPEGencSpec`**
   - Compile and simulate your model in SCE
     - **`Validation->Compile`**
     - **`Validation->Simulate`**

     Successful?

Assignment 3

3. Analyze your JPEG Encoder model in SCE
   - Setup
     - …continued from step 2 (previous page)
   - View the structural hierachy chart
     - Select the **`Main`** behavior in the behavior browser
     - Right-click **`->Chart`**
     - Double-click the chart to add a level of hierarchy
     - **`View->Connectivity`**
     - **`...`**
     - **`Window->Print…`** to file **`"jpegencoder.ps"`**
- Deliverables
  - SpecC source file
    - **`"jpegencoder.sc"`**
  - Hierarchy chart
    - **`"jpegencoder.ps"`**

Model "V2.1"?
Successful?
Complete?

- Due
  - by Friday, Oct 24, 2008, at noon
  - by email to **`doemer@uci.edu`** with subject "EECS222C HW3"

Assignment 4

1. Become familiar with the System-on-Chip Environment (SCE)
   - Setup
     - Note that we will use the 2003 version of SCE for the tutorial:
     - **`source /opt/sce-20030530/bin/setup.csh`**
     - **`rm -rf ~/.sce`**
     - **`mkdir demo`**
     - **`cd demo`**
     - **`setup_demo`**
   - Open the SCE Tutorial document
     - **`acroread SCE_Tutorial/sce-tutorial.pdf &`**
     - To protect the environment and save some trees, please *do not print* the tutorial document! It contains 250 pages and you will likely read it only once… ;-)
   - Follow the SCE Tutorial instructions
     - **`sce &`**
     - **`...`**
   - Cleanup
     - When done (or to start over), clean up your demo directory
     - **`cd ..`**
     - **`rm -rf demo`**

Assignment 4

2. Complete JPEG Encoder application into SpecC Model
   - Version 0
     - Compile JPEG Encoder with SpecC compiler
     - **`scc jpegencoder.sc –vv -ww`**
   - Version 1
     - Introduce test bench
       - Stimulus behavior (**`ReadBmp`**)
       - Design-under-Test behavior (**`JPEGencoder`**)
         - » Seq. child behaviors (**`DCT1`**, **`DCT2`**, **`Quantize`**, **`Zigzag`**, **`Huffman`**)
         - » Communication through variables mapped to ports
       - Monitor behavior (**`DiffGolden`**)
   - Version 1.1
     - Add timing to test bench
       - Print encoding time for each block (in Stimulus and/or Monitor)
   - Version 2.0
     - Create a parallel model
       - Change DUT execution to '**`par { }`**'
       - Change communication to typed **`double_handshake`** channels
   - Version 2.1
     - Create a pipelined model
       - Change communication to typed **`queue`** channels

Assignment 4

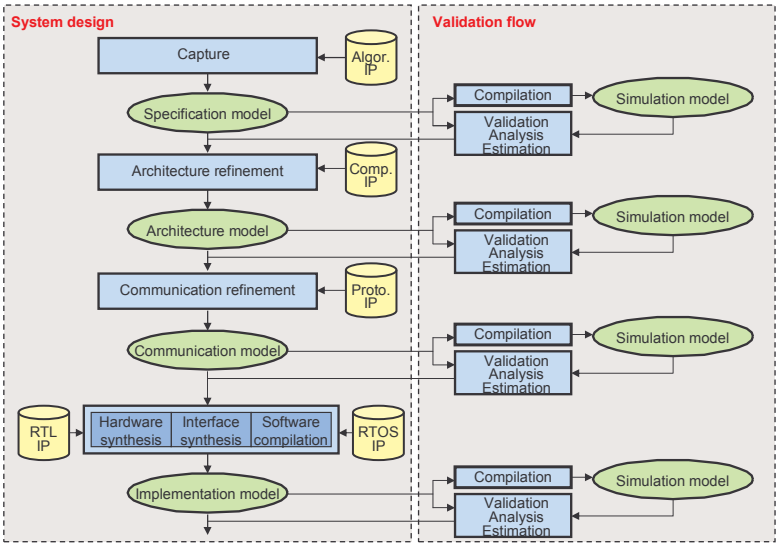
3. Simulate your JPEG Encoder model "V2.1" in SCE
   - Setup
     - Note that we will use the 2008 version of SCE for the JPEG Encoder:
     - **`source /opt/sce-20080601/bin/setup.csh`**
     - **`rm –rf ~/.sce`**
     - **`cd jpegencoder`**
     - **`sce`**
   - Create a new project in SCE
     - **`Project->New`**
     - **`Project->Settings`**
       - Set verbosity level to 3 and warning level to 2
       - Adjust any other options the compiler may need to compile your model
     - **`Project->SaveAs "jpegencoder.sce"`**
   - Load your design model into SCE
     - **`File->Import "jpegencoder.sc"`**
     - **`Project->AddDesign`**
     - Right-click on **`jpegencoder.sir`** in the project window, and **`Rename`** the model to **`JPEGencSpec`**
   - Compile and simulate your model in SCE
     - **`Validation->Compile`**
     - **`Validation->Simulate`**

No warnings!
Successful!

Assignment 4

4. Analyze your JPEG Encoder model in SCE
   - Setup
     - …continued from step 2 (previous page)
   - View the structural hierachy chart
     - Select the **`Main`** behavior in the behavior browser
     - Right-click **`->Chart`**
     - Double-click the chart to add a level of hierarchy
     - **`View->Connectivity`**
     - **`...`**
     - **`Window->Print…`** to file **`"jpegencoder.ps"`**
- Deliverables
  - SpecC source file
    - **`"jpegencoder.sc"`** **One single/complete file!**
  - Hierarchy chart
    - **`"jpegencoder.ps"`** **One chart with connectivity!**
- Due
  - by Friday, Oct 31, 2008, at noon
  - by email to **`doemer@uci.edu`** with subject "EECS222C HW4"

SCE System Design Flow



SCE Demonstration

- Design example: GSM Vocoder
  - Enhanced full-rate voice codec
  - GSM standard for mobile telephony (GSM 06.10)
  - Lossy voice encoding/decoding
    - Incoming speech samples @ 104 kbit/s
    - Encoded bit stream @ 12.2 kbit/s
    - Frames of 4 x 40 = 160 samples (4 x 5ms = 20ms of speech)
  - Real-time constraint:
    - max. 20ms per speech frame
      (max. total of 3.26s for sample speech file)
  - SpecC specification model
    - 29 hierarchical behaviors (9 par, 10 seq, 10 fsm)
    - 73 leaf behaviors
    - 9139 formatted lines of SpecC code
      (~13000 lines of original C code, including comments)

Embedded Software

- Chapter 4, part 1, of
  "Embedded System Design"
  by P. Marwedel (Univ. of Dortmund, Germany),
  Kluwer Academic Publishers, 2003.

  - **`Lecture5-es-marw-4a-aperiodic.ppt`**