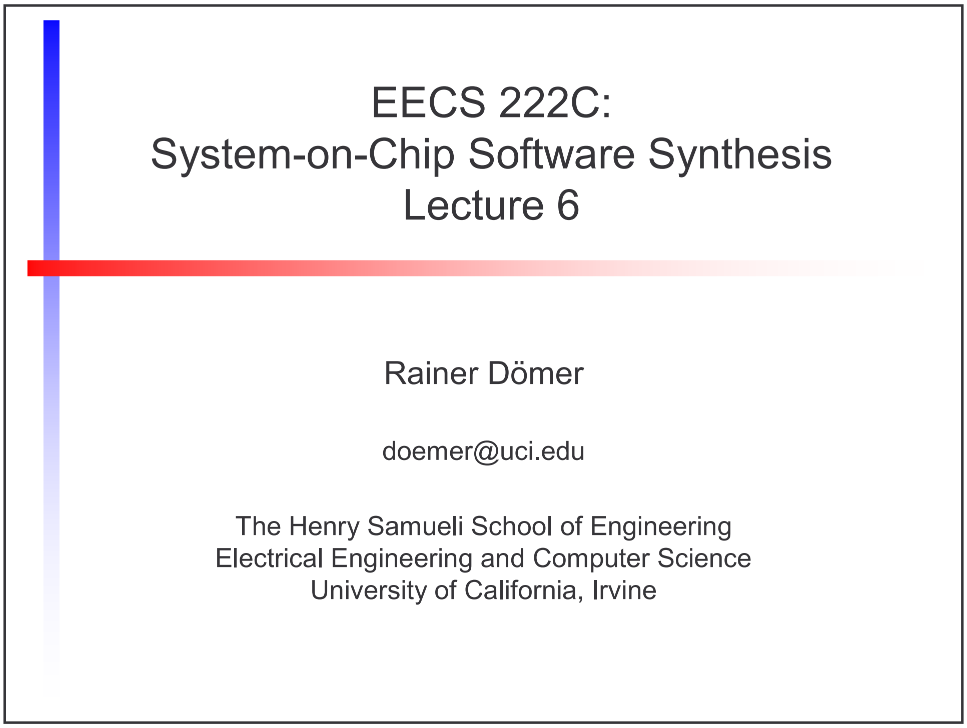
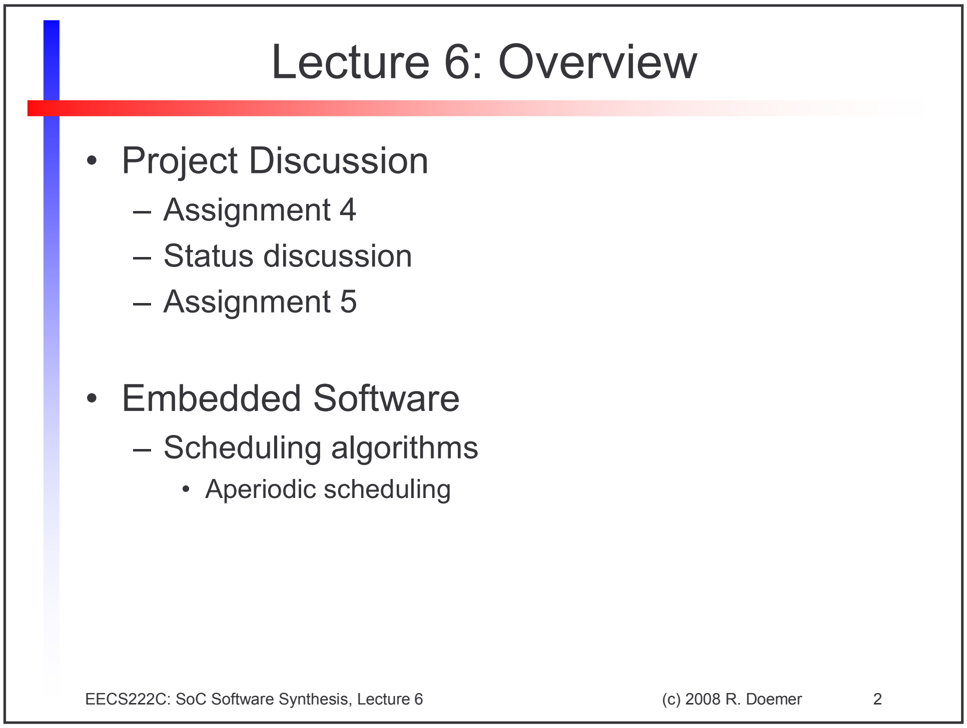


EECS 222C: System-on-Chip Software Synthesis Lecture 6

Rainer Dömer

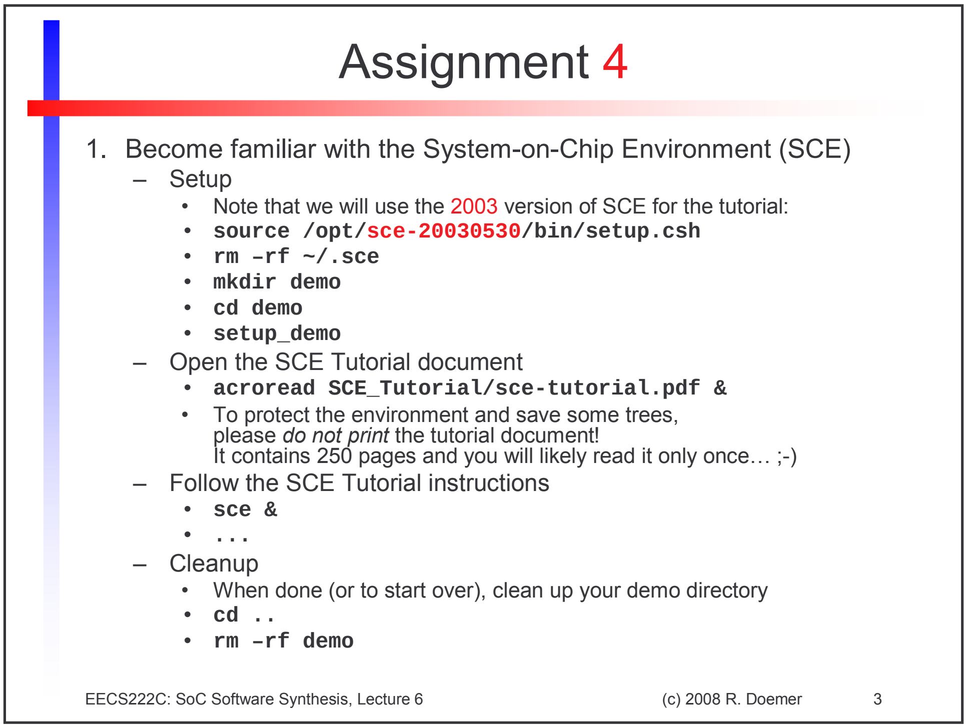
doemer@uci.edu

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



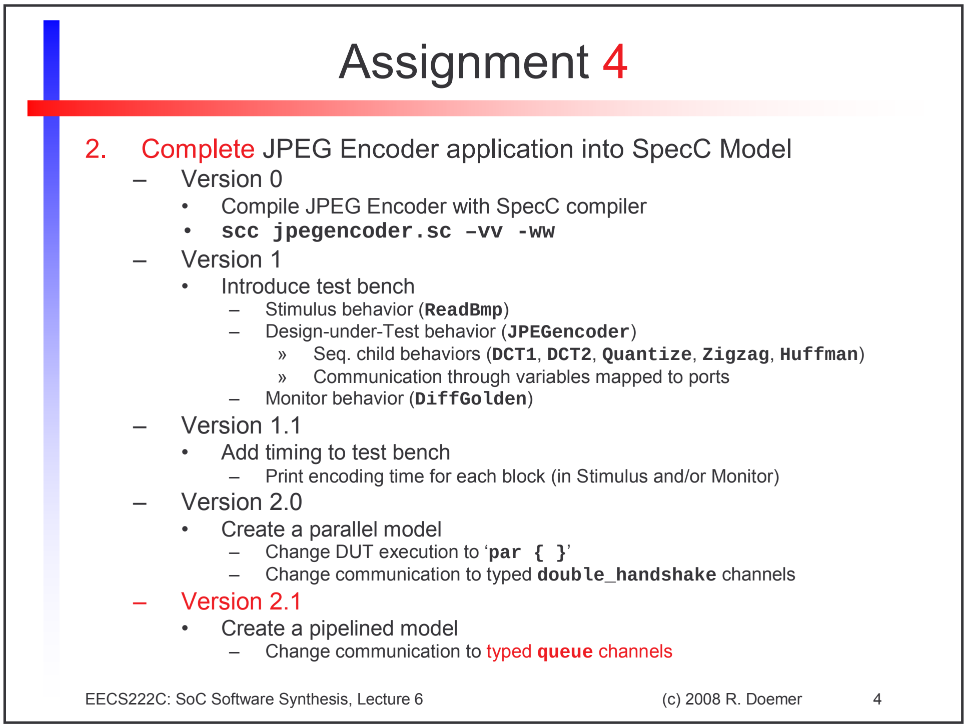
Lecture 6: Overview

- Project Discussion
  - Assignment 4
  - Status discussion
  - Assignment 5
- Embedded Software
  - Scheduling algorithms
    - Aperiodic scheduling



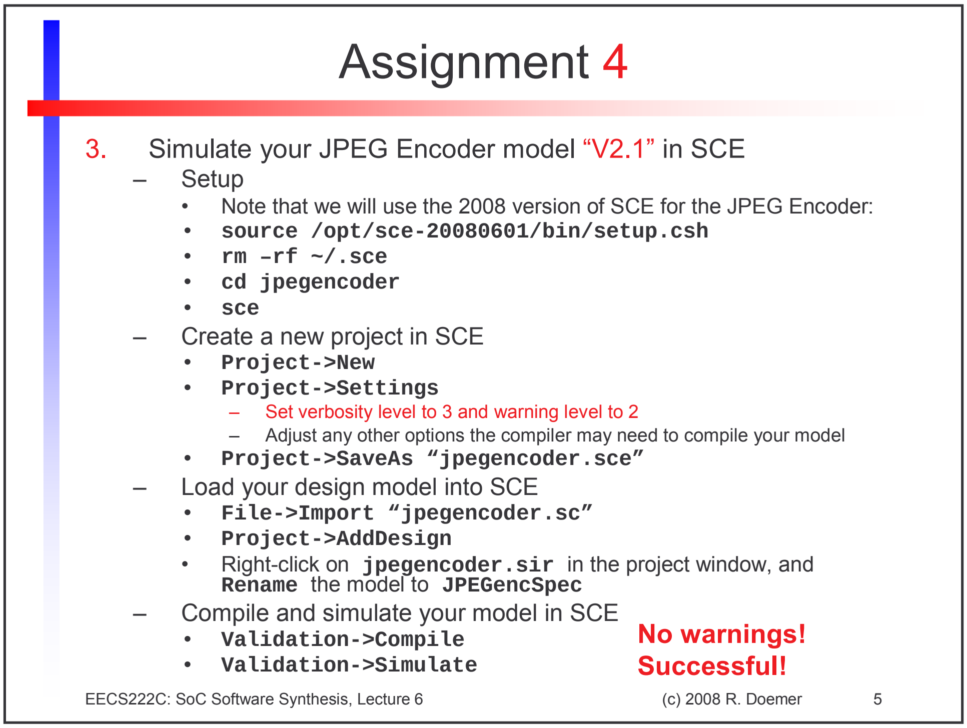
Assignment 4

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

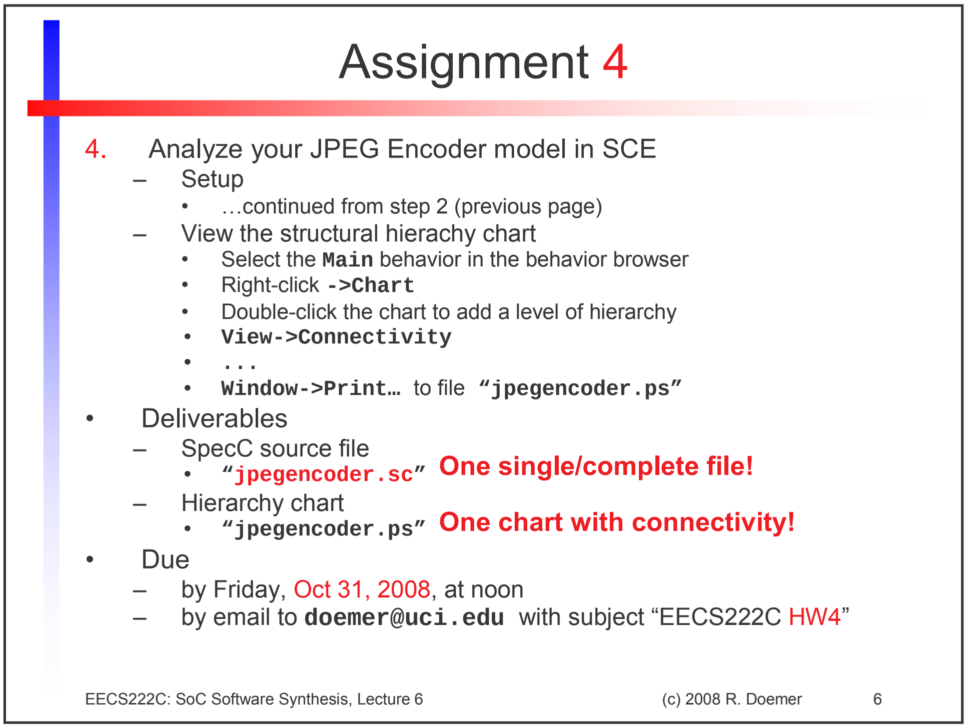


Assignment 4

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!

EECS222C: SoC Software Synthesis, Lecture 6 (c) 2008 R. Doemer 5

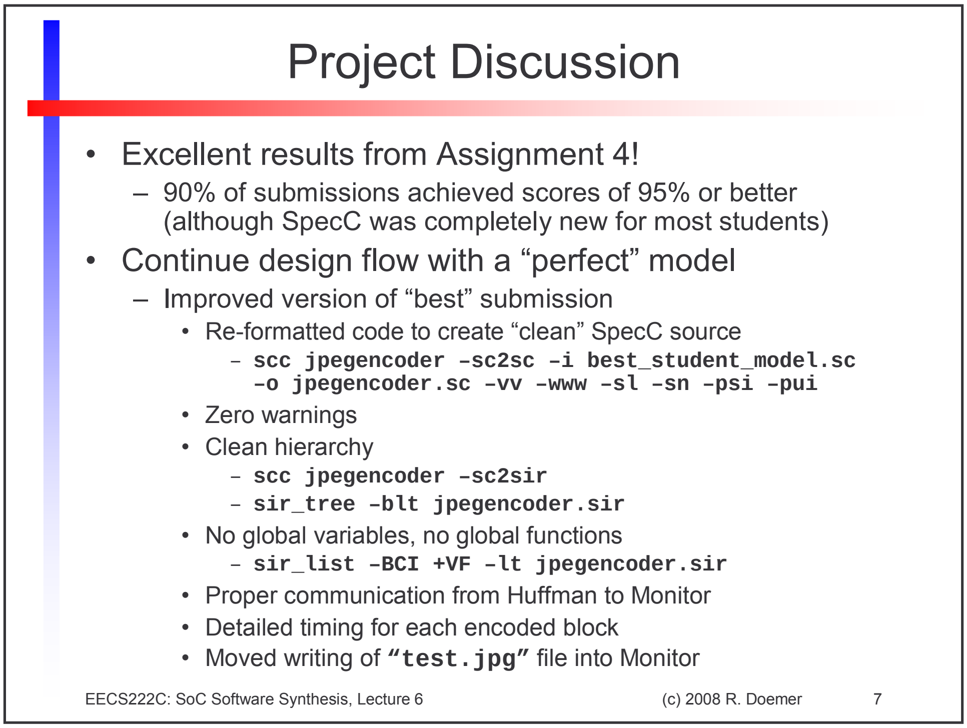


Assignment 4

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”`**

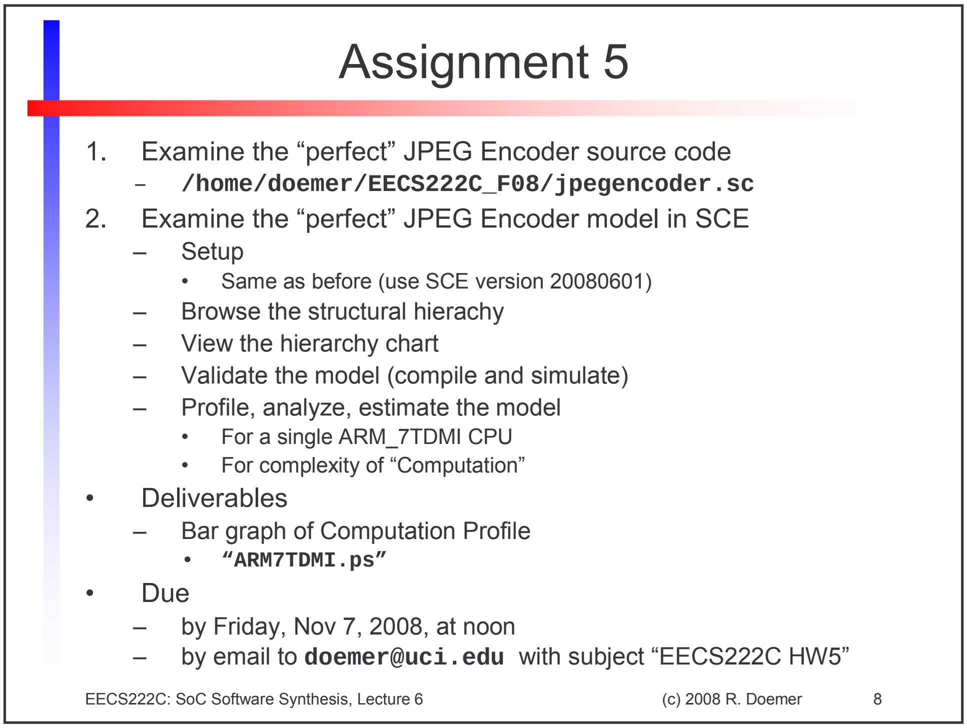
- Deliverables
  - SpecC source file
    - **`“jpegencoder.sc”`** **One single/complete file!**
  - Hierarchy chart
    - **`“jpegencoder.ps”`** **One chart with connectivity!**
- Due
  - by Friday, Oct 31, 2008, at noon
  - by email to **`doemer@uci.edu`** with subject “EECS222C HW4”

EECS222C: SoC Software Synthesis, Lecture 6 (c) 2008 R. Doemer 6



Project Discussion

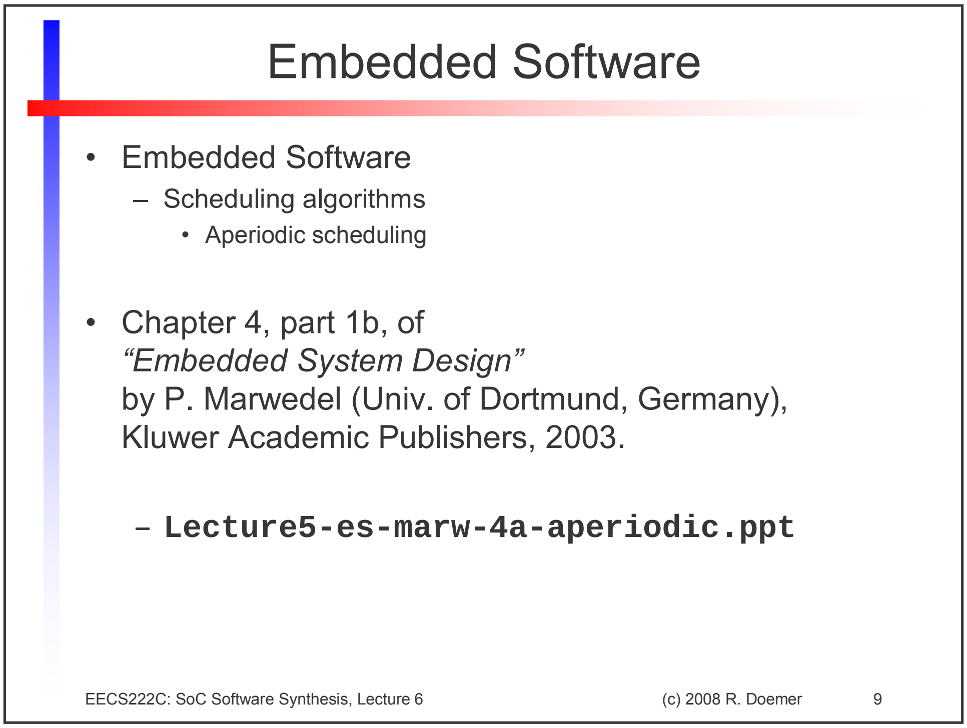
- Excellent results from Assignment 4!
  - 90% of submissions achieved scores of 95% or better (although SpecC was completely new for most students)
- Continue design flow with a “perfect” model
  - Improved version of “best” submission
    - Re-formatted code to create “clean” SpecC source
      ```
      – scc jpegencoder –sc2sc –i best_student_model.sc
        –o jpegencoder.sc –vv –www –sl –sn –psi –pui
      ```
    - Zero warnings
    - Clean hierarchy
      ```
      – scc jpegencoder –sc2sir
      – sir_tree –blt jpegencoder.sir
      ```
    - No global variables, no global functions
      ```
      – sir_list –BCI +VF –lt jpegencoder.sir
      ```
    - Proper communication from Huffman to Monitor
    - Detailed timing for each encoded block
    - Moved writing of **“test.jpg”** file into Monitor



Assignment 5

1. Examine the “perfect” JPEG Encoder source code
   - **/home/doemer/EECS222C_F08/jpegencoder.sc**
2. Examine the “perfect” JPEG Encoder model in SCE
   - Setup
     - Same as before (use SCE version 20080601)
   - Browse the structural hierachy
   - View the hierarchy chart
   - Validate the model (compile and simulate)
   - Profile, analyze, estimate the model
     - For a single ARM_7TDMI CPU
     - For complexity of “Computation”

- Deliverables
  - Bar graph of Computation Profile
    - **“ARM7TDMI.ps”**
- Due
  - by Friday, Nov 7, 2008, at noon
  - by email to **doemer@uci.edu** with subject “EECS222C HW5”



Embedded Software

- Embedded Software
  - Scheduling algorithms
    - Aperiodic scheduling

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

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