Lecture 11: Overview

- Embedded System Design Flow
  - Top-down design methodology
  - Refinement-based design flow
    - Specify
    - Explore
    - Refine

- System-on-Chip Environment (SCE)
  - Application example: GSM Vocoder
  - Interactive demonstration (part 1)
Top-Down Design Methodology

- **Requirements**: Capture
  - Pure functional
  - Transaction level
  - Bus functional
  - RTL / IS

- **Product Specification**
  - Specification model
  - Architecture refinement
  - Communication refinement
  - Implementation model

- **Constraints**
  - Untimed
  - Estimated timing
  - Timing accurate
  - Cycle accurate

Specify, Explore, Refine - Methodology

- **System Design**
  - Capture
  - Specification model
  - Architecture refinement
  - Communication refinement
  - Implementation model
  - Hardware synthesis
  - Interface synthesis

- **Validation Flow**
  - Compilation
  - Simulation model
  - Validation analysis estimation
Specify, Explore, Refine - Design Flow

- **Refinement steps**
  - Architecture refinement (Specification -> Architecture)
  - Communication refinement (Architecture -> Communication)
  - Cycle-accurate refinement (Communication -> RTL/IS)
    - HW / SW / interface synthesis

- **Levels of abstraction**
  - Specification model: untimed, functional
  - Architecture model: estimated, structural
  - Communication model: timed, bus-functional
  - Implementation model: cycle-accurate, RTL/IS

- **Component data bases**
  - Algorithms for specification
  - Components for architecture
  - Busses for communication
  - RTOS for SW
  - RTL components for HW

 Specify, Explore, Refine - Design Flow

- **Refinement Step 1: System Architecture**
  - Allocation of Processing Elements (PE)
    - Type and number of processors
    - Type and number of custom hardware blocks
    - Type and number of system memories
  - Mapping to PEs
    - Map each behavior to a PE
    - Map each channel to a PE
    - Map each variable to a PE

  ➢ **Result**
  - System architecture of concurrent PEs
    - with abstract communication via channels
Specify, Explore, Refine - Design Flow

- Refinement Step 2: PE Scheduling
  - For each PE, serialize the execution of behaviors to a single thread of control
  - Option (a): Static scheduling
    - For each set of concurrent behaviors, determine fixed order of execution
  - Option (b): Dynamic RTOS scheduling
    - Choose scheduling policy, e.g. round-robin or priority-based
    - For each set of concurrent behaviors, determine scheduling priority
  - Result
    - System model with abstract scheduler inserted in each PE

System-on-Chip Environment (SCE)

- Integrated Development Environment (IDE) with support of:
  - Graphical frontend (sce, scchart)
  - SLDL-aware editor (sced)
  - Compiler and simulator (scc)
  - Profiling and analysis (scprof)
  - Architecture refinement (scar)
  - RTOS refinement (scos)
  - Communication refinement (sccr)
  - RTL refinement (scrtl)
  - Software refinement (sc2c)
  - Scripting interface (scsh)
  - Tools and utilities (sir_list, sir_tree, ...)

(c) 2019 R. Doemer
SCE Main Window

SCE Source Editor
SCE Hierarchy Displays

SCE Compiler and Simulator
SCE Profiling and Analysis

SCE Demonstration

- Application 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)