The CAP Laboratory of Seoul National University presents

"Reconfigurable Codesign Environment for Modeling, Simulation, and Synthesis of Multimedia Embedded Systems"

Seamless codesign flow from Functional simulation to system synthesis

Block diagram specification of system behavior reusing library blocks

Optimal architecture, synthesized HW/SW code, estimated performance

CAP Lab, Seoul National University, Korea.  http://peace.snu.ac.kr/peace
Need of HW/SW Codesign Methodology

This is a conventional design flow of embedded systems. It has the following problems that the HW/SW codesign methodology aims to solve.

1. System evaluation is done after the SW code is ported on the hardware prototype. It forms a very expensive and inefficient design loop if further debugging or optimization is requested. HW/SW codesign evaluates the system performance by HW/SW co-simulation on a virtual prototyping environment without building a real hardware prototype.
2. System architecture is decided based on the past experiences or simple profiling. HW/SW codesign provides a systematic way of design space exploration.

Current HW/SW Codesign Practice

This is a current HW/SW codesign practice. It has the following problems that PeaCE aims to solve.

1. Each design step is isolated so that integration of design tools is burdensome. PeaCE provides seamless codesign flow from functional simulation to system synthesis.
2. Design flow heavily depends on which design tool is selected. PeaCE is a reconfigurable framework to which 3rd party design tool can be easily integrated.
3. Design space exploration should be done manually using a TLM simulation tool. PeaCE includes an interactive DSE framework that consists of HW/SW partitioning and communication architecture optimization.

CAP Lab, Seoul National University, Korea. http://peace.snu.ac.kr/peace
The key feature of PeaCE codesign flow is reconfigurability. The green rectangular boxes describe the design steps while the orange rounded boxes indicate the input and output xml files of the design steps. Each design step is modularized so that various algorithms or other CAD tools can be integrated with a minimal effort of wrapper design that translates the xml files. The numberings indicate the sequence of the design steps associated with design taps in the user interface.

CAP Lab, Seoul National University, Korea. [http://peace.snu.ac.kr/peace](http://peace.snu.ac.kr/peace)
(1) System Specification and Functional Simulation

Objective:
This demo shows how system-level specification of a complicated multi-mode multi-media embedded systems can be accomplished in PeaCE. And PeaCE automatically synthesizes a multi-threaded software from the specification for functional simulation. The demo system supports three mode of operations (or tasks) : a video phone, DivX player, and MP3 player.

Feature:
✓ Extended SDF (SPDF+FRDF) specification of signal processing tasks
  - H.263 Encoder / Decoder
  - G.723 Encoder / Decoder
  - MP3 Decoder
✓ FSM specification of run-time system control task
✓ Task-level specification in the codesign backplane
  - Signal processing tasks + Control task + Network interface tasks

System Specification:
✓ Four modes of operation
  - Video Phone, Self Camera, DivX Player, and MP3 Player
✓ Demo platforms
  - Pentium3 notebook

Self Camera demo on Linux notebook

Top-level schematic

Three task schematics

ffSM schematic

CAP Lab, Seoul National University, Korea. http://peace.snu.ac.kr/peace
Objective:
This demo shows the generation of three interface XML files from the top-level specification for the subsequent design steps. In the architecture selection tab, we select a candidate architecture for the final implementation. And then we generate XML files in the graph analysis tab. clustered.xml describes block dependency, TimeCost.xml has block execution time on processing elements and mode.xml defines timing constraints of the system.
(3) Automatic / Manual Partitioning

**Objective:**
This demo shows that PeaCE can generate sched.xml file and a Gantt chart as the partition result for DivX player. PeaCE can provide not only automatic partitioning but also manual partitioning capability for a given architecture model. The automatic partitioning algorithm used in PeaCE considers resource sharing between tasks and loop structures. If manual partitioning is selected, PeaCE still gives you the schedule of blocks based on the partitioning decision.

**Interface Files:**
- Input files: clustered.xml, TimeCost.xml and mode.xml
- Output result: sched.xml, Gantt chart

This ‘sched.xml’ represents how function blocks are partitioned and scheduled for each processing component.

CAP Lab, Seoul National University, Korea. [http://peace.snu.ac.kr/peace](http://peace.snu.ac.kr/peace)
Objective:
In PeaCE, HW/SW cosimulation is performed at different design stages for different purposes; memory trace generation for communication architecture DSE and cosimulation for performance verification. In this demo, Divx player with HW IDCT is cosimulated for verification purpose.

Multi-purpose Cosimulation
- Inputs
  - Generated C codes, VHDL codes
  - Interface codes (OS API, bus wrapper)
- Output
  - Simulated cycles for each task, or
  - Memory traces

Features:
- Backplane approach
- Virtual Synchronization technique
  - Fast and accurate cosimulation.

Performance Comparison:
- H.263 decoder with HW IDCT cosimulation
- Total 3 frames

<table>
<thead>
<tr>
<th></th>
<th>cycles</th>
<th>time</th>
<th>perf.</th>
</tr>
</thead>
<tbody>
<tr>
<td>*Virtual Sync.</td>
<td>9,539,078</td>
<td>11 sec</td>
<td>867 KIPS</td>
</tr>
<tr>
<td>**Seamless CVE</td>
<td>9,756,226</td>
<td>10,489 sec</td>
<td>0.93 KIPS</td>
</tr>
</tbody>
</table>

* arm922T  ** arm946e-s

Objective:
This demo shows how performance estimation is obtained and how it is used for determining partitioning and mapping.

Performance Estimation
- Computation cycle of each block
- Memory access count of each block

CAP Lab, Seoul National University, Korea.  http://peace.snu.ac.kr/peace
Objective:
This demo shows the SW and HW cosynthesis capability after partitioning is completed. In this demo, the system specification of Divx player is partitioned into 2 sub-graphs for a software and a hardware. C code for SW and VHDL code for HW are automatically generated from the partitioned sub-graphs.

SW/ HW Cosynthesis
✓ Inputs
  - System specification
  - Schedule (sched.xml)
  - SW/HW block libraries
✓ Intermediate data
  - Partitioned graphs for each components including SND/RCV blocks for communication
✓ HW synthesis
  - Automatic controller synthesis
✓ Output
  - HW/SW codes for each components

CAP Lab, Seoul National University, Korea. http://peace.snu.ac.kr/peace
Objective:
This demo shows the interface code generation and prototyping on an Altera Excalibur™ based system.

Interface Code
- Synchronization path
  - Sync. controller
  - Interrupt controller
- Data path
  - Bus architecture : AHB
  - Bus wrapper for HW IP

Prototyping on Altera Excalibur based Prototyping System
- CPU : ARM922T core, 166Mhz
  - H.263 Decoder
  - mp3 player
- FPGA
  - HW IP : Motion compensation block
  - Interface codes
    - AHB Bus & AHB wrapper
    - Sync. controller

Prototyping Result: Divx player
- H.263 Decoder : 7 frame/sec , QCIF
- mp3 player : 24kbps, 22050hz, Mono
- FPGA
  - HW Area : MC + Interface
    - 26689 LUT : 69% of 1M gate FPGA
  - HW clock speed
    - CPU : 166Mhz, FPGA : 35Mhz

CAP Lab, Seoul National University, Korea. [http://peace.snu.ac.kr/peace](http://peace.snu.ac.kr/peace)
Key Features & SW Structure

Key Features

- **Open-source framework** to promote collaboration of system level design researches.
- **Separate Java based GUI (Hae) from kernel.** The user interface is platform independent and web-based user interface will be provided.
- **Object-oriented C++ kernel inherited from the Ptolemy project.** Decoupled from the user interface, the kernel is designed for high performance with more maintenance requirements.
- **Formal and compositional system specification. PeaCE** uses different models of computations for functional and control representations: dataflow (SPDF) graphs for function representation and FSM (fFSM) for control representation. They reside in yet another model of computation for the codesign backplane.
- **Automatic hardware/software/interface synthesis** from SPDF and fFSM specifications.
- **Ease of customization.** By adding target specific modules and adjusting the user interface, PeaCE can be converted into your tool, specific to your target application.

PeaCE has a server-client architecture as shown below. PeaCE server is developed in a Linux platform though it can run in a Solaris or a Windows machine. PeaCE server is programmed in C++. PeaCE client is written in Java to provide a platform independent user interface. We named the client program as “Hae” which has double meanings in Korean: “sun” and “solution”.

The server consist of three modules

- Daemon process
- Server Kernel
- Block/Component Library

The GUI (Hae) runs in a client machine.

When you start the client program, it is connected to the daemon process of the server side via the designated port. Then the daemon process forks the server process and makes the server run.

CAP Lab, Seoul National University, Korea. [http://peace.snu.ac.kr/ peace](http://peace.snu.ac.kr/peace)
PeaCE vs. Well-known Tools

PeaCE vs. Matlab/ Simulink:
Matlab/Simulink is the most widely used tool for embedded system design, particularly for behavior synthesis. It has rich libraries for reusable blocks and demos as well as numerous supporters. Since there is no connection to the system synthesis tool from Matlab, you need to start from scratch for developing the real system. PeaCE has a seamless workflow starting from the behavior simulation up to the high quality software and hardware synthesis. Moreover, PeaCE can work with Matlab for system simulation.

PeaCE vs. Ptolemy II:
Both works are brought up from the Ptolemy classic project. But, Ptolemy II is being totally redesigned in Java after developing the theory for integrating models of computation based on the Ptolemy’s experience. Ptolemy II has strong simulation and modeling capability. PeaCE is being built on top of Ptolemy classic but significantly extended for the sake of efficient and convenient system synthesis.

PeaCE vs. Other CAD Tools:
Recently C-based design tools appear to the market for system simulation and interface synthesis after mapping between function and architecture is somehow completed. PeaCE advocates higher-level specification method using SPDF and FSM for data processing and control tasks. Since PeaCE can produce C and HDL (including SystemC and VHDL) code from the initial specification, PeaCE is complementary to those CAD tools, raising the level of abstraction in the synthesis. One of the major design concerns in PeaCE is tool integration.

PeaCE vs. CASE Tools:
Since PeaCE can synthesize the C code from the initial block diagram specification, it can be regarded as a CASE tool for embedded software, which is natural since software engineering is a subset of system engineering. There is no widely-recognized CASE tool for embedded software yet while extensive researches are being performed: Through our experiences, PeaCE is viable as CASE tool for embedded software especially for multi-media application.

PeaCE is an open-source research product for embedded system design:
So, we would like to invite you to use our tool and to check out the claims that we make above. One thing we can guarantee is that we are working hard and your feedback will be a great help to us.

Platform:

<table>
<thead>
<tr>
<th>PeaCE</th>
<th>Disk Size</th>
<th>About 600 MBytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>OS &amp; Compiler</td>
<td>Linux Redhat 7.x &amp; gcc 2.95.3</td>
<td></td>
</tr>
<tr>
<td>Tcl/Tk</td>
<td>Tcl 7.6 Tk 4.2 &amp; itcl/itk 2.2</td>
<td></td>
</tr>
<tr>
<td>Hae</td>
<td>Disk Size</td>
<td>About 5 MBytes</td>
</tr>
<tr>
<td>JVM</td>
<td>JDK 1.4.1 or above</td>
<td></td>
</tr>
</tbody>
</table>

CAP Lab, Seoul National University, Korea. http://peace.snu.ac.kr/peace
# CAP People (Prof. Soonhoi Ha)

<table>
<thead>
<tr>
<th>Name</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dr. Hyunok Oh</td>
<td>Cosynthesis, FRDF, memory optimization, H.263, Hae</td>
</tr>
<tr>
<td>Dr. Dohyung Kim</td>
<td>System level specification &amp; co-verification</td>
</tr>
<tr>
<td>Chaeseok Im</td>
<td>Low-power system design techniques</td>
</tr>
<tr>
<td>Sungchan Kim</td>
<td>Design space exploration</td>
</tr>
<tr>
<td>Hyunuk Jung</td>
<td>Automatic hardware synthesis</td>
</tr>
<tr>
<td>Wookchul Jeun</td>
<td>Fixed point simulation, parallel programming</td>
</tr>
<tr>
<td>Choonseung Lee</td>
<td>Hardware /software cosynthesis</td>
</tr>
<tr>
<td>Youngmin Yi</td>
<td>RTOS, Hardware /software cosynthesis</td>
</tr>
<tr>
<td>Seongnam Kwon</td>
<td>Performance estimation, TLM</td>
</tr>
<tr>
<td>Youngpyo Joo</td>
<td>Automatic interface generation</td>
</tr>
<tr>
<td>Kyoungjoo Oh</td>
<td>System architecture design</td>
</tr>
<tr>
<td>Kiseun Kwon</td>
<td>RTOS, task level simulation</td>
</tr>
<tr>
<td>Hoeseok Yang</td>
<td>Automatic hardware synthesis</td>
</tr>
<tr>
<td>Haewoo Park</td>
<td>Embedded software</td>
</tr>
<tr>
<td>Hyeyoung Hwang</td>
<td>Ubiquitous system</td>
</tr>
<tr>
<td>Dukyoung Yun</td>
<td>Ubiquitous system</td>
</tr>
</tbody>
</table>

## Sponsors
- Grant No. M1-0104-00-0015 from **National Research Laboratory** fund of Ministry of Science & Technology
- Grant No. 0501-2003001 from **Brain Korea 21 Project** fund of Ministry of Education & Human Resources Development

## Contact
- The **CAP** laboratory
  - School of Computer Science and Engineering
  - Seoul National University,
  - San 56-1, Sinlim-dong, Gwanak-gu, Seoul 152-742, Korea
- TEL: +82-2-880-7292
- FAX: +82-2-879-1532
- Email: peacemakers@iris.snu.ac.kr

**CAP Lab, Seoul National University, Korea.** [http://peace.snu.ac.kr/peace](http://peace.snu.ac.kr/peace)