Configurable SID-based Multi-core Simulators for Embedded System Education

Chung-Wen Huang,

Wei-Kuan Shih,

Yarsun Hsu,

Jenq Kuen Lee (jklee@cs.nthu.edu.tw)

National Tsing-Hua University, Taiwan.

Embedded Software Consortium, Taiwan.

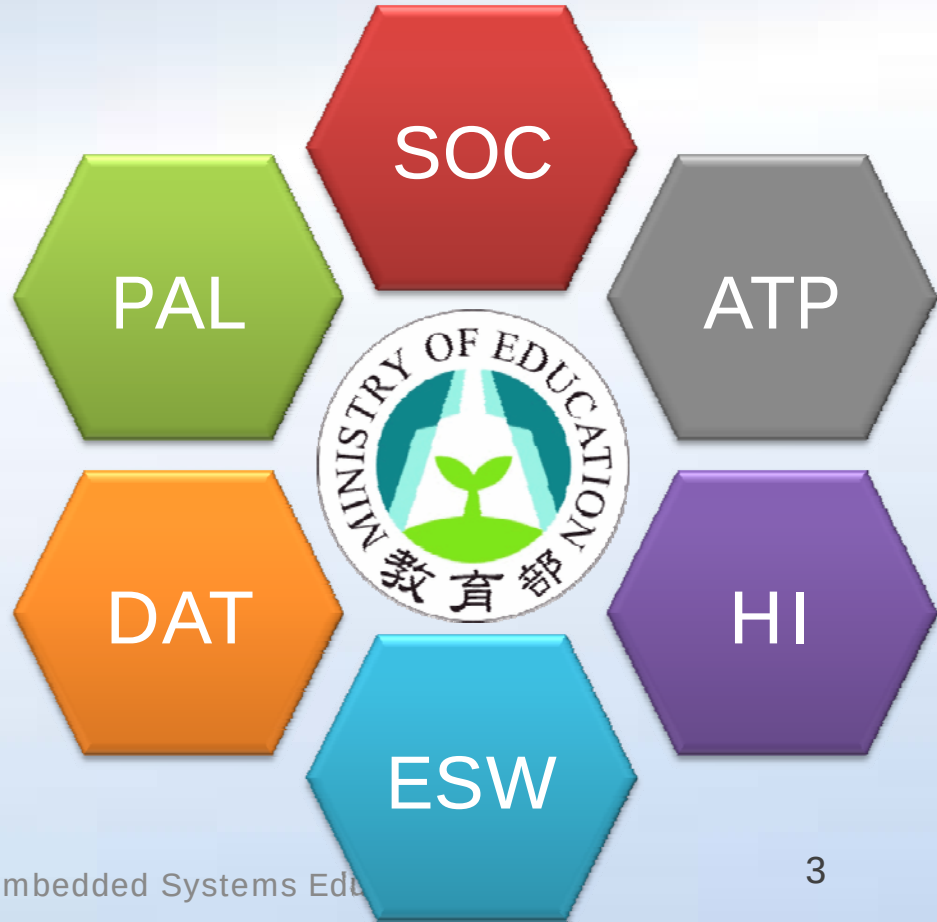
Outline

- Background
- SID Simulation Framework
- Heterogeneous Multi-core Simulation Platform
- Embedded System Courses/Projects based on SID Framework
- Conclusion

Background of Taiwan Embedded Software Consortium

- The Embedded Software Consortium, established in February 2004 in Taiwan, is the consortium funded by the Ministry of Education (Other consortiums including EDA, heterogeneous integrations, etc).

  - The ESW Consortium focuses the development of embedded software curriculum.
  - We hope to provide a reference curriculum for universities in Taiwan to develop their embedded program.
  - Developing embedded multi-core curriculum is becoming a focus for this program.



CourseWare: Multi-core Simulation Tools

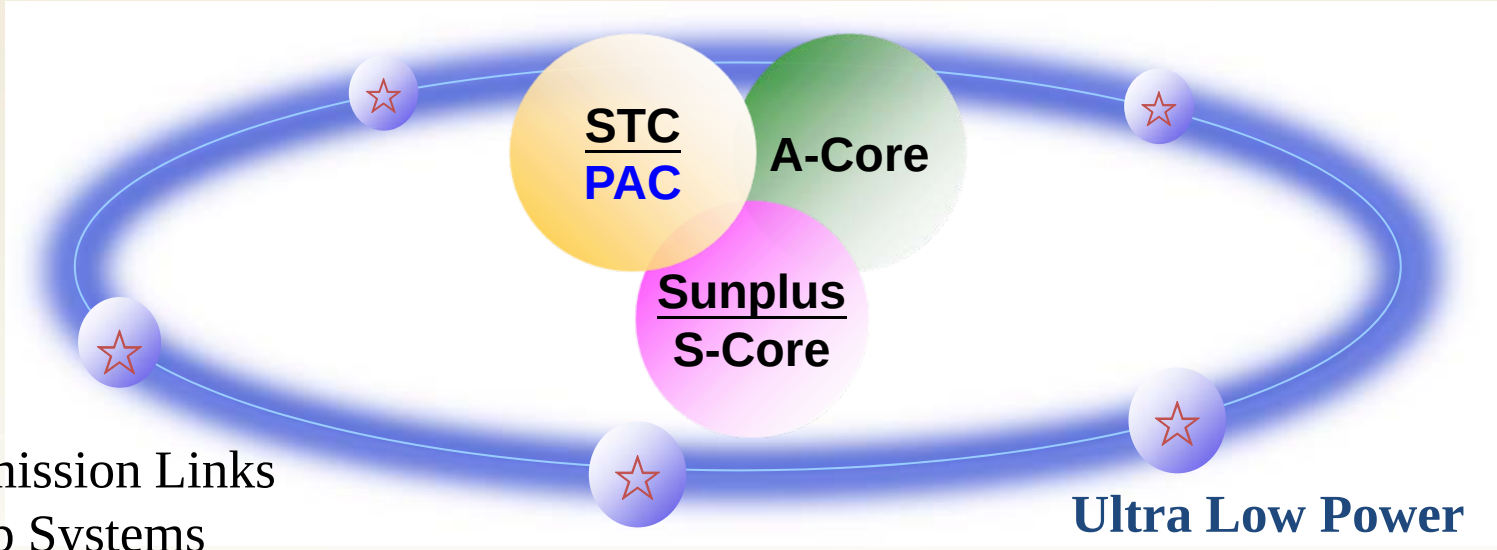
- With the emerging of multi-core designs for embedded systems, there is a need of multi-core simulation tools for courseware and class experiments.

- Several issues are considered.
  1. To have a sequence of course work based on similar toolkits
  2. To be able to configure the architectures with a variety of processor IPs (including MPU and DSP processors).
  3. To accommodate ingenious local IPs for experiments.
  4. Experiment with multi-core programming models.
  5. Experiment with interconnection networks.
  6. Experiment with embedded multi-core applications.
  7. Learn how to track and debug multi-core programs and performance tuning.
  8. Experiment with middleware experiments.

StarIP Programs

Key Techniques in Chip Systems

Messenger – Distributed Radio Transmitter System



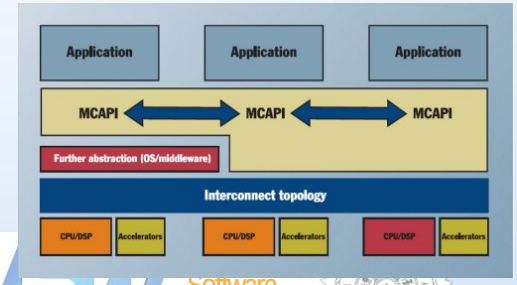
Transmission Links in Chip Systems

Ultra Low Power (ULP) DSP Core

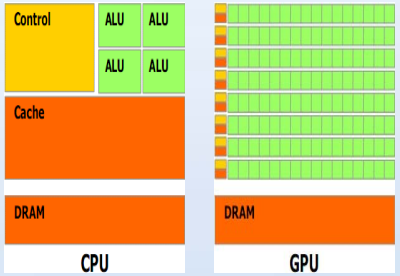
Low-Power Dual-Processor System

Experiments with Various Parallel Programming Models

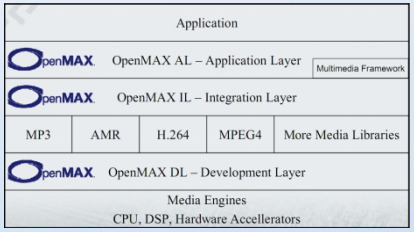
- MCAPI
  - Released the first spec at 2008 by the Multicore Association
  - A standardized API for communication and synchronization
  - Focus on high performance and low memory footprint
  - Defines three communications type: Messages, Packet channels, and Scalar channels
- CUDA
  - Introduced by NVIDIA at 2006 for GPU architecture
  - C extensions and need compiler support
  - SPMD-like programming model
  - Fine-grained data parallelism and thread parallelism
- OpenCL
  - Released the first spec at 2008 by Khronos Groups with many industry-leading companies and institutions
  - Target for heterogeneous multicore environment
  - For general purpose applications
- OpenMAX
  - Released the first spec at 2007 by Khronos Groups
  - Defined three layers of API interface to abstract software and hardware difference for portability
  - For multi-media applications
  - Lacked of multicore support
- Streaming RPC
  - ICPP 2008
  - Integration of Remoting with streaming mechainisms



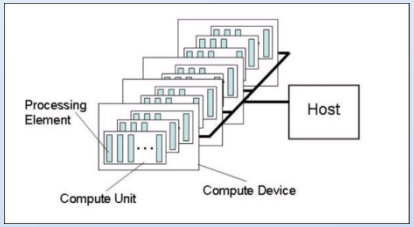
Source: http://www.multicore-association.org/



Source: NVIDIA Corp., "NVIDIA CUDA Compute Unified Device Architecture Programming Guide V2.0", 2008



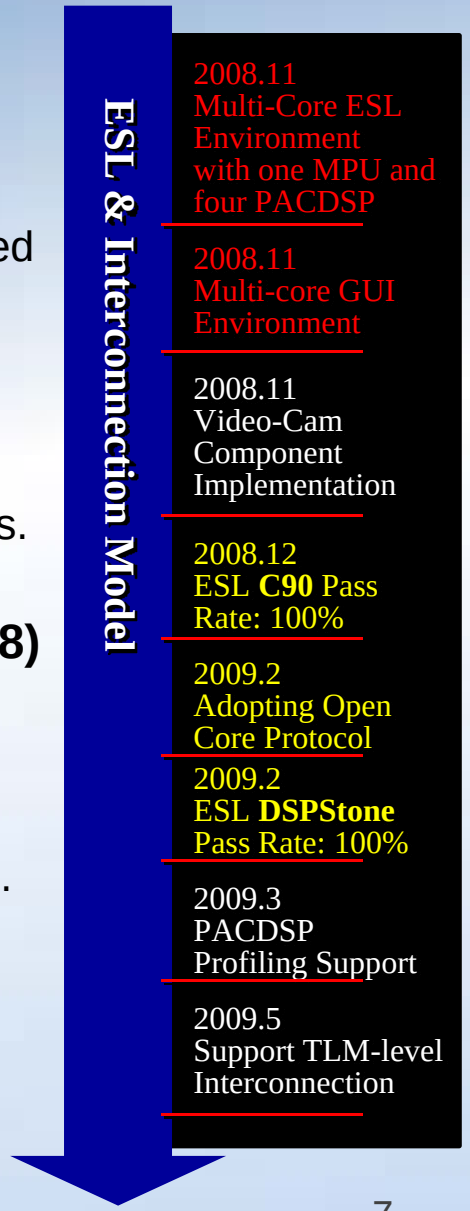
Source: Giulio Urlini, "The OpenMAX Integration Layer Standard," CELF Embedded Linux Conference, 2007



Source: Neil Trevett, "OpenCL the Open Standard for Heterogeneous Parallel Programming," SIGGRAPH Asia, 2008

SID Multicore Simulation Milestones

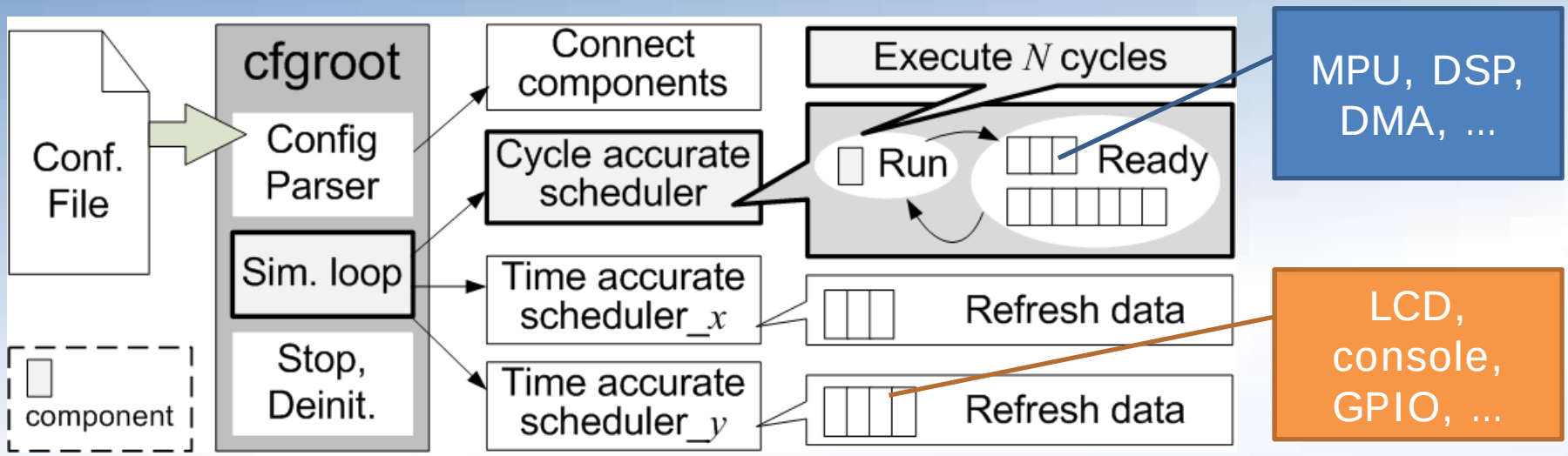
- **SID Origins (<2001)**
  - SID is an open source framework for building computer system simulations.
  - The simulation framework is comprised of a set of loosely coupled components (in C, C++ or TCL).
  - SID was first released at 2001 Embedded System Conference.
- **SID in the Public Domain (http://sourceware.org/sid/)**
  - 100+ components are source code available in public domains.
    - such as processors, memory, DMAs, LCDs, and peripherals.
  - 10+ architecture description files available for references.
- **SID Components of Taiwan Dual-core Platform (2007-2008)**
  - 5-way issues VLIW DSP component (NTHU, ITRI STC)
  - Virtual I/Os of Video Camera and Network Interface. (NTHU)
- **SID Multicore Simulation (2008~2009) (NTHU)**
  - Configurable multi-core simulation with one MPU and four DSPs.
  - Multi-core GUI development environment.
  - OCP compliant interconnect implementation.
  - Multi-core data profiling and tracing.



Comparison of ESL Simulation

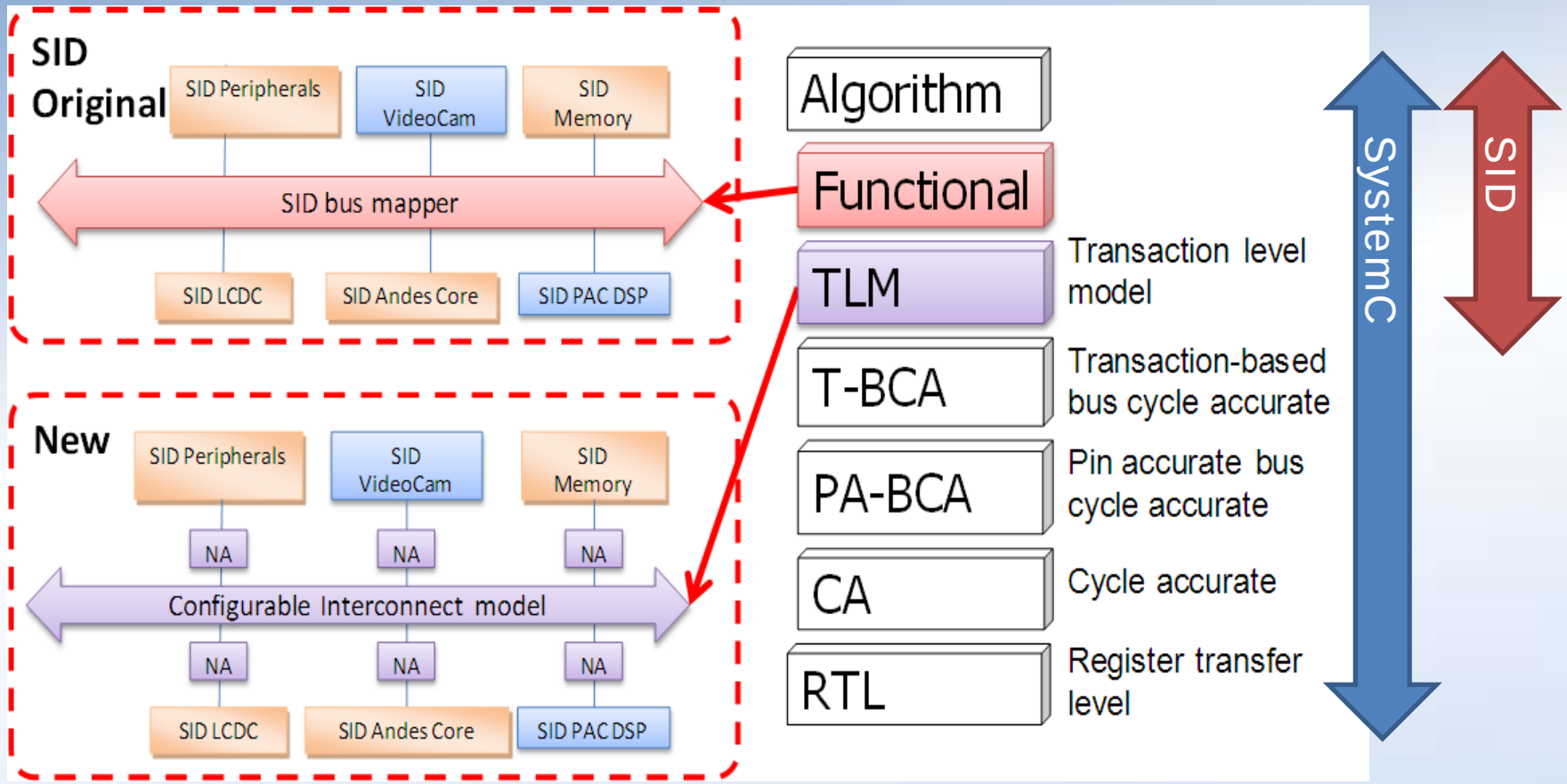
Items	CoWare™ Architecture Designer	SID Simulation Framework	QEMU open source processor emulator
Language	C++/SystemC	C++/C/Tcl	C
IP-based (Configurable)	Yes	Yes	No
Free	No	Yes	Yes
Source code	No	Yes	Yes
Scheduler / Clock Gen	Central clock source	Individual time quantum	none
Performance Accuracy of Simulation	High (Most IPs are cycle-accurate.)	Middle (Processor IPs are cycle-accurate.)	Less (Most IPs are functional-accurate.)

SID Simulation Framework



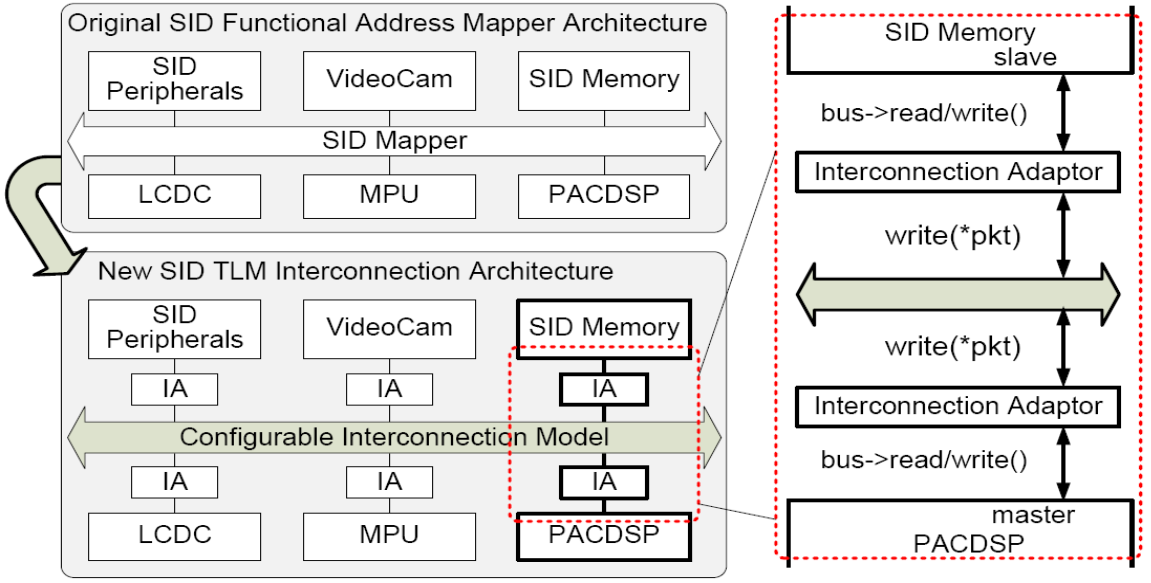
- According to the types of modeling targets, SID components can be classified into hardware, software, and system components.
- The most important system components are the main controller (*cfgroot*) and the *scheduler* component.
  - The *cfgroot* controls the compositions of components and the main execution of simulation.
  - The *scheduler* then arranges the execution sequences of components.

Layers of Simulation Models



Configurable Interconnection Model

- For the original SID framework, the bus mechanism is emulated by look-up tables and the memory addresses are mapped to the read/write functions of components.



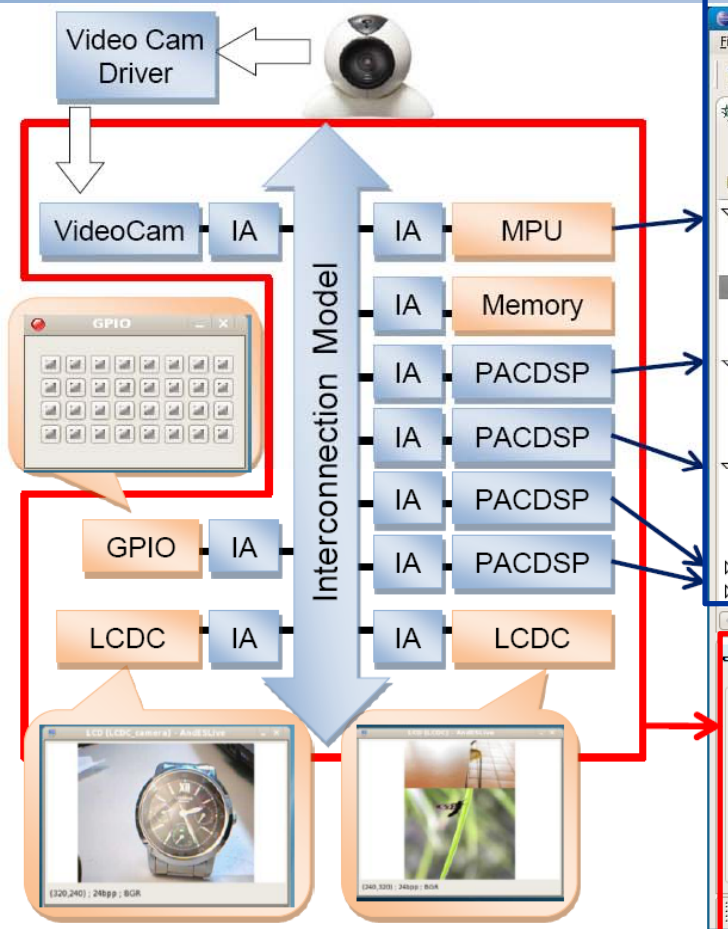
- We add new data communication flows of the interconnection networks.
  - An new interconnection adaptor (IA) accepts the bus control and transforms data packages.
  - The new interconnection component is a package of mathematical models with the statistics of package transmission history and a probabilistic analysis of communication predictions.
  - The interconnection model now can be configured as bus, crossbar, and on-chip network.

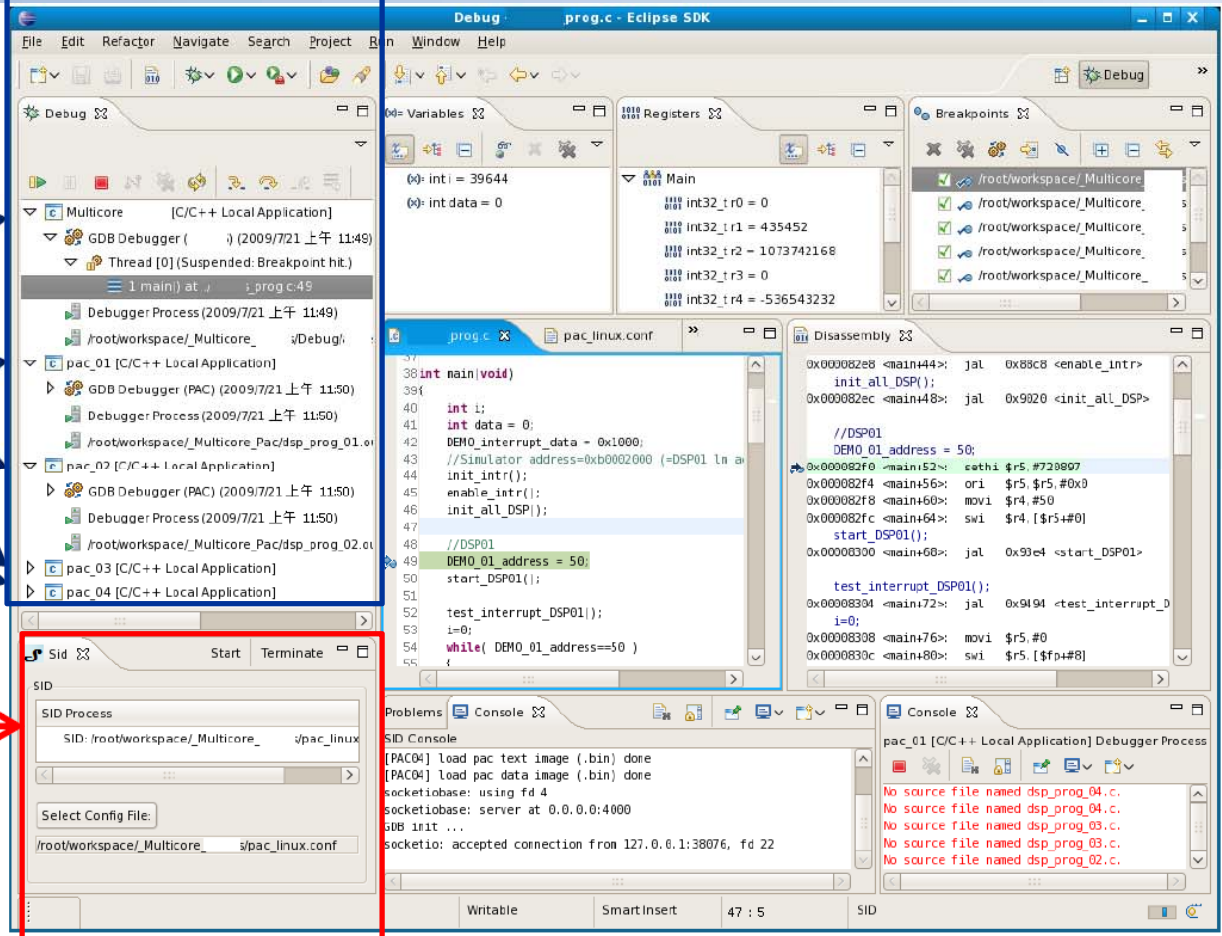
Heterogeneous Multicore Simulation Platform

SID simulation framework

Control simulator by GDB protocol

Multi-core Eclipse-based IDE

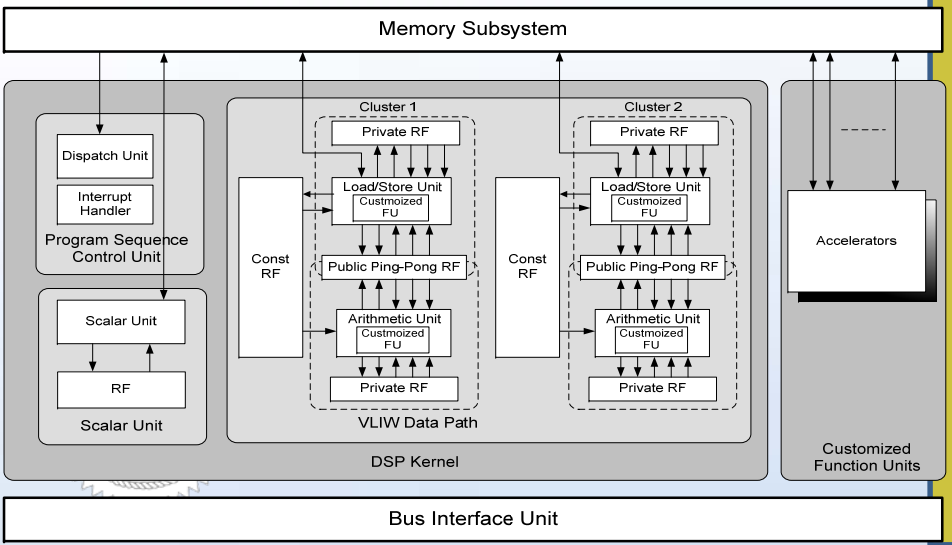


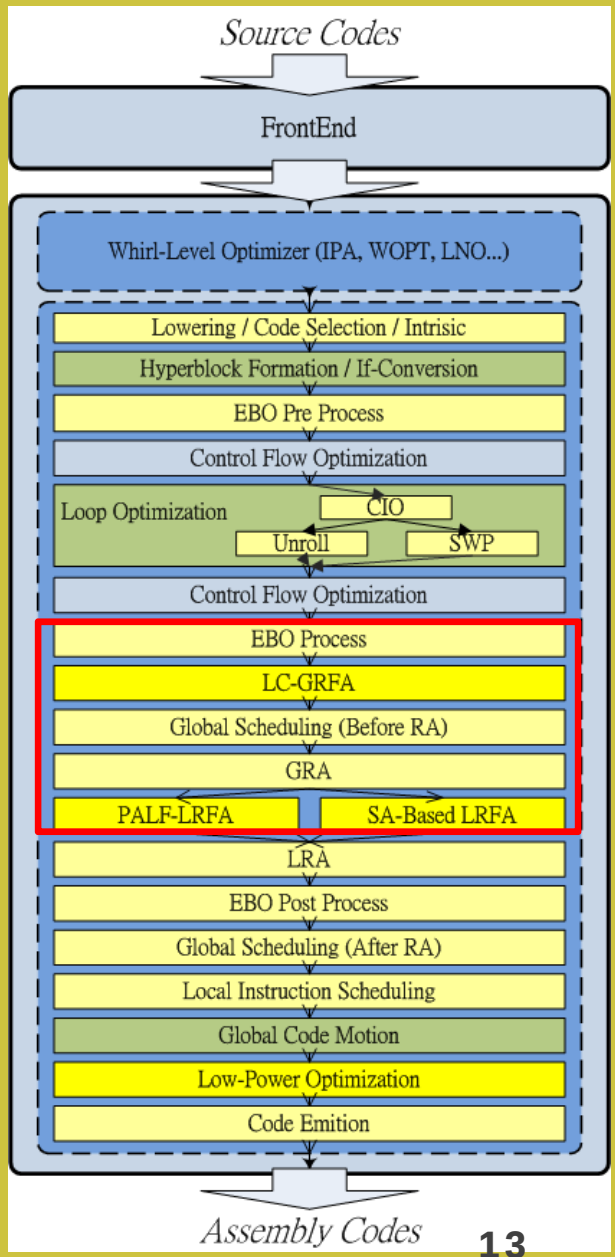


Simulation launcher

PACDSP Compiler

- Based on Open64 compiler
- VLIW DSP compilers for distributed register files
- PALF scheduling policies for ILP (CPC 2006)
- GRA scheme for distributed register files (CPC 2007)
- SIMD compiler optimizations + intrinsics/extrinsics
- Copy propagations for distributed register architectures (LCPC 2006)
- Register spills among distributed register banks (CPC 2009)

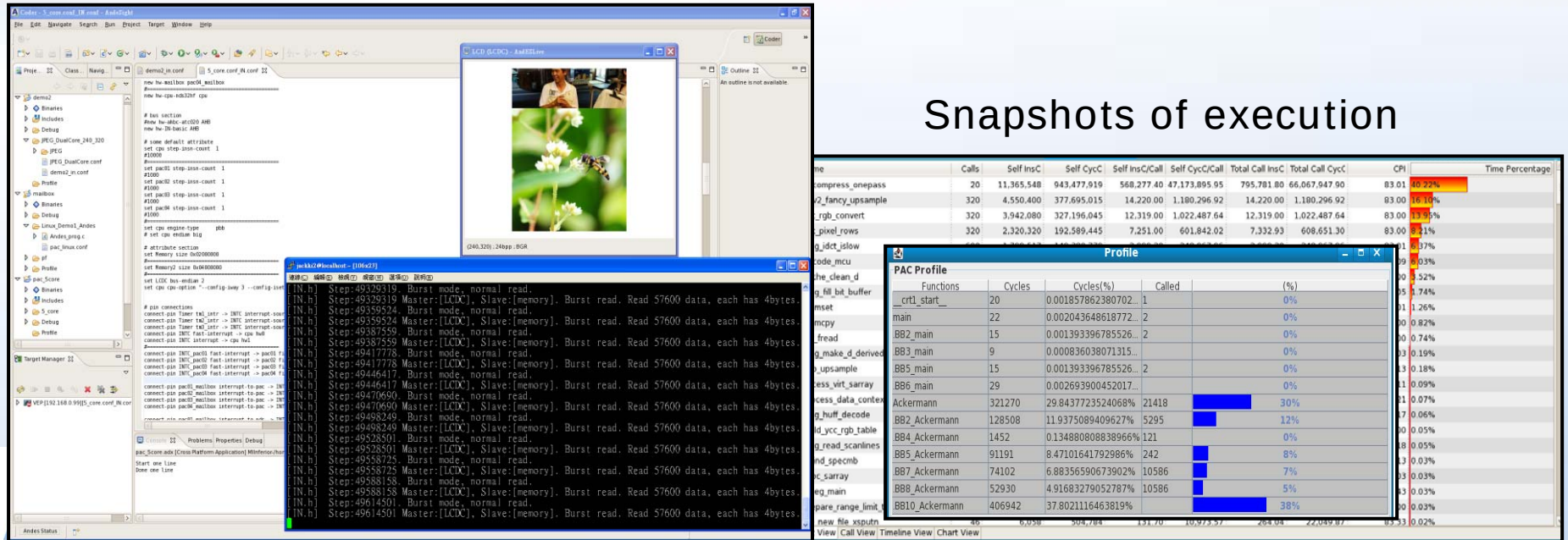




SID Framework for Embedded System Course

- The tool is developed to hope to be able to provide labs for several graduate embedded courses.
  1. *Toolchain for Embedded Software (ES-Y04-2)*
  2. *Embedded Compiler Design (ES-Y05-1)*
  3. *Embedded Multimedia Design (ES-Y08-1)*
  4. *Embedded Hw/Sw Co-design and Analysis (ES-Y08-3)*
  5. *Heterogeneous Multi-cores Course (ES-Y09-1)*

Snapshots of execution



SID Framework for Embedded Courses

- The on-going courses and lab modules on the SID simulation framework at NTHU EE/CS.
  1. Category of System Tool: (Prof. Jenq Kuen Lee)

     Embedded Compiler Design (ES-Y05-1), Special Topics on Advanced Programming Languages

     - Streaming RPC on Multi-core Platform
     - Enable the Software Cache on DSP Processors

  2. Category of Architecture: (Prof. Yarsun Hsu)

     Advanced Computer Architecture, Embedded Hw/Sw Co-design and Analysis (ES-Y08-3)

     - Comparisons of SID Interconnection to SystemC TLM IP

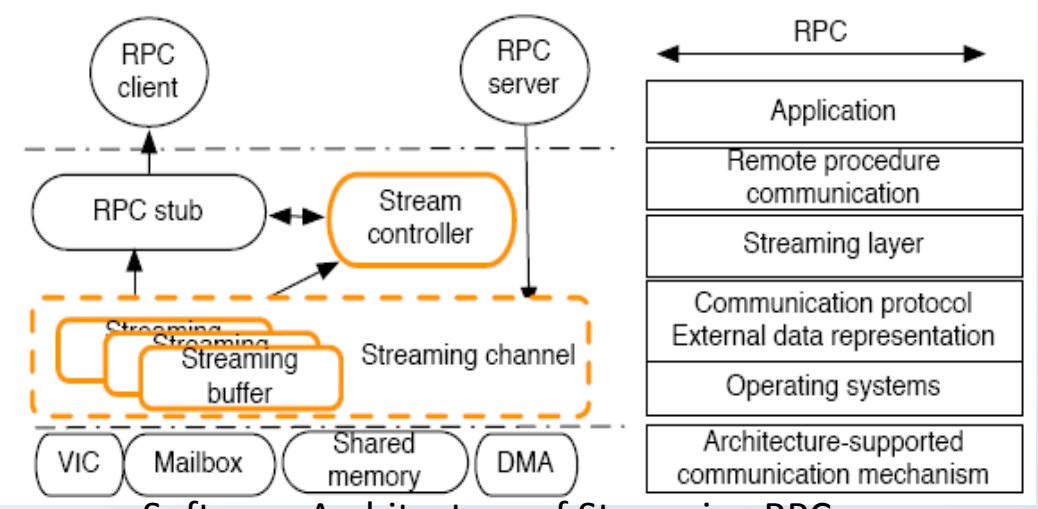
  3. Category of Applications: (Prof. Wei-Kuan Shih and Prof. S. H. Lai)

     Embedded Multimedia Design (ES-Y08-1), Toolchain for Embedded Software (ES-Y04-2), Special Topics on Embedded System Designs

     - Face Recognition Applications
     - Belief Propagation Method for Stereo Vision

Streaming RPC on Multi-core Platform

- Key environment setting:
  MPU, DSP, Interconnection, Interrupts, and OS
- Project goal:
  Experiment with DMA effects with streaming remoting programming model on SID multi-core platforms



Software Architecture of Streaming RPC

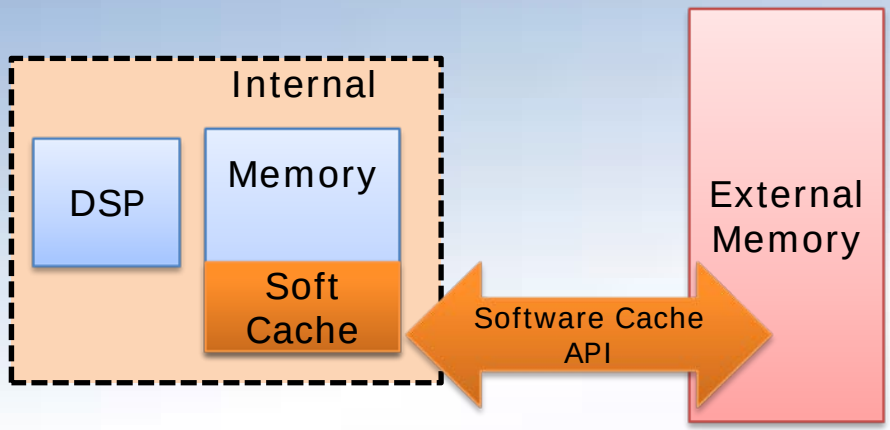
** Enabling Streaming Remoting on Embedded Dual-core Processors, Kun-Yuan Hsieh, Yen-Chih Liu, Po-Wen Wu, Shou-Wei Chang, Jenq Kuen Lee, ICPP 2008, Portand, Oregon, Sep. 8-12, 2008.

Experiments with Software Cache on DSP Processor

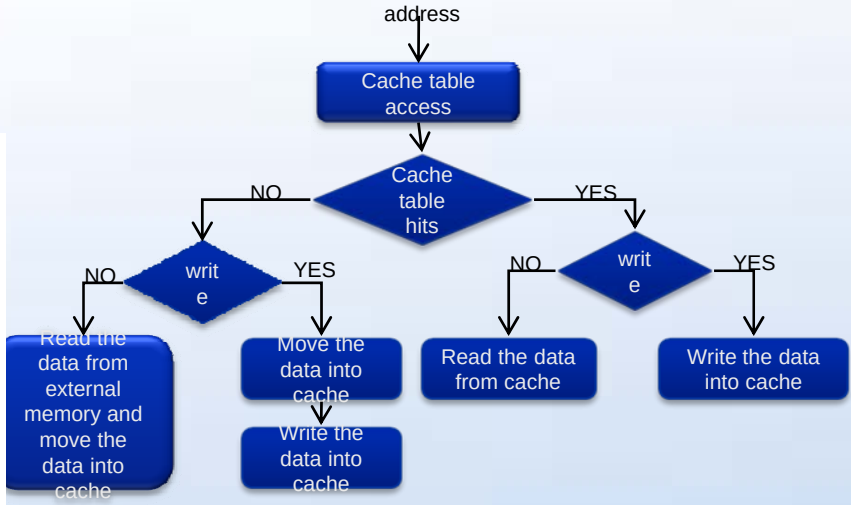
- Key environment setting:

  MPU, DSP, Interconnection, DMA, and Local/External Memory

- Project goal:
  1. Implement Software Cache on the non-data-cached PAC DSP.
  2. Experiment with DMA to accelerate the software cache.

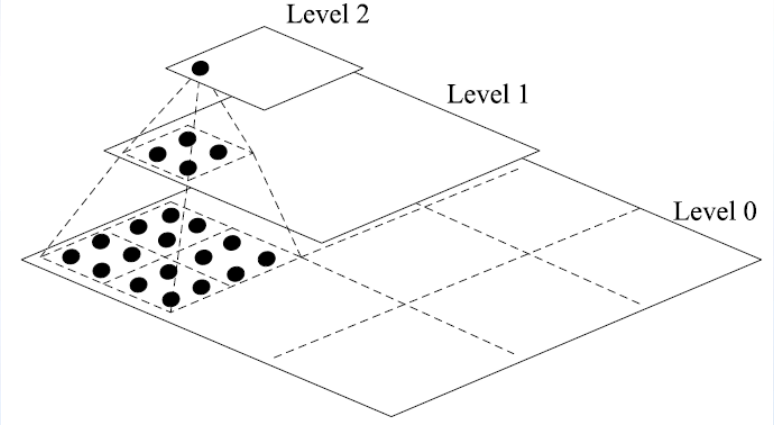


Name	Description
pCore_sw_cache_read()	Return the data value from the cache.
pCore_sw_cache_write()	Write the value into the cache.
pCore_sw_block_read()	Fast data block read function reducing address lookup overhead.
pCore_sw_block_write()	Fast data block write function to the external memory.
pCore_sw_block_trans_done()	Indicate that the block transaction is done.
pCore_sw_cache_lock()	Function that locks the specified cache line.
pCore_sw_cache_unlock()	Function that unlocks the specified cache line.

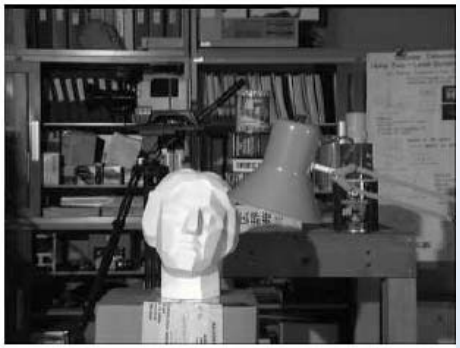


Belief Propagation Method on Embedded Multi-core Processors for Stereo Vision

- Key environment setting:
  MPU, DSP, DMA, and Local/External Memory
- Project goal:
  Parallelize the belief propagation method on multi-core PAC DSP environment
- Strategies:

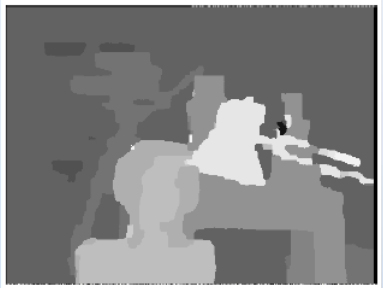


Building the data pyramid from lowest level data layer



(a) Original image pair.

(b) Ground-truth.



(b) Result of parallelized BP.

Experiments with SID Interconnection and SystemC TLM IP

- Key environment setting:
  Processor, Interconnection, Memory, and CoWare™ tools
- Project goal:
  1. Evaluate performance variations of benchmarks in SID and CoWare™ SystemC IPs.
  2. Evaluate accuracy variations of benchmarks in SID and CoWare™ SystemC IPs.
- Strategies:
  - Trace the accessing patterns of SID interconnections and compare to the SystemC TLM model.

Feedbacks and Experiences

- Virtual I/O components provide the graphical experience to attract students' interests on the multimedia projects.
- Embedded multi-core platforms can be configured for embedded multi-core compiler testbed.
- The interconnection of the multi-core platform in our testbed can reflect different topology and communication types.
- For debugging multi-core applications, the multi-core IDE can provide a source-code-level debugging.
- For a difficult bug or a hardware bug, students can use *trace unit component* to record the detailed activities of the whole system.
- Bugs and version fixes slow down student project progresses.
- TAs efforts are needed to help move lab. forwards.

Conclusion

- We developed an embedded multi-core platform based on SID framework.
- It included MPU, DSP, interconnection, and trace unit. .
- It accommodated ingenious local IPs such as MPU and DSP.
- It's useful for a variety of experiments and lab modules with embedded multi-core topics.
- Several courses are designed based on this tool for lab experiments.