

# New ADL and Its Platform for Computer Architecture Simulation

Christopher Barnes and Jaehwan John Lee, PhD  
Department of Electrical and Computer Engineering  
Purdue School of Engineering and Technology  
Indiana University Purdue University Indianapolis

## Abstract

Computer architecture simulation has always played a pivotal role in continuous innovation of computers. Thus, the Computer Architecture Lab in the ECE department of IUPUI is dedicated to develop the most promising computer architecture simulation framework for the multithreaded simulation of both uni-core and multi-core processors. Along this line of research, the CADL Compiler was developed to extend the multithreaded simulator currently in development at the ECE Department Computer Architecture Lab. The purpose of the compiler is to process the proprietary Computer Architecture Description Language (CADL) and output a fully functioning multithreaded simulator program. Utilizing an industry standard extensible markup language called XML, CADL is a language developed at the ECE Department of IUPUI to describe the functionality and architecture of the processor for simulation. The CADL compiler is necessary to extend the simulator by allowing users to easily and quickly modify the structure, instruction set, and execution of the processor modeled within the multithreaded simulator.

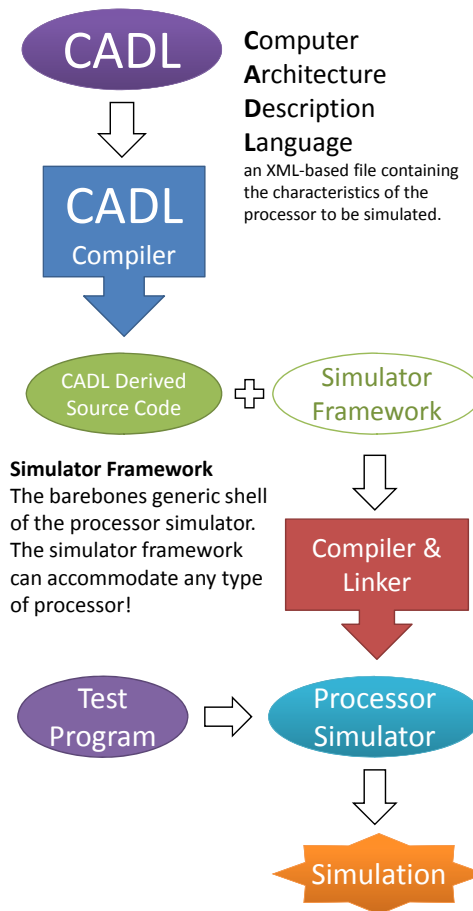
## Motivation & Objective

Researchers in both industry and academia leverage hardware simulation to evaluate new designs and novel ideas of computer architecture [1]. Further development of multithreaded hardware simulation is needed as more and more computers adopt a multiprocessing approach [2]. Multithreaded simulation means that each processing core of the host executes one or a set of modules of the simulator to reduce simulation time. To fulfill this need, the Computer Architecture Lab of the ECE Department of IUPUI is developing a multithreaded simulator framework for researching computer architecture.

Moreover, most current simulators are monolithic, meaning they simulate predefined computer architectures. To address the need of researchers to simulate different computer architectures, a new language was developed. The new language developed by the ECE Department of IUPUI was termed Computer Architecture Description Language (CADL). CADL utilizes an extensible markup language called XML [3], a standardized text-based data format, to describe a computer architecture for the purpose of simulation. XML is used to easily read and modify data format, which allows researchers to quickly alter a CADL document, thus changing the characteristics of the hardware to be simulated. With the aid of the CADL, a simulator designer can specify every aspect of the simulated hardware and automatically construct simulators [4].

## Methodology

The CADL Compiler inputs a user-defined ADL file and outputs the derived C++ source code. The resulting source code is combined with the simulator framework and compiled in to the final processor simulator. Test programs can then be executed on the newly created simulator.



## Development

- The CADL Compiler and Processor Simulator were developed on Microsoft Windows 64 Bit (Vista and XP) using Microsoft Visual Studio 2008 (C++.)
- Intel Core 2 Quad processor based systems were used for testing and development.
- Development of the CADL file specification, CADL Compiler, and processor simulator framework was completed over two months.

## Contributions

- Processor simulators are used to test new processor concepts and designs.
- The easy-to-use CADL enables the user to quickly understand & modify the simulator.
- Students can use the CADL Compiler & Processor Simulator for learning about Computer Architecture by adding to and modifying the processor's resources.

## Future Research

- Research is currently underway to produce a visual display of running processor simulations.
- Additional processor types will be provided in future simulator distributions.

## References

1. Kevin Skadron, Margaret Martonosi, David I. August, Mark D. Hill, David J. Lilja and Vijay S. Pai, "Challenges in Computer Architecture Evaluation," IEEE Computer, 36(8), pp. 30-36, Aug. 2003.
2. D. Geer. "Industry Trends: Chip Makers Turn to Multicore Processors," IEEE Computer, 38(5), pp. 11-13, 2005.
3. Extensible Markup Language (XML), <http://www.w3.org/XML/>, visited March 2008.
4. S. Onder and R. Gupta, "Automatic generation of microarchitecture simulators," IEEE Int. Conf. Comput. Languages, pp. 80-89, 1998.

# New ADL and Its Platform for Computer Architecture Simulation

Christopher Barnes, Pranav Vaidya and Jaehwan Lee  
Department of Electrical and Computer Engineering  
Purdue School of Engineering and Technology, IUPUI

## Abstract

The Computer Architecture Lab of the ECE Department at IUPUI has developed a novel computer architecture description language (CADL) for computer architecture teaching and research using simulation. The CADL utilizes the industry standard XML language to enable researchers and students to easily reconfigure and experiment with simulations of uniprocessors and multiprocessors. The easily approached CADL provides an opportunity to challenge students with computer architecture class projects and an ideal support tool for teaching computer architecture concepts.

## 1. Introduction

As processor technology becomes more complex, researchers are turning to hardware simulation to evaluate new designs and novel ideas. Most current simulators are monolithic, and thus, they simulate predefined computer architectures. To address the need of researchers to rapidly explore various computer architectures using simulation, a new language has been developed, which is termed Computer Architecture Description Language (CADL).

CADL utilizes an extensible markup language called XML [1], a standardized text-based data formatting language, to describe a computer architecture for the purpose of simulation. XML is used to easily read and modify data format, which allows researchers to quickly alter a CADL source file and thus change the characteristics of the hardware to be simulated. With the aid of CADL, a simulator designer can specify every aspect of the simulated hardware and automatically construct simulators with various configurations.

Students and teachers of computer architecture stand to benefit from the approachable CADL source format. CADL enables students to take part in an active learning process, which complements the traditional computer architecture learning environment. CADL provides a platform for students of every experience level to take part in design decision and experimenting with different processor configurations. Additionally, once a simulator is constructed, students can assemble different programs to test their simulator's functionality and performance.

CADL provides an entirely open and configurable environment for processor simulation. Since the CADL source format enables quick and easy modification of the computer architecture, entirely new and radical instruction sets and processors can be developed. For example, the

simulator framework places no restrictions on the amount or purpose of processor stages. This sort of open environment encourages creativity for researchers choosing to delve deeper in to the inner-workings of the processor simulator.

```

- <Channels>
  - <Channel name="IF2ID">
    <Output>IF</Output>
    <Input>ID</Input>
    - <Data>
      <uint32>fetchedInstruction</uint32>
      <uint32>originPC</uint32>
    </Data>
  </Channel>
  - <Channel name="ID2EX">
    <Output>ID</Output>
    <Input>EX</Input>
    - <Data>
      <DecodedInstruction>decodedInstruction</DecodedInstruction>
      <uint32>rs_data</uint32>
      <uint32>rt_data</uint32>
      <uint32>base</uint32>
      <uint32>originPC</uint32>
      <bool>nop</bool>
    </Data>
  </Channel>

```

Figure 1: CADL Source Code Example, Structure View

contains the details of the intra-stage communication, memory system and cache system (see Figure 1). The execution view contains the details of the stages and processor execution concurrency on the host machine. Additionally, the execution view contains a link to the stage logic specific to each instruction set architecture. Lastly, the instruction view contains the details of the registers in the processor. Further details on the usage of the CADL source code are provided in the CADL platform distribution.

```

- <ExecutionView>
  <Processor run="InParallel" cores="1" name="Simple MIPS" />
  - <Stages run="InParallelOneThread">
    <IF code="sim\adl_input\stage_if.cpp" />
    <ID code="sim\adl_input\stage_id.cpp" />
    <EX code="sim\adl_input\stage_ex.cpp" />
    <MEM code="sim\adl_input\stage_mem.cpp" />
    <WB code="sim\adl_input\stage_wb.cpp" />
  </Stages>
</ExecutionView>

```

Figure 2: CADL Source Code Example, Execution View

Each pipeline stage corresponds to one C++ source file, referenced in the execution view section of the CADL source code (see Figure 2). These stage files contain the stage behavior code needed to execute the ISA in the simulator. Source code for a five stage MIPS64 ISA is included with the CADL platform distribution. Researchers who wish to alter the stage behavior or construct a new ISA can do so by altering these files.

### 3. Building a Simulator with CADL

The CADL source code passes through several phases to before becoming the executable simulator. The CADL compiler eases the difficulties of this process by handling each process in

## 2. CADL Source Files

The CADL source code is an XML-based [1] file containing the core description of the computer architecture to be simulated. The source code can be viewed and modified with a standard text editor or a specialized XML editing tool.

At a high level, the CADL source code contains three main sections: the structure view, execution view and instruction view. The structure view

Each pipeline stage corresponds to one C++ source file, referenced in the execution view section of the CADL source code (see Figure 2). These stage files contain the stage behavior code needed to execute the ISA in the

simulator. Source code for a five stage MIPS64 ISA is included with

a single step. Though this process requires a one-time configuration by the researcher, future modifications to the CADL document are easily implemented by simply executing the CADL compiler.

Upon execution, the CADL compiler imports and processes the CADL source code and outputs the corresponding C++ code. The newly formed C++ code is placed in to a generalized simulator framework which contains the minimum amount of logic needed to execute the processor-specific logic. The CADL compiler then invokes a command-line compiler and linker yielding a final simulator executable from the combined source code. Finally, the CADL compiler can (optionally) execute the final simulator once the linking is completed.

Researchers who choose to debug or perform extensive development with the simulator may choose to use the CADL compiler-generated C++ source code with an integrated development environment (IDE.) Instructions for development of the CADL compiler and simulator within Microsoft Visual Studio [2] are included with the CADL distribution.

#### 4. Using CADL for Computer Architecture Education

The CADL compiler provides a unique platform for students to experiment with computer architecture. Students can modify the processor's high level architecture such as registers, memory and cache configuration using the CADL framework. In addition, students can write and simulate new programs to complement their new configurations. Additionally, students seeking a greater challenge can also modify the low level details of the simulator by modifying the ISA's stage logic.

The CADL platform presents a wide range of potential projects to challenge students. Initially, students can become familiar by manipulating high level configuration without being overwhelmed with low level details. For example, students can test the effectiveness of different cache mapping functions such as direct and set associative, as well as replacement schemes. As familiarity with the simulator grows, students can be given more advanced tasks such as the implementation of a branch predictor or the development of a modified or even new ISA.

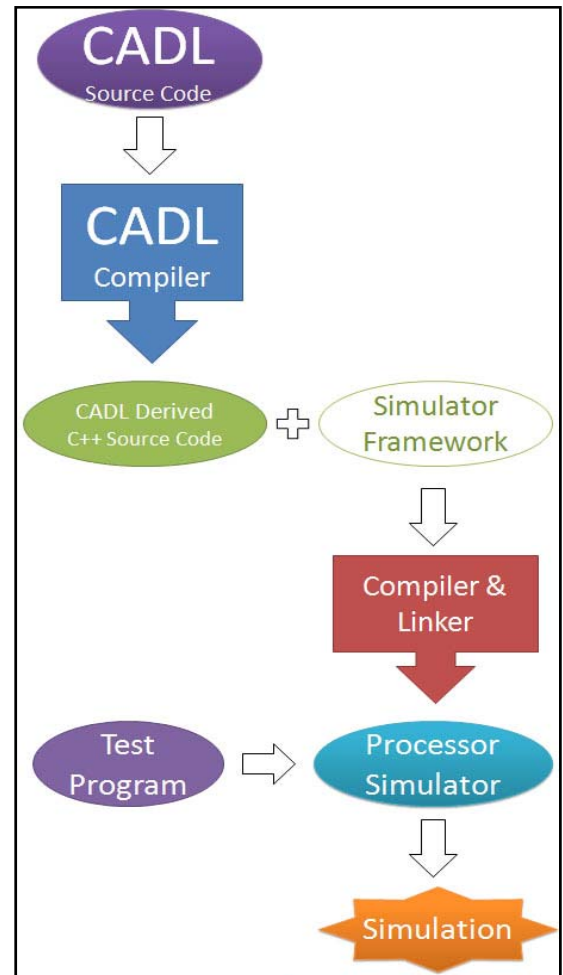


Figure 3: CADL Source Code to Simulation Data Flow

## **5. Future Research**

Research is currently underway on a visual component which will enable researchers and students to see the propagation of data through each stage of the processor. Future versions of CADL will be ported to different operating systems. Lastly, additional instruction sets will be developed to enable students and researchers get started quickly in their ISA of choice.

## **6. Conclusions**

CADL is an emerging computer architecture research platform developed by the ECE department at IUPUI. The CADL platform removes the steep learning curve of writing a processor simulator and encourages researchers and students to experiment with and learn computer architecture. CADL is approachable to computer architecture novices with the easily modifiable CADL source code files. Additionally, CADL is relevant to advanced users who wish to extend the simulator's feature set or construct new ISAs.

The CADL platform is an ideal support tool for computer architecture education. The CADL simulator framework with the CADL compiler removes the low-level complexity of writing processor simulators, allowing students to focus their learning effort on the processor's functionality. Supplementing the computer architecture class material with the CADL platform will help reinforce class material and engage students in the learning process.

## **7. References**

- [1] Extensible Markup Language (XML), <http://www.w3.org/XML/>, visited July 2008.
- [2] Visual Studio Developer Center, <http://msdn.microsoft.com/en-us/vstudio/default.aspx>, visited July 2008.