CS 6291: Embedded Software
Summer 2020

Instructor
Santosh Pande
E-mail
santosh@cc.gatech.edu

Office: Klaus 2338
Office Hrs: Google Hangout (TBD)

TAs E-mails:
(1) Tyson Bailey (tbailey38@gatech.edu)
(Head TA)
(2) Diane Nguyen
(dnguyen78@gatech.edu)
(3) Greg Gorlan (ggorlen3@gatech.edu)

Course Objectives

- To understand the tight coupling and synergies that exist between hardware and software in embedded processors and that are exposed through the abstraction of instruction sets including DSPs, VLIWs, etc.
- To understand how machine specific features (such as data paths, register and memory banks) of the underlying processor and machine specific code optimizations based on them for high performance, compact code and low energy.

Course Description

In 21st century, embedded systems are the systems of future with cellular phones, smart-phones, tablets becoming the dominant platforms for computing and communication. Ubiquity of information and the associated need for the computation that accompanies it, is driving this revolution only to be accelerated by the new paradigms such as the Internet-of-Things (IoT). These platforms are clearly very different in terms of their processing requirements which are very unique in terms of real-time responsiveness needs, high performance but at low energy, compact code and data segments and most importantly ever changing software stack. Such unique requirements have led to a complete redesign and reinvention of the both hardware and the software from ground up, for example, brand new processors such as ARM, DSPs, network processors were invented all the way up-to new virtual machines such as Dalvik, new operating systems such as the Android and new programming models and compiler optimizations. The goal of this course is to take a holistic view of the embedded system stack foundation with a focus on processor architectures, instruction sets and the associated advanced (machine specific) compiler optimizations that take advantage of the same.

Prerequisites

Students should have taken an undergraduate course in computer architecture. A strong background in C and/or C++ is required

Textbooks:

NOTE: The following book is available in electronic form in Georgia Tech library and enrolled students can access the book in that format

(1) Title : Embedded Computing: A VLIW Approach to Architecture, Compilers and Tools 1st Edition

by Joseph A. Fisher , Paolo Faraboschi , Cliff Young Hardcover: 712 pages
Publisher: Morgan Kaufmann; 1 edition (December 31, 2004)

ISBN-10: 1558607668
ISBN-13: 978-1558607668

NOTE: The following book is **not available i**n Georgia Tech library in electronic book format but is an excellent reference book and students should purchase a copy for their personal use.

(2) Title: Compilers: Principles, Techniques, and Tools (2nd Edition)

by Alfred V. Aho, Monica S. Lam, Ravi Sethi and Jeffrey D. Ullman

Publisher: Addison Wesley; 2nd edition (September 10, 2006)
Hardcover 1000 pages
ISBN-10: 0321486811
ISBN-13: 978-0321486813

Course Outline:

Part I: Embedded Processor Architectures

1. Introduction to instruction level parallelism: Pipelining, RISC vs CISC, Very Large Instruction Words (VLIW) instruction sets, Hardware complexity (Superscalars) vs Compiler Optimizations (VLIWs) Tradeoffs

2 Design of Instruction Set Architectures: VLIW encoding, Exposing vs Hiding Architectural Details, RISC vs CISC ISAs, Opportunities for compilers, Dependences and Independences, Instruction bundling for VLIW, Compact instruction representation.

3. Embedded Micro-architectures: Scratch-pad: software managed memory, clustered register files, special arithmetic, addressing modes for special needs (DSPs), branches in embedded domains: speculation and predication, unbundling

branches,

Part II : Software Optimizations
(Machine Specific Compiler based optimizations)

4. Introduction to Compiler phases: Overall working of the compiler, overview of phases, intermediate representation, backend code generation issues.

5. Register Allocation Foundation : RISC philosophy (load, store architecture), Live range analysis, Interference Graph, Graph Coloring Based Register Allocation, Live Range Splitting

6. Register Allocation for Embedded Processors: Post-pass register allocation, Allocation gaps and register reuse, Energy reduction due to reduced memory accesses, Differential register allocation, Register encoding, Hardware support, Increase in exposed registers, Software pipelining and energy reduction

7. Data Layouts for Embedded Processors: Auto addressing mode, Data layouts, Simple and general offset assignment problems, Address sequence optimizations, Memory coalescing, Data and code segment minimization

8. Data and Code Compaction: X-Y memory, Parallelizing Load/Stores in DSPs, Data replication, Performance vs Data Segment/ size, ARM vs Thumb code generation, Mixed code generation, Frequent values in embedded programs and their encoding, Data cache optimization via compaction.

9. Network Processors: Processing in the network, Network processors, Dual Bank Register Allocation for Network Processors, Multi-threading in network processors, Context switch and latency, Register allocation across threads to minimize latency

Emphasis and Grading

- Understand the instruction sets and micro-architectures for embedded processors
- Understand the compiler optimizations that take advantage of the specific embedded processor features
- Understand the synergies between the hardware design and software optimizations
- **Grading :** Homeworks : 30%, Project : 37%, Comprehensive Final : 33%

Course Educational Outcomes:

Upon successful completion of this course, students should be able to …

- Devise instruction sets and micro-architectures in domain specific manner optimizing common cases of execution for high performance efficiency and/or for lower energy.
- Undertake multi-objective optimization of performance, code density, code size and energy consumption through a careful demarcation of the design space between hardware and software
- Differentiate between design goals of different kinds of domain specific embedded processors: DSPs, VLIWs/GPUs, Network processors, ARM/Thumb, and Smartcards.
- Devise post-pass optimizers for removing inefficiencies of general purpose compilers that do not undertake machine specific optimizations (such as gcc)
- Devise a hard-ware/compiler solution that gets past the ISA limitations to expose more architected resources for higher efficiency.
- Devise high throughput optimizations for streaming applications such as packet processing in networks.
- Develop data layout solutions that take advantage of addressing modes in DSPs.

Academic Integrity

Academic dishonesty will not be tolerated. This includes cheating, lying about course matters, plagiarism, or helping others commit a violation of the Honor Code. Plagiarism includes reproducing the words of others without both the use of quotation marks and citation. Students are reminded of the obligations and expectations associated with the Georgia Tech Academic Honor Code and Student Code of Conduct, available online at www.honor.gatech.edu. All the software turned in as a part of this course will be thoroughly checked for plagiarism and fullest penalties will be imposed upon detection of violations of honor code.

Learning Accommodations

If needed, we will make classroom accommodations for students with documented disabilities. These accommodations must be arranged in advance and in accordance with the Office of Disability Services (http://disabilityservices.gatech.edu).

Excused Absence Policy http://www.catalog.gatech.edu/rules/4/