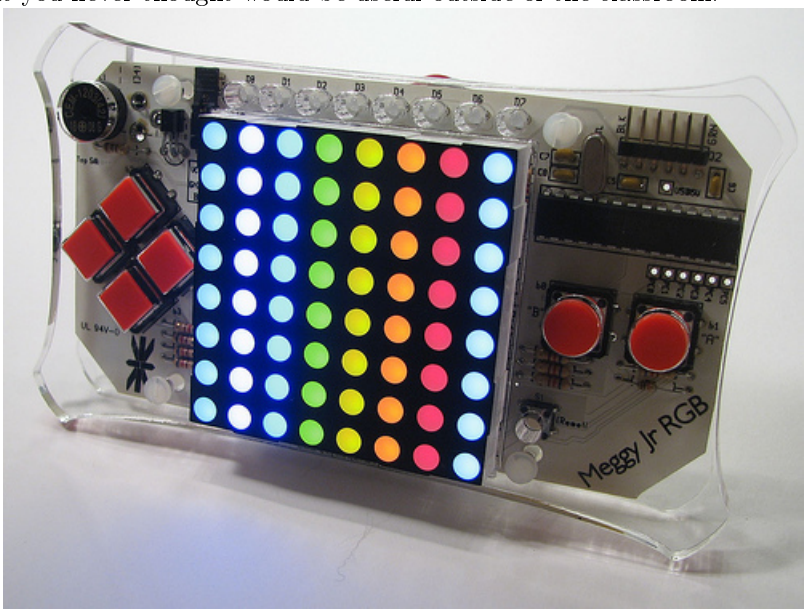
MeggyJava: Making the Compilers Course More Attractive to Undergraduate Students

Ryan Moore

September 2, 2010

Interest in the Topic

Compiler construction is an important area in Computer Science. Compilers are software packages that will compile(reduce) a higher level programming language to a machine language which computers understand and can run. Compilers have a great impact on how fast a program can run. If a compiler is not designed well it can produce code that is slow or incorrect, which makes compiler construction an important topic. Unfortunately many undergraduate students do not find the area interesting due either to the large amount of programming involved, or because they do not see how the skills learned in compilers are relevant to software development in general. I myself was not originally attracted to the idea of writing compilers mostly due to how much programming would be required, but after my adviser and I had identified this project I signed up for the compiler course the next day and loved the course. From a purely theoretical and intellectual standpoint, I found writing compilers very satisfying. It ties together a lot of theory with concrete examples. It ties together how good software engineering techniques can be used with actual Computer Science theory. It really makes all of the previous classes mean something, even the classes that you never thought would be useful outside of the classroom.



Project Scope

To make compilers more interesting I am going to modify the main project in the course so that instead of writing a compiler that generates MIPS assembly code the course will now write a compiler that generates AVR assembly code, which will run on the Evil Mad Scientist Laboratories Meggy Jr game platform.

The Meggy Jr looks similar to a GameBoy handheld system but with a 8 by 8 LED screen instead of a full screen. The reason for the switch is that currently when students generate MIPS assembly they can only run it inside the MARS Simulator. The students do not see their code doing anything particularly interesting on an actual machine. By making the main project target a machine where they can run their code after generating it, and see it work, will make the course more interesting and attract more undergraduate students.

Tasks To Support Project

There are several different major tasks that need to be completed to support the changes to the course project.

1. The first major task is to create a tool-chain, or build process, that students will use to compile their generated assembly code into an executable that will be uploaded to a Meggy Jr. This will include finding all the correct source files that target the Meggy Jr, as well as finding and installing a compiler that will assemble the generated assembly code into an executable. The final process is to then make it easy enough so that students can take their generated assembly and pass it into the tool-chain.

2. The next major task is to update the reference compiler that is used in the course to generate AVR assembly instead of MIPS assembly code. In addition to converting everything to AVR some new instructions will need to be added to the MiniJava language to support the new Meggy functions for editing the display state of the Meggy Jr as well as reading button presses from the Meggy Jr.

3. The last major task is to implement a simulator that will be used for grading as well as allowing students to debug the assembly code generated by their compiler before uploading it to the Meggy Jr device. The simulator will allow students to interactively step through their generated assembly code and watch to see what happens as each instruction is ran. This is the most important step because without this there is no way to accurately grade all of the student's compilers in a reasonable amount of time due to differences in the generated assembly. The simulator will allow for the generated assembly programs to be ran in a batch mode where the programs are ran to completion so that grading can be automated and the TA/Professor will not have to grade each generated assembly file by hand. This simulator will implement a subset of the AVR assembly such that students will be able to simulate what will happen on an actual device.

End Result

By the end of the project I will have modified the main project in the Compiler course to make it more interesting to undergraduate students, who may not otherwise want to take the course. In support of this I will have created a tool-chain that will take a AVR assembly file and build it into an executable that can be loaded onto a Meggy Jr. In addition to this I will provide a simulator that will simulate AVR assembly so that student's will be able to locally test their programs before they upload it to a Meggy Jr. The simulator will also be used to grade the student's compilers by checking the assembly code that they generate. The last part that will be completed is to update the reference compiler to include the new Meggy Jr instructions as well as compiling to AVR instead of MIPS assembly.

Time Line

- Tool-Chain Completed - September 8, 2010
- Constant Expressions - September 14, 2010
- Function Calls - September 22, 2010
- Simulator Batch Mode - September 29, 2010
- Control Flow Constructs - October 6, 2010
- Basic Simulator Gui - October 13, 2010
- Classes And Arrays - October 20, 2010
- Penultimate Draft/Reflection Paper - Will Present to Thesis Adviser and Committee Member on October 20, 2010
- Stepping and Breakpoint Control in Simulator - November 6, 2010
- Oral Presentation - November 12, 2010 Computer Science Building 345 4:00pm - 5:00pm

Resources

- Michelle Strout - Thesis Adviser and Compiler Professor
- Current CS453 Website - Outline of Compiler Development Iterations - http://www.cs.colostate.edu/~cs453
- MiniJava Grammar Web page - Description of the MiniJava Language - http://www.cambridge.org/resources/052182060X/MCIIJ2e/grammar.htm
- ATmega328p Feature Document - Outline of the AVR Instruction Set and the Major Features of the Atmega328p processor - http://www.atmel.com/dyn/resources/prod_document
- ATmega328p In-depth Feature Document - Full Description of Instruction Set and Processor Design - http://www.atmel.com/dyn/resources/prod_documents/doc8271.pdf
- Meggy Jr Web page - Description of Meggy Jr - http://www.evilmadscientist.com/article.php/meggyjr